Surface mesh fitting of 3D SMLM Datasets

Rusty Nicovich

July 22, 2019

1 Introduction

This document describes the algorithm and implementation of a method to generate surface reconstructions for 3D point cloud datasets with noise characteristics representative of single molecule localization microscopy (SMLM) datasets. Code referenced in this document is available in the associated Git repository, located at https://bitbucket.org/nicovich12/smlmmeshfitti

1.1 Motivation

Biological membranes are structures rich in molecular interactions driving cellular- and organism-level changes. Proteins, lipids, and other molecules surrounding the cell membrane are regulated spatially within signaling cascades leading to these larger events. Many membrane-based systems include components associated with certain topographic features of the membrane, be it protrusions, invaginations, or areas of high or low curvature, both positive and negative. This spatial regulation occurs on length scales not possible to observe using conventional light microscopy. The length scales for observation made available by super-resolution microscopy reveals additional details about these spatial regulations. However, current analyses are insufficient to effectively model the underlying structure of the plasma membrane.

Were it to exist, an effective method for analysis in this context would return a surface model of the membrane with resolution on the order of 10 to 100 nm. Current surface reconstruction algorithms designed for 3D scanning or LiDAR applications are not tailored to the noise characteristics of SMLM datasets. Additionally it is currently unclear how the resolution of such a surface would be evaluated.

1.2 Single Molecule Localization Microscopy (SMLM)

SMLM datasets are spatial point clouds arising from serially-detected isolated single molecules collected in a time-lapse dataset. Emitters within the specimen stochastically activate, are detected on a camera for one or more frames, and then bleach or cease to emit. These emitters are fluorophores associated with a biological target (usually a protein). Many tens to millions of emitters will become emissive, be detected, and bleach over the course of an experiment. Each single emitter localized in a single frame is referred to as a 'detection' or 'detection event' and makes a single point in the final dataset.

The increased precision of a SMLM dataset arises from the localization of these single emitters. As they are isolated, diffraction-limited spots, it is possible to fit these emitters to a prior (approximated as a 2D Gaussian) and retrieve a localization of each emitter to 10x higher precision than the width of the underlying width of the emitter's spot on the camera. This localization precision is approximated as the width of the emitter peak divided by the square root of the number of photons collected for that peak. In practice, localization precisions of 20-50 nm are common.

In the case of an emitter that persists for multiple frames or re-activates over multiple cycles, each single frame this emitter is detected yields a new detection event. As a result, a single emitter will give rise to one or more detection events. A series of detection events from a single emitter will typically be observed as an area of local density in the full point cloud dataset, with multiple detection events normally distributed around a centroid. The centroid will be close to the emitter's true location and the width of the local point cloud distribution represented by the localization precision of the points.

The experimental realization of SMLM datasets yields characteristics to consider for further analysis:

1. **Noisy** - Imprecision of point cloud is on order of desired feature size
2. **Sparse** - Emitters (and therefore point cloud) will contain local gaps on order of desired feature size
3. **Spatially inhomogeneous** - Local precision and density will vary across dataset
4. **Unordered** - Points will not be ordered by any value relevant to analysis
5. **Contain significant off-target signal** - Emitters and false positive detections can occur far from desired target object
6. **Best defined by local center of mass** - Analysis should capture local features
7. **Not defined by a priori model for shape** - Target features are amorphous and have no reference for underlying shape

This project strives to generate surface reconstructions from point clouds including these above features to yield insights regarding biological membrane topography.

2 Data Simulation

2.1 SMLM data simulation

SMLM data simulation proceeds by:

1. Defining an underlying, 'ground truth' structure.
2. Scattering emitters across this structure at some pre-defined density.
3. Adding additional 'off-target' emitters randomly through the detection volume at some pre-defined density.
4. Generating detection events around each emitter, with some pre-defined distribution modeling the number of events per emitter and localization precision of each event.
5. Outputting the list of detection events to a file or array.

These steps, represented in Figure 1, proceed for both 2D and 3D simulation. Currently 3D simulation only is supported by the repo, but 2D could be included if desired.

