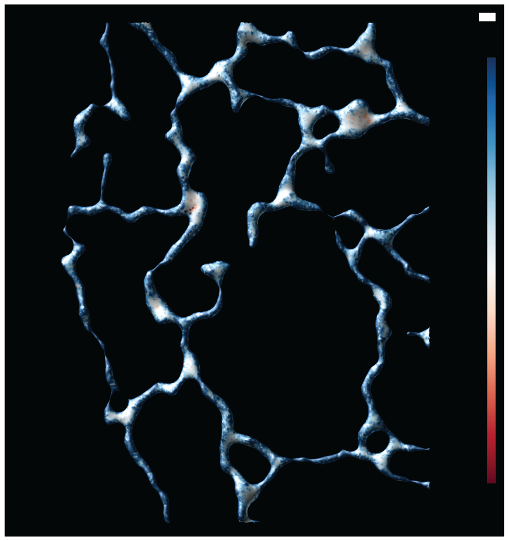
SUPPLEMENTARY MATERIAL



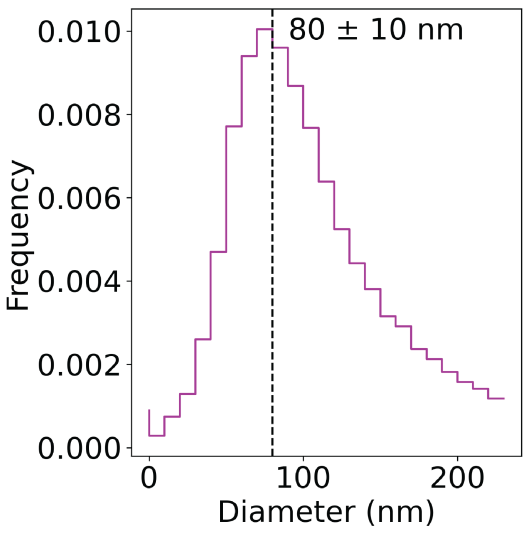


Figure S1: **Left,** Tubular ER from the lower left portion of Figure 3, colored by mean curvature. Lookup table: -0.01 to 0.01 nm^{-1}. Scale bar is 200 nm. **Right,** Histogram of $D = 2/k_{\max,l}$ where $k_{\max,l}$ is the maximal principal curvature of vertex l. The mean and standard error of the mean for the diameter distribution are reported.

Point cloud simulation

In order to demonstrate the effectiveness of the algorithm, it was necessary to generate biologically-motivated test structures. To do this, we established a constructive solid geometry library where each primitive is represented as a signed distance function (SDF).

Points were generated from a test structure using Monte-Carlo sampling of an octree of the structure's SDF. The octree subdivides space that straddles where the SDF goes to zero until a fixed sampling rate dx is reached. The center of each octree box with an absolute SDF value of less than dx is kept with a probability of p and rejected otherwise. This results in a point set of size S.

Next, non-specific localizations are simulated. For a given noise fraction n, $J = \frac{nS}{1-n}$ additional points are generated with uniform randomness over the bounding box of the point set. This ensures $J/T = n$ for $T = J + S$ total points.

To simulate localization precision, each point i in the resulting point set is jittered by sampling values from a Gaussian with mean μ_i equivalent to the location of point $i \in 0\,..\,T-1$ and standard deviation $\boldsymbol{\sigma}_{i,e}$ for $e \in x, y, z$. Up to N samples are generated per point. The total point set is then clipped to size T via random uniform selection of points.

The uncertainty $\boldsymbol{\sigma}_{i,e}$ is calculated by simulating an exponential distribution with a mean value of ρ, corresponding to the average photon count of a simulated localization. This distribution is clipped to only contain values greater than a background photon count b, simulating the noise floor for data collection. For the remaining counts, the first S are selected and

$$\sigma_{i,e} = \frac{r_e}{2.355\sqrt{\rho_i}}$$

where r_e is the resolution of the simulated system along axis e.

