Hidden Markov Modeling for Maximum Probability Neuron Reconstruction

Thomas Athey (tathey1@jhu.edu)
Johns Hopkins University https://orcid.org/0000-0002-8987-3486
Daniel Tward
University of California Los Angeles https://orcid.org/0000-0002-4607-6807
Ulrich Mueller
Johns Hopkins University
Joshua Vogelstein
Johns Hopkins University https://orcid.org/0000-0003-2487-6237
Michael Miller
Johns Hopkins University

Article

Keywords: Hidden Markov, Neuron Reconstruction, Python package brainlit

Posted Date: October 5th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-934665/v1

Version of Record: A version of this preprint was published at Communications Biology on April 25th, 2022. See the published version at https://doi.org/10.1038/s42003-022-03320-0.

HIDDEN MARKOV MODELING FOR MAXIMUM PROBABILITY NEURON RECONSTRUCTION

THOMAS L. ATHEY

Center for Imaging Science & Kavli Neuroscience Discovery Institute
& Department of Biomedical Engineering
The Johns Hopkins University

DANIEL J. TWARD

Department of Computational Medicine & Department of Neurology
University of California at Los Angeles

ULRICH MUELLER

Department of Neuroscience
The Johns Hopkins University

JOSHUA T. VOGELSTEIN

Center for Imaging Science & Kavli Neuroscience Discovery Institute
& Department of Biomedical Engineering
The Johns Hopkins University
301 Clark Hall
Baltimore, MD 21218

MICHAEL I. MILLER

Center for Imaging Science & Kavli Neuroscience Discovery Institute
& Department of Biomedical Engineering
The Johns Hopkins University
301 Clark Hall
Baltimore, MD 21218

E-mail addresses: tathey1@jhu.edu, DTward@mednet.ucla.edu, umuelle3@jhmi.edu, jovo@jhu.edu, mim@cis.jhu.edu.

Date: September 27, 2021.

ABSTRACT. Recent advances in brain clearing and imaging have made it possible to image entire mammalian brains at sub-micron resolution. These images offer the potential to assemble brain-wide atlases of neuron morphology, but manual neuron reconstruction remains a bottleneck. Several automatic reconstruction algorithms exist, but most focus on single neuron images. In this paper, we present a probabilistic reconstruction method, ViterBrain, which combines a hidden Markov state process that encodes neuron geometry with a random field appearance model of neuron flourescence. Our method utilizes dynamic programming to compute the global maximizers of what we call the "most probable" neuron path. Our most probable estimation method models the task of reconstructing neuronal processes in the presence of other neurons, and thus is applicable in images with several neurons. Our method operates on image segmentations in order to leverage cutting edge computer vision technology. We applied our algorithm to imperfect image segmentations where false negatives severed neuronal processes, and showed that it can follow axons in the presence of noise or nearby neurons. Additionally, it creates a framework where users can intervene to, for example, fit start and endpoints. The code used in this work is available in our open-source Python package `brainlit`.

1. INTRODUCTION

Neuron morphology has been a central topic in neuroscience for over a century, as it is the substrate for neural connectivity, and serves as a useful basis for neuron classification. Technological advances in brain clearing and imaging have allowed scientists to probe neurons that extend throughout the brain, and branch hundreds of times [Winnubst et al., 2019]. It is becoming feasible to assemble a brainwide atlas of cell types in the mammalian brain which would serve as a foundation to understanding how the brain operates as an integrated circuit, or how it fails in neurological disease. One of the main bottlenecks in assembling such an atlas is the manual labor involved in neuron reconstruction.

In an effort to accelerate reconstruction, many automated reconstruction algorithms have been proposed, especially over the last decade. In 2010, the DIADEM project brought multiple institutions together to consolidate existing algorithms, and stimulate further progress by generating open access image datasets, and organizing a contest for reconstruction algorithms [Peng et al., 2015]. Several years later, the BigNeuron project continued the legacy of DIADEM, this time establishing a common software platform, Vaa3D, on which many of the state of the art algorithms were implemented [Peng et al., 2010b]. Acciai et al. [2016] offers a review of notable reconstruction algorithms up to, and through, the BigNeuron project.

Previous approaches to automated neuron reconstruction have used shortest path/geodesic computation [Peng et al., 2010a, Wang et al., 2011, Turetken et al., 2013], minimum spanning trees [Yang et al., 2019], Bayesian estimation [Radojević and Meijering, 2017], tracking [Choromanska et al., 2012, Dai et al., 2019], and deep learning [Friedmann et al., 2020, Zhou et al., 2018, Li et al., 2017]. Methods have also been developed to enhance, or extend existing reconstruction algorithms [Wang et al., 2019, 2017, Peng et al., 2017, Chen et al., 2015]. Also, some works focus on the subproblem of resolving different neuronal processes that pass by closely to each other [Xiao and Peng, 2013, Li et al., 2019, Quan et al., 2016].

Both DIADEM and BigNeuron initiatives focused on the task of single neuron reconstruction so most associated algorithms fail when applied to images with several neurons. However, robust reconstruction in multiple neuron images is essential in order to assemble brainwide atlases of neuron morphology.

We propose a probabilistic model-based algorithm, ViterBrain, that operates on imperfect image segmentations to efficiently reconstruct neuronal processes. Our estimation method does not assume that the image outside the reconstruction is background, and thus allows for the existence of other neurons. Our approach draws upon two major subfields in Computer Vision, appearance modeling and hidden Markov-models, and

generates globally optimal solutions using dynamic programming. The states of our model are locally connected segments. We score the state transitions using appearance models such as exhibited by Kass and Cohen's early works [Kass et al., 1988, Cohen, 1991] on active shape modeling and their subsequent application by Wang et al. [Wang et al., 2011] for neuron reconstruction. For our own approach we exploit foreground-background models of image intensity for the data likelihood term in the hidden-Markov structure. We quantify the image data using KL-divergence and intensity autocorrelation in order to validate our model assumptions.

Our probabilistic models are hidden Markov random fields, but we reduce the computational structure to a hidden Markov model (HMM) since the latent axonal structures have an absolute ordering. Hidden Markov modeling (HMM) involves two sequences of variables, one is observed and one is hidden. A popular application of HMM's is in speech recognition where the observed sequence is an audio signal, and the hidden variables a sequence of words [Rabiner and Juang, 1986]. In our setting, the observed data is the image, and the hidden data is the contour representation of the axon or dendrite's path.

The key advantages to using HMMs in this context are, first, that neuronal geometry can be explicitly encoded in the state transition distribution. We utilize the Frenet representation of curvature and torsion in our transition distribution, which we have studied previously in Athey et al. [2021], Khaneja et al. [1998]. Secondly, globally optimal estimates can be computed efficiently using dynamic programming in HMMs [Forney, 1973, Khaneja et al., 1998]. The well-known Viterbi algorithm computes the MAP estimate of the hidden sequence in an HMM. Our approach, inspired by the Viterbi algorithm, also computes globally optimal estimates. Thus, our optimization method is not susceptible to local optima that exist in filtering methods, or gradient methods in active shape modeling.

In this work, we apply our hidden Markov modeling framework to the output of low-level image segmentation models. Convolutional neural networks have shown impressive results in image segmentation [Ronneberger et al., 2015], but it only takes a few false negatives to sever neuronal processes that are often as thin as one micron (Figure 1). Our method strings together the locally connected components of the binary image masks into a global neuron reconstruction with a global ordering. Thus, our method is modular enough to leverage state of the art methods in machine learning for image segmentation.

We apply our method to data from the MouseLight project at Janelia Research Campus [Winnubst et al., 2019], and focus on the endpoint control problem in a single neuronal process i.e. start and end points are fixed. We introduce the use of Frechet distance to quantify the precision of reconstructions and show that our method has comparable precision to state of the art, when the algorithms are successful.

2. Results

2.1. **Overview of ViterBrain.** Our reconstruction method takes in an image, and associated neuron mask produced by some image segmentation model. The algorithm starts by processing the mask into neuron fragments and estimating fragment endpoints and orientation to generate the state space. Next, both the prior and likelihood terms of the transition probabilities are computed to construct a directed graph reminiscent of the the trellis from the HMM work in Forney [1973]. Lastly, the shortest path between states is computed with dynamic programming. We have proven that the shortest path is the "most probable" estimate state sequence formed by a neuronal process. Our most probable estimation procedure is different from maximum likelihood or maximum a posteriori estimation because it only considers the observed data located at the given state sequence. Thus, it does not assume that observed data outside of the state sequence is background, allowing for the possibility of other neurons. An overview of our algorithm shown in Figure 2 and details are given in the Methods section.

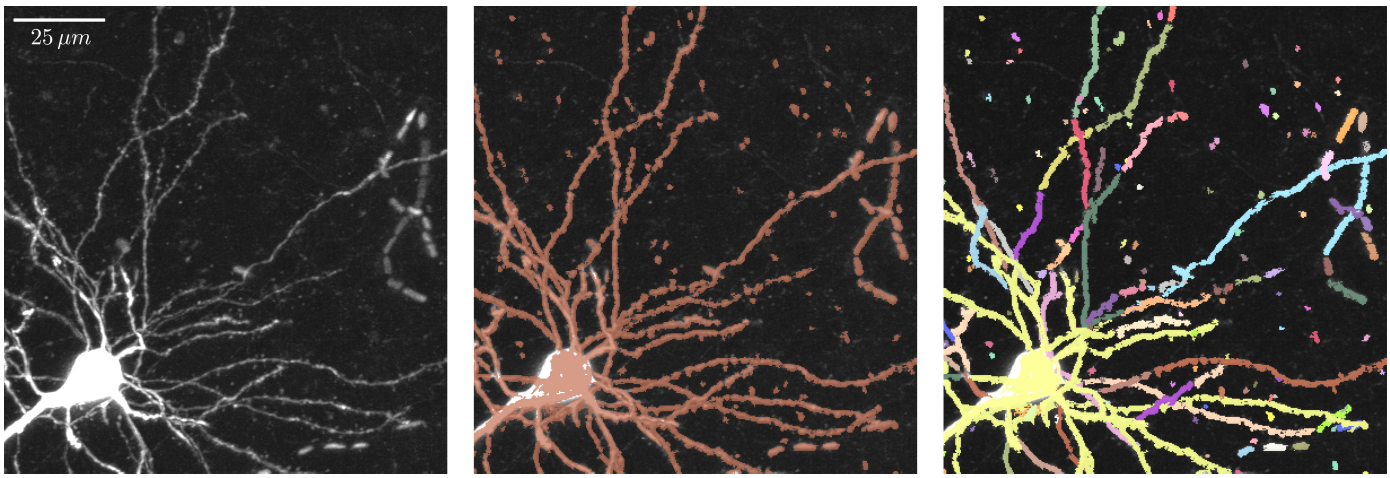

FIGURE 1. Left panel shows an image subvolume from the MouseLight project containing a single neuron. The middle panel shows the same image overlaid with a binary image mask in brown. This mask was generated by a neural network and illustrates the typical output of an image segmentation model. The right panels shows the same binary image mask, with a different color for each connected component. The colors show that the neuron has been severed into several pieces. All panels are maximum intensity projections (MIPs).

