small methods
www.small-methods.com

NanoLocz: Image Analysis Platform for AFM, High-Speed AFM, and Localization AFM

George R. Heath, Emily Micklethwaite, and Tabitha M. Storer*

Atomic Force Microscopy (AFM), High-Speed AFM (HS-AFM) simulation AFM, and Localization AFM (LAFM) enable the study of molecules and surfaces with increasingly higher spatiotemporal resolution. However, effective and rapid analysis of the images and movies produced by these techniques can be challenging, often requiring the use of multiple image processing software applications and scripts. Here, NanoLocz, an open-source solution that offers advanced analysis capabilities for the AFM community, is presented. Integration and continued development of AFM analysis tools is essential to improve access to data, increase throughput, and open new analysis opportunities. NanoLocz efficiently leverages the rich data AFM has to offer by incorporating and combining existing and newly developed analysis methods for AFM, HS-AFM, simulation AFM, and LAFM seamlessly. It facilitates and streamlines AFM analysis workflows from import of raw data, through to various analysis workflows. Here, the study demonstrates the capabilities of NanoLocz and the new methods it enables including single-molecule LAFM, time-resolved LAFM, and simulation LAFM.

1. Introduction

Open-source tools such as RELION[1,2] and numerous super-resolution microscopy software developments[3–8] have played a crucial role in the "resolution revolutions" of cryogenic electron microscopy (Cryo-EM) and super-resolution fluorescence microscopy (such as photoactivated localization microscopy (PALM), Stochastic optical reconstruction microscopy (STORM), and DNA points accumulation for nanoscale topography (DNA-PAINT)), respectively. The open-source and collaborative nature of these developments in data analysis have fostered a vibrant community of researchers, democratizing access to advanced imaging capabilities and driving significant progress in understanding biological structures and processes. AFM and HS-AFM can provide video rate 3D visualization of surfaces at nanometer resolution across a range of length scales from whole cells[9] down to single molecules.[10] These unique capabilities enable insight into structural biology questions under various physiological conditions including protein-protein interactions[11] and in situ dynamics of single molecules in response to force,[12] light,[13] ligands,[14] and drugs.[15] Furthermore, the application of single molecule localization microscopy analysis concepts to AFM data has led to the development of Localization AFM (LAFM)[16] which enables 4Å resolution to be achieved on single biomolecules. With the significant advancements in AFM methods and hardware, the need for accessible software to harness the full potential of AFM data becomes ever more important.

The wide range of proprietary file formats provided by AFM companies (which additionally change over time) makes it particularly difficult to access, share, and compare raw AFM data and apply new analysis methods. Current open-source software such as Gwyddion[17] and WSxM[18] enable a vast range of AFM image formats to be opened, catering to general scanning probe microscopy users but are focused on individual images. AFM image analysis tools aimed at biomolecular analysis such as TopoStats[19] and trace_y[20] concentrate on tracing analysis, proving powerful for the analysis of DNA and filament topology respectively. Additionally, to aid interpretation and analysis of biological structures in experimental data, simulation AFM tools such as BioAFMviewer have been developed to simulate and fit AFM topographies from and to PDB structures.[21,22] To better understand the influence of force on biomolecules such as deformation, simulation software dForce has been developed to simulate dynamic tapping of the AFM probe.[23,24] Despite these significant advancements, there remains a critical need for user-friendly AFM and HS-AFM post-processing and analysis tools that enable users of different instruments and expertise to browse, interact, and analyze raw or processed AFM data, especially HS-AFM data, with high-throughput.

G. R. Heath, E. Micklethwaite, T. M. Storer
School of Physics & Astronomy
Bragg Centre for Materials Research
University of Leeds
Leeds LS2 9JT, UK
E-mail: g.r.heath@leeds.ac.uk
G. R. Heath
School of Biomedical Sciences
Astbury Centre for Structural Molecular Biology
University of Leeds
Leeds LS2 9JT, UK

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/smtd.202301766

DOI: 10.1002/smtd.202301766

Table 1. Current AFM manufacturer file formats read by NanoLocz including access to different data channels and support for video processing.

AFM Manufacturer	Format Used	Read Height	Read All Channels	Video Support	Video Format	Author/Source
Bruker™	.spm	✔	✔	✔	Folder	J. D. Groot[26]
	.jpk	✔	✔	✔	Folder	R. D. Ortuso[27]
RIBM™	.asd	✔	x	✔	Single file	G. Tagiltsev & S. Scheuring
Oxford Instruments™	.aris	✔	✔	✔	Single file	G. R. Heath
	.ibw	✔	x	✔	Folder	J. Bialek[28]
Nanosurf™	.nhf	✔	✔	✔	Folder	G. R. Heath
	.gyw	✔	✔	✔	Folder	E. L. Fricke[29]
Park Systems™	.tiff	✔	x	✔	Single file	G. R. Heath

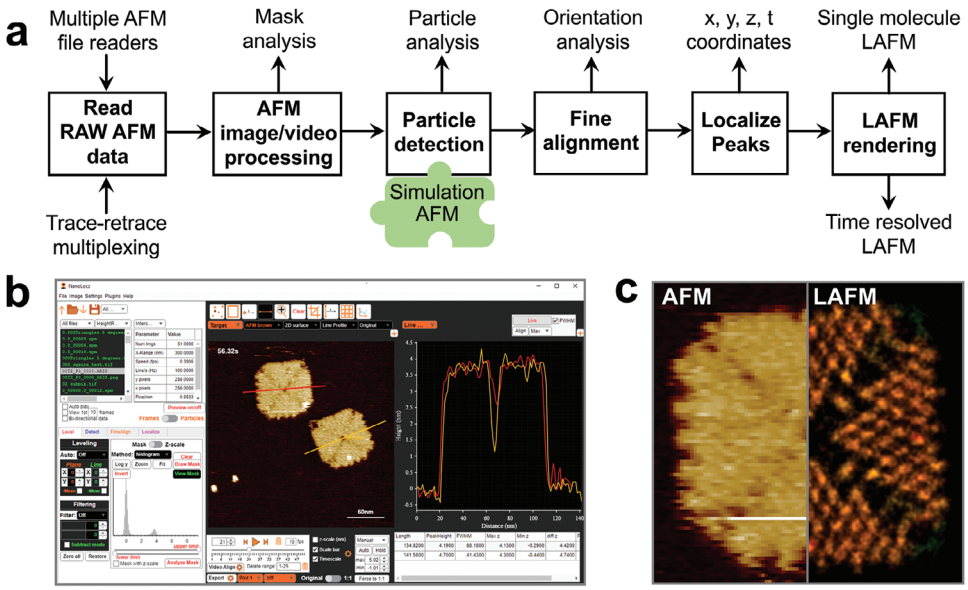

Figure 1. NanoLocz workflow. a) Overview of NanoLocz workflow: multiple AFM file readers allow data to be imported from different instruments and processed through the same workflow. Simulation AFM is incorporated as a plugin allowing the import of simulated topographies to use in particle detection or to simulate LAFM imaging. Multiple branch points demonstrate the flexibility and options for various analysis output. b) GUI of NanoLocz demonstrating multiple line profile analysis (red and yellow lines and traces). c) Comparison of a single AFM image (left) and a NanoLocz LAFM map (right) of a DNA origami tile.