Meshing parameters

To generate the parameter space for evaluation of SPR meshes derived from simulated data in Figure 2, we followed recommendations in (18). For SPR we estimated normals from 10, 30, 50 and 100 nearest neighbors, set the smoothing parameter $\alpha = 0, 1, 2$ and 4, and set the samples-per-node to $1.5, 5, 10, 20$ and 30. We used an octree depth of 8, 8 Gauss-Seidel relaxations, and a scale factor of 1.1.

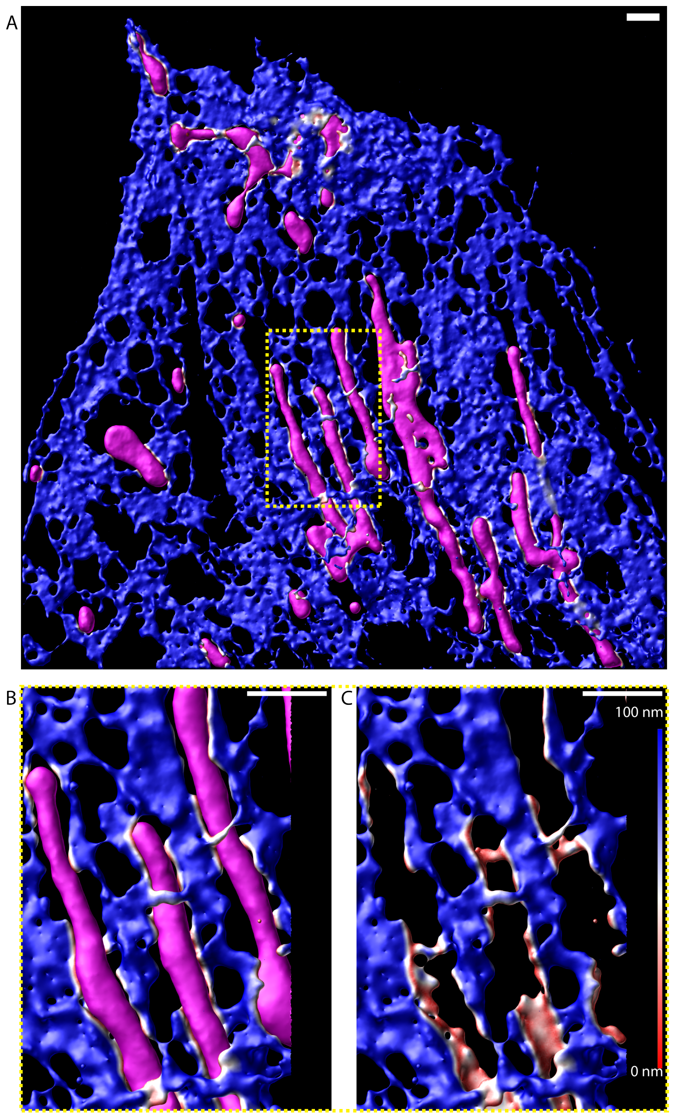


Figure S2: Distance between two surfaces. Surface generated from Sec61β localizations (red-white-blue) and surface generated from TOMM20 localizations (magenta). The Sec61β surface is colored by its distance from the TOMM20 surface (0 to 100 nm; red to blue). **A,** Full field of view, displaying both surfaces. **B,** Region of interest, showing both surfaces. **C,** The Sec61β surface alone. Scale bars are 1 µm.

For our method, we compared maximum number of iterations 19 and 39, a variety of starting threshold densities, 5×10^{-6}, 1×10^{-5}, 2×10^{-5}, 5×10^{-5}, 1×10^{-4}, 2×10^{-4}, 4×10^{-4}, 1×10^{-3}, 2×10^{-3}, and $\lambda = 10, 15, 25$. We remeshed every 5 iterations.

