Aberration-Corrected Multifocal Microscope Data Processing and Analysis Software Manual

Jesse Aaron
Advanced Imaging Center
HHMI Janelia Research Campus
Version 6/2017

Contents

List of Figures

INTRODUCTION

The aberration-corrected multifocal microscope (acMFM) can acquire fluorescent images of up to 9-focal planes in a sample simultaneously. This technology is particularly well-suited for dynamic imaging of very rapid biological processes in all three dimensions, including 3D single particle tracking (SPT). It can perform this function at high acquisition rates – up to *ca.* thirty 9-plane stacks per second. It can also obtain images of two different colors at this rate simultaneously. See Abrahamsson et al. *Nat. Meth.* **10** 60-63 (2013) for a technical description of this system.

Interpreting acMFM data requires several processing and analysis steps. These algorithms have been implemented in a Matlab-based graphical user interface (GUI) that is freely available to users of the Advanced Imaging Center (AIC). In summary, the major steps are:

1. **Calibration:** images of standard fluorescent beads are taken across a range of focal positions to assess several parameters needed to reconstruct and analyze final images. These parameters include (i) inter-plane spacing, (ii) relative emission intensity distribution across planes, (iii) affine transform matrices for alignment of each plane relative to each other, and (iv) the measured point-spread function (PSF) in each plane. *Calibration images must be taken using the same objective lens and excitation wavelength(s) that will be used for the experiment being performed.*

2. **Image Alignment/Processing:** The parameters calculated in step 1 are used to properly re-construct MFM images such that they can be analyzed. The software allows for batch processing of multiple time-course movies (comprising up to two color channels), but can only process images using a single calibration file for each color channel. Several options are available. The only required step is image alignment – transforming each raw 3D data set into a properly aligned 4D (x/y/z/t) "hyperstack" that can be opened in ImageJ/FIJI. Options also exist to correct for non-uniform intensity distribution across each focal plane, and to perform a background subtraction and Richardson-Lucy deconvolution.

3. **Particle Localization/Tracking:** Once images have been properly aligned and processed, single particle localization and tracking can be performed to assess the transport behavior of particles or single molecules within the cell. The algorithm is based on the Mosaic ImageJ/Fiji plugin to estimate the location of single point-sources within the image to a precision higher than the diffraction limit. These locations can be linked together using a LAP tracker (see Jaquaman, et. al, *Nat. Meth.* 2008). These trajectories can be visualized either as a 2D overlay with the original data, or as a 3D Matlab plot. Trajectories can also be analyzed to calculate individual diffusion constants, either with a Brownian or anomalous diffusion model. An option also exists to calculate a 2D diffusion constant in cases where particles are tightly confined within a 2D structure such as a cell membrane.

SOFTWARE REQUIREMENTS, CONFIGURATION, AND INSTALLATION

Matlab Requirements

The acMFM software is platform independent (Windows, Mac OS, or Linux) but requires installation of Matlab R2014a or later. It also requires the following toolboxes to be installed:

- Curve Fitting
- Image Processing
- Parallel Computing
- Statistics

For users that do not have access to these products, the AIC can provide a stand-alone compiled package, but this has not been tested extensively. We recommend that the software be installed on a high-end workstation with >16GB of RAM and a multi-core processor.

Graphical Processing Unit (GPU) and CUDA Requirements

To take advantage of the software's GPU-accelerated computational capabilities, your system must have a recent CUDA-compatible NVIDIA video card.

To see a list of CUDA-compatible video cards, visit: https://developer.nvidia.com/cuda-gpus.

To download the latest CUDA driver, go to: http://www.nvidia.com/Download/index.aspx.

To ensure that Matlab can communicate with your GPU device, type the following into the Matlab command prompt:

```
>> D = gpuDevice
```

An example output is shown below. Of particular importance is the `ComputeCapability` field. For proper performance, this value should be 1.3 or greater.

```
D =

  CUDADevice with properties:

                      Name: 'GeForce GTX TITAN Black'
                     Index: 1
         ComputeCapability: '3.5'
            SupportsDouble: 1
             DriverVersion: 6.5000
            ToolkitVersion: 5.5000
        MaxThreadsPerBlock: 1024
          MaxShmemPerBlock: 49152
        MaxThreadBlockSize: [1024 1024 64]
               MaxGridSize: [2.1475e+09 65535 65535]
                 SIMDWidth: 32
               TotalMemory: 6.4425e+09
                FreeMemory: 5.8405e+09
      MultiprocessorCount: 15
             ClockRateKHz: 1071500
              ComputeMode: 'Default'
     GPUOverlapsTransfers: 1
   KernelExecutionTimeout: 1
         CanMapHostMemory: 1
          DeviceSupported: 1
           DeviceSelected: 1
```

The MFM software relies on calls to a Java package in order to read Nikon-formatted (.nd2) data files. To avoid memory errors, it is generally necessary to increase the Java Heap Memory allocation in Matlab from its default value. See the following instructions to perform this configuration:

http://www.mathworks.com/matlabcentral/answers/159220-how-i-can-increase-java-heap-memory

In general, the Java Heap Memory should be set to at least 4GB or more if you are working with very large data sets.

The particle tracking feature requires a newer version of Java (v1.8) than what is typically used in the latest Matlab releases (v1.7). You can download the latest version of Java here:

https://www.java.com/en/download/

You then need to tell Matlab to use this latest version. Operating specific Instructions for doing this can be found here:

MAC OS: https://www.mathworks.com/matlabcentral/answers/103056-how-do-i-change-the-java-virtual-machine-jvm-that-matlab-is-using-for-mac-os

Windows: https://www.mathworks.com/matlabcentral/answers/130359-how-do-i-change-the-java-virtual-machine-jvm-that-matlab-is-using-on-windows

Linux: https://www.mathworks.com/matlabcentral/answers/130360-how-do-i-change-the-java-virtual-machine-jvm-that-matlab-is-using-for-linux

To install the MFM GUI, unzip the ***MFM.zip*** file into your Matlab working directory. Add the resulting directory into Matlab's search path. See here for instructions:
http://www.mathworks.com/help/matlab/matlab_env/what-is-the-matlab-search-path.html

ImageJ/Fiji Requirements

Miji

The automated particle tracking aspect of the MFM package uses the MOSAIC ToolSuite. See: http://mosaic.mpi-cbg.de/?q=downloads/imageJ for details on this package. However, the most recent version of MOSAIC is not available as a standalone Matlab code. Thus, we make use of an interface between Matlab and Fiji/ImageJ called Miji. See here for a tutorial:

http://bigwww.epfl.ch/sage/soft/mij/ .

You will need to add the path that contains the Miji.m file to Matlab's search path. Importantly, be sure that this path is below the MFM software path in Matlab's search directory list. You will also need to add the java class path for miji.jar and ij.jar. The above website explains how to do this, and AIC staff can help you.

MOSAIC ToolSuite

The final feature of the acMFM software package allows users to perform particle tracking analysis on their data. This package currently utilizes the Mosaic Particle Tracking software.

See: http://mosaic.mpi-cbg.de/?q=downloads/imageJ . Users will need to make sure the Mosaic plugin suite is installed in ImageJ/Fiji, using the following steps.

Fiji

1. To install in Fiji, click Help → Update Fiji.
2. Click Manage update sites
3. Check "MOSAIC ToolSuite"
4. Click Close
5. Click Apply Changes. You will be prompted to re-start Fiji.

ImageJ

1. Go to http://mosaic.mpi-cbg.de/Downloads/Mosaic_ToolSuite_imagej_new.jar
2. Download into the ImageJ plugins folder.
3. Restart ImageJ if needed.

You should now be ready to start the MFM software.

MAIN GUI WINDOW

To start the MFM software, simply type `MFM` at the Matlab command prompt:

```
>> MFM
```

The following window should appear:

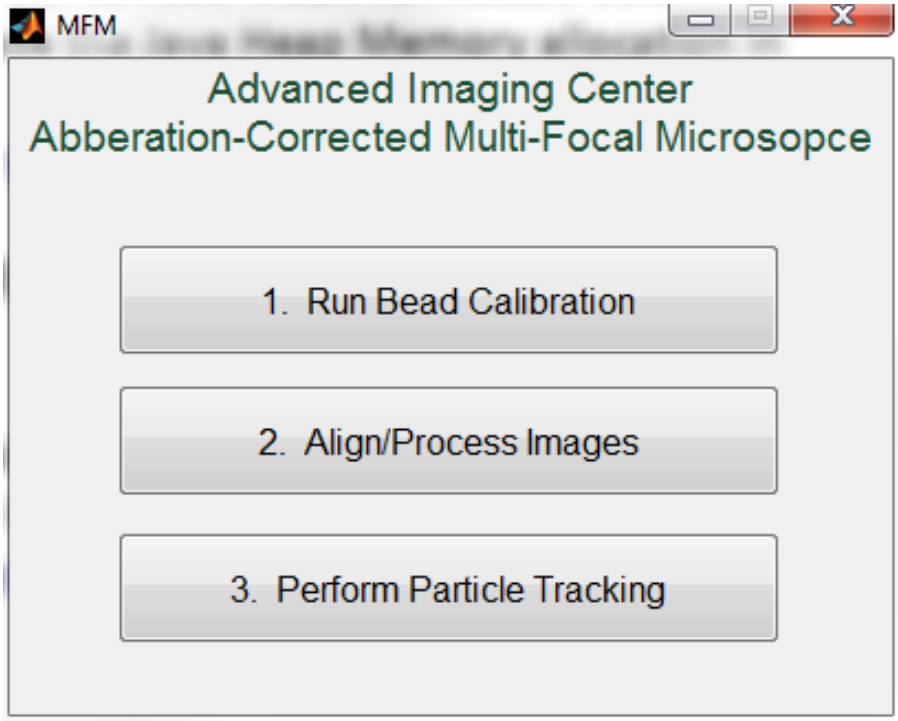

Figure 1. MFM software main window

The three buttons refer to the 3 steps outlined in the introduction section of this manual. In general, they should be performed in the order given.