2.2 3D point clouds based on test surfaces

Input objects for 3D SMLM point cloud analysis can be provided in the STL format. This file format represents a surface as a series of locally-flat triangles with associated nodes. This format is commonly used for 3D printers and computer-generated objects and many programs exist for generating and manipulating these files and represented objects. AutoDesk Fusion 360 is a freely-available CAD program I have used for generating test objects.

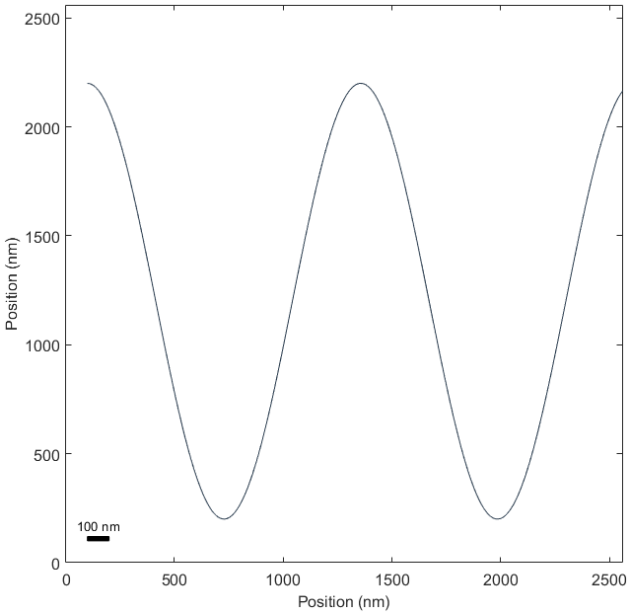
Code exists in the repository to read STL files and then generate SMLM point clouds with these as the ground truth. Function `.\GenerateTestDataFromSTLFile.m` loads a defined STL file (using `.\stlread.m`), scales by user-defined factor, shifts center of mass to defined position, and then generates N points randomly positioned on the loaded surface. These then constitute the 'emitter' locations.

Function `.\pointsToSMLMPointCloud.m` generates detection events around these emitters. The user supplies the (x,y,z) coordinates of all emitter locations, along with the number of emitters, the mean number of detection events per emitter, and the number of detected photons and localization precision distribution to sample for this dataset. The actual number of detection events for a given emitter is chosen as a Poisson-distributed random number with the specified mean. The number of detected photons and localization precision distribution is pulled from an example experimental dataset (loaded using `.\Import1File.m`). Those simulated points whose position falls outside the specified ROI bounds are removed using `.\enforceROIBounds.m`. This action concludes the simulation actions, yielding a simulated 3D SMLM dataset for further processing.

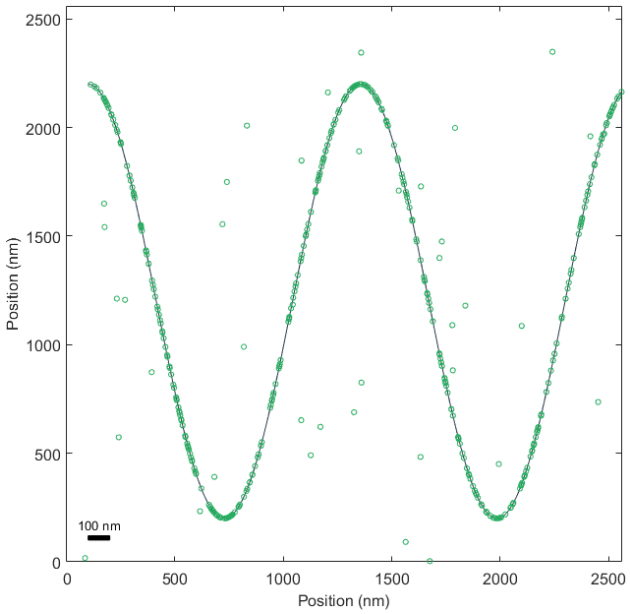
2.3 Experimentally-relevant parameters

Within the input parameters of the simulation code, there are select variables with direct experimental corollaries. These following parameters (and `.\MeshFitting3d.m` variable names) would be most useful in simulating various experimental outcomes:

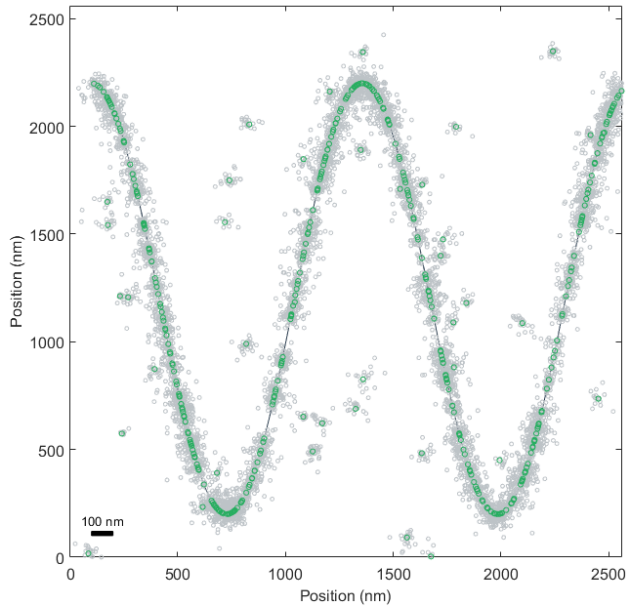
1. Emitter density (`nPoints`) - Currently defined as number of emitters per dataset. Most relevant if converted from emitters per μm^2 of surface in test object. Experimentally this pertains to specificity of label molecule, efficiency of labeling, and acquisition duration.
2. Detections per emitter (`nDetectsPerPoint`) - Mean number of detection events per molecule specified. This value can be modulated experimentally by acquisition buffer chemistry, laser power, and fluorophore selection.
3. Localization precision (implicit in `exampleDataFile`) - Values for these distributions are given *via* an example data set. It would be straightforward to model the necessary distribution as a log-Gaussian with given mean and width, or by manipulating the provided values by some scalar factor. Experimentally the precision is a function of photons collected per detection event, which is a function of the laser power, buffer chemistry, fluorophore, and acquisition camera parameters chosen.
4. Off-target labeling (currently not included in 3D analysis) - The density of emitters or false positive detection events randomly distributed through the simulated ROI. Experimentally a function of specificity of label molecule, buffer chemistry, and acquisition camera parameters chosen.



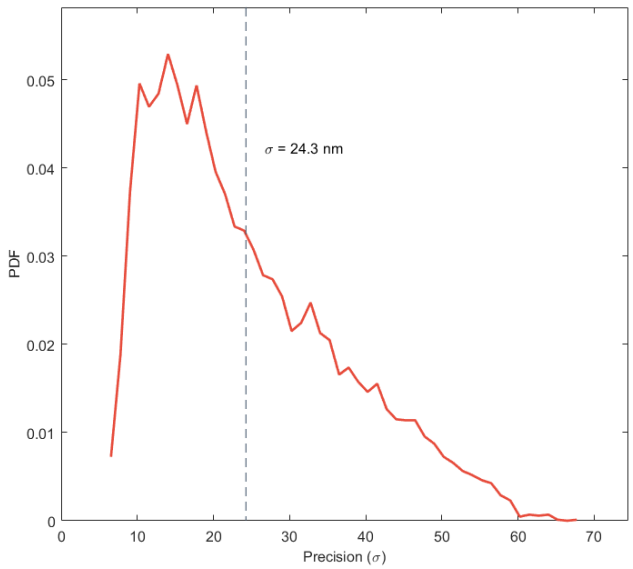
(a) Ground truth shape.



(b) Emitters (green) sparsely scattered around ground truth shape with additional points randomly distributed.



(c) Detection events (gray) localized around each simulated emitter.



(d) Histogram of localization precision of simulated detection events, drawn from a distribution of experimental values, with scaling.

Figure 1: Steps for simulating a SMLM datset.

The inter-dependence of these parameters on the same experimental conditions makes the joint analysis of these values interesting. For example, a given label could have high specificity, high localization precision (low values), but low emitter density and number of detections per emitter. Another label could have lower specificity and precision, but higher emitter density and detections per emitter. Any label could trade localization precision for additional detections per emitter (or the opposite - fewer detections per emitter but each at a higher precision) by tuning buffer chemistry and acquisition parameters.