	Max iters	Curvature weight	Remesh frequency	Neck first iter	Punch frequency	Kc	Minimum edge length	Smooth curvature	Truncate at
Figure 3	19	20	5	0	0	1.0	10	True	1000
Figure 4 A-C	29	20	5	0	0	1.0	10	True	1000
Figure 4 D-F (ER)	29	20	5	0	0	1.0	10	True	1000
Figure 4 D-F (Mito)	49	20	5	0	0	1.0	10	True	1000
Figure 4 G-I	9	20	5	0	0	1.0	10	True	1000

Table S1: Parameters used for iterative surface fitting of experimental data in this paper.

USER GUIDE

This tutorial will walk you through using the shrink-wrapping method within PYMEVisualize to create 3D surfaces based on a provided 3D point-cloud data set (`user-guide-data.hdf`). After completing this tutorial, you should be able to apply the same approach to shrink-wrap your own 3D point-cloud data sets. If you are unfamiliar with using PYMEVisualize, we strongly encourage you to first install PYME and go through the PYMEVisualize User Guide to understand the basics of the program before moving on to this tutorial (24).

Install the Shrink-wrap Plugin

You can find the GitHub repository for the shrink-wrap plugin and installation instructions here.

Download and import tutorial data set

Download the data set `user-guide-data.hdf` found in the supplementary materials. Launch a PYMEVisualize window by entering `visgui` into an Anaconda prompt with the PYME environment activated. Navigate to **File** → **Open** and select the `user-guide-data.hdf` file. To improve the visualization of our data, under the "Layers" side tab change "Method" to *pointsprites*, "Colour" to *z*, "Alpha" to *0.2*, and "Point size" to *10*. Your PYMEVisualize window should look identical to Figure S3 now.

Create initial isosurface

You can learn about PYMEVisualize isosurfaces and how to make them here. Once you're familiar with how to create an isosurface, use the parameters shown in Figure S4 to create an isosurface based on the `user-guide-data.hdf` points. Once finished, you should see a 3D surface rendered with your points. Change "Method" to *wireframe* to more easily visualize how well the isosurface adheres to the point cloud. It should look something like Figure S5. At this point, it is a good idea to assess the overall accuracy of this initial isosurface. You can alter the parameters for the isosurface in the **Data Pipeline** tab on the left under **DualMarchingCubes**. It is typical to test several parameters before settling on the best options with "N points min" and "Threshold density" being the two major parameters requiring tweaking. It should be noted that this data set is a small cropped region of interest (ROI) from a much larger data set. It is highly recommended that you test parameters for the initial isosurface (and the eventual shrink-wrapped surface) on a smaller ROI that has enough points to test the structure you're trying to create a surface from, but no more than that (about 20,000 points is ideal). The time required to create the surfaces scales roughly with the number of localizations, so testing parameters on large data sets is impractical. Once you establish which parameters are best with your smaller ROI, you can apply the same ones to the full data set. Learn how to create an ROI in PYMEVisualize here.

Create shrink-wrapped surface

Now that we have our initial isosurface, we can use it to create a shrink-wrapped surface. You will find this option under **Mesh** → **Shrinkwrap membrane surface**. Once selected, a new window will pop up for you to select the parameters you'd like to use. Refer to Table S2 for descriptions of what each parameter is and what typical values are for it. For this tutorial, use the parameters shown in S6. This process can take a minute or so. Check the Anaconda prompt for output to confirm it is running. Once finished, your shrink-wrapped surface should look similar to S7.