RUN BEAD CALIBRATION

Press the **Run Bead Calibration** button, and the following window should appear:

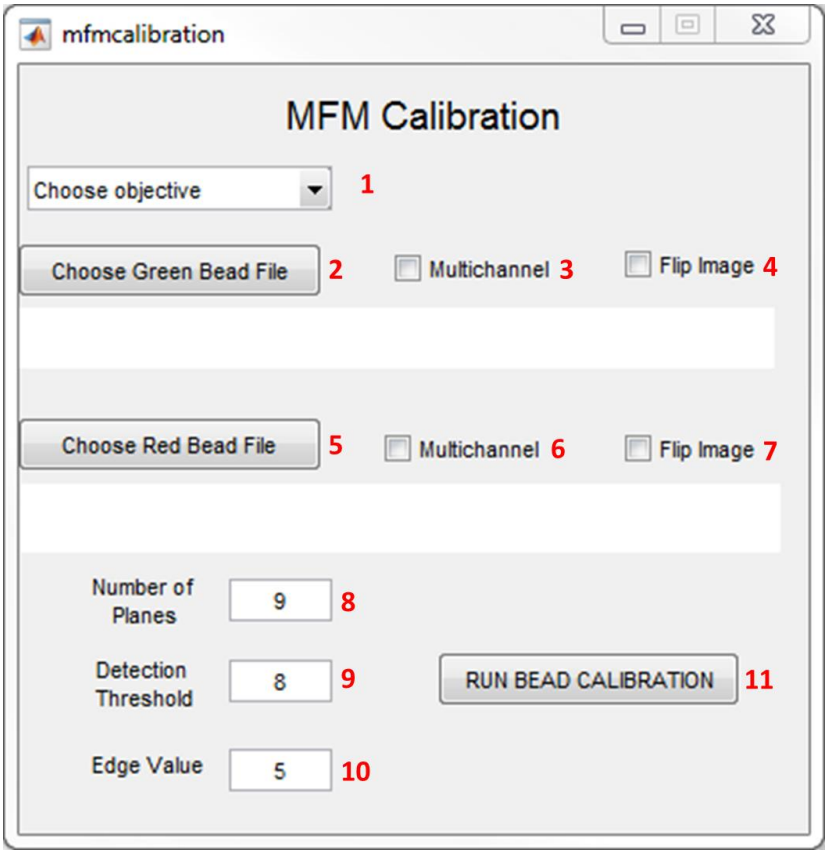

Figure 2. MFM calibration window. Red numbers reference the steps outlined below.

To perform a calibration:

1. Choose the correct objective lens from the drop-down menu at top (this will generally be 100x oil).

2. Click the **Choose Green Bead File** (if applicable), and navigate to the folder containing the .nd2 file corresponding to the green calibration image, and choose the file.

3. If the data contains two color data, check the Multichannel box.

4. If the data needs to be flipped left to right, click the Flip Image box. This is not generally needed for the AIC MFM.

5. Click the **Choose Red Bead File** (if applicable), and navigate to the folder containing the .nd2 file corresponding to the red calibration image, and choose the file. If the calibration data is multicolor, you will choose the same file as in step 2.

6. If the data contains two color data, check the Multichannel box.

7. If the data needs to be flipped left to right, click the Flip Image box. This is not generally needed for the AIC MFM.

8. Ensure that the number of planes is set to 9

9. Set the detection threshold to an appropriate value. Assuming that calibration images were acquired with sufficient signal-to-noise ratio, the value should be 5-40 (typically 8 is sufficient).

10. The Edge Value refers to the number of pixels from the edge of each image to ignore when detecting particles. This may be increased if there are particles very near the image edge, but should not be decreased below 5 in order to accurately measure the PSF.

11. Once all parameters have been specified, click **RUN BEAD CALIBRATION.**

The software will then attempt to read the .nd2 files corresponding to each color channel. It will automatically detect if the calibration is single or multi-color.

Either one (single color calibration) or two (multi-color calibration) Matlab figure Windows should appear. In the case of multi-color calibration, the first window shows the green channel bead image, and the second is the red channel bead image. Each of the 9 planes (in each color channel) will have a number next to each detected bead. Visually inspect all 9 images (across each color channel if applicable) to make sure that (i) each image has the same number of beads and (ii) each number references the same bead in all images. The figure below shows a multi-color calibration where these two conditions are satisfied for both color channels.

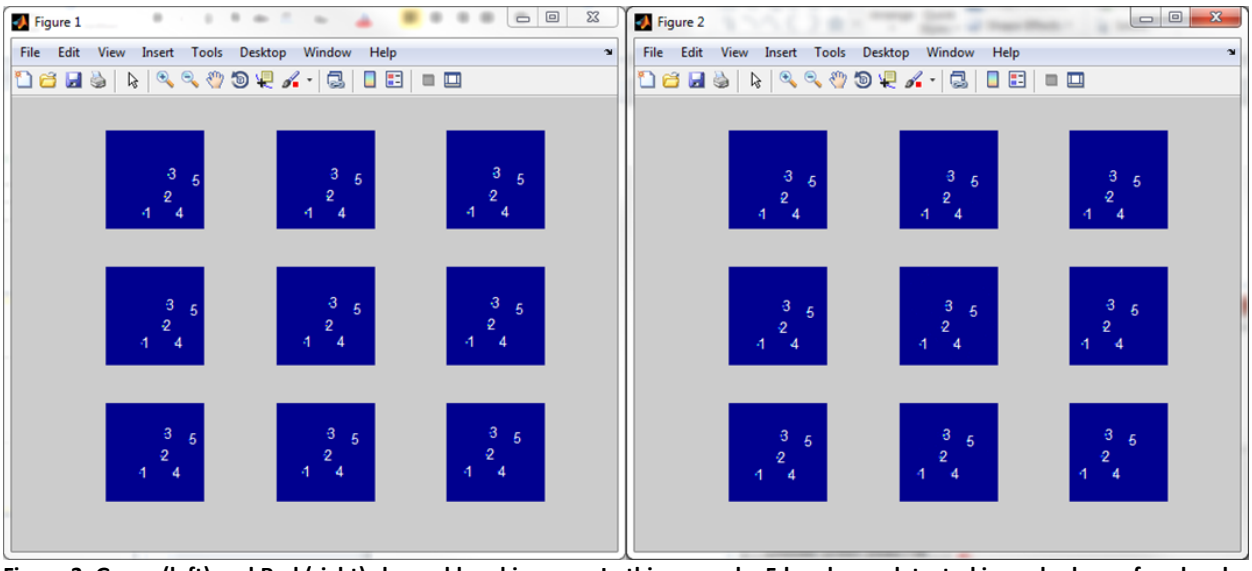

Figure 3. Green (left) and Red (right) channel bead images. In this example, 5 beads are detected in each plane of each color channel. Also, the relative position of each number with respect to the other numbers is consistent in each image/channel.

The software will prompt the user to confirm that all bead images satisfy these two conditions:

```
>>Beads ok? (y/n)
```

Type `y` if this is the case. If the bead images do not satisfy conditions (i) and (ii) above, then there are a few options. First, type `n` at the command prompt. This will exit out of the calibration GUI running mode. If a bead is apparent in the images, but there is no number next to it that indicates it was detected, try increasing or lowering the detection threshold. This is not a particularly sensitive parameter – increasing from 10 to 20 (or vice versa) will generally be necessary to see a difference. Also note if there are beads near the edges of any images. If this is the case, try increasing the Edge value parameter from 5 to 10. Once all parameters have been specified, click the RUN BEAD CALIBRATION button again, and follow the guidelines above. If the calibration procedure continues to fail even after adjustment of the threshold and edge parameters, it may be necessary to repeat the calibration image acquisition.

Once the bead detection has been verified (and you have typed `y` at the command prompt), the algorithm will proceed to calculate the relevant parameters needed for acMFM image processing/analysis. A number of windows will appear to indicate which parameters are being determined. In the case of a single color calibration only one of each of the following windows will appear. In the case of a multi-color calibration, each of the following windows will appear twice (one for each color).