Here we present NanoLocz, an open-source software developed for advanced AFM, HS-AFM, and LAFM data analysis integrated with simulation AFM and accessed through an interactive graphical user interface. We show and explain the fundamental and new methods enabled by NanoLocz which provide users with an AFM specific tool for seamless and in-depth analysis of AFM image data.

2. Results and Discussion

Starting from raw data AFM topography images or movies, NanoLocz is has been developed to have multiple possible workflows and branch points depending on the desired analysis (**Figure** 1a). All workflows can be performed in a user-friendly graphical user interface (GUI) (Figure 1b).

To read in raw AFM image or HS-AFM video files, we modified existing and developed new file format readers allowing topography trace, retrace scans, or other data channels to be processed. The workflow of NanoLocz treats AFM data in either single image or video format, i.e., a stack of images, whilst allowing seamless integration with various AFM and HS-AFM data formats, making it suitable for data obtained from different experimental setups. See **Table** 1 for the full list of accepted file formats, access to different data channels, and file organization required for video format. Raw or processed AFM file formats are automatically recognized and imported as either single images or movies along with metadata of each frame. To increase the amount of data included in the AFM analysis, we develop a routine to load and intercalate trace and retrace movie frames (**Figure** 2). Trace-retrace intercalated movies enable a twofold increase in frames, this is particularly powerful when applied to reduce noise by averaging the trace-retrace signals or to increase the number of particles for LAFM analysis and therefore increase spatial resolution or temporal resolution of time-resolved LAFM. Trace-retrace intercalated movies typically require frame alignment to account for substantial offsets in the x direction.

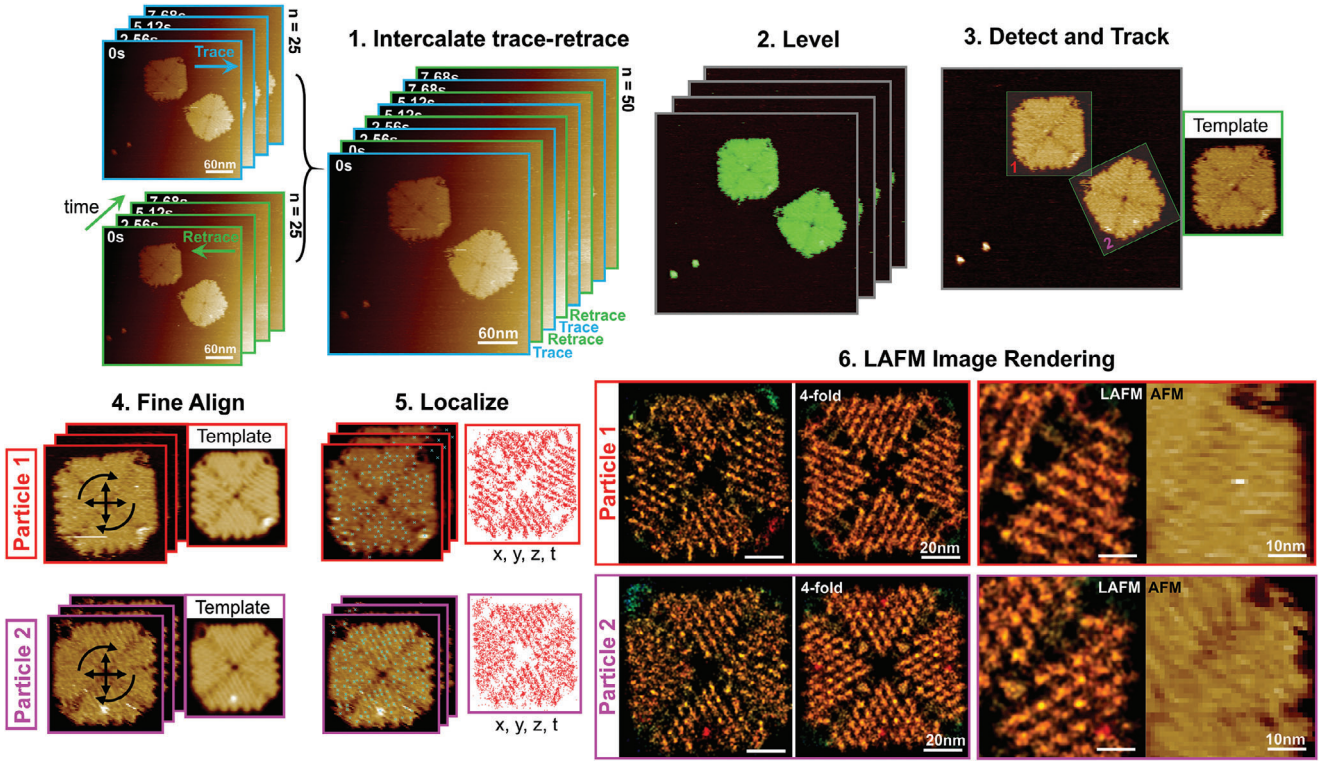

Figure 2. Example single particle LAFM workflow. 1) Loading of raw data and intercalation of trace and retrace frames. 2) Iterative leveling of the movie frames with height thresholding. 3) Detection of 4FS DNA origami tiles using a user selected reference image as a template (a single 4FS Origami tile image, insert on the right). 4) Iterative fine alignment of each individual particle using an averaged image as template updated after each iteration. 5) Localization of local height fluctuations to generate a 3D point cloud of localizations in x, y, z, and time. 6) Rendered localization AFM images without (left) and with (right) fourfold rotational symmetry averaging. Right: zoomed comparison between symmetrized LAFM map and single particle AFM images.

Combining trace-retrace data has been previously used to eliminate parachuting artifacts,[25] however since LAFM picks local maxima, regions containing parachuting are already eliminated from image reconstruction. Alternatively, data that has been acquired in bi-directional scanning modes that combine trace and retrace lines into one image can be used to increase temporal resolution. Bi-directional data capture can double the imaging frame rate, however, the method typically uses computational real-time data processing to align individual scan lines to build an image and is prone to contain errors that will propagate in the further data analysis particularly at higher scan rates or with higher levels of parachuting at low forces. We develop two methods for automatic alignment of line-shift in bi-directional AFM images, a line correlation method that uses correlation between scan line pairs (trace-retrace) and a peak matching method that calculates shift in peak positions between scan line pairs. To evaluate the performance of line shift correction in the presence of parachuting we incorporate simulation AFM capabilities into NanoLocz and develop routines to simulate image parachuting (Figure S1, Supporting Information). In the presence of parachuting the line correlation method overcorrects even if no line shift is present due the additional parachuting height signal in opposing directions (Figure S2, Supporting Information). However, this can be reduced using the peak matching method which analyzes line maxima to ignore parachuting.

