Neural Process Reconstruction from Sparse User Scribbles

Mike Roberts[1], Won-Ki Jeong[1], Amelio Vázquez-Reina[1], Markus Unger[2], Horst Bischof[2], Jeff Lichtman[1], and Hanspeter Pfister[1]

[1] Harvard University
[2] Graz University of Technology

Abstract. We present a novel semi-automatic method for segmenting neural processes in large, highly anisotropic EM (electron microscopy) image stacks. Our method takes advantage of sparse scribble annotations provided by the user to guide a 3D variational segmentation model, thereby allowing our method to globally optimally enforce 3D geometric constraints on the segmentation. Moreover, we leverage a novel algorithm for propagating segmentation constraints through the image stack via optimal volumetric pathways, thereby allowing our method to compute highly accurate 3D segmentations from very sparse user input. We evaluate our method by reconstructing 16 neural processes in a $1024 \times 1024 \times 50$ nanometer-scale EM image stack of a mouse hippocampus. We demonstrate that, on average, our method is 68% more accurate than previous state-of-the-art semi-automatic methods.

1 Introduction

Mapping neural circuitry is an important ongoing challenge in neurobiology. Current approaches to this task involve tracing neural processes through segmented nanometer-scale EM (electron microscopy) image stacks of brain tissue. Since our understanding of neural circuitry is often limited by our ability to reconstruct neural processes from EM image stacks, accurately segmenting neural processes is an important open problem in the medical image analysis community.

Dense reconstruction algorithms [1, 7, 11–13, 15, 20] generally rely on supervised learning methods to automatically classify every pixel in an image stack according to the type of cellular structure to which it belongs. However, no dense reconstruction algorithm can reliably produce segmentations that are completely free of topological errors. In practice, these methods often require significant user effort to correct errors in the automatically generated segmentations.

On the other hand, *sparse reconstruction algorithms* rely on the user to interactively guide the segmentation of individual neural processes. Most existing sparse algorithms compute 3D reconstructions as sequences of locally optimal 2D segmentations after the user provides an initial 2D contour [4, 8, 10, 14, 19]. However, these approaches do not optimally enforce 3D geometric consistency constraints on the resulting segmentation, and therefore often require frequent user intervention. The recent *Markov Surfaces* algorithm [16] requires user-defined

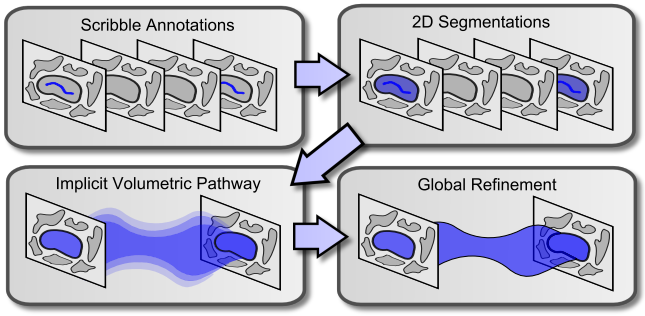

Fig. 1. Overview of our method. We assume that we are given scribble annotations indicating a neural process of interest on the first and last slices of an image stack (top left). We compute 2D segmentations that contain the scribble annotations and align with strong image edges; these 2D segmentations define hard constraints on our 3D segmentation (top right). We propagate the 2D segmentations through the image stack according to an implicitly represented volumetric pathway, which we compute based on the dense optical flow between image slices; the interior level sets of this volumetric pathway define soft constraints on our 3D segmentation (bottom left). We compute the final 3D segmentation by globally refining the volumetric pathway according to an anisotropic variational segmentation model that aligns with strong in-plane image edges and enforces 3D smoothness (bottom right).

2D contours on the first and last slices of an image stack. This algorithm automatically tessellates a set of globally optimal paths between these contours, relying on 2D Bézier interpolation to produce smooth surfaces. However, since Bézier interpolation does not take into account the underlying image data, the resulting segmentations may ignore important image features.