The first window(s) to appear will denote the inter-plane spacing data. This is a plot of focal position vs. plane number. The slope of the red fit curve will correspond to the spacing between each focal plane. Note the R^2 value at top. A good calibration should have $R^2 > 0.98$. Note that the slope of the curve will be either positive or negative, depending on whether data was acquired in ascending or descending z-position. The software will accommodate either situation by taking the absolute value of the slope.

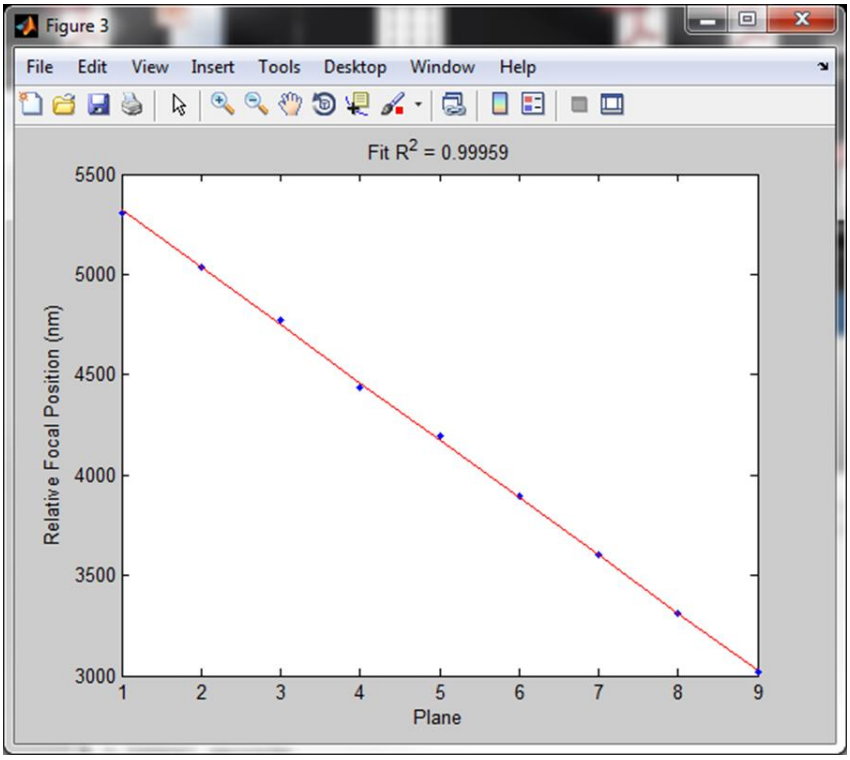

Figure 4. Plot of focal position vs. plane number. The slope of the red fit line is the interplane spacing. The R^2 value should be greater than 0.99.

The algorithm will then attempt to find the sub-pixel location of every bead in every plane, across each color channel (if applicable). The progress of this procedure will be displayed in the Matlab command window, indicating the bead and plane currently being localized.

Once the localization step has completed, the second window(s) to appear will indicate the alignment accuracy computed by the algorithm. In the case of a single color calibration, all images are aligned to the central plane. In the case of a multi-color calibration, all images are aligned to the central green-channel plane. Each image plane is shown as a two-color overlay, with the reference plane in green and the image being aligned in red. Atop each image will be the average alignment error for that plane, given in pixels. A value of *ca.* <0.2 is generally acceptable (corresponding to <25nm with the 100x objective). Figure 5 below shows an example alignment image for the red channel in a multi-color calibration.

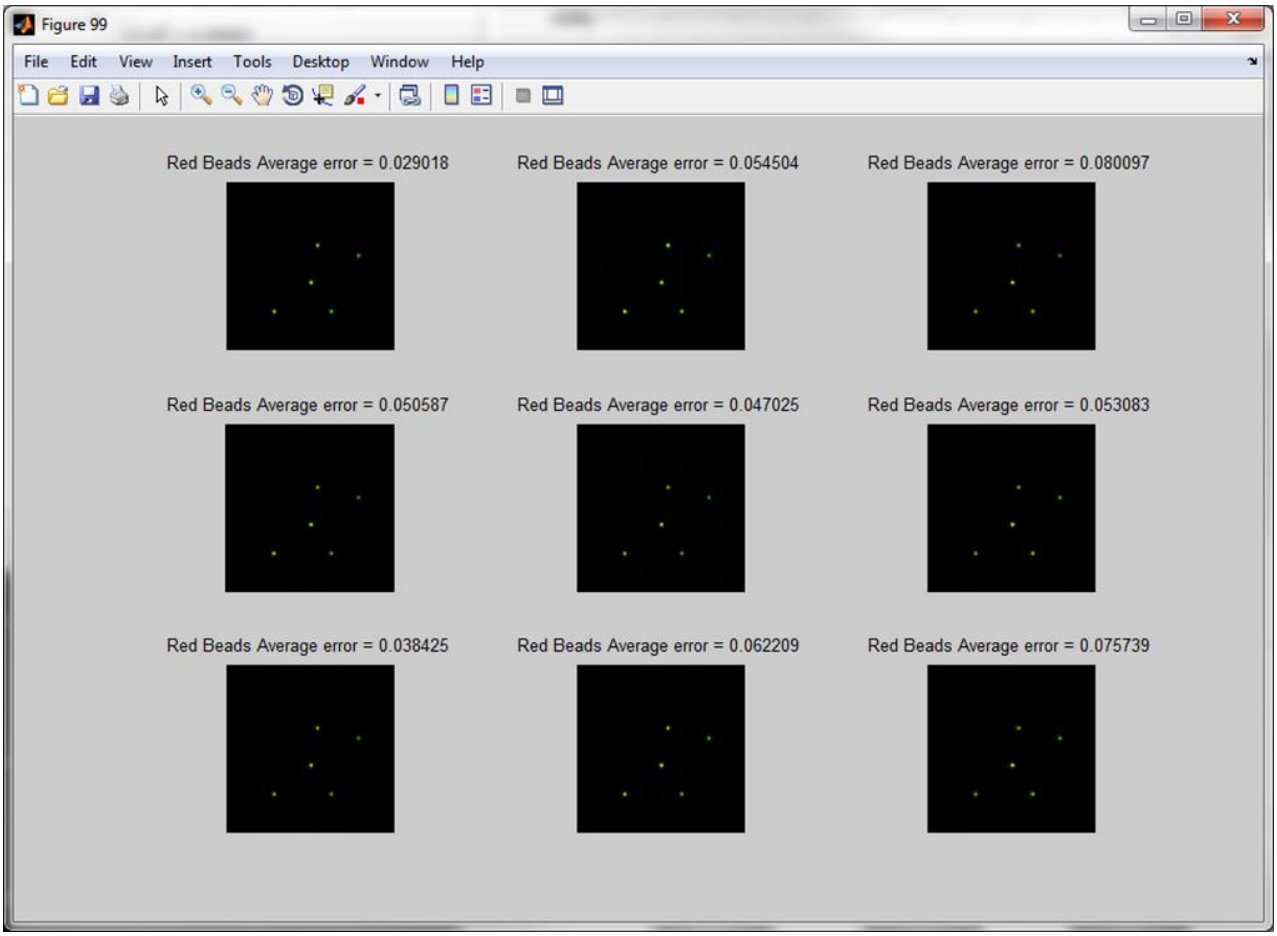

Figure 5. Example alignment image, in this case for the red channel in a multi-color calibration. The average alignment error (in pixels) for each plane is indicated on top of each corresponding image. A value of <0.2 for each plane is acceptable.

Finally, the calibration algorithm will produce three more images (for each color) that display the average point-spread function (PSF) in each acMFM plane. Images denote the xy, xz, and yz projections of the PSF, respectively. While there are no quantities that need to be monitored for this step, it is useful to observe the general shape of the PSF, and noting any large asymmetries or curvatures, particularly with the xz and yz projections. Figure 6 shows an example.

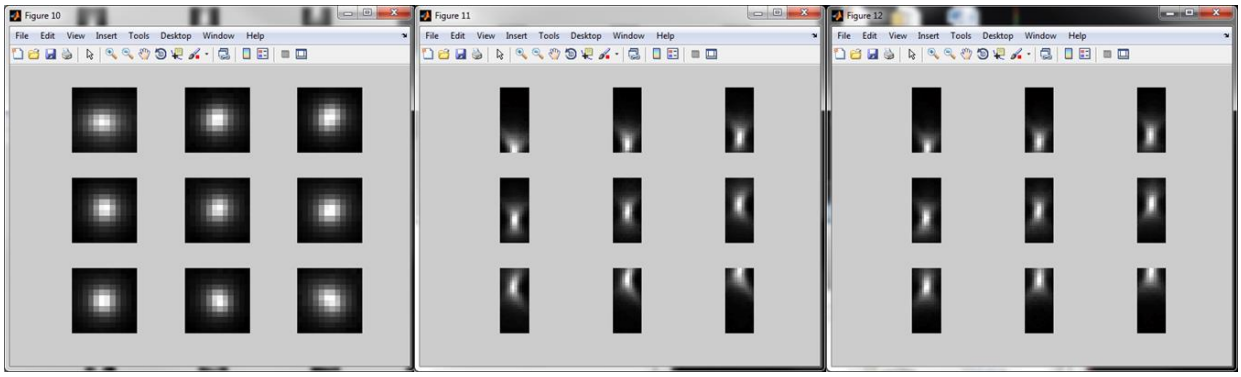

Figure 6. Example images of the point-spread function (PSF) in each of the nine planes. The xy (left), xz (middle) and yz (right) projections are each displayed.