Raw AFM images typically require background and line leveling to correct for sample tilt and height variations between scan lines. This process often requires multiple iterations to fit data with higher or lower height features excluded to avoid skewing the leveling.[26] To increase throughput we implement automatic image leveling with pre-set routines and real-time previews during movie processing, eliminating the need for slow batch processing and leveling errors. Leveling can also be performed manually. The manual and automatic thresholding algorithms account for height throughout a movie or set of images to avoid changes in baseline and improve accuracy. Accurate bassline height is particularly important for analysis in which heights between frames are being measured. At this step in the image processing pipeline one can perform standard AFM analysis such as threshold area/height analysis and line profiling or advance further toward single particle analysis. Example leveling and area/height analysis using time-lapse AFM data of antimicrobial peptide interactions with supported lipid membranes using previous data[30] is shown in Figure S3 (Supporting Information).

Depending on the sample and resolution achieved in AFM images, single particles may appear as featureless amorphous blobs or display visible substructure. Single particle detection and analysis in NanoLocz can be performed using either local maxima (for featureless objects) or using cross-correlation searches with a reference image (for particles with visible substructure). For the

local maxima method, we develop automatically height profiling to generate statistics on particle height, location, and width using full-width half maxima. Alternatively for applications that require analysis of the particle substructure such as LAFM, reference-based single particle detection can be performed. We optimize template matching algorithms that enable detection using a reference region in the image using either: a user selected typical particle, or using Simulation AFM to generate a reference image from 3D coordinates (for example a protein structure from the Protein Data Bank, i.e., PDB file) with user defined noise, parachuting, tip radius and pixel sampling taken directly from the image/video metadata. To develop a multi-rotation search algorithm, we combine correlation coefficient maps from the different reference images taking their maxima at each x, y position, the local maxima in the combined map is then matched back to the rotation parameters used to create the template for detection. The user interface facilitates interactive tuning of particle detections by employing a slider to visualize in real-time the effect of changing the minimum/maximum correlation coefficient and/or height thresholds, enabling rapid visual feedback for the inclusion or exclusion of detected particles. Controlling the correlation thresholds can also be used to select for specific conformations in the correlation distribution. If required, particle detection can then be refined using an averaged image of the particles detected in this first round of cross-correlation searching. We integrate single particle tracking algorithms to enable tracking of the detection coordinates of particles over time. At this stage statistics of particle angle, height, spatial distribution and movements over time can also be output.

Once positions have been detected particles can be rotationally and translationally aligned further. Translational and rotational alignment to the reference image is performed in multiple iterations in which the detection positions and reference are refined in each iteration. To prevent data loss, each alignment iteration updates the detection's x, y coordinates and angle only. This approach allows for a one-time particle extraction from the full image, avoiding losses that may occur due to multiple pixel interpolations in the sequential iterations. If tracking has been performed, alignments can be made to either all particles or a single particle over time. Particle coordinates (x, y, z), frame, time, angle, and track id are stored for analysis, LAFM imaging and can be exported at any stage.

Once the particle stack (either many individual molecules, or many observations of one or several molecules over time) is aligned, it can be utilized for LAFM. Previously the LAFM analysis workflow was performed using bicubic interpolation of the images to a higher pixel sampling, typically 3–4x expansion to reach 0.5Å per pixel.[16] This expansion was used to improve image alignment and improve precision of localizations. Local maxima were then found and rendered into a LAFM map using these expanded movies in a single step in an ImageJ plugin. To enable greater optimization and reproducibility of LAFM we develop a modular approach that decouples the localization analysis from the image rendering process. The first step of LAFM is finding local height maxima caused by height fluctuations whilst avoiding background noise. To achieve this, we create real-time visualization of the local maxima detected in response to non-destructive (without modification of the image data) image filters, height thresholds, and peak prominence thresholds. A typical recommended filter strength is a Gaussian filter with a strength of at least 0.6. To allow refinement of the peak detection positions to subpixel precision, bicubic interpolation is applied to a 5×5 subset of pixels surrounding each of the identified local maxima peaks, expanding it 10x. Rather than expanding the entire image set limiting expansion to subsets of pixels for each localization calculation reduces memory requirements and reduces the number of spurious localizations generated through the interpolation process itself. The LAFM image can then be rendered using these localizations on an expanded pixel grid with points rendered using 2D Gaussian profiles and 2D false-color scale options which combine height and localization probability. For each localization, information of x, y, z, time, peak prominence, and particle id is stored or can be exported, enabling LAFM rending to be reproduced independently of the image data. For molecules with symmetry we developed an automatic centering based on the alignment of the rotated localization point cloud, enabling a symmetrized LAFM map to be generated. At this stage LAFM images can be rendered to produce time-resolved LAFM movies and/or single molecule LAFM images. Figure 1c demonstrates a comparison between a single AFM image and the LAFM map obtained through NanoLocz processing.

Processed images and movies in NanoLocz can be saved and opened later or exported to a range of file formats. Exporting as .tiff, .csv, .txt, or .xls enables export without loss of image information whereas export as .gif, .avi, .png, .jpeg, or .pdf gives movies/images at presentation or publication quality with automatic scale bars and timestamps. All other data obtained through NanoLocz analysis, such as height and width distributions, particle orientations, tracking coordinates, and localizations can also be exported and/or plotted within the user interface.

2.1. Single Particle LAFM

To demonstrate new single particle LAFM capabilities of NanoLocz, Figure 2 shows an example workflow to obtain single particle LAFM images of two different copies of the same 4FS DNA origami tile design.[31,32] The raw data contains 25 images in which the two DNA origami tiles are visible. Typically, LAFM imaging requires at least 100 images of the same or different particles to get a reasonable sampling of localizations.[16] To increase the number of images for LAFM analysis, trace-retrace frame intercalation was used to give 50 unique frames and therefore 50 images for each particle. Given that the 4FS Origami tile has fourfold symmetry, we actually pool the information of 100 (trace only) or 200 (trace and retrace) redundant structural elements in our LAFM maps. After leveling the movie, DNA origami tiles were detected using a reference region from the leveled images with rotational freedom enabled. Particles were then tracked allowing separation of particle 1 and 2 for image fine alignment using single particle averages to iteratively refine alignment further. Each particle can then be treated separately in the localization analysis to generate independent localization point clouds and subsequent LAFM images. The LAFM images show positions of defects unique to each origami. fourfold symmetrization of the two single particle LAFM images shows the matching structural features of the DNA origami organization at high-resolution. Comparison of the LAFM images to the raw