2.2. **Modeling Image Intensity.** Figure 3a shows kernel density estimates of the foreground and background image intensity distributions. The distributions vary greatly between the three image subvolumes, implying that modeling the image process as homogeneous throughout the whole brain would be inappropriate. Additionally, many of the distributions, especially for the manual labels, do not appear to be either Gaussian or Poisson. Indeed, Kolmogorov-Smirnov tests rejected the null hypothesis for both Gaussian and Poisson goodness of fit in all cases, with all p-values below 10^{-4}.

For that reason, we do not assume that the image is generated by a Poisson process or a Gaussian distribution. However, the K-L divergence remains the important test statistic describing image quality and performance of the fragment generation through the appearance modeling.

We also investigated the covariance between nearby voxels in the image background (Figure 3b) and found that covariance decayed rapidly with increasing distance, lending support to our conditionally independent spatial increment assumption.

2.3. **Maximally Probable Axon Reconstructions.** Figure 4 demonstrates the reconstruction method on both a satellite image of part of the Great Wall of China, and part of an axon. Different image segmentation models were used to generate the fragments in the two cases, but the process of joining fragments into a reconstruction was the same. The algorithm successfully reconstructs the one dimensional structure in both cases.

Since the state transition probabilities are modeled with a Boltzmann distribution, reconstructions are driven by relative energies of different transitions. Thus, our method can still be successful in the presence of luminance or fragment dropout, as long as the neuronal process is relatively isolated (Figure 5).

Figure 6a shows various examples of maximally probable reconstructions. The algorithm was run with the same hyperparameters in all cases: $\alpha_d = 10$ and $\alpha_\kappa = 1000$.

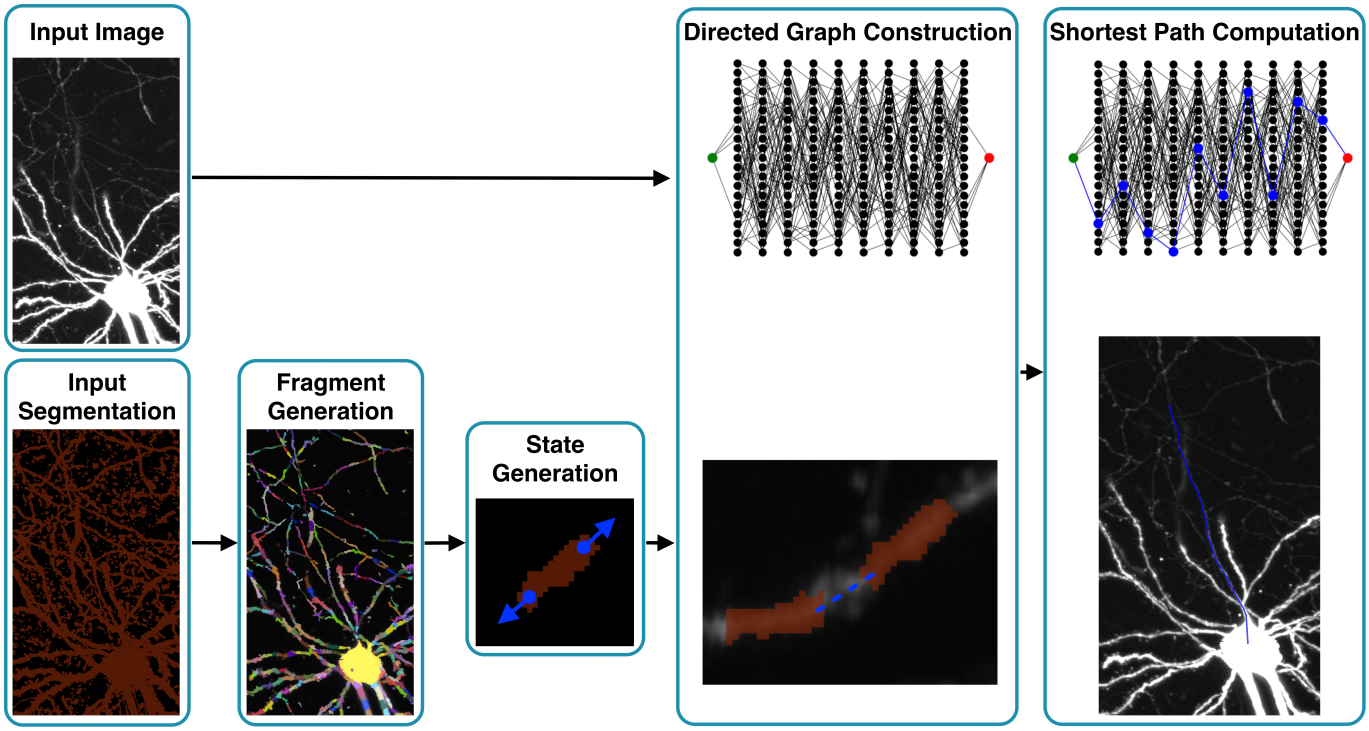

FIGURE 2. Summary of our algorithm. The algorithm takes in an image and a binary mask that might have severed, or fused neuronal processes. First, the mask is processed into a set of fragments. For each fragment, the endpoints and endpoint orientations are estimated and added to the state space. Next, transition probabilities are computed from both the image and state data to generate a directed graph reminiscent of the trellis graph in classic hidden Markov modeling. Finally, a shortest path algorithm is applied to compute the maximally probable state sequence connecting the start to the end state.

In some cases, reconstruction accuracy is sensitive to hyperparameter values (Figure 6b). Higher values of α_d penalize transitions between fragment states with large gaps; higher values of α_κ penalize state transitions with sharp angles as measured by their discrete curvature. It is important to note that the transition distributions depend on the exact values of α_d and α_κ, not just, for example, the ratio between them. We found $\alpha_d = 10$ and $\alpha_\kappa = 1000$ to be effective for the reconstructions shown throughout the paper, but these values should be adjusted according to the quality of fragments, and the geometry of the neurons being reconstructed.

2.4. **Signal Dropout causes Censored Fragment States.** Though our method is robust to fragment dropout in certain contexts, it is less robust when there is dropout near a parallel neuronal process (Figure 7). Since the reconstruction is ultimately a sequence of fragments, missing fragments will not allow the algorithm to fully follow the trajectory of the neuronal process. Instead, the algorithm will follow a different sequence of fragments, often on a nearby neuronal process.

The most obvious source of fragment dropout is low image luminance $I(y)$, leading to false negatives in the initial image segmentation. However the reason for fragment dropout depends on the underlying segmentation model. Empirically, we found that the reconstruction algorithm fails when when the fragment generation process neglects portions that are greater than $\sim 10\mu m$ in length.

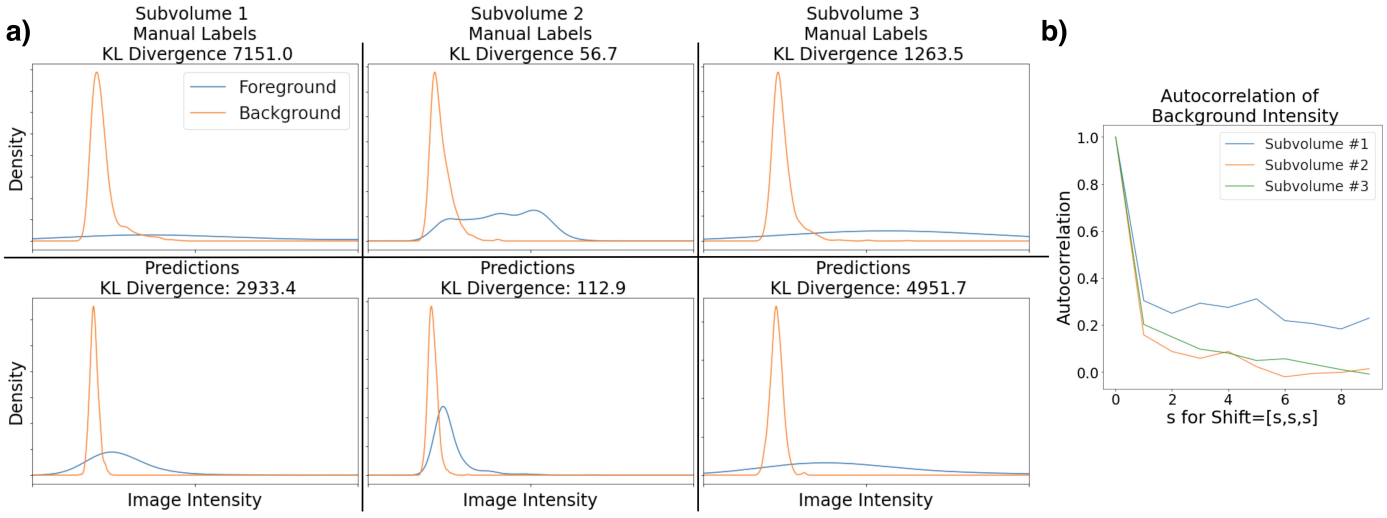

FIGURE 3. a) Kernel density estimates of foreground and background intensity distributions of three different subvolumes (three columns) of one of the MouseLight images. A subset of the voxels in each subvolume was manually labeled (top row), then used to train an Ilastik model to classify the remaining voxels (bottom row). In each panel, we computed the KL Divergence between the foreground and background kernel density estimates. These panels show that the intensity distributions vary across different regions of the image. b) Autocorrelation of image intensities between background voxels at varying displacements from each other. Different displacements (all in the direction of $[1, 1, 1]$) are plotted along the horizontal axis. In each case, 1000 voxel pairs were sampled. The three lines represent three different image subvolumes, and in all three cases, the covariance drops to zero rapidly, suggesting that image intensity between nearby background voxels are not correlated.

2.5. **Comparison to State of the Art.** We examined the accuracy of ViterBrain compared to existing state of the art reconstruction algorithms. Out of the 26 reconstruction algorithms available in Vaa3D version 3.2 [Peng et al., 2010b], we chose the two algorithms that have accompanying publications, most closely resemble our approach, and performed reasonably well on the MouseLight data. The first method is APP2 which starts with an oversegmentation of the neuron using shortest path algorithm, then prunes spurious connections [Xiao and Peng, 2013]. The second is Snake which is based on active contour modeling [Wang et al., 2011]. Both algorithms were run with their default settings and hyperparameters.