All of the parameters calculated by the calibration algorithm are stored in a .mat file that is saved in the same directory as the raw calibration images. In the case of multi-color calibration, separate values for each color channel are stored. To view the contents, you can open the .mat file in Matlab. Although not necessary for any subsequent processing and analysis, viewing the results of the calibration may help identify potential problems. The following is a summary of the parameters saved in each calibration .mat file. In the case of a multi-color calibration, the names of some of the following variables will have either a ***g*** or an ***r*** appended to the end of their name to indicate whether it is specific for the green or red channel, respectively. You can view these values by simply dragging the calibration .mat file into the Matlab workspace.

- ***d***: the interplane spacing, in nanometers. The result will depend on wavelength and objective. At 515nm and 100x magnification, it should be ca. 350nm.

- ***fitparams***: The Gaussian fit parameters (in the Z-direction) used to estimate the relative focal position and average intensity across each MFM plane. This variable will be in the form of a cell array. Each cell in the array corresponds to each bead in the image. Within each cell, there is a matrix with 9 rows and 3 columns. Each row corresponds to each MFM plane. Each column corresponds to the intensity, focal-position, and z-width (in terms of 2x standard deviation) of the resulting Gaussian fit of the PSF in the z-direction.

- ***intscorrect***: derived from the data in *fitparams*, this variable lists the relative intensity distribution across each plane, normalized to unity. These values can vary widely with wavelength, with some MFM planes only reaching ~0.5 relatively intensity.

- ***objmag:*** The selected objective magnification

- ***psfs:*** a three dimensional matrix that represents the complete psf in each MFM plane. The matrix will be 12 pixels wide by 12x9 pixels tall. It will have a third dimension corresponding to the number of z-steps that contain an image with at least one psf in focus.

- ***tform:*** A nine-member cell array that contains the affine transformation matrices for each MFM plane. Each member of the cell array is stored as a structured array as per requirement for the Matlab *imwarp* function.

- ***zstepsize:*** The step size, in nanometers, used to take the z-series acquisition to acquire the calibration data set.

This completes the acMFM calibration process. To proceed to the next step, click the ***Align/Process Images*** button on the MFM software main window (see figure 1).

ALIGN/PROCESS IMAGES

MFM Image Processing

Once any pre-processing is completed, you can proceed to the Align/Process window described above, and shown in the figure below:

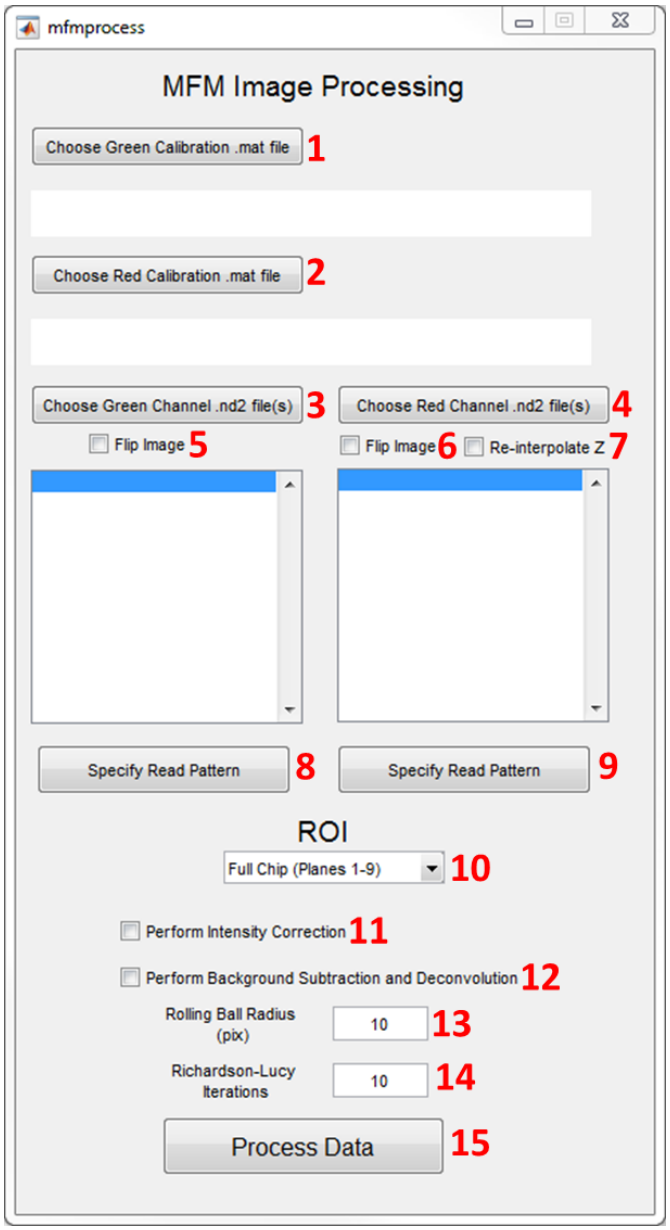

Figure 7. MFM Image Processing window. The red numbers correspond to the detailed steps outlined below.

1. Choose the green channel .mat file (if applicable) that was generated in the Calibration steps above.

2. Choose the red channel .mat file (if applicable) that was generated in the Calibration steps above. In the case of multi-color calibrations, this will be the same file as in step 1.

3. Now, choose the green channel .nd2 files (if applicable) that correspond to the imaging data you acquired. If the data is multicolor, the software will automatically read only the channel 1 images.

4. Then, choose the red channel .nd2 files (if applicable) that correspond to the imaging data you acquired. If the data is multicolor, the software will automatically read only the channel 2 images.

5. If the green data needs to be flipped left to right, click this box. This is not generally necessary for the AIC MFM.

6. If the red data needs to be flipped left to right, click this box. This is not generally necessary for the AIC MFM.

7. In the case of multicolor data, you can choose the re-interpolate the red channel data such that its z-spacing matches the green channel. This is useful for direct overlaying of green/red data. However, because the green data spans a smaller z-depth, doing this will reduce the red channel depth accordingly.

8. If the green channel data has been acquired using an interlaced or other custom illumination mode, then users may want to only process a portion of the total data set. In such cases click the Specify Read Pattern button. A new window will open that prompts users for 3 pieces of information:

    a. The pattern length: This is the total length of the illumination/acquisition pattern, as defined in the custom illumination plugin in the Nikon Elements software. In the case of 2-color interlaced acquisition, the pattern length is simply 2.

    b. The frames to keep within the specified pattern. This is expressed in Matlab vector notation. For example, if a custom illumination pattern is defined over 12 frames, and the data of interest is contained in the odd numbered frames up to the 9^{th} acquisiton, you would type 1:2:9 (start frame 1, increment by 2, end frame 9). In the case of 2-color interlaced acquisition, you can simply type 1 for the odd frames or 2 for the even frames.

    c. You can specify a file suffix to be added to the processed data file name that indicates which “subpopulation” of timepoints that were processed.

9. Same as 8 above, but applied to the red channel data.

10. AIC recommends using the full chip (planes 1-9) in order to maximize the z-depth of imaging. However, in special cases, users may acquire data using only a portion of the EMCCD chip in order to attain higher acquisition speed. In these cases, the user should specify which sub-region (corresponding to which planes) was used for the selected data. There are 3 possible sub-regions

which must be specified prior to acquisition of the data. Ask AIC staff how to set this up prior to image collection.

a. "Top 6" corresponds to a subregion containing pixel columns of 1-512, and pixel rows of 1-340.
b. "Middle 3" corresponds to a subregion containing pixel columns of 1-512, and pixel rows of 171-340.
c. "Bottom 6" corresponds to a subregion containing pixel columns of 1-512, and pixel rows of 171-512.

11. The MFM system does not perfectly distribute light intensity across each plane. To correct for this, check the box entitled "**Perform Intensity Correction**". This should make your images appear more uniform, and is should always be done unless troubleshooting the system.

12. To improve image quality and apparent resolution, you may perform background subtraction and Richardson-Lucy Deconvolution on your data sets by checking the "**Perform Background Subtraction and Deconvolution**" box. This step will improve the sharpness of your images, though will take longer. This will create a new directory within your data folder called **Deconvolution**, and all data will be placed in this new location. Checking this option will forgo saving non-deconvolved data.