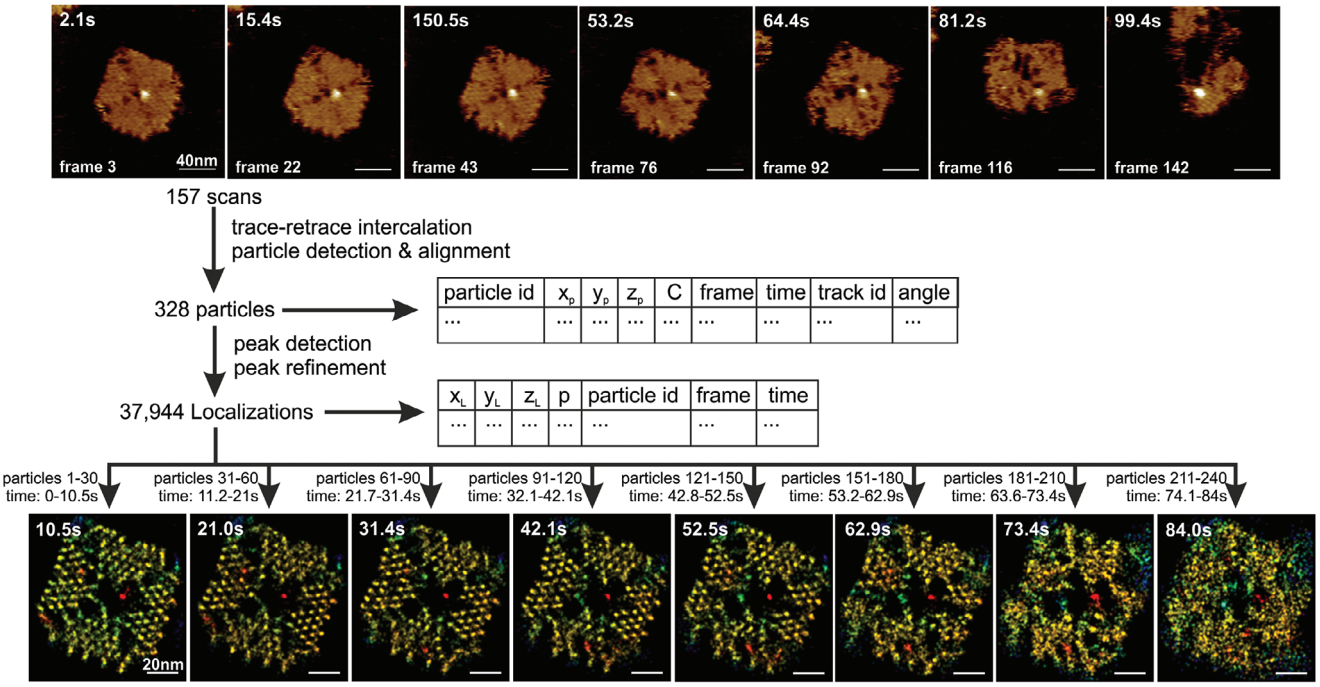

Figure 3. Example time-resolved LAFM workflow. Demonstration of time-resolved LAFM processing workflow for a single 4FS DNA origami tile degrading over multiple scans due to AFM tip interactions. Trace-retrace intercalation is performed on HS-AFM imaging data of the 4FS DNA origami tile captured at 1.4 frames per second. 328 origami tiles are detected and aligned using template matching with rotational freedom. Particle position, cross-correlation (C), time, frame, and angle are stored. Localization analysis is then performed on the detected particles to obtain localization position, prominence (p) whilst transferring information of which particle, frame, and time point each localization is obtained from. Localizations are separated by time range to render multiple LAFM images and generate a LAFM time series while taking into account a single height color scaling.

AFM data images highlight the significant improvement in resolution and contrast gained through the LAFM processing.

2.2. Time Resolved LAFM

To move from single LAFM imaging to time-resolved LAFM movies requires tracking the time points of each localization so LAFM maps can be rendered according to time. **Figure 3** shows an example workflow for time-resolved LAFM using a HS-AFM movie of a single 4FS DNA origami tile imaged over time. In Figure 3 we utilize trace-retrace interaction to double the number of particles and hence the time-resolved LAFM time resolution. Particles are then detected and aligned as previously demonstrated (Figure 2). From 157 scans at 1.4 frames per second, we acquired 328 particles (from a single origami tile). The time point and particle id of each detection is stored. Upon localization analysis to find fluctuations in each particle detection, the corresponding particle ids and time points are transferred to each localization coordinate. To render a time-resolved LAFM movie the stored localizations are selected across specified time frames or a given number of particles. The time-resolved LAFM of DNA origami degradation uses 30 particles per LAFM map giving a time resolution of 10.5 s per frame (Figure 3). The LAFM time series resolves stable origami junction positions where DNA staples are bound or removed during imaging eventually leading to complete destabilization of the origami tile.

2.3. Simulation AFM and LAFM

Development and integration of Simulation AFM into NanoLocz enables the generation and import of simulated AFM surfaces, using pixel sampling matching experimental data, from protein structures in repositories such as the Protein Data Bank. In brief, AFM surface topography prediction of the proteins is based on scanning in silico an estimated tip radius over the PDB structure to find the points of contact. To model fluctuation dynamics, random fluctuations can be added to the atomic coordinates with constraints limiting atomic positions overlapping within a set distance. By adding fluctuations, simulations can be used to rapidly test LAFM analysis performance on simulated protein surfaces. **Figure 4** shows simulations of a 2D lattice of bacteriorhodopsin viewed from the cytoplasmic side of the protein using existing known structures from X-ray crystallography.[33] By combining simulations of open and closed state structures we simulate AFM imaging and time-resolved LAFM mapping of a conformational change. Features observed in the simulated AFM images and simulated LAFM (Figure 4b,c) show the increase in structural information obtained by the LAFM method. Simulated LAFM may also be used to support interpretation of experimental LAFM imaging of conformational changes such as those previously obtained for bacteriorhodopsin dynamics.[13,34]

Alternatively, simulated surfaces can be used as a reference image for particle detection, to find protein location and orientation