It is not clear which, if any, of these options is most effective to yield the highest-resolution surface as output of the next step. One goal of this study is to determine which parameters are most essential to yield the best dataset for subsequent analysis.

3 Data Processing

Data processing requires as input a 3D point cloud and a few parameters highlighted below. The output is a mesh representation of the reconstructed surface.

3.1 Algorithm description

3.1.1 2D point cloud fitting

The reconstruction algorithm follows the scheme outlined in [1]. Starting with an unsorted point cloud, a minimum spanning tree (*via* the Delaunay triangulation) is calculated. This acyclic graph connects all points while minimizing the total distance traversed by all edges. From this, the shortest path along the minimum spanning tree is calculated between the two most distant points in within this graph. The ordered position of these points is a trajectory through the dataset roughly approximating a central path through the point cloud. This path is smoothed and sub-sampled to more closely approximate a smooth path through the enter of the point cloud. Note : this smoothing step as implemented is somewhat crude and could possibly be improved for higher-fidelity reconstructions.

In the implementation provided, the function `meshfigAndSmooth2d.m` calculates each 2D point cloud reconstruction using the steps above. This is called within the function `fitPointSetLoop.m`, which loops through all intersecting planes along a given axis.

A special case exists where a complex object is intersected along a plane at a location that results in two or more distinct 2D point clouds. The step(s) between these point clouds along the smoothed path calculated above is significantly longer than those in the remainder of the step size distribution. These exceptionally long linkages should not persist in the analysis and are detected by the `rosinThreshold.m` function. This function returns the threshold value for a 'long' step in a set of smoothed paths. The `splitByLargeDistance.m` function then separates the curve containing these distant point clouds into individual reconstruction curves.

3.1.2 3D $\rightarrow$ 2D planes

The above portion of the algorithm can be performed on N dimensional data, but scales poorly with a large number of points and edges. The Prim algorithm used in MATLAB has scaling $O(E * log(N))$, where E is the number of edges and N the number of points in the minimum span tree. SMLM datasets can have millions of points in the full 3D dataset, so downsampling or parcellating the calculations is prudent.

Here this is accomplished by approximating the 3D surface as a set of 2D lines of intersection between the point cloud and perpendicular sampling planes. In a single instance, a set of points close to the plane intersecting the point cloud is collapsed into a 2D plane. This 2D point cloud is fit using the algorithm above. The resulting reconstructed and smoothed curve, with points in 3D space representing 2D curve on intersecting plane, is returned. This repeats at a user-specified spacing along this axis, followed by the same calculation but along the perpendicular axis. In practice this is repeated along two more orthogonal axes, but at $\frac{\pi}{4}$ radians rotated from the first two axes. The set of all reconstructed and smoothed curves is carried forward.

3.1.3 2D fit planes $\rightarrow$ 3D volume

From here, the surface could be passed to a surface reconstruction algorithm. However, sampling along a third axis, orthogonal to all intersecting planes, has not occurred. To avoid losing information due to not sampling in this direction, a further calculation is performed.

Though originally motivated by desiring a somewhat isotropically-sampled point cloud for the surface reconstruction, it is not clear how necessary this step truly is. It may be better to remove the Z resampling step. This would simplify calculations while retaining the emitter density information still maintained in the intersecting plane analysis. However, the point cloud entering the surface reconstruction step may be more noisy. One aspect of this project should evaluate these two options to see if continuing with the resampling step here is required.

The assembled set of all reconstructed 2D intersection lines (resampled or no) is carried forward for surface reconstruction.

3.1.4 Surface reconstruction

Surface reconstruction on the smoothed point cloud is accomplished using Poisson reconstruction [2] in MeshLab. This is called through a MATLAB script invoking the MeshLab command line interface.

The surface reconstruction algorithm returns a set of points and connectivity constituting a reconstructed surface. This reconstructed surface is the final output of the calculation and should best-approximate the ground truth object sampled through the SMLM point cloud.

4 Results

5 References

References

[1] In-Kwon Lee. Curve reconstruction from unorganized points. *Computer aided geometric design*, 17(2):161–177, 2000.

[2] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006.