In this paper, we introduce a novel method for neural process reconstruction that only requires very sparse scribble annotations as input (Fig. 1). We evaluate our method by reconstructing 16 neural processes in a $1024 \times 1024 \times 50$ nanometer-scale EM image stack of a mouse hippocampus. We demonstrate that, on average, our method is 68% more accurate than Markov Surfaces [16], 91% more accurate than Geo-Cuts [2], and 263% more accurate than Marker-Controlled Watersheds [6].

2 Our Method

We observe that the problem of reconstructing neural processes through highly anisotropic EM image stacks is conceptually similar to the problem of tracking moving objects in video sequences. Based on this observation, our work is inspired by the recent *Anisotropic Total Variation* model proposed by Unger *et al.* [17], which tracks objects through video sequences based on sparse constraints provided by the user. However, the absence of color information in EM image data and poor spatial continuity across EM image slices prevent the direct ap-

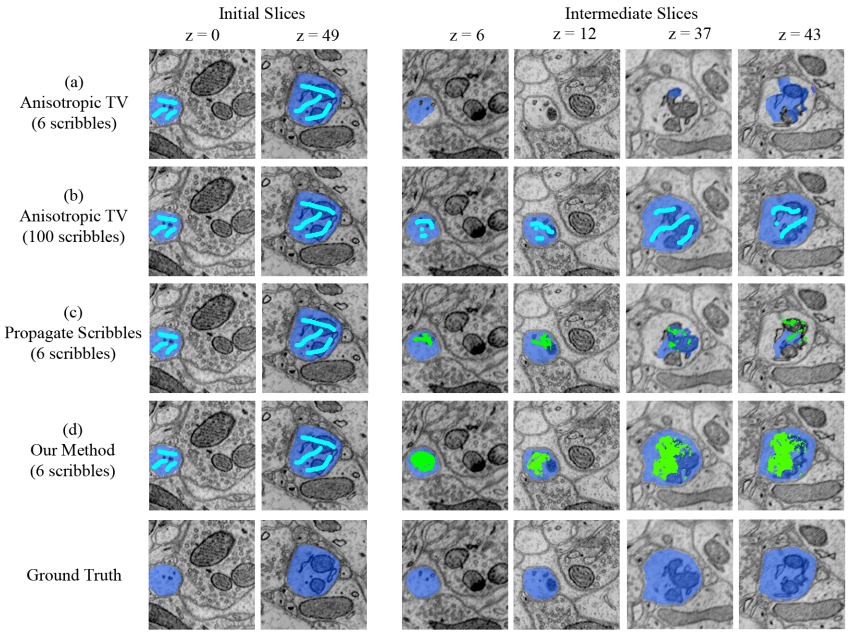

Fig. 2. Key observations motivating our method. Anisotropic Total Variation [17] fails to segment this neural process from sparse scribble annotations (a), but succeeds if scribble annotations are given on every slice (b). Our method only requires scribble annotations on the first and last slices because we automatically propagate segmentation constraints through the image stack. However, propagating scribble annotations as segmentation constraints results in a significant under-segmentation of this neural process (c). Instead, we compute 2D segmentations from the scribble annotations and propagate the 2D segmentations, resulting in an accurate segmentation of this neural process (d). Scribble annotations are shown in light blue, segmentations are shown in dark blue, and automatically propagated segmentation constraints are shown in green.

plication of this method to neural process reconstruction (Fig. 2a). We observe that this model can robustly track the neural process of interest if the user provides constraints on each slice of the image stack (Fig. 2b). This observation motivates our method for automatically propagating segmentation constraints through the image stack based on the dense optical flow between slices.

If we propagate the user-provided scribble annotations through the image stack as soft segmentation constraints, we observe an uneven distribution of propagated constraints. This can result in a significant under-segmentation of the neural process of interest (Fig. 2c). In contrast, if we propagate accurate 2D segmentations (instead of scribble annotations) through the image stack, we observe more evenly distributed segmentation constraints. This results in an accurate 3D segmentation of the neural process of interest (Fig. 2d). These observations motivate our method for computing 2D segmentations and subsequently propagating them through the image stack as soft segmentation constraints.