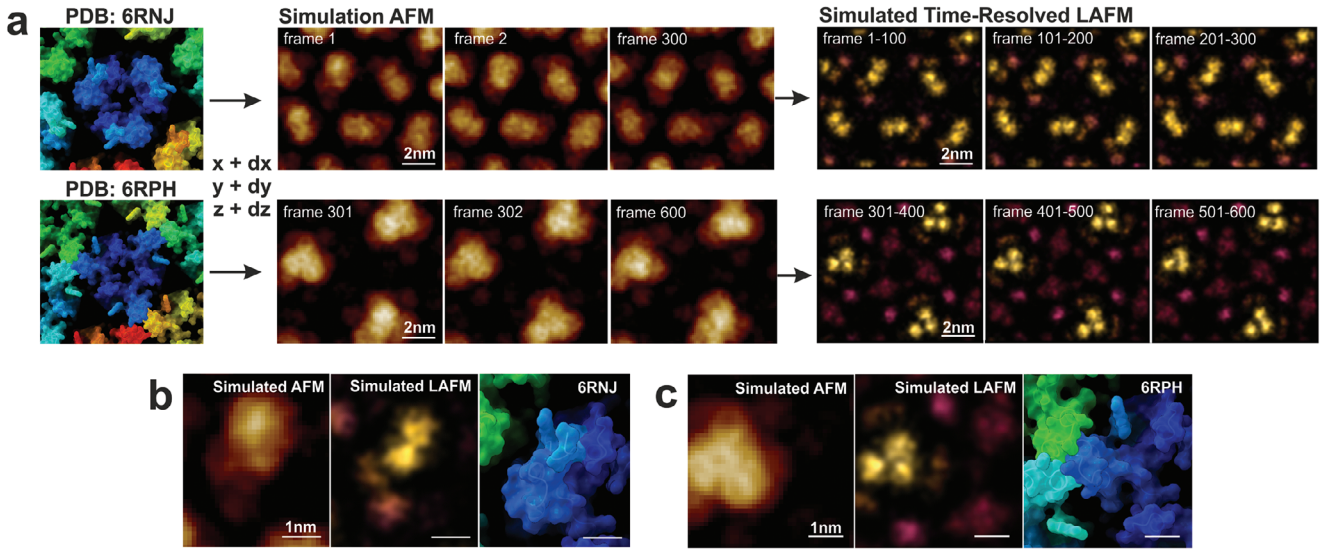

Figure 4. Example Simulation LAFM workflow. a) Left: Structures of bacteriorhodopsin trimers assembled into a 2D lattice viewed from the cytoplasmic side in closed (PDB: 6RNJ) and open (PDB: 6RPH) states.[33] Simulation AFM of 6RNJ and 6RPH performed using 300 unique sets of fluctuated coordinates per conformational state. Right: Time-resolved LAFM analysis of the 600 simulation AFM frames using 100 particles per LAFM map. Comparisons between simulated AFM, simulated LAFM, and the surface structure of the bacteriorhodopsin monomer in b) closed and c) open conformations respectively.

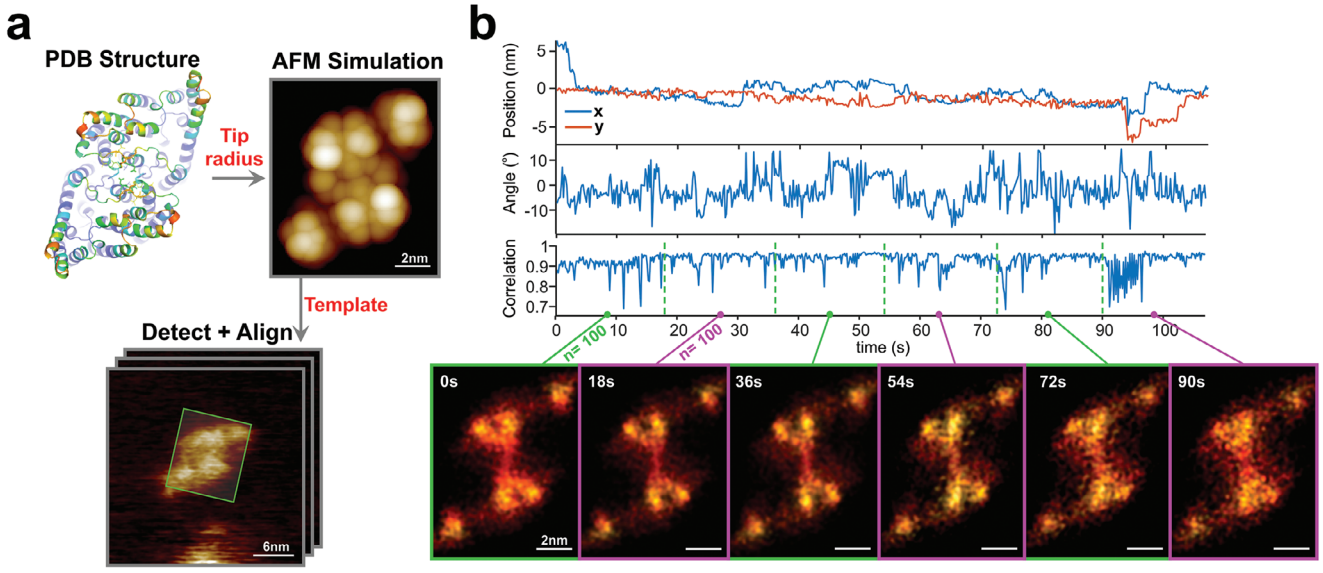

Figure 5. Example Simulation AFM for particle detection Workflow. a) Demonstration of the use of an AFM simulation image, the extracellular face of CLC-ec1 (PDB 1OTS[35]), as reference, imported into NanoLocz for particle detection and fine alignment. b) x, y detection position, angle, and normalized cross-correlation of the HS-AFM experimental particle over time compared to the simulated structure reference. Time-resolved LAFM maps of the CLC at 18s intervals using 100 independent frames per LAFM map. LAFM maps have twofold symmetry applied.

in experimental data. In **Figure** 5a, a CLC-ec1 antiporter structure (PDB: OST1[35]) is used to produce a simulated molecular topography. This simulated surface is then used as a template for cross-correlation searches and particle detection in previous HS-AFM experimental data. The simulated template is also used as reference for fine rotational and translational alignment. The data obtained from the fine alignment such as x, y position, rotation angle, and normalization cross-correlation value (with the simulated surface) gives information about the positional dynamics and conformational variations (or image quality) over time (Figure 5b). Such analysis, potentially using several simulation topographies of the protein in different states, could also be used to detect conformational changes over time. Whilst the CLC-ec1 molecule is not undergoing significant conformational changes on the 18s timescale, such a time-resolved LAFM workflow can be applied to HS-AFM which contains multiple copies of the same particle in each frame to increase the effective LAFM time resolution.

3. Conclusion

NanoLocz's advanced capabilities and user-friendly interface make it a valuable tool for AFM image analysis. By streamlining and integrating AFM, HS-AFM and LAFM data processing workflows, NanoLocz significantly reduces manual efforts which should lead to increased productivity, more access to single molecule analysis methods and help pave the way for future advances in AFM analysis. The ongoing development and future enhancements to NanoLocz including python versions aim to further support new improvement in AFM data analysis capabilities. By encouraging the development of novel analysis algorithms through plugin integration, NanoLocz aims to create a collaborative environment for further advancements in AFM research. Such an open-source software package used by the AFM community at large will automatically lead to common standards and enhance data reproducibility and exchange.