Shown in Table 8 are spatial distances and Frechet distances metrics between automatic reconstructions and manual ground truth for the samples shown in Figure 6a. Spatial distance has been defined in neuron reconstruction literature, and Frechet distance is a mathematical metric that is invariant to reparameterization (see section 4.5 for derivation of Frechet distance). The spatial distance between ViterBrain and manual reconstructions are all between 1 and 2 microns which is roughly the resolution of the image. Successful reconstructions by the other algorithms are also in this range.

Frechet distances are larger, since they indicate maximal distance, not average distance, between curves. We observe that the ~ 5 micron deviations occur most often near the axon hillock where the axon broadens to merge with the soma.

Metrics from unsuccessful reconstructions are not reported. Snake was observed to dramatically oversegment images when it failed, generating dense curves in background space. APP2 often had the opposite problem when it failed, stopping reconstruction prematurely along some neuronal processes.

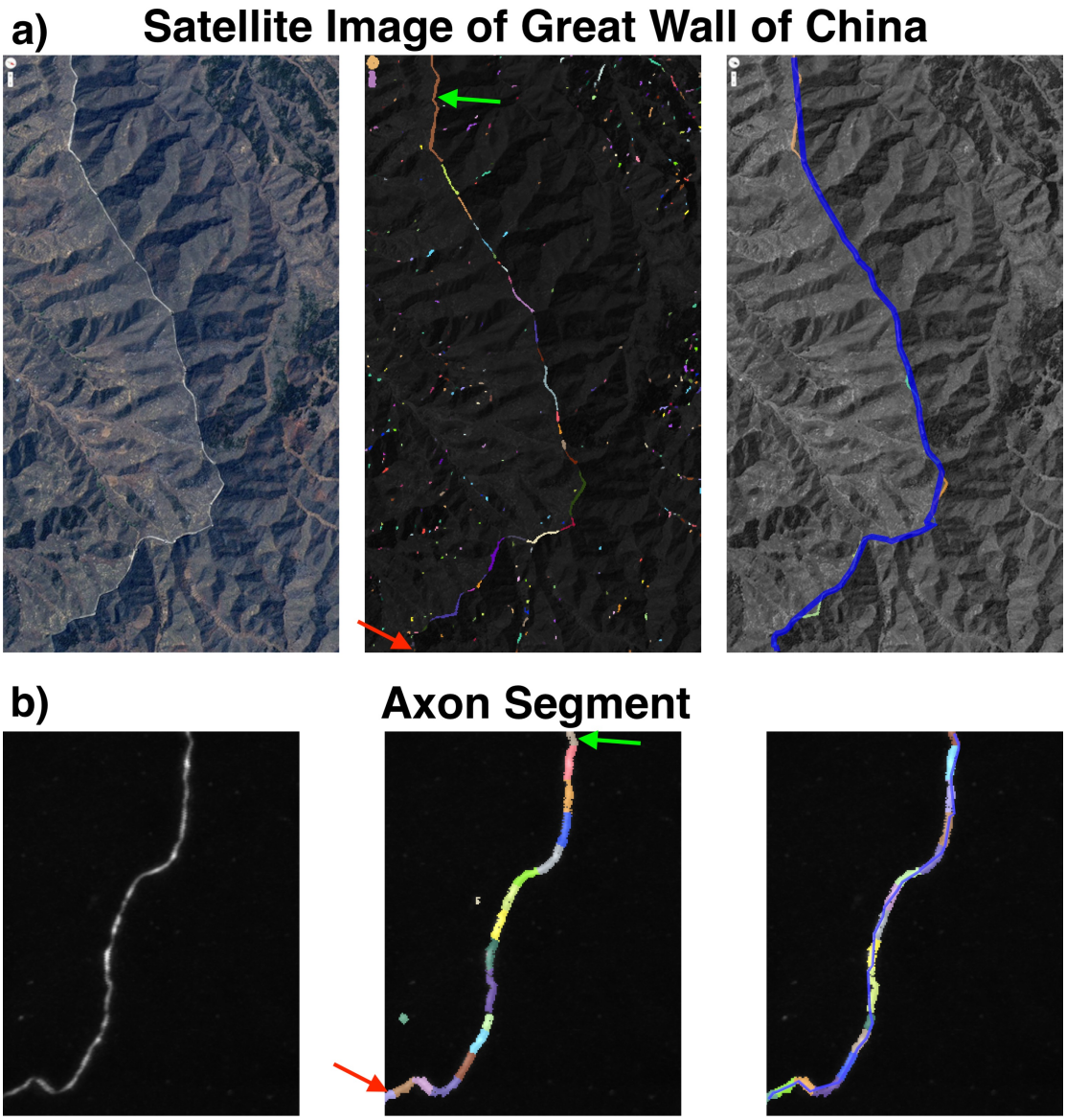

FIGURE 4. Demonstration of maximally probable reconstruction on isolated linear structures from a) a satellite image of part of the Great Wall of China and b) a neuronal process from the MouseLight dataset (MIP). Left panels show the original images. Middle panels shows the space of fragments, $\mathcal{F}$, pictured in color. The green and red arrows indicate the start and end states of the reconstruction task, respectively. The right panels show the most probable fragment sequences, where the fragments are colored and overlaid with a blue line connecting the endpoints of the fragments.

3. DISCUSSION

This paper presents a hidden Markov model based on axon fragments generated from an appearance modeling. Our method involves converting the connected components of an image mask into a set of fragments. These fragments are assembled based on the associated appearance model score of the observed image and

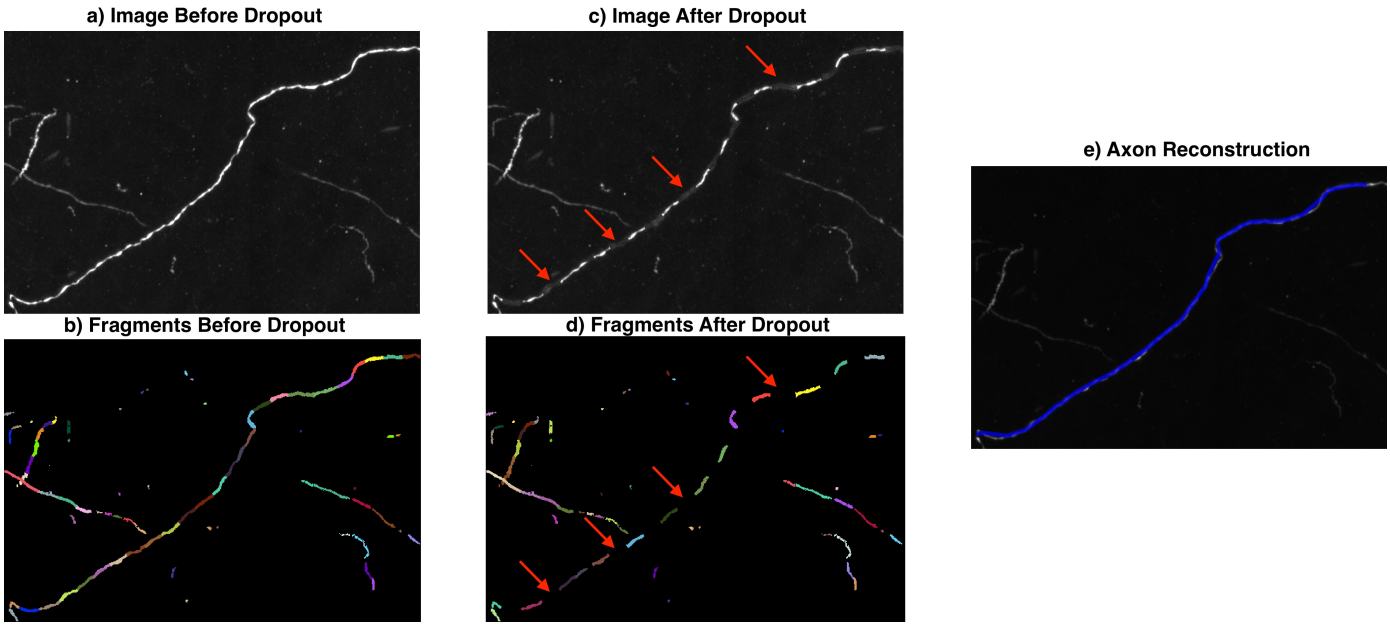

FIGURE 5. ViterBrain is robust to image intensity and fragment dropout when axons are relatively isolated. In this image, we censored the image intensity and the fragment space periodically along an axon path (red arrows in panels c and d show censored sections). Nonetheless, our algorithm was able to jump over the censored regions to reconstruct this axon (panel e). All images are MIPs

the discrete numerical curvature of adjacent fragments selected during assembly. In the Methods, we derive the Bayes posterior for maximum a-posteriori distribution estimation of the hidden state sequence encoding the axon reconstruction. We also show that applying the $O(n|\mathcal{S}|^2)$ Viterbi algorithm is not possible for the MAP estimator since the state grows for the MAP solution due to cycles in the graph formed by having the axon return to the same location in the image. We describe a modified procedure for defining the path probabilities making it feasible to efficiently calculate the globally optimal neuron path dealing efficiently with cycles. The solution for efficiently generating the globally optimum in probability path implies that the local minimum associated to gradient and active appearance models solutions is resolved in this setting.

