Candidate Sampling for Neuron Reconstruction from Anisotropic Electron Microscopy Volumes

Jan Funke[1,2], Julien N. P. Martel[1], Stephan Gerhard[1,4], Bjoern Andres[2], Dan C. Cireşan[3], Alessandro Giusti[3], Luca M. Gambardella[3], Jürgen Schmidhuber[3], Hanspeter Pfister[2], Albert Cardona[4], Matthew Cook[1]

[1] Institute of Neuroinformatics, UZH/ETH Zürich
[2] School of Engineering and Applied Science, Harvard Universiy
[3] IDSIA, Lugano
[4] HHMI Janelia, Ashburn (VA)

Abstract. The automatic reconstruction of neurons from stacks of electron microscopy sections is an important computer vision problem in neuroscience. Recent advances are based on a two step approach: First, a set of possible 2D neuron candidates is generated for each section independently based on membrane predictions of a local classifier. Second, the candidates of all sections of the stack are fed to a neuron tracker that selects and connects them in 3D to yield a reconstruction. The accuracy of the result is currently limited by the quality of the generated candidates. In this paper, we propose to replace the heuristic set of candidates used in previous methods with samples drawn from a conditional random field (CRF) that is trained to label sections of neural tissue. We show on a stack of *Drosophila melanogaster* neural tissue that neuron candidates generated with our method produce 30% less reconstruction errors than current candidate generation methods. Two properties of our CRF are crucial for the accuracy and applicability of our method: (1) The CRF models the orientation of membranes to produce more plausible neuron candidates. (2) The interactions in the CRF are restricted to form a bipartite graph, which allows a great sampling speed-up without loss of accuracy.

1 Introduction

To study the structure and function of nervous systems, neuroscientists need to image volumes of neural tissue that are large enough to contain complete neural circuits, with high enough resolution to resolve individual synapses. Currently, serial section electron microscopy (EM) is the only technique that meets these requirements [1], resulting in multi-terabyte anisotropic image stacks (that is, volumes with high xy-resolution and low z-resolution) even for small organisms like *Drosophila melanogaster*. The biggest bottleneck in the processing of these image stacks is the time needed to manually reconstruct the neuron morphologies. Addressing this issue, automatic reconstruction methods for anisotropic volumes became the subject of a vivid branch of research at the intersection between computer vision and neuroscience [2–10].

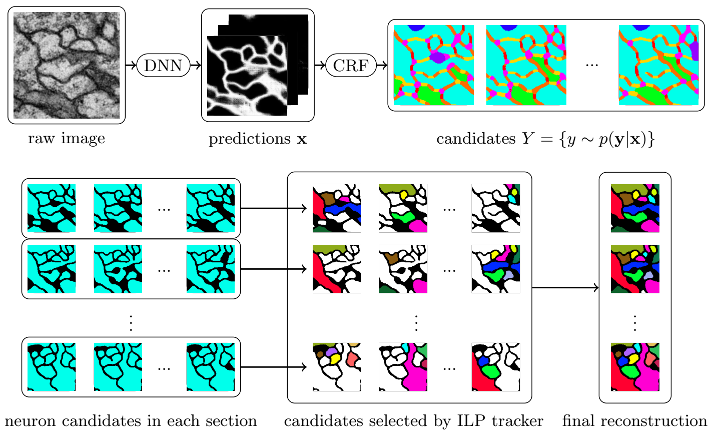

DNN
CRF
raw image
predictions $\mathbf{x}$
candidates $Y = \{y \sim p(\mathbf{y}|\mathbf{x})\}$
neuron candidates in each section
candidates selected by ILP tracker
final reconstruction

Fig. 1: Overview of the proposed method. **Top:** For each section, 2D neuron candidates are generated by sampling from a CRF. **Bottom:** An integer linear program (ILP) tracker is used to find a consistent subset of the candidates for the whole stack at once.

The de facto standard for anisotropic neuron reconstruction methods consists of two steps [7, 8, 10]: In the first step, a set of possible 2D neuron candidates is generated for each section of the stack individually. These candidates are obtained by identifying membranes using predictions of a local image patch classifier like a deep neural network (DNN) [9] or a random forest [7, 8, 10]. Due to ambiguities in the data and imprecisions of the predictions, many plausible and possibly contradictory candidates are extracted for each section to increase the chance that the correct candidates are among them. In the second step, the 2D neuron candidates of the whole stack are fed into an integer linear program (ILP) tracker, which connects them across adjacent sections. The ILP tracker ensures that a non-contradictory subset of candidates is chosen and that the candidates are connected to optimize a global criterion like a smooth continuation.

The accuracy and efficiency of the reconstruction is limited by the quality of the generated 2D neuron candidates. It has been noted that the generation of 2D neuron candidates is responsible for about 50% of the final error [8]. Although the ILP tracker can ignore wrong candidates to some extent, it can not fantasize missing correct candidates. Thus, it is important to generate enough candidates with high variation to make sure that the correct candidates are among them. However, having too many incorrect candidates increases the computational overhead and may lead to spurious results. Therefore, a careful generation of candidates is crucial for accuracy and tractability.

Our Approach We are proposing a novel method to generate 2D neuron candidates that greatly improves the reconstruction accuracy. The key of our method is the generation of 2D neuron candidates as samples from a pairwise conditional random field (CRF) that is designed and trained to label 2D sections of neural tissue. An overview of the proposed method is shown in Fig. 1.