4. Experimental Section

AFM Data Import: To read raw AFM image or HS-AFM video data, various file readers algorithms for common AFM file types were modified from existing codes (.spm, .asd, .jpk, .ibw, .gwy) or written in NanoLocz (.asd, .aris and .nhf). See Table 1 for the capabilities and source of each file reader. All file readers were modified and designed to store standardized metadata including scan size, scan speed, time, and available data channels. Within NanoLocz image data was stored as 2D arrays for single images or as 3D arrays for video or multiple frame data. The use of 3D arrays in general allowed for faster computation and lower memory requirements but required homogenous data, i.e., with the same number of x and y pixels in each frame.

Image Leveling: Image leveling could be performed either manually or through an automatic image leveling routine. For manual leveling procedures, the order of the polynomial fit was defined by the user (see script "filter_img_v2.m").

Plane Fitting: To remove tilt or bow in images, a polynomial was fit to the average of all rows (for y tilt removal) and/or the average of all the columns (for x tilt removal), these polynomial trends were then subtracted from the original image.

Line Fitting: To remove height variations between scan lines, polynomials were fit to each line in the x or y direction, these polynomial trends in each line were then subtracted from the original image.

Line or Plane Fitting with a Mask: To level images and movies that contained a variety of height levels required certain regions (referred to as masked regions) to be excluded from the polynomial fits. Typically the included region should be a surface that was expected to be level and has a high coverage. Masks could be automatically generated using the Otsu method or by fitting a Gaussian to all the values in the movie and using 1.5 x standard deviation as the threshold values. Manual masks could be generated using upper and lower threshold values or manually drawn regions. To level with a mask, the function in NanoLocz (thresholder.m) set masked (excluded) pixel values to Nan and then line fitting or plane fitting was run as before but fitting was performed using only the real values. The fitting was then subtracted from the whole of the original image.

Automatic Leveling: Leveling typically requires multiple iterations, to achieve this automatically several pre-set leveling routines were implemented in NanoLocz. For example the "Iterative Holes" and "Iterative Peaks" routines that used the following routine:

1) Subtract a plane fit in x and y with 2nd order polynomial then subtract the median line in x.
2) Fit a Gaussian to all the heights in the movie and set a mask threshold to $z > 1.5\sigma$ (Iterative Peaks) or $z < -1.5\sigma$ (Iterative Holes) to generate a mask.
3) Repeat leveling step 1. with the threshold mask applied.
4) Generate new mask by repeating step 2.
5) Subtract a plane fit in x and y with 2nd order polynomial then subtract a 1st order polynomial line in x with the threshold mask applied.

"Iterative Holes" and "Iterative Peaks" automatic leveling routines were designed for image set with a fraction of lower features such as membrane defects (Iterative Holes) or a fraction of objects with higher features such as features on a mica surface (Iterative Holes). Other iterative leveling routine in NanoLocz include basic leveling followed by Otsu mask leveling or leveling followed by two leveling iterations with Gauissan fitted masks for z outside the range of $+/-1.5\sigma$.

Bi-directional Line Shift Correction: Bi-directional line shift correction was performed per image to find the pixel lag between scan directions using one of the following methods:

Line Correlation Method: The line correlation method computed the normalized cross-correlation between each trace and retraced line pair and calculated the lag with the highest correlation. A Gaussian was then fit to all the lags in the image and rounded to the nearest whole value.

Peak Matching Method: The peak matching method first Gaussian filtered the trace and retrace lines and then found peak positions with a minimum prominence of 0.2 nm (to reduce selection of noise peaks). The differences between peak positions between each trace and retrace line pair were calculated to give lag values. A Gaussian was then fit to all the lags in the image and rounded to the nearest whole value.

Trace-Retrace Intercalation: To intercalate trace-retrace HS-AFM frames, the image stack containing n trace images was first loaded. Retrace frames were then loaded and combined with trace frames into a stack of $2n$ frames with trace frames at $2i+1$ and retrace frames at $2i$ for i ranging from 0 to $n-1$.

Particle Detection: Two methods were implemented for particle detection using either local maxima (peak method) or template matching to a reference image (ROI method).

Peak Method: The peak method of particle detection found local maxima above a threshold value within a circular neighborhood of pixel distances. To achieve this every pixel was searched to check if it was the highest in the local neighborhood, pixel positions in x, y, z and frame are stored (peak2D.m). To reduce the selection of noisy peaks a user defined Gaussian could be applied to filter the image for detection purposes only. Particles were then extracted from the images based on a user defined box size. To eliminate spurious detections all particles were cross-correlated with the particle average enabling the user to select a minimum normalized correlation coefficient.

ROI Method: The ROI (region of interest) method used image cross-correlation of a reference image with the full image or video data set to detect particles. Reference images could be manually selected, simulated using simulation AFM of a pdb file, using the average particle of a previous detection round or using the down sampled LAFM map. The cross-correlation of the reference image with the image data was performed with Gaussian image filters on the image and the resulting normalized cross-correlation map (default strength, $\sigma = 1$), local maxima in the cross-correlation map were then used as detection points and positions in x, y, z and frame were stored. Particles were then extracted from the images based on the pixel dimensions of the reference image. To detect different rotations of a particle the detection could be run with rotation enabled, in this mode the cross-correlation was run with different rotations the reference image to different angles. At each rotation the normalized cross-correlation coefficient map was calculated (CCR) between the rotated reference image and the target image. This was then assembled into a 3D array with each image in the 3rd dimension corresponding to a CCR of each rotation. A maxima search was performed along the third dimension to search for the maximum correlation values across rotations while keeping track of which slice (rotation) it originated from.

Particle Tracking: Particle coordinates were input into the simpletracker algorithm (https://github.com/tinevez/simpletracker) to track particle positions over time. In brief particle pairs identified between frames as the closest (based on Euclidean distance) were connected to form links. Using the Hungarian algorithm the sum of these pair distances

was minimized across all particles between successive frames. Subsequently, a second iteration examined track endings. If a track's start closely aligned with another track's end in the subsequent frames, a link spanning multiple frames was established, effectively bridging the gap and reestablishing the track. The method allowed tracking to bridge a user defined number of frames where a particle was missing/undetected.

Particle Fine Alignment: Subpixel translational and rotational alignment was performed to improve alignment of detected particles. The process could be performed iteratively alternating between translational and rotational while improving the reference image average. Translational alignment was performed using 2D image cross-correlation between each particle image and the reference image. To achieve subpixel translational alignment a 5×5 pixel region surrounding maximum normalized cross-correlation value in the CCR map was expanded with bicubic interpolation to interpolate the maxima position with greater precision. Rotational alignment was performed by calculating the 2D correlation coefficient between the reference image and particle image at different image rotations. For angle searches greater than +/- 20° particles were rotated at 0.5° intervals, below 20° angle searches, rotation was in 0.2° intervals. Spline interpolation of the correlation coefficient values versus angle was used to improve precision of the angle at which the maximum lied. Rotational and translational alignment could be performed with a threshold height applied to exclude lower pixels from the alignment.