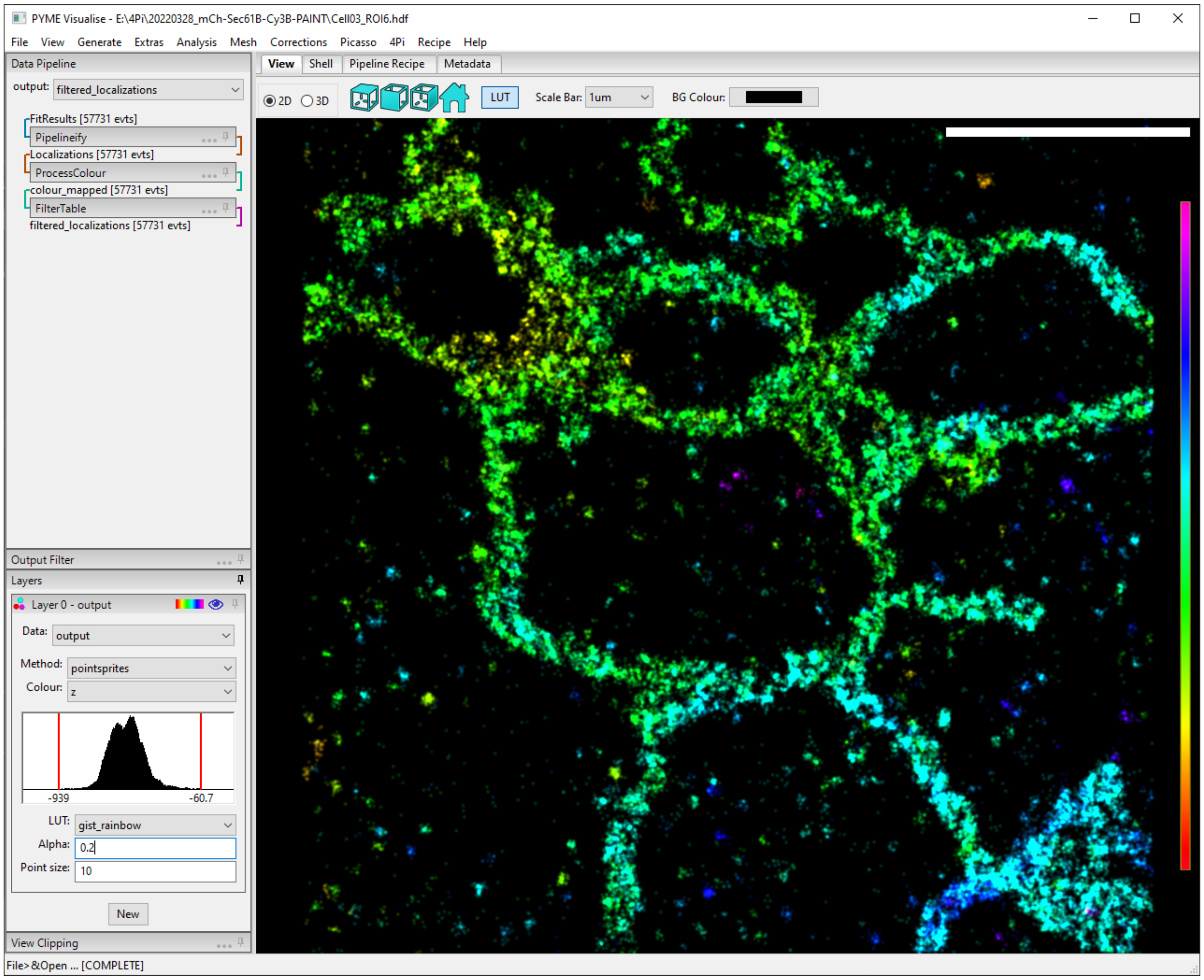
Additional tips for shrink wrapping surfaces

The following are tips for situations that did not come up in the tutorial, but occur frequently enough to warrant mentioning:

- An error that can occur when attempting to shrink wrap is the "singular matrix" error. This simply means that the operation reached a point that was mathematically impossible to calculate. It is most often caused by several surfaces being present after creating an isosurface, especially small surfaces that are often capturing background localizations. Most of the time, this can be fixed be removing all but the largest surface present. To do so, enter the following line the the **Shell** tab in PYMEVisualize: `pipeline.dataSources['surf0'].keep_largest_connected_component()`.