Input. We assume that the user marks the neural process of interest with a few foreground and background scribbles on the first and last image slices. For the sake of notational clarity, we assume that the neural process of interest is roughly orthogonal to the image stack. In practice, we allow the user to segment neural processes that run in any direction by projecting the segmentation problem along any 1D path through the image volume.

Computing 2D segmentations. We compute 2D segmentations of the neural process of interest by globally minimizing the following variational segmentation energy [18]:

$$\underset{u_i}{\operatorname{argmin}} \int_{\mathbf{x}\in\Omega_i} (g_i|\nabla u_i| + f_i u_i)\, d\mathbf{x}, \tag{1}$$

where i refers to the indices of the first and last slices, Ω_i is the 2D image domain. The function $u_i : \Omega_i \to [0, 1]$ encodes the resulting 2D segmentation for each slice, where $u_i > \frac{1}{2}$ is foreground and $u_i \leq \frac{1}{2}$ is background. The function $g_i : \Omega_i \to [0, 1]$ encodes strong image edges as small values, and the function $f_i : \Omega_i \to (-\infty, \infty)$ is defined according to the user-provided scribble annotations, where we set $f_i := -\infty$ for foreground scribbles, $f_i := \infty$ for background scribbles, and $f_i := 0$ otherwise. Minimizing (1) results in 2D segmentations that respect the user-provided scribbles annotations and align with strong image edges.

The foreground and background regions of these 2D segmentations define hard foreground and background constraints on our 3D segmentation, respectively.

Computing an Optimal Volumetric Pathway. Once we have obtained hard constraints on the first and last slices of our image stack, we generate soft constraints on all the other slices by automatically propagating the previously computed 2D segmentations through the stack. One way to accomplish this would be to advect each foreground pixel in each 2D segmentation through the image stack according to the dense optical flow between image slices. However, this approach is unreliable since small errors in the pairwise optical flow between images accumulate quickly, as noted previously by Pan *et al.* [16].

Instead, we define an optimal volumetric pathway through the image stack that connects the previously computed 2D segmentations and encloses the pixels that are most likely to belong to the neural process of interest. In this formulation, the optimal volumetric pathway is given by the interior level sets of a *cost volume* that encodes the probability of each pixel in the image stack belonging to the neural process of interest.

We define the cost of each pixel $\mathbf{p}$ in the cost volume as the length of the shortest path that connects the previously computed 2D segmentations via $\mathbf{p}$, as described in Fig. 3. We compute the length of each path through the image stack as a function of the dense optical flow between image slices as follows. For the pixels $\mathbf{p} \in \Omega_i$, $\mathbf{q} \in \Omega_{i+1}$, and the optical flow vector $v(\mathbf{p})$, we define the length from $\mathbf{p}$ to $\mathbf{q}$ as $d(\mathbf{p}, \mathbf{q}) = |\mathbf{p} + v(\mathbf{p}) - \mathbf{q}|$.

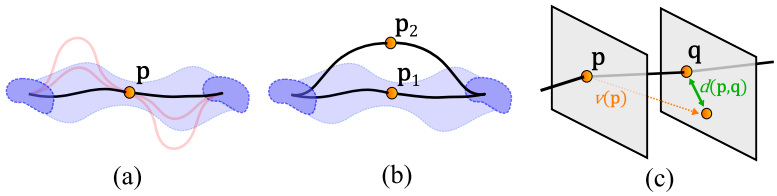

Fig. 3. Computing the cost volume. There are many possible paths through the image stack (shown in red) that connect the 2D segmentations on the first and last slices (shown in dark blue) via $\mathbf{p}$, but there is only one shortest path (shown in black); we set the cost of each pixel in the cost volume to be the length of this path (a). For example, $\mathbf{p}_2$ will be assigned a higher cost than $\mathbf{p}_1$, since the length of its shortest path is longer; this means $\mathbf{p}_2$ is less likely to belong to the neural process of interest than $\mathbf{p}_1$ (b). When computing the length of each path, we model the distance (shown in green) between pixels on adjacent slices as a function of the dense optical flow vectors (shown in orange) between the pixels; in this formulation, paths that agree strongly with the dense optical flow field have very short lengths (c).