We apply the algorithm in the case of fixed start and end points to two-photon image data of neurons in mice. In a dataset of 10 partial axon reconstructions, our algorithm is competitive with existing state of the art algorithms (Table 8). We observed that the most common failure mode in this dataset was when there are extended ($>\sim$ 10 micron) stretches of the putative axonal path where there is significant loss of luminance signals resulting in fragment generation being highly censored. This implies that the maximally probable HMM procedure is only effective if paired with effective voxel level modelling tools. The algorithm can also fail in areas densely populated with neuronal processes. We demonstrate that proper selection of the hyperparameters to reflect the density of the fragments and the geometry of the underlying neurons can resolve these issues. Our algorithm is specifically adapted for reconstructing axons in projection neurons in datasets such as MouseLight, in two ways. First, the high image quality as indicated by large KL-divergence values (Figure 3a), makes it straightforward to build an effective foreground-background classifier which is an essential part of our state generation for the HMM structure. Secondly, our algorithm encodes the geometric

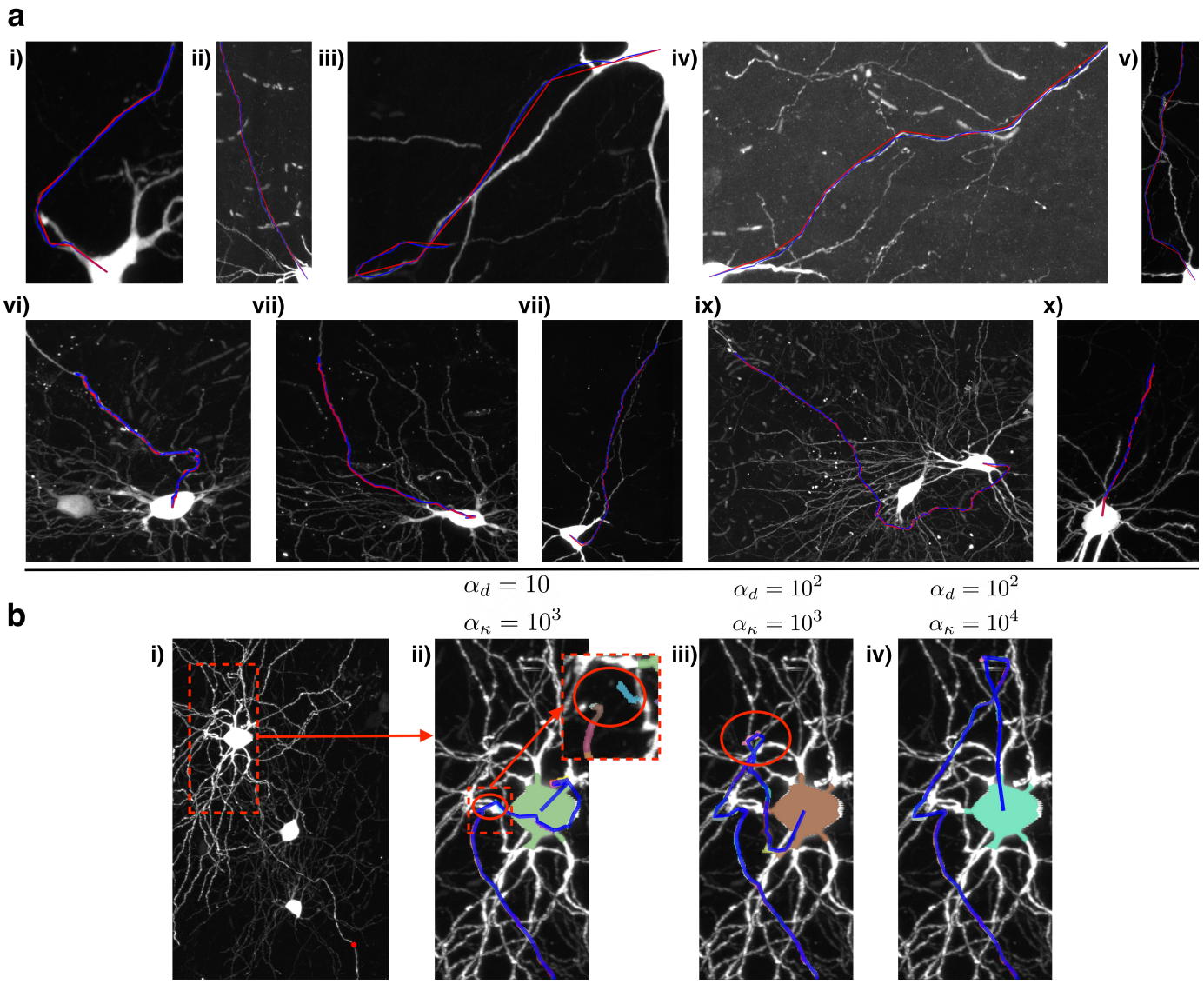

FIGURE 6. a) Successful axon reconstructions; the ViterBrain reconstructions are shown by the blue line; the manual reconstructions are shown by the red line. The algorithm was run with the same hyperparameters in each case: $\alpha_d = 10$ and $\alpha_\kappa = 1000$. b) Different hyperparameter values lead to different results. Panel i) shows the neuron of interest. Panels ii-iv) are close-up views of reconstructions with different hyperparameter values that weigh transition distance (α_d) and transition curvature (α_κ). The red circle in Panel ii) indicates where the reconstruction deviated from the true path by jumping $\sim 10\ \mu m$ to connect the gray fragment to the light blue fragment. Panel iii) shows how a higher α_d value avoids the jump in panel ii), but takes a sharp turn to deviate from the true path (red circle). Finally, in panel iv), the reconstruction avoids both the jump from panel ii) and the sharp turn from panel iii) and follows the true path of the axon back to the cell body. All images are MIPs.

properties of axons such as curvature, which allows our solutions to adapt to the occasional sharp turns in projection axons.

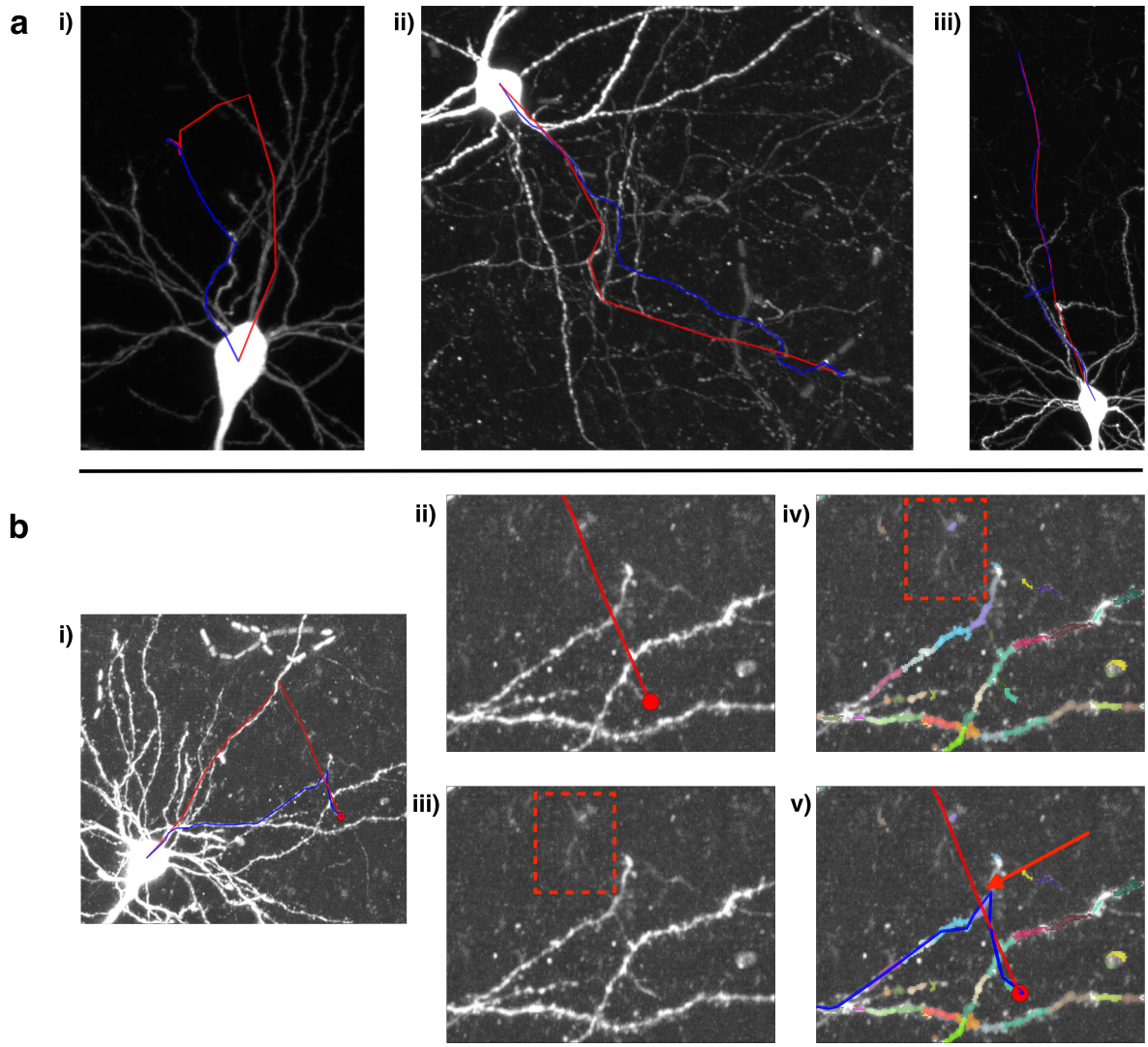

FIGURE 7. The most common failure mode is an inadequate fragment space due to signal dropout. a) shows maximally probable reconstructions (blue lines) that deviate from the manual reconstructions (red lines). b) shows an example with an inadequate fragment space in more detail. The full image subvolume is shown in panel i) with the blue line depicting the most probable reconstruction and the red line depicting the manual reconstruction. The close up views in panels ii,iii) show that the image signal under the manual reconstruction is weak (red dotted box). Panel iv) shows the fragments in different colors. Indeed, there are only a few fragments generated in the region delineated by the red dotted box, where the manual reconstruction is. Panel v) shows the manual reconstruction in red and the algorithm's reconstruction in blue. The algorithm starts on the right path, then veers away from the true path at the red arrow.

It is important to note that the images chosen for Figure 6/Table 8 were not chosen randomly and thus they should not be construed to indicate success rates of the different algorithms. The examples were chosen to illustrate the class of images where ViterBrain is successful, and to show that the precision of ViterBrain is

Discrepancies Between Reconstruction Algorithms and Manual Traces										
Example No. (Figure 6a)	i	ii	iii	iv	v	vi	vii	viii	ix	x
Spatial Distance (μm)										
ViterBrain	1.5	**1.3**	**1.3**	**1.5**	**1.8**	**1.2**	**1.4**	**1.3**	**1.2**	1.6
APP2	**0.9**	✗	✗	✗	✗	1.9	1.6	✗	✗	1.7
Snake	✗	✗	✗	✗	✗	✗	1.8	2.4	✗	**1.0**
Frechet Distance (μm)										
ViterBrain	5.9	**4.1**	**3.5**	**4.5**	**8.1**	**3.1**	**4.1**	**5.0**	**4.1**	5.2
APP2	**2.2**	✗	✗	✗	✗	5.7	7.4	✗	✗	**3.0**
Snake	✗	✗	✗	✗	✗	✗	23.2	27.7	✗	**3.0**

FIGURE 8. Spatial distance and Frechet distance metrics comparing manual axon reconstructions to successful ViterBrain, APP2, and Snake reconstructions. Spatial distance indicates the average distance between reconstruction points, and frechet distance indicates the maximal distance between reconstruction points. All samples calculated are from Figure 6. The lowest error metrics are bolded.

comparable to that of state of the art. We do not estimate the success rates of the different algorithms because they strongly depend on the quality of the images at hand, and the hyperparameter values.