NOTE `surf0` should be the name of **your** isosurface. By default it is `surf0`, but can be a different name for various reasons.

NOTE This solution clearly isn't always reasonable to use. For data from an ER protein, it is reasonable since the ER is continuous. However, data from a mitochondrial protein is not so reasonable since mitochondria are discontinuous.


PYME Visualise - E:\4Pi\20220328_mCh-Sec61B-Cy3B-PAINT\Cell03_ROI6.hdf
File View Generate Extras Analysis Mesh Corrections Picasso 4Pi Recipe Help
Data Pipeline
View Shell Pipeline Recipe Metadata
2D 3D
LUT
Output Filter
Layers
New
View Clipping
-939
-60.7

output: filtered_localizations

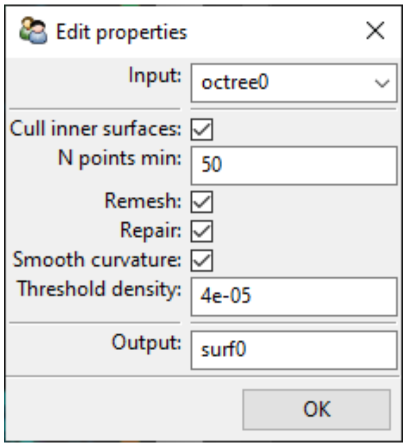
FitResults [57731 evts]
Pipelineify
Localizations [57731 evts]
ProcessColour
colour_mapped [57731 evts]
FilterTable
filtered_localizations [57731 evts]

Scale Bar: 1um
BG Colour:

Layer 0 - output
Data: output
Method: pointsprites
Colour: z
LUT: gist_rainbow
Alpha: 0.2
Point size: 10

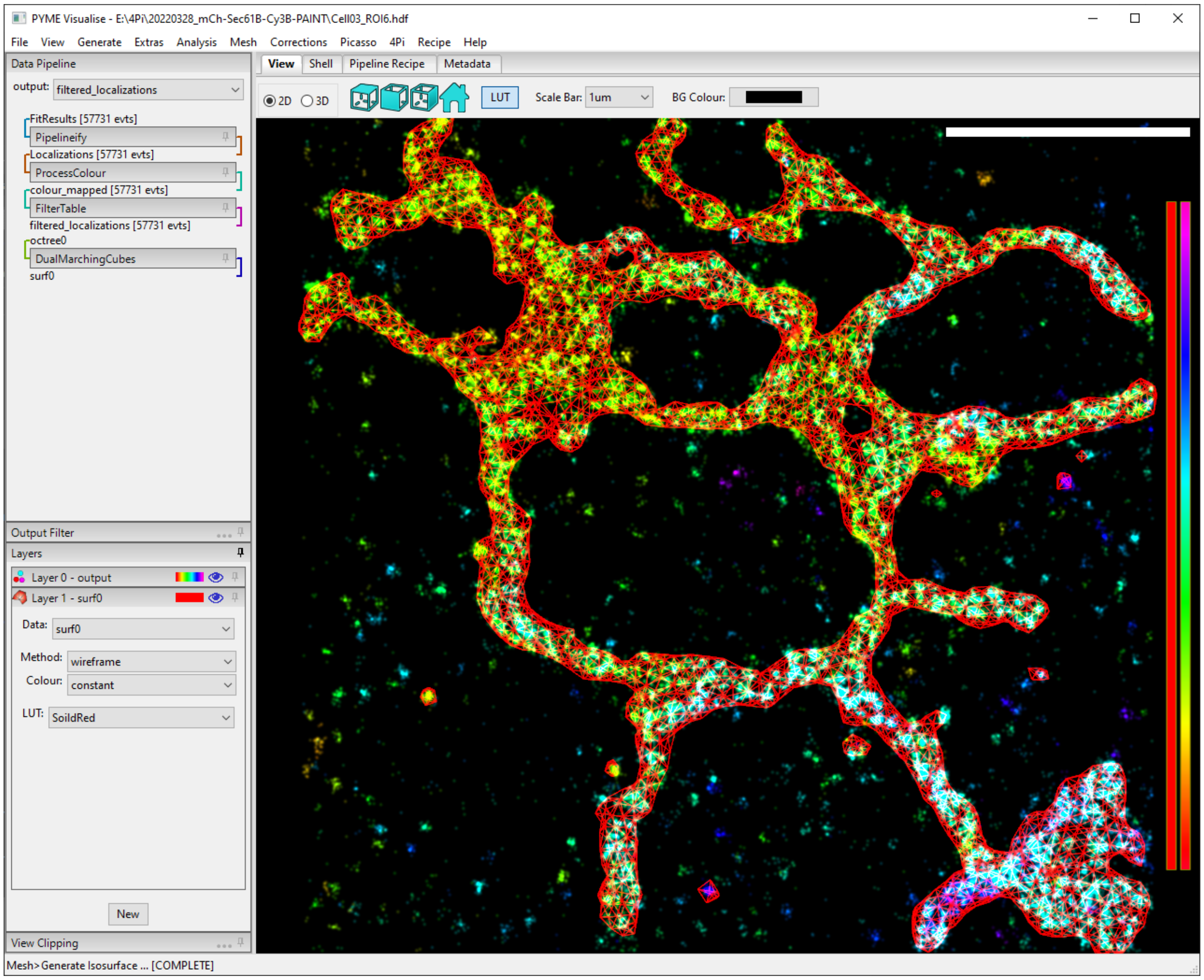
File>&Open ... [COMPLETE]