To compute our cost volume, we find the minimum distances from each pixel to the 2D segmentations on the first and and last image slices in two distinct passes, using the dynamic programming algorithm proposed by Pan et al. [16]. We set the cost of each pixel to be the sum of both distances, as proposed by Jeong *et al.* [9]. We compute dense optical flow using use the open-source implementation of Farnebäck's algorithm [5] in The OpenCV Library [3].

The interior level sets of our cost volume define soft foreground constraints on our 3D segmentation.

Computing the 3D Segmentation. Once we have obtained hard constraints on the first and last slices of the image stack, and soft constraints on all other slices, we obtain the final 3D segmentation by globally minimizing the following variational segmentation energy [17]:

$$\underset{u}{\operatorname{argmin}} \int_{\mathbf{x}\in\Omega} (g|\nabla^{\mathrm{xy}} u| + |\nabla^{\mathrm{z}} u| + fu)\, d\mathbf{x}, \tag{2}$$

where Ω is the 3D image domain corresponding to the entire image stack, and $\nabla^{\mathrm{xy}} u$ and $\nabla^{\mathrm{z}} u$ are the in-plane and out-of-plane gradients of u, respectively. As in (1), the function u encodes the resulting segmentation, the function g encodes strong in-plane image edges as small values, and the function f encodes constraints on the segmentation. Using the hard and soft constraints computed in the previous sections, we set $f := -\infty$ for hard foreground constraints, $f := \infty$ for hard background constraints, $f := c$ for some scalar value $c \in \mathbb{R}^-$ for soft foreground constraints, $f := c$ for some scalar value $c \in \mathbb{R}^+$ for soft background constraints, and $f := 0$ otherwise. Minimizing (2) results in a smooth 3D segmentation that respects the previously computed constraints, follows the neural process of interest, and aligns with strong in-plane image edges.

Minimizing the 2D and 3D Segmentation Energies. We compute the global minimum of (1) and (2) using the iterative parallel algorithm proposed by

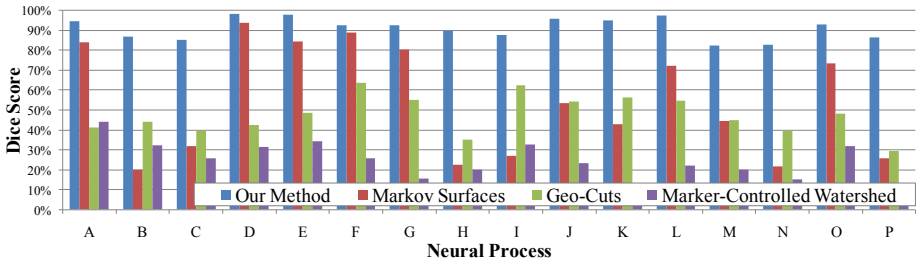

Fig. 4. Accuracy of our method, Markov Surfaces [16], Geo-Cuts [2], and Marker-Controlled Watersheds [6] while segmenting 16 neural processes in an annotated $1024 \times 1024 \times 50$ mouse hippocampus EM image stack.

Unger *et al.* [17, 18]. We begin by reformulating these equations as optimization problems of two variables to preserve their differentiability. In this two-variable formulation, (1) and (2) become strictly convex, so we use a projected gradient descent strategy to obtain a globally optimal solution for the segmentation variable u.

3 Results and Discussion

We evaluated our method, Markov Surfaces [16], Geo-Cuts [2], and Marker-Controlled Watersheds [6] by segmenting 16 neural processes in a $1024 \times 1024 \times 50$ mouse hippocampus image stack for which the ground truth classification of each neural process was known.

We implemented our method in CUDA and C++ on a PC with an Intel Xeon 3 GHz CPU (12GB of memory), and an NVIDIA GTX 480 graphics processor (1.5GB of memory). In all cases, computing each 2D segmentation took at most 3 seconds, computing the cost volume took at most 5 seconds, and computing the final 3D segmentation took at most 10 seconds. Total segmentation times, including all user interaction and computation time, varied between 45 and 70 seconds, with an average of 50 seconds per neural process.