We propose that this method can be used in semi-automatic neuron reconstruction where a tracer could click on two different fragments along a neuronal process, and this algorithm would fill in the gap. The biggest hurdle in extending this algorithm to fully-automated setting would be designing the distributions in the model so they accurately model the underlying neuron generation process. Our future work will be focused on this endeavor, ensuring that the state transition distribution and image emission distribution lead to robust reconstruction of neuronal processes.

The code used in this work is available in our open-source Python package `brainlit`: http://brainlit.neurodata.io/, and a tutorial on how to use the code is located at: http://brainlit.neurodata.io/link_stubs/hmm_reconstruction_readme_link.html.

4. METHODS

4.1. The Bayesian Appearance Imaging Model.

Our Bayes model is comprised of a prior which models the axons as geometric objects and a likelihood which models the image formation process.

We model the axons as simply connected curves in $\mathbb{R}^3$ written as a function of arclength

$$c(\ell), \ell \geqslant 0 \, , c(\ell) \in \mathbb{R}^3 \, .$$

We denote the entire axon curve in space as $c(\cdot) \coloneqq \{c(\ell), \ell \geqslant 0\}$. To interface the geometric object with the imaging volume we represent the underlying curve $c(\cdot)$ as a delta-dirac impulse train in space. We view the imaging process as the convolution of the delta-dirac impulse train with the point-spread kernel of the imaging platform. We take the point-spread function of the system to be roughly one micron in diameter, implying that the axons are well resolved. The fluorescence process given the axon contour is taken as a relatively narrow path through the imaging domain ($\sim 1 \ \mu m$ diameter) with relatively uniform luminance.

We take the the image to be defined over the voxel lattice $D = \cup_{i \in \mathcal{Z}^{m^3}} \Delta y_i \subset \mathbb{R}^3$ with centers $y_i \in \Delta y_i$. We model the image as a random field I_{y_i}, $\Delta y_i \in D$ which is an inhomogeneous Poisson process with mean field $E\{I_{y_i}\} = \lambda_{y_i}$ [Snyder and Miller, 2012]. The Poisson assumption implies the random field has conditionally independent spatial increments across the image, implying the marginal probability at any fixed location

encodes the joint probability for the fragments. This is consistent with the covariance plot in Figure 3b. We adopt a foreground-background model for the images giving the marginal probability of a voxel intensity as:

$$p(I_y) = \alpha_1(I_y), y \in \text{foreground} \tag{1a}$$

$$p(I_y) = \alpha_0(I_y), y \in \text{background} \tag{1b}$$

We denote the image random field associated to any subset of sites $Y \subset D$, as

$$I_Y = \{I_{y_i} : \Delta y_i \in Y\} .$$

Modelling the image as a random field with independent increments, the appearance probability conditioned on the set being in foreground or background is given by the product of probabilities of (1a):

$$p(I_Y) = \prod_{y \in Y} \alpha_1(I_y),\ Y \subset \text{foreground} , \tag{2a}$$

$$p(I_Y) = \prod_{y \in Y} \alpha_0(I_y),\ Y \subset \text{background} ; \tag{2b}$$

the shorthand notations for the joint probability of the set in foreground or background

$$\alpha_k(Y) := \prod_{y \in Y} \alpha_k(I_y),\ k = 0, 1 . \tag{2c}$$

There can be many simple transformations between the original luminance measurement and the pre-processed data including scaling and shifting which affects the Poisson probability. To accomodate arbitrary scaling and shift we estimate the foreground-background intensity distributions, including all of its moments, from the data itself. We estimate $\alpha_0(\cdot)$, $\alpha_1(\cdot)$ by labeling a subset of the data as foreground/background then fitting Gaussian kernel density estimates to the labeled data (see Figure 3a). This allows us to empirically fit the data while maintaining the independent increments property of inhomogeneous spatial Poisson process, which is consistent with the covariance curve depicted in Figure 3b. This is the key property that allows us to factor joint probabilities into products of marginals for sets of voxels.

The foreground-background imaging model allows us to estimate the error rate of classifying a voxel as either foreground or background. In the Neyman-Pearson framework, foreground-background classification is a simple two hypothesis testing problem and the most powerful test at a given type 1 error rate is the log likelihood ratio test. The Kullback-Leibler (KL) divergence between the foreground and background distributions gives the exponential rate at which error rates converge to zero as the number of independent, identically distributed samples increases [Cover Thomas and Thomas Joy, 1991]. In the case of using Gaussians to model the foreground-background distributions, the KL divergence reduces to the squared signal to noise ratio. In the absence of normality, the KL Divergence is the general information theoretic measure of image quality for arbitrary distributions.

4.2. **The Prior Distribution via Markov State Representation on the Axons Fragments.** Our representation of the observed image is as a hidden Markov random field with the axon as the hidden latent structure, exploiting the simplicity of conditional independence conditioned on the axon. Given the complexity of sub-micron resolution images, we build an intermediate data structure at the micron scale that represents pieces of axons called fragments $F \subset D$ defined as collections of voxels without any assumed global ordering between them. Depicted in Figure 1 are the fragments shown via different colors. The fragments are a coarser scale voxel representation that gives us an efficient data structure upon which we reassemble and can be viewed analogously as higher order features.

The axon reconstruction problem becomes the reassembly of the fragments along with the imputation of the censored fragments. In the image exmaples presented in this work, the complexity of the space of fragments is approximately $|\mathcal{F}| = 100{,}000$ for a cubic millimeter of projection neuron image data. The probability of

a fragment under the foreground-background appearance model is the simple product probability from Eq. (2c), for $F \subset$ foreground or background, $\alpha_k(I_F) = \prod_{y \in F} \alpha_k(I_y)$, $k = 0, 1$.

We exploit the computational structures of hidden Markov models (HMMs) when the underlying latent structure is absolutely ordered so that dynamic programming can efficiently compute globally optimal state sequence estimates. For each fragment $F \in \mathcal{F}$ we assume that there is a natural identification or correspondence to the states, $F : s \in \mathcal{S} \mapsto F(s) \in \mathcal{F}$ with the state containing the minimum information required to specify the HMM. The state is augmented for all of the calculations to include the annotation of the endpoints of the fragments $x^0, x^1 \in \mathbb{R}^3$ so that we can compute "deletion" or "gap" probabilities, along with unit length tangents $\tau^0, \tau^1 \in \mathbb{R}^3$ associated to the endpoints so we can compute curvature.

As the fragment generation has no implied orientation we generate two states for each fragment, one for each orientation. The states are identical with their endpoints and tangents swapped.

The collection of states $s \in \mathcal{S}$ are the hidden variables in the HMM.

Our algorithms exploit two splitting properties, the Markov nature of the state sequence and the splitting of the random field image conditioned on the state sequence. We use the notation $s_{i:j} := (s_i, s_{i+1}, \ldots, s_j)$ for partial state sequences. We model the state sequence $s_1, \ldots, s_n$ as Markov with splitting property:

$$p(s_{i+1:n}, s_{1:i-1}|s_i) = p(s_{i+1:n}|s_i)p(s_{1:i-1}|s_i) \ , \ i = 2, \ldots, n-1 \ ,$$

which implies the 1-order Markov property $p(s_i|s_{i-1}, s_{1:i-2}) = p(s_i|s_{i-1})$.

We construct the Markov chain with the Boltzmann distribution for transition probabilities:

$$p(s_i|s_{i-1}) = \frac{e^{-U(s_{i-1},s_i)}}{Z(s_{i-1})} \ , \text{ with } Z(s_{i-1}) = \sum_{s_i \in \mathcal{S}} e^{-U(s_{i-1},s_i)} \ ,$$

and energy

$$U(s_{i-1}, s_i) = \alpha_d |x_i^0 - x_{i-1}^1|^2 + \alpha_\kappa \kappa(s_{i-1}, s_i)^2 \ ,$$

where $|\cdot|$ is the standard Euclidean norm. The term $\kappa(s_{i-1}, s_i)$ approximates the curvature of the path connecting s_{i-1} to s_i as follows:

Define $\tau_c := \frac{x_{i-1}^1 - x_i^0}{||x_{i-1}^1 - x_i^0||}$ which is the normalized vector connecting s_{i-1} to s_i and τ_i, τ_{i-1} the tangent vectors:

$$\kappa(s_{i-1}, s_i)^2 = \frac{\kappa_1(s_{i-1}, s_i)^2 + \kappa_2(s_{i-1}, s_i)^2}{2} \ ,$$

with $\kappa_1(s_{i-1}, s_i)^2 = 1 - \tau_{i-1}^1 \cdot \tau_c$, $\kappa_2(s_{i-1}, s_i)^2 = 1 - \tau_c \cdot (-\tau_i^0)$. We derive these terms from the formula for curvature as modeled in Athey et al. [2021] in Supplement S2.

We control the computational complexity associated to computing prior probabilities for all $|\mathcal{S}|^2$ states by restricting the possible state transitions. We set to 0 probability any transitions where the distance between endpoints is greater than $15\,\mu m$ or the angle between states is greater than 150°. We also set the probability that a state transitions to itself to zero.

4.3. **Global Maximally Probable Solutions via $O(n|\mathcal{S}|^2)$ Calculation.** The maximum a-posteriori (MAP) state sequence is defined as the maximizer of the posterior probability (MAP) over the state sequences $s_{1:n} \in \mathcal{S}^n$, with $|\mathcal{S}|$ finite:

$$\hat{s}_{1:n} := \text{argmax}_{s_{1:n} \in \mathcal{S}^n} \log p(s_{1:n}|I_D) \ . \tag{3}$$

The solution space has cardinality $|\mathcal{S}|^n$ so it is infeasible to compute the global maximizer by evaluating all possibilities. Our approach is to rewrite the probability recursively in order to use the Viterbi algorithm

and dynamic programming with $O(n|\mathcal{S}|^2)$ time complexity. We rewrite the MAP estimator in terms of the joint probability:

$$\hat{s}_{1:n} = \text{argmax}_{s_{1:n}\in\mathcal{S}^n} p(s_{1:n}|I_D) = \text{argmax}_{s_{1:n}\in\mathcal{S}^n} p(s_{1:n}, I_D) \ .$$

The image random field is split or conditionally independent conditioned on the fragment states:

$$p(I_{F(s_i)}, I_{D\setminus F(s_i)}|s_i,) = p(I_{F(s_i)}|s_i)p(I_{D\setminus F(s_i)}|s_i) \ ,$$

which implies $p(I_{F(s_i)}|s_i, I_{D\setminus F(s_i)}) = p(I_{F(s_i)}|s_i)$.