Figure S3: Data from `user-guide-data.hdf` displayed as 10 nm point sprites with 0.2 alpha. The points are colored by their location in the z-direction according to the lookup table on the right.



Input: octree0
Cull inner surfaces: ✓
N points min: 50
Remesh: ✓
Repair: ✓
Smooth curvature: ✓
Threshold density: 4e-05
Output: surf0

Figure S4: Ideal parameters to use to create an isosurface based on `user-guide-data.hdf`.



PYME Visualise - E:\4Pi\20220328_mCh-Sec61B-Cy3B-PAINT\Cell03_ROI6.hdf

Data Pipeline

output: filtered_localizations

FitResults [57731 evts]
Pipelineify
Localizations [57731 evts]
ProcessColour
colour_mapped [57731 evts]
FilterTable
filtered_localizations [57731 evts]
octree0
DualMarchingCubes
surf0

Layers

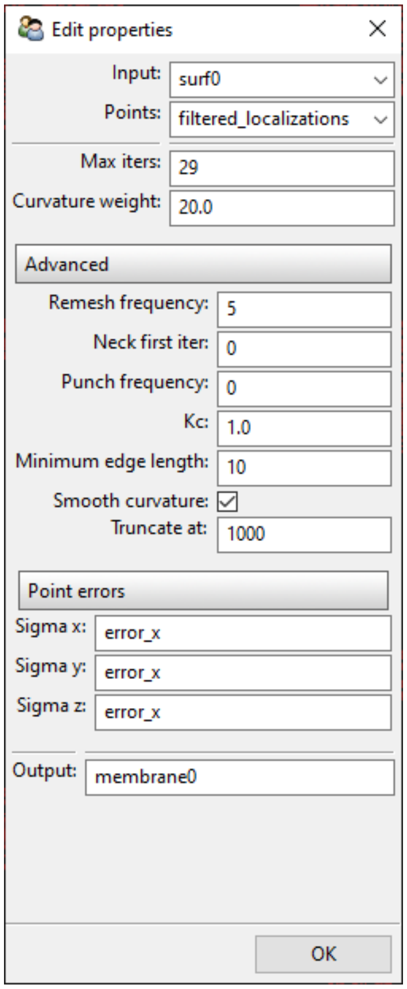
Layer 0 - output
Layer 1 - surf0

Data: surf0
Method: wireframe
Colour: constant
LUT: SoildRed

Scale Bar: 1um
BG Colour:

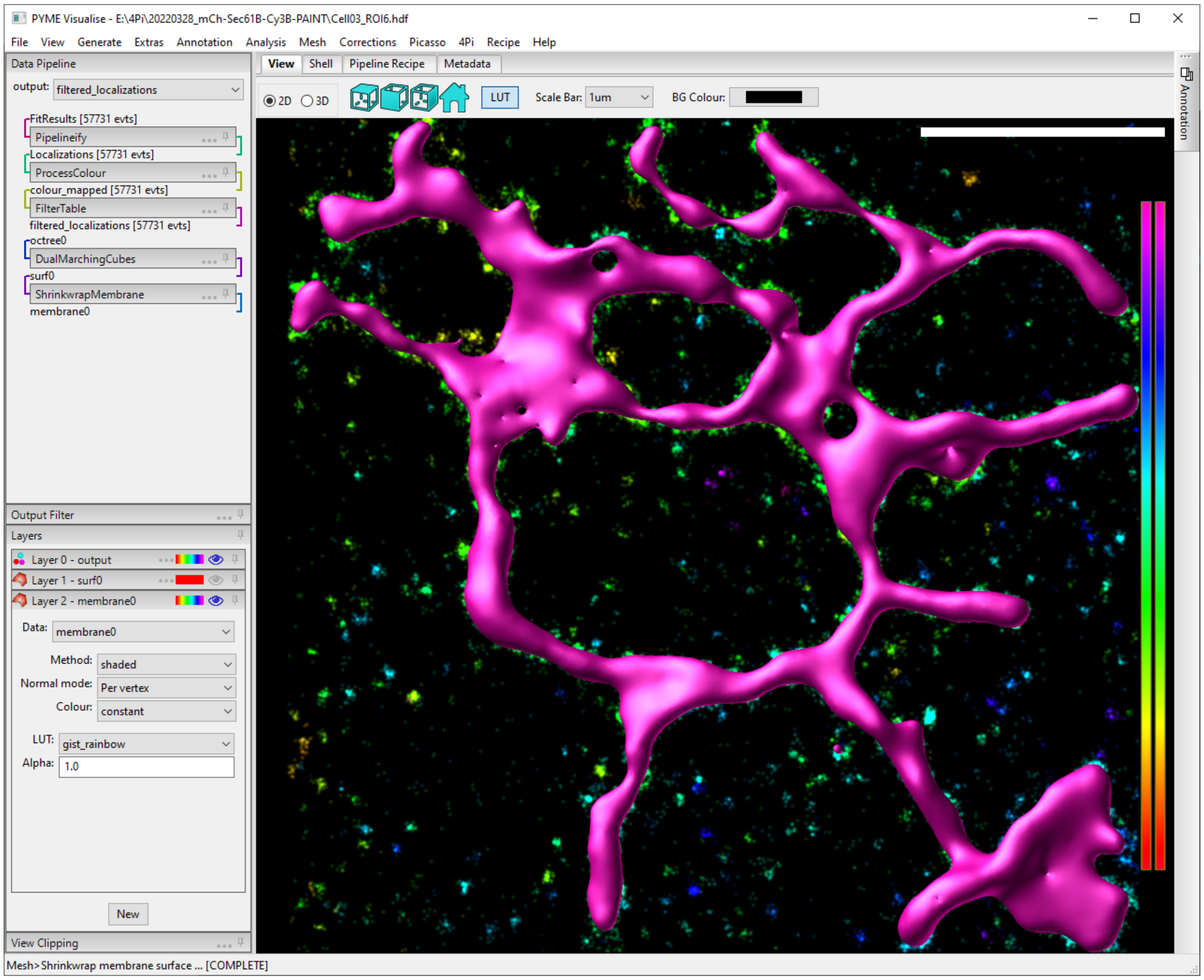
Mesh>Generate Isosurface ... [COMPLETE]

Figure S5: The isosurface generated using the parameters in S4. The isosurface is displayed using the *wireframe* method to make it easy to assess how well it fits the point cloud.