13. Choose the rolling-ball background subtraction radius. This value should be just larger than the largest features of interest in your image, but generally smaller than the size of your cells.

14. Choose the number of Richardson-Lucy iterations. Larger number of iterations will increase the resolution of your images. However, more iterations on relatively low SNR images will begin to look "grainy" as the algorithm attempts to deconvolve noise spikes. Also, more iterations will increase processing time. Typical values are between 5 and 15.

15. Once all files and options have been chosen, click the **Process Data** button. This will initiate the MFM image alignment/processing algorithm. Depending on your computer's speed this can take some time (up to several hours in the case of many Gigabytes of data). A progress bar will denote the frames being processed for the current data file.

All processed data will be saved as a high-density format v5 file (.h5). This format is open source, and plugins exist in ImageJ and FIJI to view this data (explained below). A metadata file is also included in .csv format that contains the x, y, and z position of the image, as well as the time stamp for each time point in the image. Do not modify the .csv file, as it will be necessary for computation of diffusion coefficients in later steps (if applicable).

Viewing Data in ImageJ/FIJI

To view the 4D data created as described above in ImageJ or FIJI, you may select the following from the main FIJI window: File → Import →HDF5. This will open a dialog box to select the file of interest. Once the file has been selected, the plugin will identify the dimensions and number of time points in the file. A new window will open entitled "**Select Data Sets**", as shown in Figure 8. The top part of the dialog box allows you to choose which time point(s) to view. You can manually select particular time points of interest via ctrl+click or ctrl+shift+click. Alternatively, select the first time point, and then ctrl+A to select all time points. Then, ensure that the hyperstack (time series) option is selected, and press load. The data will be imported, and displayed as a 4D data set, as illustrated in Figure 9. This window will have two scroll bars beneath the image. The top scroll bar (indicated by green arrow in the figure) allows selection of the Z-focus plane. The bottom scroll bar (indicated by blue arrow in the figure) allows selection of the time point.

Further processing can be performed at this point, and will depend on the application in question. For example, users may wish to Gaussian filter, despeckle, Fourier filter, or create a 3D projection of the data at each time point.

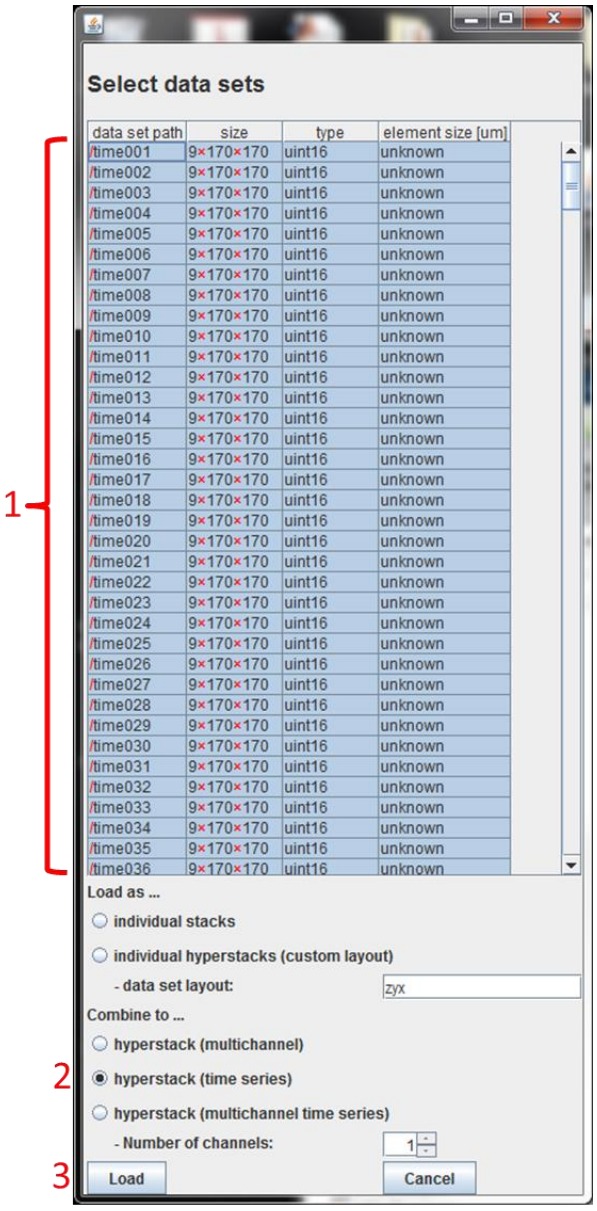

Figure 8. The ImageJ/Fiji HDF5 import plugin. After the .h5 file of interest has been selected, the Select Data Sets window will open. (1) Select the time points you wish to view, or simply click on any row in the table and then ctrl+A to select all. (2) Ensure that the resulting image is compiled as a hyperstack (time series). (3) Press Load button.

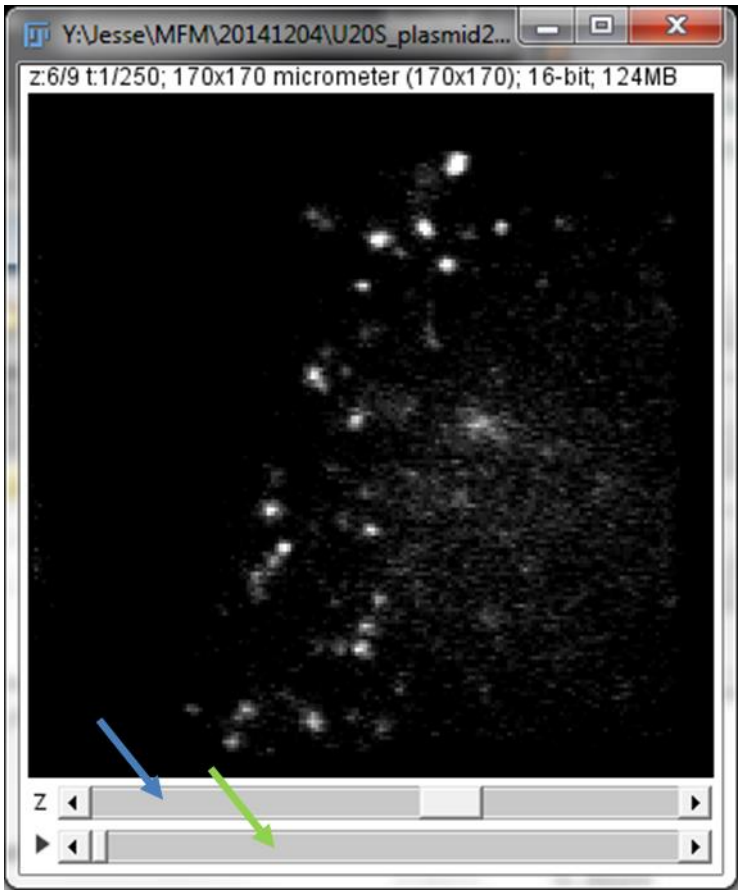

Figure 9. 4D data visualization in ImageJ/FIJI. The upper scroll bar (blue arrow) controls the z-focus plane to be displayed. The lower scroll bar (green arrow) controls the time point to display.

PARTICLE TRACKING

Considerations and Assumptions

Several parameters will determine the success of this analysis, including (i) signal to noise ratio, (ii) particle density, (iii) particle speed, and (iv) particle size. Higher SNR will increase the sensitivity and accuracy of the particle detection and localization process. Achieving an optimal particle density can be challenging from a sample preparation standpoint; ideally, however, particles should be spatially well separated from each other to avoid PSF overlap or ambiguity when trying to link particle locations into a trajectory. Similarly, particle speed can be a factor. While the acMFM can track particles with high acquisition rates, the combination of high particle density and fast velocity can produce ambiguity in forming trajectories.

Optimizing Particle Tracking Parameters in Fiji

Users will generally want to optimize the particle tracking parameters in Fiji using test data sets before performing automated tracking and analysis in Matlab. To do this, first open the test (.h5) data sets in Fiji/ImageJ, as described previously. Then click on Plugins → Mosaic → Particle tracker 2D/3D. You should see a window like that shown in Figure 10.