We introduce the shorthand notation identifying fragment sequences with the state sequence:

$$F_{1:n} \coloneqq (F(s_1), \ldots, F(s_n)) \ .$$

Define the indicator function $\delta_A(x) = 1$ for $x \in A$, 0 otherwise.

Lemma 1. *For $n > 1$ we have the joint probability:*

$$p(s_{1:n}, I_D) = \prod_{i=2}^{n} \left(\frac{\alpha_1(I_{F_i})}{\alpha_0(I_{F_i})}\right)^{\delta_{D\setminus F_{1:i-1}}(F_i)} p(s_i|s_{i-1}) \ p(s_1, I_D) \ . \tag{4}$$

See Supplement S3 for the proof which rewrites the probability recursively giving the factorization.

It is natural to identify the negative logarithm of the joint probability and the total cost of a path through a directed trellis graph [Forney, 1973], for which several algorithms exist that solve the shortest path problem in $O(n|\mathcal{S}|^2)$ complexity. However, we cannot use these algorithms directly because the cost function is not "sequentially-additive" due to the dependence of the indicator function on previous states in the sequence. In the Supplement, section S4, we offer a example demonstrating that directly applying the Viterbi algorithm to this problem does not generate the MAP estimate.

We adjust our probabilistic representation on the $|\mathcal{S}|^n$ paths in order to utilize shortest path algorithms such as Bellman-Ford or Dijkstra's [Bellman, 1958, Dijkstra, 1959]. For this we note that the $\frac{\alpha_1(I_{F_i})}{\alpha_0(I_{F_i})}$ term in Eqn. (4) may often be greater than 1. In the directed graph formulation in which arcs represent distance between states via the negative logarithm of the probability, this can lead to negative cycles in the graph of states. When negative cycles exist, the shortest path problem is ill-posed. To avoid this phenomena we remove the background component of the image from the joint probability, which converts $\frac{\alpha_1(I_{F_i})}{\alpha_0(I_{F_i})}$ to $\alpha_1(I_{F_i})$, and converts our global posterior probability to our path probability formulation.

Theorem 1. *Define the most probable solution $s_{1:n} \in \mathcal{S}^n$ by the joint probability* $\text{argmax}_{s_{1:n}\in\mathcal{S}^n} p(s_{1:n}, I_{F_{1:n}})$. *Then we have*

$$\max_{s_{1:n}\in\mathcal{S}^n} p(s_{1:n}, I_{F_{1:n}}) = \max_{s_{1:n}\in\mathcal{S}^n} \prod_{i=2}^{n} (\alpha_1(I_{F_i}))^{\delta_{D\setminus F_{1:i-1}}(F_i)} \, p(s_i|s_{i-1}) \, p(s_1, I_{F_1}) \ . \tag{5}$$

See Supplement S3 for proof.

We note that since the path $(s_{1:n})$ defines the subset of the image in the joint probability $(I_{F_{1:n}})$ we can define the probability as a function of only the state sequence $\bar{p}(s_{1:n}) \coloneqq p(s_{1:n}, I_{F_{1:n}})$ emphasizing that we are solving the most probable path problem.

We can maximize this expression using dynamic programming since $\alpha_1(\cdot)$ satisfies the condition that it is a probability.

Proposition 1. *Suppose that $\alpha_1(I_y) \leqslant 1$ for all y. Then the globally optimal solution to the fixed start and end point problem is a nonrepeating sequence and can be obtained by computing the shortest path in a directed graph where the vertices are the states, and the edge weight from state s_{i-1} to s_i is given by:*

$$e(s_{i-1}, s_i) = -\log \alpha_1(I_{F_i}) - \log p(s_i|s_{i-1}) \tag{6}$$

Proof. First, we want to show that the globally optimal sequence is nonrepeating. This is clear because if $\alpha_1(I_y) \leqslant 1$, then every term in the product in equation (5) is bounded by 1. Thus, for any state sequence that repeats states, if we remove all elements between the two instances of the repeated states, then this new sequence will have at least as much probability $p(s_{1:n}, I_{F_{1:n}})$.

For nonrepeating state sequences, the probability $p(s_{1:n}, I_{F_{1:n}})$ of (5) can be simplified:

$$p(s_{1:n}, I_{F_{1:n}}) = \prod_{i=2}^{n} \left(\alpha_1(I_{F_i})\right) p(s_i|s_{i-1}) p(s_1, I_{F_1}) \ .$$

Taking the logarithm yields a sequentially additive cost function that can be solved with shortest path algorithm on a graph with edge weights given by Equation (6).

□

4.4. **Implementation.**

4.4.1. *Fragment generation:* Fragments are collections of voxels, or "supervoxels," and can be viewed analogously as higher order features such as edgelets or corners. As described in section 4.2, identifying the subset of fragments that compose the axon, then ordering them becomes equivalent to reconstructing the axon contour model.

The first step of fragment generation is obtaining a foreground-background mask, which could be obtained, for example, from a neural network, or by simple thresholding. In this work, we use an Ilastik model that was trained on three image subvolumes, each of which has three slices that were labeled [Berg et al., 2019]. During prediction, the probability predictions from Ilastik are thresholded at 0.9, a conservative threshold that keeps the number of false positives low.

The connected components of the thresholded image are split into fragments of similar size by identifying the voxel v with the largest predicted foreground probability and placing a ball B_v with radius $5\,\mu m$ on that voxel. The voxels within B_v are removed and the process is repeated until the component is covered. The component is then split up into pieces by assigning each voxel to the center point from the previous step, v, that is closest to it. This procedure ensures that each fragment is no larger than a ball with radius $5\,\mu m$. At this size, it is reasonable to assume that each fragment is associated with only one axon branch since no fragment is large enough to extensively cover multiple branches.

Next, the endpoints x^0, x^1 and tangents τ^0, τ^1 are computed as described in Supplement S1. Each fragment is simplified to the line segment between its endpoints which is rasterized using the Bresenham algorithm [Bresenham, 1965]. Briefly, the Bresenham algorithm identifies the image axis along which the line segment has the largest range and samples the line once every voxel unit along that axis. Then, the other coordinates are chosen to minimize the distance from the continuous representation line segment.

4.4.2. *Imputing Fragment Deletions.* In practice the imaging data may exhibit significant dropouts leading to significant fragment deletions. While computing the likelihood of the image data, we augment the gaps between any pair of connected fragments in $F_1, F_2, \ldots$ by augmenting the sequence with imputed fragments $F_1, \bar{F}_1, F_2, \bar{F}_2, \ldots$., with $\bar{F}_i \subset D$ the imputed line of voxels which forms the connection between the pair F_i, F_{i+1}. For this define the start and endpoint of each fragment as $x^0(F) \in \mathbb{R}^3, x^1(F) \in \mathbb{R}^3$ with line segment connecting each pair:

$$L_{i,i+1} = \{y : y = ax^1(F_i) + (1-a)x^0(F_{i+1}),\ a \in [0,1]\} \ .$$

The imputed fragment $\bar{F}_i \subset D$ for each pair (F_i, F_{i+1}) is computed by rasterizing $L_{i,i+1}$ with the Bresenham algorithm.

The likelihood of the sequence of fragments augmented by the imputations becomes

$$p(s_{1:n}, I_{F_{1:n}}) = \prod_{i=2}^{n} \alpha_1(I_{F_i})^{\delta_{D\setminus F_{1:i-1}}(F_i)} \alpha_1(I_{\bar{F}_i}) p(s_i|s_{i-1}) p(s_1, I_{F_1})$$

4.4.3. *Initial and Endpoint Conditions.* We take the initial conditions to represent

$$p(s_1, I_{F_1}) = \pi(s_1) p(I_{F_1}|s_1) \ ,$$

with π the prior on initial state. For all of our axon reconstructions we specify an axonal fragment as the start state s_{start} and set $\pi(s_1) \coloneqq \delta_{s_{start}}(s_1)$.

The endpoint conditions are defined via a user specified terminal state s_{term} where the path ends giving the maximization:

$$\max_{s_{1:n} \in \mathcal{S}^n} p(s_{1:n}, I_{F_{1:n}}|s_n = s_{term}) \ .$$

The marginal probability on the terminal state always transitions to itself, so that $p(s_n = s_{term}) = \delta_{s_{term}}(s_n)$. Thus, a state sequence solution of length n may end in multiple repetitions of s_{term}, such as

$$s_{1:n} = \{s_1, s_2, ..., s_{n'}, s_{term}, s_{term}, ..., s_{term}\}.$$

4.5. **Accuracy Metrics.** We applied several state of the art reconstruction algorithms to several neurons in the brain samples from the MouseLight Project from HHMI Janelia [Winnubst et al., 2019]. In this dataset, projection neurons were sparsely labeled then imaged with a two-photon microscope at a voxel resolution of $0.3 \times 0.3 \times 1\mu m$. Each axon reconstruction is generated semi-automatically by two independent annotators. The MouseLight reconstructions are sampled roughly every $10\mu m$, so in some cases we retraced the axons at a higher sampling frequency in order to obtain more precise accuracy metrics.

We quantified reconstruction accuracy using two metrics, the first of which is Frechet distance. Frechet distance is commonly described in the setting of dog walking, where both the dog and owner are following their own predetermined paths. The Frechet distance between the two paths then is the minimum length dog leash needed to complete the walk, where both dog and owner are free to vary their walking speeds but are not allowed to backtrack. In our setting we compute the Frechet distance between two discrete paths $P : \{1, ..., L_p\} \to \mathbb{R}^3$, $Q : \{1, ..., L_q\} \to \mathbb{R}^3$ as defined in Eiter and Mannila [1994]. In this definition, a coupling between P and Q is defined as a sequence of ordered pairs:

$$(P[a_1], Q[b_1]), (P[a_2], Q[b_2]), ..., (P[a_K], Q[b_K])$$

where the following conditions are put on $\{a_k\}, \{b_k\}$ to ensure that they enumerate through the whole sequences P and Q:

- $a_1, b_1 = 1$
- $a_N = L_p$, $b_N = L_q$
- $a_k = a_{k-1}$ or $a_k = a_{k-1} + 1$
- $b_k = b_{k-1}$ or $b_k = b_{k-1} + 1$

Then the discrete Frechet distance is defined as:

$$\delta_{dF}(P, Q) = \min_{\text{coupling } \{a_k\},\{b_k\}} \max_{k \in \{1,...,K\}} |P[a_k] - Q[b_k]|$$

We use the standard Euclidean norm for $|\cdot|$. The discrete Frechet distance is an upper bound to the continuous Frechet distance between polygonal curves, and it can be computed more efficiently. Further, if

we take a discrete Frechet distance of zero to be an equivalence relation, then δ_{dF} is a metric on this set of equivalence classes and thus is a natural way to compare non-branching neuronal reconstructions. In this work, all reconstruction are sampled at at least one point per micron.