Input:	surf0
Points:	filtered_localizations
Max iters:	29
Curvature weight:	20.0
Remesh frequency:	5
Neck first iter:	0
Punch frequency:	0
Kc:	1.0
Minimum edge length:	10
Smooth curvature:	✓
Truncate at:	1000
Sigma x:	error_x
Sigma y:	error_x
Sigma z:	error_x
Output:	membrane0

Figure S6: Ideal parameters to create a shrink-wrapped surface based on `user-guide-data.hdf`.



PYME Visualise - E:\4Pi\20220328_mCh-Sec61B-Cy3B-PAINT\Cell03_ROI6.hdf

Data Pipeline

output: filtered_localizations

FitResults [57731 evts]
Pipelineify
Localizations [57731 evts]
ProcessColour
colour_mapped [57731 evts]
FilterTable
filtered_localizations [57731 evts]
octree0
DualMarchingCubes
surf0
ShrinkwrapMembrane
membrane0

Scale Bar: 1um
BG Colour:

Layers

Layer 0 - output
Layer 1 - surf0
Layer 2 - membrane0

Data: membrane0
Method: shaded
Normal mode: Per vertex
Colour: constant
LUT: gist_rainbow
Alpha: 1.0

Mesh>Shrinkwrap membrane surface ... [COMPLETE]

Figure S7: The shrink-wrapped surface generated using the parameters in S6.

Parameter	Description	Standard values
input	The data source containing the coarse isosurface.	`surf0`
Points	The data source containing the points to fit.	`filter_localizations`
Max iters	Maximum number of fitting iterations.	10 - 100
Curvature weight	The contribution of curvature (vs. point attraction force) to the fitting procedure. Higher values create smoother surfaces.	10 - 100
Remesh frequency	Remesh the isourface every N iterations. Helps keep the fitting numerically stable. Should be often.	5
Neck first iter	Every neck first iter iterations, check for and remove necks in the mesh.	9 (0 to not use necking)
neck_threshold_low	Vertices with Gaussian curvature below this threshold are necks.	-1.00E+03
neck_threshold_high	Vertices with Gaussian curvature above this threshold are necks.	1.00E+02
Punch frequency	Every punch frequency iterations, check for and add holes in regions of the mesh where there is a continuous empty area in between two "sides" of the mesh.	0
Kc	Lipid stiffness coefficient of membrane in eV (can be looked up in the literature).	0 - 1 (20k_bT)
Minimum edge length	Small length that the edges joining surface vertices can be. Smaller means the surface is more finely sampled. Setting it to 10 is usually sufficient. Setting it to -1 removes any limit.	10
Smooth curvature	Replace the fit curvature of a vertex with the average curvature of it and its neighbors	`True`
Truncate at	Stop after this many iterations no matter what. Useful for visualizing the behavior of a shrink wrap over time	1000
sigma_x	The variable in the points data source containing localization precision in the x-direction. If sigma is only known for one direction, supply it here and it will be assumed for all directions.	
sigma_y	The variable in the points data source containing localization precision in the y-direction.	
sigma_z	The variable in the points data source containing localization precision in the z-direction.	
output	The name of the data source that will contain the fit isosurface.	`membrane0`

Table S2: Descriptions and standard values for all parameters used in the shrink-wrapping algorithm.

In the latter case, try to change the initial isosurface parameters, especially *N points min* and *Threshold density*, to resolve the error when shrink wrapping.

- If attempting to create several separate shrink-wrapped surfaces from multi-color data, make sure you are shrink wrapping to only the points belonging to the desired channel. By default, the algorithm uses `filtered_localizations` which includes all points in the data. To learn how to extract color channels from multi-color data in PYMEVisualize, please read the relevant documentation here.