Localization AFM: Localization AFM is performed in three main steps, 1) peak identification, 2) sub-pixel localization, and 3) LAFM image rendering.

Peak Identification: To identify peaks that corresponded to molecular height fluctuations, local maxima above a threshold value within a circular neighborhood of pixel distances were searched (peaks2D.m) to give pixel positions in x, y, z, time and particle id. To reduce the identification of noisy peaks the search could be performed with an image filter and/or a minimum peak prominence.

Peak prominence was calculated by calculating the Euclidean distance between the current peak and all other peaks. The height of the peaks was checked against neighboring peaks. If the current peak's height was greater than or equal to the height of any other peak, the prominence was assigned as the height. Otherwise, the prominence was given as the difference between the current peak's height and the minimum intensity along the line profile between the current peak and its nearest neighboring peak with greater height.

Sub-pixel Localization: To allow refinement of the peak detection positions to subpixel localizations, bicubic interpolation was applied to a 5×5 subset of pixels surrounding each of the identified local maxima peaks, expanding it to 50×50 pixels. The expanded region was then cropped to the central 30×30 pixels and the maxima of this subsections was then used as the final localization position.

LAFM Image Rendering: LAFM images were rendered using the stored x, y, z, time and particle id localization information. The LAFM image canvas size was determined by the maximum positions of the localizations in x and y before expanding the pixel grid (typically 5x). To render each localization with a specific color, z values were normalized between 0 and 1. The algorithm then looped through each color in the selected color map also scaled between 0 and 1, identifying all localizations that should be rendered that color depending on their normalized z value. Within the loop an image in generated with 1s at localization positions, these are then Gaussian smoothed and the corresponding color RGB values in that height "slice" applied. Each color/height slice was then added to the previous one to produce the full LAFM image. The z-scale color limits could be set by the user in terms of percentage of the full localization z-range to alter the z-range of the LAFM maps.

For molecules that have symmetry, a symmetrized LAFM maps can rendered by adding a set of duplicated localizations which were rotated and combined with the original localization list. This was achieved by duplicating the localization data and rotating it ***n*** times by angles depending the number of the degrees of symmetry ***n***. To ensure correct alignment of the localizations after rotation, the relative center of the localization point cloud could be found by rotating each additional point cloud and aligning it with the original. Point cloud alignment was performed using the iterative closest point (ICP) algorithm. The average of all the alignments for each rotation gave the new center point. The combined localization list can was then rendered into an LAFM image as before.

Time Resolved LAFM: Time information of every localization was stored from the point of particle detection in the original starting AFM frames through to the localization peak identification. To render a time-resolved LAFM image sequence, localization values were separated depending on their time value and then rendered into images separately. To implement this users defined the number of particles to be included in each LAFM image. A sliding window value could be used to overlap localizations in subsequent LAFM images. The sliding window value determined the number of particles to step forward, for example if the number of particles was 50 and the sliding window was 50 there will be no overlap. However, if the sliding window value was 25 particle there would be a 50% overlap of particle localizations between frames in the LAFM Movie.

Simulation AFM: Simulated AFM images were produced by calculating collision points between a model AFM tip and set of 3D coordinates. The image was produced by taking the highest values at each pixel position. The AFM tip was modeled as a cone with a hemisphere shaped apex. The angle of the cone corresponded to the sidewall angle, θ, of the AFM probe which was typically available from probe manufactures. The radius, r, of the hemisphere could be considered as the tip radius. For each coordinate $A(x, y, z)$, such as the atomic coordinates obtained from a protein data bank file, the heights, $z(i)$, on a surrounding set of grid positions $dx(i)$ and $dy(i)$ are calculated based on geometric arguments:

$$dh\,(i) = r - \sqrt{r^2 - \left(dx(i)^2 + dy(i)^2\right)} \tag{1}$$

where $dh(i)$ values correspond to the heights on the hemisphere. For grid positions within the hemisphere radius this results in real values dh which are then used to calculate heights at each grid position:

$$z\,(i) = A_z\,(x, y, z) - dh\,(i) \tag{2}$$

For imaginary values of dh (i.e., where the distance is outside of the hemisphere) the height is calculated using the sidewall angle, θ, as follows:

$$z\,(i) = A_z\,(x, y, z) - r - \left(\sqrt{dx(i)^2 + dy(i)^2} - r\right)\tan\,(\theta) \tag{3}$$

The final image takes the highest $z(i)$ values calculated across all coordinates at each pixel position.

Modeling Fluctuations: To model fluctuation dynamics in a simplistic way, the 3D coordinates were randomly offset in x-y and/or z using normally distributed random numbers:

$$A\,(x, y, z) = (x + dx,\ y + dy,\ z + dz) \tag{4}$$

To prevent unrealistic overlap of atomic positions a clash prevention routine was implemented which checked for coordinates within a 2.8Å distance of each other based the typical 1.4Å van der Waals radius for solvent-accessible surface area. Coordinate pairs residing <2.8 Å distance of each other were not accepted and were both iteratively forced to move apart in opposite directions at 20% of the overlap per iteration. After fluctuation were added and clashes prevented each whole set of coordinates was then used to produce simulated AFM images of those coordinates undergoing positional fluctuations.

Simulating Parachuting: To simulate parachuting artifacts observed in AFM imaging, particularly at high scan speeds, simulation AFM image were generated first and then checked for decreasing gradients in the x direction above a user defined threshold level. Gradients were calculated per scan line using a sobel convolution kernel. If the gradient at any pixel was above the threshold all subsequent pixels were replaced with linearly decreasing values using the threshold gradient until the original height value is reached. The process was repeated for all positions that exceed the

gradient threshold. To simulate retrace parachuting the image was flipped in the x direction for calculation and then flipped back. For bi-directional parachuting calculation alternating lines were flipped for calculation and flipped back.

Algorithm and User Interface Implementation: All the algorithms and app designer files for the GUI design of NanoLocz were written in MATLAB (MathWorks, Natick, MA) and are available to download at https://github.com/George-R-Heath/NanoLocz. The GUI file (.mlapp) was built using MATLAB App Designer (version 2023b) which can be modified and built upon. The standalone Windows and Mac executables of NanoLocz were prepared using the MATLAB compiler (2023b), the MATLAB app version (.mlappinstall) of NanoLocz was packaged using the MATLAB App Designer.

Code Availability: The source-code, standalone versions for PC or Mac, MATLAB app, test data and installation instructions for updated versions of NanoLocz can all be found at https://github.com/George-R-Heath/NanoLocz.

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

The authors thank Abeer Alshammari and Maya Tekchandani for extensive testing of the software, Grigory Tagiltsev and Simon Scheuring for providing .asd file opening algorithms, Hector Corte for assisting with .nhf and .gwy file opening algorithms, Christoph Walti and Ilaria Sandei for providing 4FS DNA origami tiles. The authors gratefully acknowledge funding from the Engineering and Physical Science Research Council in an EPSRC Open fellowship (EP/W034735/1) to G.R.H.