Various other performance metrics have been proposed, including an arc-length based precision and recall [Wang et al., 2011], a critical node matching based Miss-Extra-Scores (MES) [Xie et al., 2011] and a vertex matching based spatial distance (SD) [Peng et al., 2010b]. We chose to compute SD since it gives a picture of the *average* spatial distance between two reconstructions. This complements the Frechet distance described earlier, which computes the *maximum* spatial distance between two reconstructions.

The first step in computing the SD from reconstruction P to reconstruction Q is, for each point in P, finding the distance to the closest point in Q. Directed divergence (DDIV) of P from Q is then defined as the average of all these distances. Then, SD is computed by averaging the DDIV from P to Q and the DDIV from Q to P.

5. Acknowledgements

We thank the MouseLight team at HHMI Janelia for providing us with access to this data, and answering our questions about it.

6. Author Contributions

MIM helped to develop the HMM and probabilistic model and DT advised on the theoretical direction of the manuscript. UM coordinated the data acquisition for the experiments. TA designed the study, implemented the software, and managed the manuscript text/figures. All authors contributed to manuscript revision.

7. Conflict of Interest Statement

MIM owns a significant share of Anatomy Works with the arrangement being managed by Johns Hopkins University in accordance with its conflict of interest policies. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

8. Funding

This work is supported by the National Institutes of Health grant RF1MH121539.

References

L. Acciai, P. Soda, and G. Iannello. Automated neuron tracing methods: an updated account. *Neuroinformatics*, 14(4):353–367, 2016.

T. L. Athey, J. Teneggi, J. T. Vogelstein, D. J. Tward, U. Mueller, and M. I. Miller. Fitting splines to axonal arbors quantifies relationship between branch order and geometry. *Frontiers in Neuroinformatics*, 2021.

R. Bellman. On a routing problem. *Quarterly of applied mathematics*, 16(1):87–90, 1958.

S. Berg, D. Kutra, T. Kroeger, C. N. Straehle, B. X. Kausler, C. Haubold, M. Schiegg, J. Ales, T. Beier, M. Rudy, K. Eren, J. I. Cervantes, B. Xu, F. Beuttenmueller, A. Wolny, C. Zhang, U. Koethe,

F. A. Hamprecht, and A. Kreshuk. ilastik: interactive machine learning for (bio)image analysis. *Nature Methods*, Sept. 2019. ISSN 1548-7105. doi: 10.1038/s41592-019-0582-9. URL https://doi.org/10.1038/s41592-019-0582-9.

J. E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems journal*, 4(1):25–30, 1965.

H. Chen, H. Xiao, T. Liu, and H. Peng. Smarttracing: self-learning-based neuron reconstruction. *Brain informatics*, 2(3):135–144, 2015.

A. Choromanska, S.-F. Chang, and R. Yuste. Automatic reconstruction of neural morphologies with multi-scale tracking. *Frontiers in neural circuits*, 6:25, 2012.

L. D. Cohen. On active contour models and balloons. *CVGIP: Image Understanding*, 53(2): 211–218, 1991. ISSN 1049-9660. doi: https://doi.org/10.1016/1049-9660(91)90028-N. URL https://www.sciencedirect.com/science/article/pii/104996609190028N.

M. Cover Thomas and A. Thomas Joy. *Elements of information theory*, volume 2. Wiley, 1991.

T. Dai, M. Dubois, K. Arulkumaran, J. Campbell, C. Bass, B. Billot, F. Uslu, V. De Paola, C. Clopath, and A. A. Bharath. Deep reinforcement learning for subpixel neural tracking. In *International Conference on Medical Imaging with Deep Learning*, pages 130–150. PMLR, 2019.

E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

T. Eiter and H. Mannila. Computing discrete fréchet distance. Technical report, Citeseer, 1994.

G. D. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

D. Friedmann, A. Pun, E. L. Adams, J. H. Lui, J. M. Kebschull, S. M. Grutzner, C. Castagnola, M. Tessier-Lavigne, and L. Luo. Mapping mesoscale axonal projections in the mouse brain using a 3d convolutional network. *Proceedings of the National Academy of Sciences*, 117(20):11068–11075, 2020.

M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International journal of computer vision*, 1(4):321–331, 1988.

N. Khaneja, M. I. Miller, and U. Grenander. Dynamic programming generation of curves on brain surfaces, 1998.

R. Li, T. Zeng, H. Peng, and S. Ji. Deep learning segmentation of optical microscopy images improves 3-d neuron reconstruction. *IEEE transactions on medical imaging*, 36(7):1533–1541, 2017.

R. Li, M. Zhu, J. Li, M. Bienkowski, N. N. Foster, H. Xu, T. Ard, I. Bowman, C. Zhou, M. Veldman, X. W. Yang, H. Hintiryan, J. Zhang, and H. Dong. Precise segmentation of densely interweaving neuron clusters using g-cut. *Nature Communications*, 10, 2019.

H. Peng, Z. Ruan, D. Atasoy, and S. Sternson. Automatic reconstruction of 3d neuron structures using a graph-augmented deformable model. *Bioinformatics*, 26(12):i38–i46, 2010a.

H. Peng, Z. Ruan, F. Long, J. H. Simpson, and E. W. Myers. V3d enables real-time 3d visualization and quantitative analysis of large-scale biological image data sets. *Nature biotechnology*, 28(4):348–353, 2010b.

H. Peng, E. Meijering, and G. A. Ascoli. From diadem to bigneuron, 2015.

H. Peng, Z. Zhou, E. Meijering, T. Zhao, G. A. Ascoli, and M. Hawrylycz. Automatic tracing of ultra-volumes of neuronal images. *Nature methods*, 14(4):332–333, 2017.

T. Quan, H. Zhou, J. Li, S. Li, A. Li, Y. Li, X. Lv, Q. Luo, H. Gong, and S. Zeng. Neurogps-tree: automatic reconstruction of large-scale neuronal populations with dense neurites. *Nature methods*, 13(1):51–54, 2016.

L. Rabiner and B. Juang. An introduction to hidden markov models. *ieee assp magazine*, 3(1):4–16, 1986.

M. Radojević and E. Meijering. Automated neuron tracing using probability hypothesis density filtering. *Bioinformatics*, 33(7):1073–1080, 01 2017. ISSN 1367-4803. doi: 10.1093/bioinformatics/btw751. URL https://doi.org/10.1093/bioinformatics/btw751.

O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

D. L. Snyder and M. I. Miller. *Random point processes in time and space*. Springer Science & Business Media, 2012.

E. Turetken, F. Benmansour, B. Andres, H. Pfister, and P. Fua. Reconstructing loopy curvilinear structures using integer programming. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1822–1829, 2013.

C.-W. Wang, Y.-C. Lee, H. Pradana, Z. Zhou, and H. Peng. Ensemble neuron tracer for 3d neuron reconstruction. *Neuroinformatics*, 15(2):185–198, 2017.

Y. Wang, A. Narayanaswamy, C.-L. Tsai, and B. Roysam. A broadly applicable 3-d neuron tracing method based on open-curve snake. *Neuroinformatics*, 9:193–217, 03 2011. doi: 10.1007/s12021-011-9110-5.

Y. Wang, Q. Li, L. Liu, Z. Zhou, Z. Ruan, L. Kong, Y. Li, Y. Wang, N. Zhong, R. Chai, et al. Teravr empowers precise reconstruction of complete 3-d neuronal morphology in the whole brain. *Nature communications*, 10(1):1–9, 2019.

J. Winnubst, E. Bas, T. A. Ferreira, Z. Wu, M. N. Economo, P. Edson, B. J. Arthur, C. Bruns, K. Rokicki, D. Schauder, et al. Reconstruction of 1,000 projection neurons reveals new cell types and organization of long-range connectivity in the mouse brain. *Cell*, 179(1):268–281, 2019.

H. Xiao and H. Peng. App2: automatic tracing of 3d neuron morphology based on hierarchical pruning of a gray-weighted image distance-tree. *Bioinformatics*, 29(11), June 2013.

J. Xie, T. Zhao, T. Lee, E. Myers, and H. Peng. Anisotropic path searching for automatic neuron reconstruction. *Medical image analysis*, 15(5):680–689, 2011.

J. Yang, M. Hao, X. Liu, Z. Wan, N. Zhong, and H. Peng. Fmst: an automatic neuron tracing method based on fast marching and minimum spanning tree. *Neuroinformatics*, 17(2):185–196, 2019.

Z. Zhou, H.-C. Kuo, H. Peng, and F. Long. Deepneuron: an open deep learning toolbox for neuron tracing. *Brain informatics*, 5(2):1–9, 2018.

SUPPLEMENTARY MATERIAL

S1. COMPUTING ENDPOINTS AND TANGENTS OF FRAGMENTS

As explained in section 4.2, each fragment is a subset of the image $F \subset D$. Each fragment is assumed to be associated with a segment of an underlying neuron curve, and we want to estimate the locations of the two endpoints of the fragment. First, we compute the length of the diagonal of the bounding box that contains the fragment. We divide this length by two to get R. Then, with each voxel y in the fragment, we associate a set of voxels N_y which is the intersection of the voxels in the fragment and the voxels within distance R from y. The voxel with the smallest set N_y (by cardinality) is chosen to be the first endpoint. The second endpoint is the voxel that has the smallest set N_y but is also farther than R away from the first endpoint.

Currently the tangents at the endpoints is just approximated by the difference of the endpoints i.e. $\tau^0 = \frac{x_0 - x_1}{||x_0 - x_1||}, \tau^1 = -\tau^1$. This method is based on the assumption that fragments are small enough that they are approximately straight. We also experimented with approximating the endpoint tangents by computing principal components of voxels near the endpoints, but found it to be less robust for downstream reconstruction.