Fig. 4 shows the Dice Scores[1] of all the methods used and neural processes segmented in our evaluation. On average, our method is 68% more accurate than Markov Surfaces [16], 91% more accurate than Geo-Cuts [2], and 263% more accurate than Marker-Controlled Watersheds [6]. Fig. 5 shows the segmentation results for one neural process from Fig. 4 across several 2D image slices.

As indicated in Fig. 5, Marker-Controlled Watersheds tended to under-segment neural processes, since it places segmentation boundaries at local intensity maxima. Geo-Cuts performed reasonably well on slices containing scribble annotations, but generally tended to miss the correct segmentation on other slices, since it does not propagate segmentation constraints through the image stack. Markov Surfaces performed well in regions with well-delineated boundaries, but tended

[1] $\frac{2|G \cap S|}{|G|+|S|}$ for a ground truth set of pixels G and a set of segmented pixels S

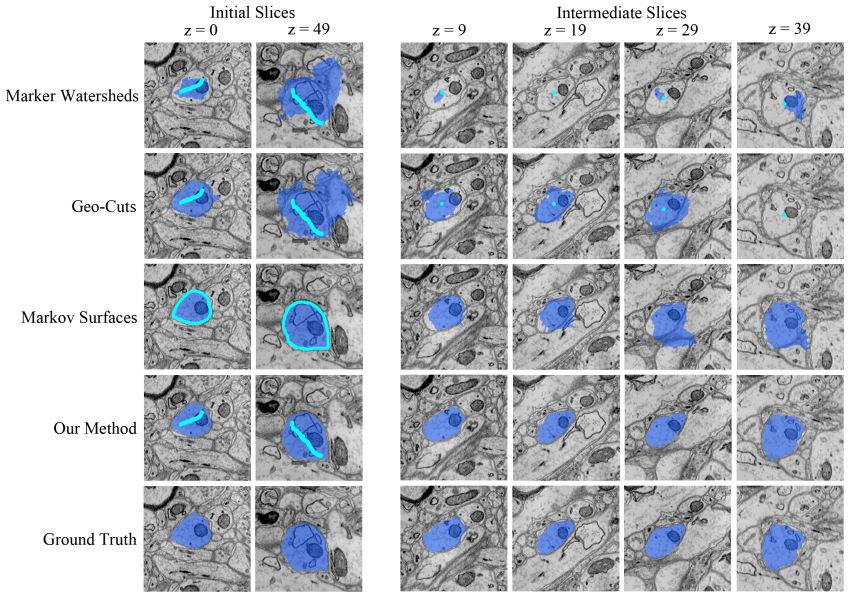

Fig. 5. Segmentation results from our method, Markov Surfaces [16], Geo-Cuts [2], and Marker-Controlled Watersheds [6] on various slices of a $1024 \times 1024 \times 50$ mouse hippocampus EM image stack. Bright blue regions indicate user-provided annotations used to initialize the algorithm, dark blue regions indicate the resulting segmentations.

to diverge heavily from image features, since it relies on Bézier interpolation to produce smooth surfaces and does not enforce 3D geometric constraints on the segmentation. Our method outperformed these methods, due to its ability to robustly propagate segmentation constraints through the image stack and optimally enforce 3D smoothness on the segmentation.

Acknowledgements. This work was partially supported by the Intel Science and Technology Center for Visual Computing, NVIDIA, the National Institute of Health under Grant No. 1P30NS062685-01 and R01 NS020364-23, the Gatsby Charitable Foundation under Grant GAT3036-Connectomic Consortium, and the National Science Foundation under Grant No. PHY-0835713. We also wish to thank Yongsheng Pan for providing us with the Markov Surfaces source code, and Manhee Lee for helping us to collect ground truth data.