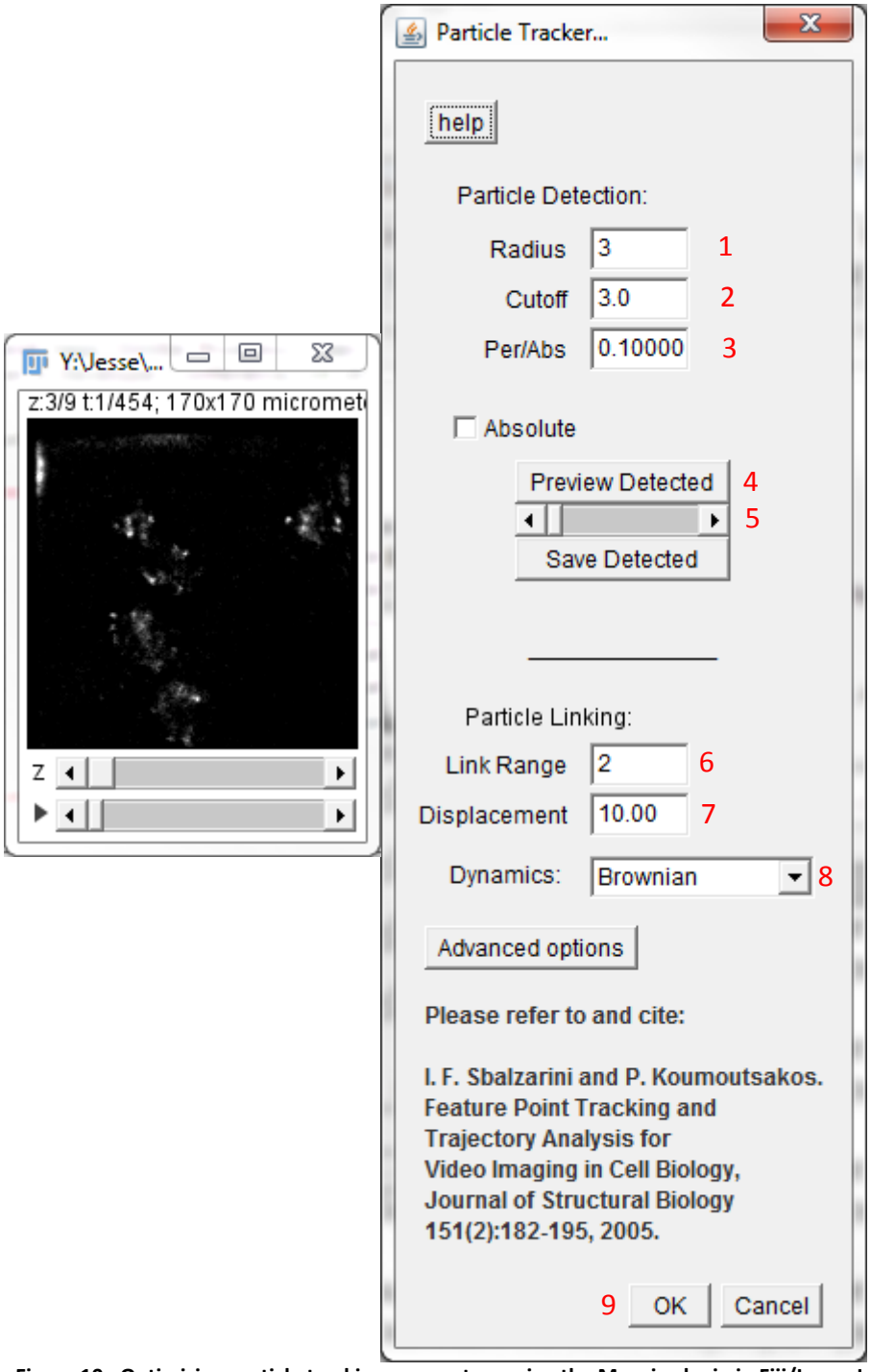

Figure 10. Optimizing particle tracking parameters using the Mosaic plugin in Fiji/ImageJ.

The dialogue box has several entries that are numbered and explained below.

1. Radius. This defines the size of the particles being tracked. This should match the features of interest, and for sub-diffraction particles (such as single molecules), the value should be set to 3.
2. Cutout. This defines a threshold score value by which the algorithm measures how symmetrical a particle is, as determined by image intensity moment analysis. Lower scores indicate a less

symmetrical particle, which may be used to discriminate features that are of the correct size, but undesirable shape. A default value of 3 is usually sufficient, but can be lowered to allow for particles with slight motion artifacts.

3. Per/Abs. This defines an intensity threshold, below which, particles are ignored. This is a relative threshold, and is thus less sensitive to photobleaching. By default, it is in terms of percentage, and refers to the top percentage of peaks that should be considered particles. Thus, lowering the value will lower the detection sensitivity (and usually increase the number of detected particles). The optimal value will depend heavily on the SNR in your image. However, values in the range of 0.025 to 5 are common.
4. Test the values in 1-3 above by pressing the preview detected button. Mosaic will then display a new window that highlights the detected particles using a red symbol, as show in Figure 11. Note that this result is for a single time point, but all z-planes in your image. The particular shape of the red-symbol denotes whether its peak z-position is 1, 2, or 3 planes away from the current view. Also note, particles whose peak z-position is in the first or last z-plane are not counted, as their z-position cannot be determined.

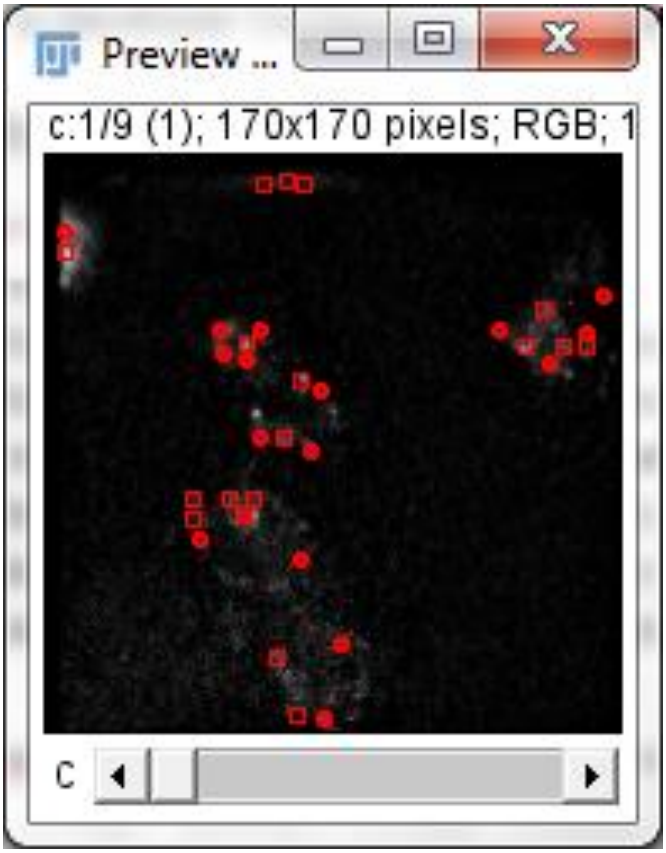

Figure 11. Preview detected particles result. Particles are denoted by a red symbol. The exact symbol (circle, square, start, etc) indicate whether the particle's peak z-position is 1, 2, or 3 frames away from the current view. Note that particles whose maximum z-intensity occurs in the first or last z-plane are not counted.

5. Scroll bar. You can advance the data in time using the scroll bar to check the particle detection values across multiple time points. This is important to gauge how sensitive your particle detection algorithm at later time points, when photobleaching may become more apparent.

Once you are satisfied with the particle detection parameters (all visible particles are being recognized), you can proceed to the particle linking step.

6. Link range. This refers to the number of frames that the algorithm will "look ahead" when attempting to link two particles in order to construct a particle trajectory. It is generally advisable to set this equal to 1, particularly in the case of higher particle density, and no fluorophore blinking.
7. Displacement. This refers to the maximum distance a particle may travel (in any direction) between successive time points. Performing the preview detection step across successive time points can help you estimate this value. Compare the position of a few particles at time = t and t+1 to gauge the maximum displacement in x, y, or z. In many applications, this value is 400-750nm, depending on diffusion constant and acquisition time.
8. Dynamics. The Mosaic plugin can optimize the linking procedure if the particle being tracked are known to behave in a certain way. In many applications, the dominant mode of motion is Brownian – that is, a random walk. However, in other applications, particles may be mainly actively transported in one particular direction over several frames. If this is the case, choosing "constant velocity" may help avoid linking ambiguity, particularly in the case of high particle density.
9. OK. This initiates the Mosaic plugin. The algorithm will proceed first by detecting particles in each track, using parameters 1-3. It will then link each particle into tracks using parameters 6-8. Once it has completed, a new window will appear, as shown in Figure 12.