S2. Curvature Calculation for the Potential in the Markov Chain

The term $\kappa(s_{i-1}, s_i)$ is an approximation of the curvature of the path that connects s_{i-1} to s_i. For a curve with the tangent vector $T(s)$, curvature is defined as $\kappa(s) = \left|\left|\frac{dT}{ds}\right|\right|$. A finite difference approximation of curvature is then:

$$\begin{aligned}\left|\left|\frac{dT}{ds}\right|\right|^2 &\approx ||T(s) - T(s-1)||^2 \\ &= ||T(s)||^2 + ||T(s-1)||^2 - 2T(s)\cdot T(s-1) \\ &= 2(1 - T(s)\cdot T(s-1))\end{aligned}$$

Thus,

$$\left|\left|\frac{dT}{ds}\right|\right|^2 \propto 1 - T(s+1)\cdot T(s) \tag{7}$$

We consider τ_i, τ_{i-1} and the normalized vector between the states $\tau_c := \frac{x^1_{i-1} - x^0_i}{|x^1_{i-1} - x^0_i|}$ as samples of $T(s)$, then use Eq. (7) to estimate the curvature induced by connecting state s_{i-1} to s_i:

$$\begin{aligned}(\kappa_1)^2 &= 1 - \tau^1_{i-1}\cdot\tau_c \\ (\kappa_2)^2 &= 1 - \tau_c\cdot(-\tau^0_i) \\ \kappa(s_{i-1}, s_i)^2 &= \frac{(\kappa_1)^2 + (\kappa_2)^2}{2} \qquad \text{Arithmetic mean}\end{aligned}$$

S3. Proof of Lemma 1 and Theorem 1

Recall the likelihood of a complete fragment under the foreground-background model (Eq. 2c):

$$\alpha_k(I_F) := \prod_{y\in F}\alpha_k(I_y)\ , k = 0, 1\ .$$

Lemma 1:
For $n > 1$ we have the recursion probability

$$p(s_{1:n}, I_D) = \left(\frac{\alpha_1(I_{F_n})}{\alpha_0(I_{F_n})}\right)^{\delta_{D\setminus F_{1:n-1}}(F_n)} p(s_n|s_{n-1})\, p(s_{1:n-1}, I_D), \tag{8a}$$

implying the factored probability:

$$p(s_{1:n}, I_D) = \prod_{i=2}^{n}\left(\frac{\alpha_1(I_{F_i})}{\alpha_0(I_{F_i})}\right)^{\delta_{D\setminus F_{1:i-1}}(F_i)} p(s_i|s_{i-1})\, p(s_1, I_D)\ . \tag{8b}$$

Proof. Factor the event $I_D = (I_{F_{1:n}}, I_{D\setminus F_{1:n}})$, then

$$\begin{aligned}p(s_{1:n}, I_{F_{1:n}}, I_{D\setminus F_{1:n}}) &= p(I_{F_n}|s_{1:n}, I_{F_{1:n-1}}, I_{D\setminus F_{1:n}})p(s_n|s_{1:n-1}, I_{D\setminus F_{1:n}})p(s_{1:n-1}, I_{F_{1:n-1}}, I_{D\setminus F_{1:n}}) \\ &= p(I_{F_n}|s_n)^{\delta_{D\setminus F_{1:n-1}}(F_n)}p(s_n|s_{n-1})p(s_{1:n-1}, I_{F_{1:n-1}}, I_{D\setminus F_{1:n}}) \\ &= \alpha_1(I_{F_n})^{\delta_{D\setminus F_{1:n-1}}(F_n)}p(s_n|s_{n-1})p(s_{1:n-1}, I_{F_{1:n-1}}, I_{D\setminus F_{1:n}}) \qquad (9)\end{aligned}$$

We rewrite the last term using the splitting property:

$$\begin{aligned} p(s_{1:n-1}, I_{F_{1:n-1}}, I_{D\setminus F_{1:n}}) &= p(I_{F_{1:n-1}}|s_{1:n-1}, I_{D\setminus F_{1:n}})p(s_{1:n-1}, I_{D\setminus F_{1:n}}) \\ &= p(I_{F_{1:n-1}}|s_{1:n-1})p(I_{D\setminus F_{1:n}}|s_{1:n-1})p(s_{1:n-1}) \\ &= p(I_{F_{1:n-1}}|s_{1:n-1})\frac{p(I_{D\setminus F_{1:n-1}}|s_{1:n-1})}{p(I_{F_n}|s_{1:n-1})^{\delta_{D\setminus F_{1:n-1}}(F_n)}}p(s_{1:n-1}) \\ &= \frac{1}{\alpha_0(I_{F_n})^{\delta_{D\setminus F_{1:n-1}}(F_n)}}p(s_{1:n-1}, I_{F_{1:n-1}}, I_{D\setminus F_{1:n-1}}) \ , \end{aligned}$$

with the last substitution following from the background model. Substituting into (9) yields the probability written as a recursion (8a):

$$p(s_{1:n}, I_D) = \left(\frac{\alpha_1(I_{F_n})}{\alpha_0(I_{F_n})}\right)^{\delta_{D\setminus F_{1:n-1}}(F_n)} p(s_n|s_{n-1})\, p(s_{1:n-1}, I_D),$$

□

Theorem 1: Define the most probable solution $s_{1:n} \in \mathcal{S}^n$ by the joint probability $\operatorname{argmax}_{s_{1:n}\in\mathcal{S}^n} p(s_{1:n}, I_{F_{1:n}})$. Then we have

$$\max_{s_{1:n}\in\mathcal{S}^n} p(s_{1:n}, I_{F_{1:n}}) = \max_{s_{1:n}\in\mathcal{S}^n} \prod_{i=2}^{n} \left(\alpha_1(I_{F_i})\right)^{\delta_{D\setminus F_{1:i-1}}(F_i)} p(s_i|s_{i-1})\, p(s_1, I_{F_1}) \ . \tag{10}$$

Proof.

$$\begin{aligned} p(s_{1:n}, I_{F_{1:n}}) &= p(s_{1:n}, I_{F_{1:n-1}}, I_{F_n}) \\ &= p(I_{F_n}|s_{1:n}, I_{F_{1:n-1}})p(s_{1:n}, I_{F_{1:n-1}}) \\ &= p(I_{F_n}|s_{1:n}, I_{F_{1:n-1}})p(I_{F_{1:n-1}}|s_{1:n})p(s_{1:n}) \\ &= p(I_{F_n}|s_{1:n}, I_{F_{1:n-1}})p(I_{F_{1:n-1}}|s_{1:n-1})p(s_n|s_{1:n-1})p(s_{1:n-1}) \\ &= \left(\alpha_1(I_{F_n})\right)^{\delta_{D\setminus F_{1:n-1}}(F_n)} p(s_n|s_{n-1})p(I_{F_{1:n-1}}|s_{1:n-1})p(s_{1:n-1}) \ . \end{aligned}$$

Applying this to factor the conditional probability , $I_{F_{1:i}}$, $i = n-1, \ldots, 1$ gives the joint product. □

S4. $O(n|\mathcal{S}|^2)$ ALGORITHM COUNTER-EXAMPLE

Here we present a simple counter example demonstrating that the indicator function in 4 cannot be ignored, implying the globally optimal solution cannot be obtained efficiently with the Viterbi algorithm. The details of the state likelihoods and transition probabilities are given in Figure S1a.

Here, we remove the indicator function in 4 and apply the Viterbi algorithm in the usual way. After three iterations, the Viterbi algorithm stores $(1, 1, 1, 2)$ as the highest probability path of length 4 that ends in state 2 (Figure S1c). The joint probability of this path is $1/800$. However, there is an alternative path that the algorithm missed, $(1, 2, 1, 2)$, which has higher joint probability $1/400$ (Figure S1b). The algorithm did not identify the globally optimal sequence because it did not "see" that the cost of transitioning from state 1 to 2 would drop from $1/200$ to $1/2$ after visiting state 2 the first time.

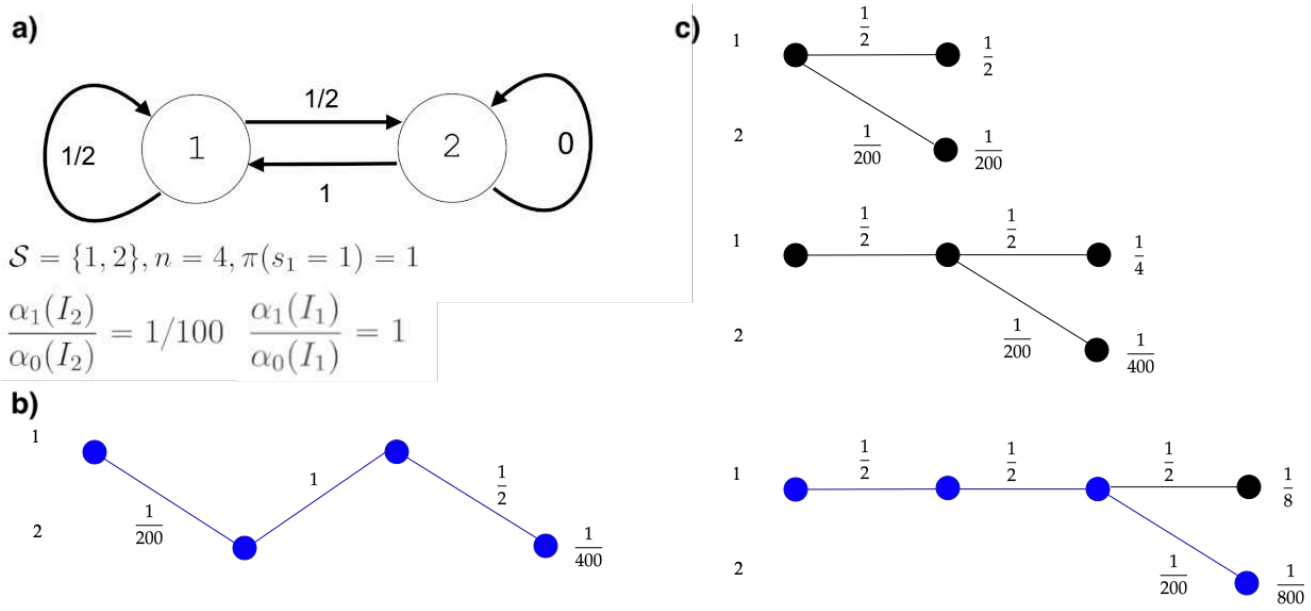

FIGURE S1. Example showing that the Viterbi algorithm is invalid in our setting. a) Transition probabilities, and image likelihood ratios of a two state Markov model where, in our probabilistic model, the globally optimal state sequence from state 1 to state 2 with length 4 cannot be computed with the classic Viterbi algorithm. Panels b) and c) depict paths through the state space. The nodes represent the states and the line segments represent state transitions, as in Figure 8 of Forney [1973]. Panel b) shows true shortest path from state 1 to state 2 with length 4. Panel c) illustrates three iterations of the Viterbi algorithm, with the lines representing the paths that are stored at each iteration. In this example, the blue path at the bottom of panel c) is identified by naive application of the Viterbi algorithm, but it differs from the true shortest path in panel b). Each transition has a number above it indicating its contribution to the joint probability - $\left(\frac{\alpha_1(I_{F_i})}{\alpha_0(I_{F_i})}\right)^{\delta_{D\setminus F_{1:i-1}}(F_i)} p(s_i|s_{i-1})$ from Eqn. (4). The number at the end of the path is the product of all such terms - the joint probability of the given path.