References

1. Andres, B., Köthe, U., Helmstaedter, M., Denk, W., Hamprecht, F.A.: Segmentation of SBFSEM volume data of neural tissue by hierarchical classification. In: DAGM Symposium on Pattern Recognition. pp. 1609–1612 (2008)
2. Boykov, Y., Funka-Lea, G.: Graph cuts and efficient N-D image segmentation. Int. J. Comp. Vis. 70 (2006)

3. Bradski, G.: The OpenCV Library. Dr. Dobb's Journal of Software Tools (2000)
4. Chklovskii, D.B., Vitaladevuni, S., Scheffer, L.K.: Semi-Automated reconstruction of neural circuits using electron microscopy. Curr. Opin. Neurobiol. 20(5), 667–675 (2010)
5. Farnebäck, G.: Polynomial Expansion for Orientation and Motion Estimation. Ph.D. thesis, Linköping University, Sweden (2002)
6. Gonzalez, R.C., Woods, R.E.: Digital Image Processing (3rd Edition). Prentice-Hall, Inc. (2006)
7. Jain, V., Bollmann, B., Richardson, M., Berger, D.R., Helmstaedter, M.N., Briggman, K.L., Denk, W., Bowden, J.B., Mendenhall, J.M., Abraham, W.C., Harris, K.M., Kasthuri, N., Hayworth, K.J., Schalek, R., Tapia, J.C., Lichtman, J.W., Seung, H.S.: Boundary learning by optimization with topological constraints. In: IEEE CVPR. pp. 2488–2495 (2010)
8. Jeong, W.K., Beyer, J., Hadwiger, M., Vazquez-Reina, A., Pfister, H., Whitaker, R.T.: Scalable and interactive segmentation and visualization of neural processes in EM datasets. IEEE Trans. Vis. Comp. Graph. 15(6), 1505–1514 (2009)
9. Jeong, W.K., Fletcher, P.T., Tao, R., Whitaker, R.T.: Interactive visualization of volumetric white matter connectivity in DT-MRI using a parallel-hardware Hamilton-Jacobi solver. IEEE Trans. Vis. Comp. Graph. 13(6), 1480–1487 (2007)
10. Jurrus, E., Whitaker, R., Jones, B.W., Marc, R., Tasdizen, T.: An optimal-path approach for neural circuit reconstruction. In: ISBI. pp. 1609–1612 (2008)
11. Kaynig, V., Fuchs, T., Buhmann, J.M.: Geometrical consistent 3D tracing of neuronal processes in ssTEM data. In: Jiang, T., Navab, N., Pluim, J., Viergever, M. (eds.) MICCAI. pp. 209–216 (2010)
12. Kaynig, V., Fuchs, T., Buhmann, J.M.: Neuron geometry extraction by perceptual grouping in ssTEM images. In: IEEE CVPR. pp. 2902–2909 (2010)
13. Lucchi, A., Smith, K., Achanta, R., Lepetit, V., Fua, P.: A fully automated approach to segmentation of irregularly shaped cellular structures in EM images. In: Jiang, T., Navab, N., Pluim, J., Viergever, M. (eds.) MICCAI. pp. 463–471 (2010)
14. Macke, J.H., Maack, N., Gupta, R., Denk, W., Schlkopf, B., Borst, A.: Contour-propagation algorithms for semi-automated reconstruction of neural processes. J. Neurosci. Methods 167(2), 349–357 (2008)
15. Mishchenko, Y.: Automation of 3D reconstruction of neural tissue from large volume of conventional serial section transmission electron micrographs. J. Neurosci. Methods 176(2), 276–289 (2009)
16. Pan, Y., Jeong, W.K., Whitaker, R.T.: Markov surfaces: A probabilistic framework for user-assisted three-dimensional image segmentation. In: PMMIA. pp. 57–68 (2009)
17. Unger, M., Mauthner, T., Pock, T., Bischof, H.: Tracking as segmentation of spatial-temporal volumes by anisotropic weighted TV. In: EMMCVPR. pp. 193–206 (2008)
18. Unger, M., Pock, T., Bischof, H.: Continuous globally optimal image segmentation with local constraints. In: CVWW (2008)
19. Vázquez-Reina, A., Miller, E., Pfister, H.: Multiphase geometric couplings for the segmentation of neural processes. In: IEEE CVPR. pp. 2020–2027 (2009)
20. Venkataraju, K.U., Paiva, A.R.C., Jurrus, E., Tasdizen, T.: Automatic markup of neural cell membranes using boosted decision stumps. In: ISBI. pp. 1039–1042 (2009)