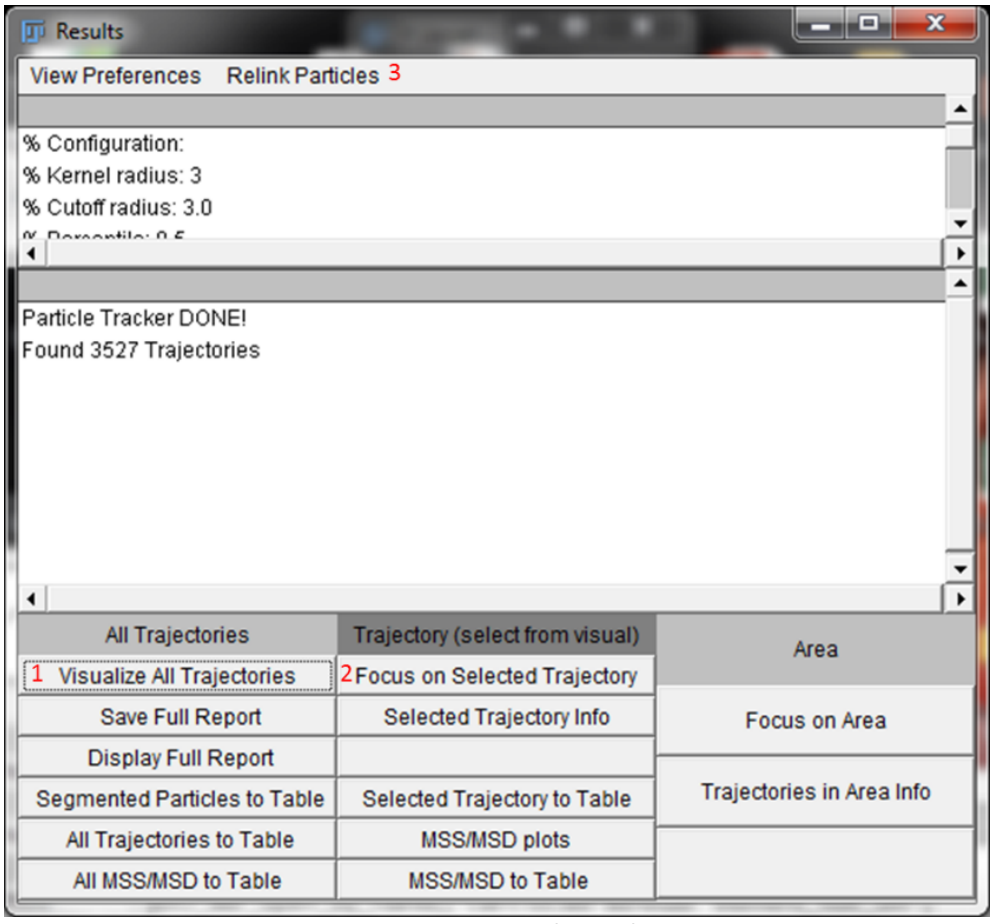

Figure 12. Mosaic Results Window.

The window will report the total number of tracks that were produced. To view a visual representation of the tracks, click the "Visualize All Trajectories" button (1), as shown in Figure 13.

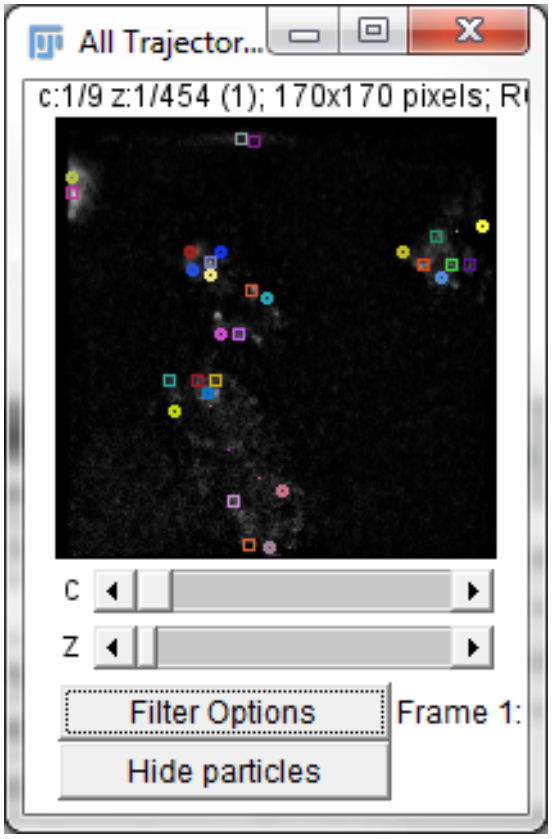

Figure 13. Mosaic Visualize trajectories window

In this window, you can scroll in z (top scroll bar) and time (bottom scroll bar). This will allow you to visualize the trajectories Mosaic has calculated. You can click on a particular track, which will create a yellow bounding box around the particle of interest. Then, in the results window, click "Focus on Selected Trajectory" (2) to see a zoomed in view of that isolated trajectory. To aid in finding good trajectories, you can click the "Filter Options" button at the bottom of the visualize trajectories window. This will allow you to remove trajectories below a certain length (in frames).

By inspection, you should be able to assess the ability of Mosaic to create faithful trajectories from the data. If you notice that the tracks are taking relatively large "jumps" between time points, which may indicate that the algorithm is linking two different particles. In this case, you may want to reduce the displacement value. You can re-link the particles by clicking the relink particles button at the top of the results window (3). This will allow you to optimize the linking parameters without having to re-detect the particles.

Automated Particle Tracking in Matlab

Although Mosaic provides a simple method to calculate particle trajectories, it does not lend itself well to batch processing, or straightforward visualization and analysis. Thus, once you have determined

appropriate parameters, you can utilize the MFM particle tracking package, by clicking on the bottom button in the main MFM Matlab window (See Figure 1). The following window will appear:

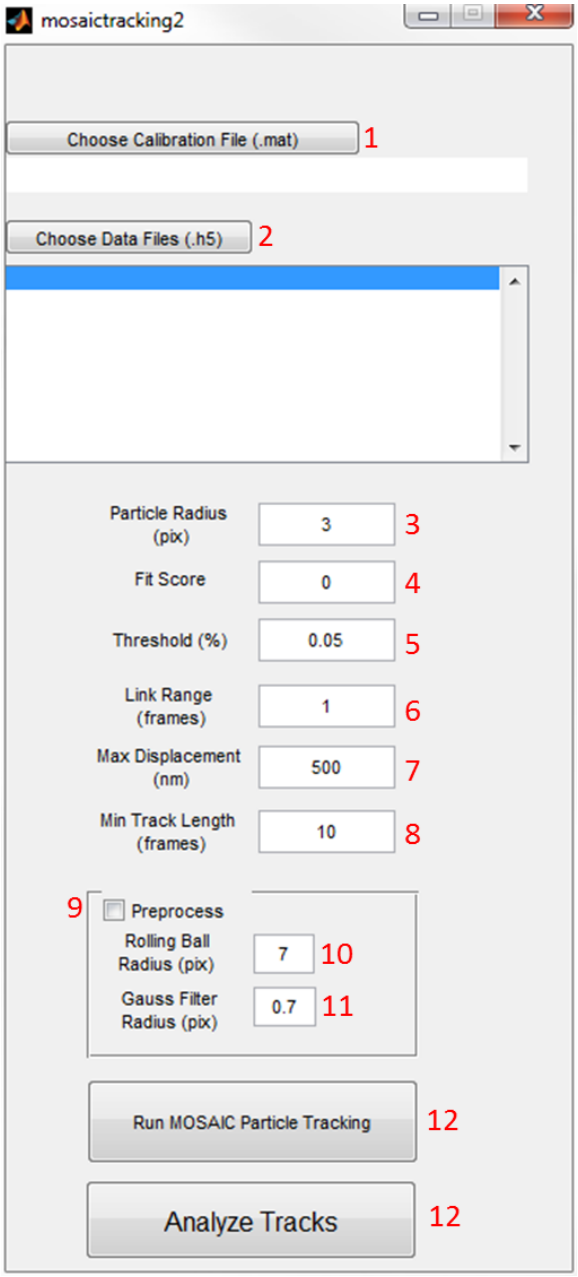

Figure 14. MFM particle tracking window

This GUI will allow you to run the Mosaic particle tracking for multiple data sets, while filtering for tracks above a certain length. The following procedure should be used (corresponding to the red numbers in Figure 14).

1. Choose the calibration (.mat) file used to process the datasets to be analyzed.

2. Choose the processed data (.h5) files that you want to analyze with the Mosaic particle tracking software.

3. Particle radius. This corresponds to parameter 1 that you optimized in the mosaic window (see Figure 10), as described in the previous section.

4. Fit score. This corresponds to parameter 2 in Figure 10.

5. Threshold %. This corresponds to parameter 3 in Figure 10.

6. Link range. This corresponds to parameter 6 in Figure 10.

7. Max Displacement. This corresponds to parameter 7 in Figure 10. However, in order to prevent confusion with magnification, this parameter is in terms of nanometers (not pixels). Thus, for instance, if it was determined that 4 pixels was sufficient in Figure 10, and the data was acquired using a 100x objective, the value should be 4*120nm = 480nm.

8. Min Track Length. This is the minimum track length (in frames) that should be considered. This will be important later should you want to compute diffusion constants, but also removes some of the "clutter" of short trajectories. A good value is 10-20.

9. Users may wish to perform pre-processing of their data before particle tracking, in order to artificially increase the signal to noise in their data. Care should be taken to balance any possible errors in particle localization introduced by such pre-processing with the increased particle detection and linking efficiency this procedure can allow. Users should first investigate and optimize the effects of pre-processing on the final particle tracking data before using on all data. If you want to pre-process your data, click the check box. This will also cause the software to save a .tif file of your processed 4D data in the same directory as the original data for later viewing.