Conflict of Interest

The authors declare no conflict of interest.

Data Availability Statement

The data that support the findings of this study are openly available in Zenodo at 10.5281/zenodo.10409523, reference number [10409523].

Keywords

AFM, DNA origami, high-speed AFM, image processing, simulation AFM, structural biology

Received: December 20, 2023
Revised: February 12, 2024
Published online:

[1] S. H. W. Scheres, *J. Struct. Biol.* **2012**, *180*, 519.
[2] J. Zivanov, J. Otón, Z. Ke, A. von Kügelgen, E. Pyle, K. Qu, D. Morado, D. Castaño-Díez, G. Zanetti, T. A. Bharat, J. A. Briggs, S. H. Scheres, *Elife* **2022**, *11*, e83724.
[3] M. Ovesný, P. Křížek, J. Borkovec, Z. Svindrych, G. M. Hagen, *Bioinformatics* **2014**, *30*, 2389.
[4] S. Wolter, A. Löschberger, T. Holm, S. Aufmkolk, M.-C. Dabauvalle, S. van de Linde, M. Sauer, *Nat. Methods* **2012**, *9*, 1040.
[5] J. Schnitzbauer, M. T. Strauss, T. Schlichthaerle, F. Schueder, R. Jungmann, *Nat. Protoc.* **2017**, *12*, 1198.
[6] J. Ries, *Nat. Methods* **2020**, *17*, 870.
[7] N. Gustafsson, S. Culley, G. Ashdown, D. M. Owen, P. M. Pereira, R. Henriques, *Nat. Commun.* **2016**, *7*, 12471.
[8] D. Sage, T.-A. Pham, H. Babcock, T. Lukes, T. Pengo, J. Chao, R. Velmurugan, A. Herbert, A. Agrawal, S. Colabrese, A. Wheeler, A. Archetti, B. Rieger, R. Ober, G. M. Hagen, J.-B. Sibarita, J. Ries, R. Henriques, M. Unser, S. Holden, *Nat. Methods* **2019**, *16*, 387.
[9] G. Benn, I. V. Mikheyeva, P. G. Inns, J. C. Forster, N. Ojkic, C. Bortolini, M. G. Ryadnov, C. Kleanthous, T. J. Silhavy, B. W. Hoogenboom, *Proc. Natl. Acad. Sci* **2021**, *118*, e2112237118.
[10] S. Lansky, J. M. Betancourt, J. Zhang, Y. Jiang, E. D. Kim, N. Paknejad, C. M. Nimigean, P. Yuan, S. Scheuring, *Nature* **2023**, *621*, 206.
[11] Y. Jiang, B. Thienpont, V. Sapuru, R. K. Hite, J. S. Dittman, J. N. Sturgis, S. Scheuring, *Nat. Commun.* **2022**, *13*, 7373.
[12] Y. C. Lin, Y. R. Guo, A. Miyagi, J. Levring, R. MacKinnon, S. Scheuring, *Nature* **2019**, *573*, 230.
[13] A. P. Perrino, A. Miyagi, S. Scheuring, *Nat. Commun.* **2021**, *12*, 7225.
[14] G. R. Heath, S. Scheuring, *Curr. Opin. Struct. Biol.* **2019**, *57*, 93.
[15] S. Tsujioka, A. Sumino, Y. Nagasawa, T. Sumikama, H. Flechsig, L. Puppulin, T. Tomita, Y. Baba, T. Kakuta, T. Ogoshi, K. Umeda, N. Kodera, H. Murakoshi, M. Shibata, *Sci. Adv.* **2023**, *9*, eadh1069.
[16] G. R. Heath, E. Kots, J. L. Robertson, S. Lansky, G. Khelashvili, H. Weinstein, S. Scheuring, *Nature* **2021**, *594*, 385.
[17] D. Nečas, P. Klapetek, *Cent. Eur. J. Phys.* **2012**, *10*, 181.
[18] I. Horcas, R. Fernández, J. M. Gómez-Rodríguez, J. Colchero, J. Gómez-Herrero, A. M. Baro, *Rev. Sci. Instrum.* **2007**, *78*, 013705.
[19] J. G. Beton, R. Moorehead, L. Helfmann, R. Gray, B. W. Hoogenboom, A. P. Joseph, M. Topf, A. L. B. Pyne, *Methods* **2021**, *193*, 013705.
[20] W.-F. Xue, (Preprint) *bioRxiv: 2023.07.05.547812*, submitted: July **2023**.
[21] R. Amyot, H. Flechsig, *PLOS Comput. Biol.* **2020**, *16*, e1008444.
[22] R. Amyot, A. Marchesi, C. M. Franz, I. Casuso, H. Flechsig, *PLOS Comput. Biol.* **2022**, *18*, e1009970.
[23] H. V Guzman, P. D. Garcia, R. Garcia, *Beilstein J. Nanotechnol.* **2015**, *6*, 369.
[24] V. G. Gisbert, R. Garcia, *Soft Matter* **2023**, *19*, 5857.
[25] S. Kubo, K. Umeda, N. Kodera, S. Takada, *Biophys. physicobiology* **2023**, *20*, e200006.
[26] B. W. Erickson, S. Coquoz, J. D. Adams, D. J. Burns, G. E. Fantner, *Beilstein J. Nanotechnol.* **2012**, *3*, 747.
[27] R. D. Ortuso, K. Sugihara, *J. Phys. Chem. C* **2018**, *122*, 11464.
[28] J. Bialek, **2013**, https://uk.mathworks.com/matlabcentral/fileexchange/42679-igor-pro-file-format-ibw-to-matlab-variable, (accessed 10 July 2023).
[29] E. L. Fricke, **2011**, https://uk.mathworks.com/matlabcentral/fileexchange/32893-gwyddion-file-importer, (accessed 12 Sep 2023).
[30] G. R. Heath, P. L. Harrison, P. N. Strong, S. D. Evans, K. Miller, *Soft Matter* **2018**, *14*, 6146.
[31] S. Confederat, I. Sandei, G. Mohanan, C. Wälti, P. Actis, *Biophys. J.* **2022**, *121*, 4882.
[32] C. Chau, G. Mohanan, I. Macaulay, P. Actis, C. Wälti, *Small* **2023**, 2308776.
[33] T. Weinert, P. Skopintsev, D. James, F. Dworkowski, E. Panepucci, D. Kekilli, A. Furrer, S. Brünle, S. Mous, D. Ozerov, P. Nogly, M. Wang, J. Standfuss, *Science* **2019**, *365*, 61.
[34] M. Shibata, H. Yamashita, T. Uchihashi, H. Kandori, T. Ando, *Nat. Nanotechnol.* **2010**, *5*, 208.
[35] R. Dutzler, E. B. Campbell, R. MacKinnon, *Science* **2003**, *300*, 108.