10. If preprocess has been selected, first specify a background subtraction rolling ball radius, in pixel units. This is equivalent to parameter 10 in figure 7, but without the deconvolution. Smaller values will emphasize smaller features (such as diffraction limited spots).

11. If preprocess has been selected, then specify a Gauss filter radius. This will apply a 2D Gaussian filter in each image (across z and time), and can be helpful to artificially increase your signal to noise. Values of 0.5-1 are typically useful for diffraction limited spots.

12. Once you have input the parameters (which should have been optimized using the procedure in the previous section), click the "Run MOSAIC Particle Tracking" button. The algorithm will read in the .h5 data, export to Fiji/ImageJ, perform the particle tracking. The results from this analysis will be stored in the same folder as the image data, and include two files:
    a. A .csv file that contains all of the tracking data found by Mosaic
    b. A .mat file that contains all tracks above the threshold length, which can be imported into Matlab.
13. Once the algorithm has completed all the files listed under box 2, you can proceed to the Analyze Tracks section to model diffusion constants and view the results. Pressing the "Analyze Tracks" button will produce the window shown in Figure 15. This window is explained in more detail in the next section.

Particle Trajectory Analysis

A multitude of parameters can be extracted from particle tracking data, depending on the model being used. One of the more common models, and the one which is used in this analysis, is based on the following equation, and is termed *Anomalous Diffusion*:

$$MSD = \gamma D \Delta t^{\alpha} \qquad (1)$$

This relation indicates that the the mean-squared displacement of the particle (*MSD*) as a function of time-lag *Δt*, is related linearly by the diffusion constant D, and exponentially by factor α. In this case, γ is equal to twice the dimensionality of the diffusion process.

This model is used because it can account for "non-Brownian" motion, which is very commonly encountered in biological systems. While the diffusion constant is a well-known quantity, the factor α is less so. Most simply, the value of α can give an indication of whether a particle is "sub-diffusing" (the particle's Brownian motion is constrained), if its value is less than one, or "super-diffusing" (also called active transport), if α is greater than one. A value near unity indicates near Brownian (or random walk) motion.

From the previous section, clicking on the "Analyze Tracks" button (as shown as button #10 in Figure 14) will produce the following window:

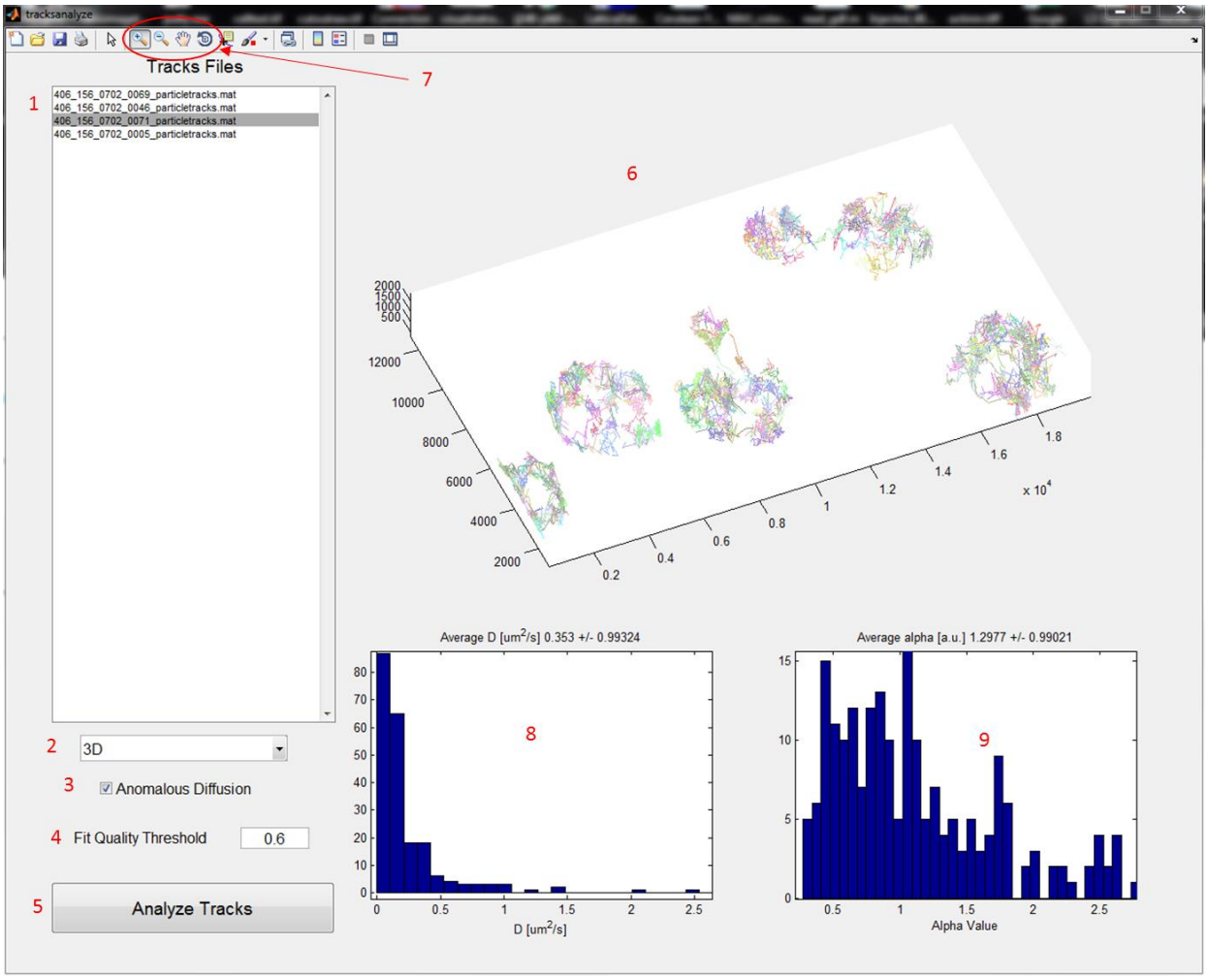

Figure 15. MFM Analyze Tracks Window.

This window will allow you to analyze the particle trajectories computed by the MOSAIC plugin. This is explained below, according to the red numbers shown in Figure 15.

1. The program will automatically search among the files listed in the previous window to find a corresponding "particletracks" file. Click on the name of the file that you want to analyze.

2. Choose the dimensionality of the diffusion process. Although the MFM tracks molecules in 3D, their motion can be confined (for example in a membrane) such that it only has 2 degrees of freedom in its motion. In situations like these, it may be appropriate to select 2D. Otherwise, choose 3D.

3. Anomalous Diffusion. Check this box to allow the value of α in equation 1 to vary. This allows for modeling constrained diffusion and active transport phenomena.

4. Fit quality threshold. Every track is modeled whereby the MSD of the particle as a function of time-lag is fit, via least-squares regression, to equation 1. By choosing a threshold for the goodness of fit of that regression, we can reject trajectories that are not well described by that

equation. This is arbitrary, but a value of 0.5-0.8 (in the anomalous case) is common to ensure high-quality fits.

5. Press the Analyze tracks button to begin the analysis and least squares regression. This may take some time, depending on the number of tracks.

6. Once the command window says the process is done, data will populate the 3 graphs shown to the right of the file list. At top is a 3D plot of the particle trajectories that were successfully modeled (given the fit quality threshold). Each trajectory is given a unique color.

7. The 3D trajectory plot can be manipulated via the figure toolbar at the top of the window. Click the corresponding buttons to zoom in/out, pan, and rotate the 3D graph to better view the data.

8. A histogram of all the computed diffusion constants is shown on the bottom left, with the average +/- standard deviation value shown in the title. This graph can also be zoom using the same buttons described in 7.

9. If anomalous diffusion was selected, the alpha values are plotted in a histogram on the bottom right.

Data Manipulation in the Matlab Workspace

The results of the analysis are saved in a new .mat file. The name of this file will be the same as the original image file, with a "_particletracks_analyzed.mat" appended to the end. This file can be opened in Matlab to give a variable called *output. Output* is a structured variable that contains the successfully modeled tracks, as well as corresponding diffusion constants and alpha values (if applicable). To access these values, use the following syntax:

```
>> tracks = output.tracks;
>> D = output.D;
>> alpha = output.alpha;
```

tracks consists of a cell array of each successfully modeled particle trajectory. Each entry in the cell array consists of a Nx4 matrix, where N is the number of frames spanning the trajectory. Each column corresponds to the x, y, z, and frame number, respectively, of the particle.

If you would like to independently plot the tracks in a separate figure (e.g. to save as a publication figure), type:

```
>> plottracks(tracks);
```

D is a vector of the calculated diffusion constants, in um^2/s. It is arranged such that the i^{th} entry in D corresponds to the i^{th} track in the *tracks* variable.

Alpha is a vector of computed alpha values, arranged similarly to the D vector.

In both cases, you can make a simple histogram of D and alpha using:

```
>> hist(D);
```

Or

```
>> hist(alpha);
```

The AIC is happy to discuss further analysis, and improvement to the current analysis!