Furthermore, we introduce two extensions for pairwise CRFs for image labeling and show that they are essential for the accuracy and applicability of our method: (1) Our CRF explicitly models the orientation of certain labels. We use this to learn shape-related priors for membranes, which are usually thin and elongated. This helps to produce more plausible section labelings, and thus better 2D neuron candidates. (2) We restrict the interactions in the CRF to form a bipartite graph. With this trick, Gibbs sampling [11] on the model can be parallelized and carried out on a GPU with a speed-up factor of 39 compared to a single core CPU implementation. This allows us to model a large number of neighbor interactions for each pixel (improving accuracy) and to use a rich set of biologically relevant labels (cell interior, mitochondria, glia cells, synapses, and different orientations of membranes) while still being fast enough to process large amounts of data.

The following section gives details about our CRF that we use to generate the 2D neuron candidates. In section 3, we show results on a stack of *Drosophila melanogaster* neural tissue, where candidates generated with our method lead to 30% fewer final errors than competing candidate generation methods.

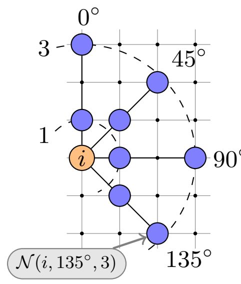
2 2D Neuron Candidate Sampling

To generate 2D neuron candidates, we repeatedly draw samples from a labeling distribution on each section individually and transform them into binary neuron/membrane segmentations. Let $\Omega \subset \mathbb{N}^2$ be the pixel domain of a single EM section. We model the distribution of labelings $\mathbf{y} = \left(y^{(i)} \in K \mid i \in \Omega\right)$ of assigning each pixel in Ω to a label of a discrete set K with a conditional random field (CRF). For the possible values in K, we distinguish between *oriented* labels K_Φ (for membranes) and *non-oriented* labels K_N (for mitochondria, glia cells, *etc.*). Each $k_\alpha \in K_\Phi$ represents a label k at a certain discrete orientation $\alpha \in \Phi$.

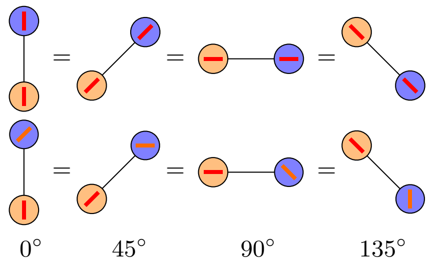
The CRF is conditioned on pixel-wise predictions $\mathbf{x} = \left(\mathbf{x}^{(i)} \in \mathbb{R}^F \mid i \in \Omega\right)$ with vectors of size F. We write $y^{(i)}$ or $\mathbf{x}^{(i)}$ to refer respectively to the label or prediction vector of pixel $i \in \Omega$ and $x_f^{(i)}$ to refer to the f$^{\text{th}}$ prediction component of $\mathbf{x}^{(i)}$. The lateral interactions of each location in the CRF are modeled with pairwise factors to neighbors in different directions Φ (as for the oriented labels) and distances $\Delta = \{d_1, \ldots, d_D\}$, approximated to the closest neighbor on the pixel grid (see Fig. 2a for an illustration). We write $\mathcal{N}(i, \alpha, d)$ to denote the closest grid neighbor of i in direction α and distance d. We achieve rotation equivariance by re-using factors of same distance for interactions in different directions to directly model the rotation invariant statistics of our data. Furthermore, the CRF is homogeneous, *i.e.*, the same factors are used at every location. Formally, we model the labeling distribution $p(\mathbf{y}|\mathbf{x})$ as

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp[-\sum_{i\in\Omega} \langle \boldsymbol{w}^{y^{(i)}}, \mathbf{x}^{(i)} \rangle - \sum_{i\in\Omega} \sum_{\alpha\in\Phi} \sum_{d\in\Delta} R_{\alpha,d}(y^{(i)}, y^{(\mathcal{N}(i,\alpha,d))})], \quad (1)$$

where $Z(\mathbf{x})$ is the partition function, the data term (first sum) is a linear combination of the components of the prediction vectors with weights $\boldsymbol{w}^k \in \mathbb{R}^F$ for



(a) neighborhood



(b) rotation invariance

Fig. 2: (a) Neighbors (blue) of a single pixel (i, orange) for orientations $\Phi = \{0°, 45°, 90°, 135°\}$ and distances $\Delta = \{1, 3\}$. Neighbors are approximated to the closest pixel on the grid in the given direction and distance. (b) Illustration of the rotation invariant interactions. The values of rotated interactions are equal for accordingly rotated labels.

each label $k \in K$, and the prior (second sum) models the pairwise interactions of each location to its neighbors. For that, $R_{\alpha,d}$ determines the costs for the joint labeling of i and its approximate neighbor $\mathcal{N}(i, \alpha, d)$ in direction α at distance d. These costs are shared across different orientations. Let k^α denote the rotated version of label k by α degrees: oriented labels change their orientation subscript and non-oriented labels do not change at all. Defining $R_{\alpha,d}(k,l) = \tilde{R}_d\left(k^{-\alpha}, l^{-\alpha}\right)$ ensures that the resulting CRF is rotationally equivariant by treating every pairwise interaction in the same way as the 0° interaction (see Fig. 2b for an illustration). Finally, $\tilde{R}_d$ is a lookup table with entries $\tilde{R}_d\left(k,l\right) = v_d^{k,l}$ for the joint labeling of neighboring pixels with distance d. The parameters $\mathbf{v} = \{v_d^{k,l}\}$ and $\boldsymbol{w} = \{\boldsymbol{w}^k\}$ of our model are obtained by maximum likelihood learning [12].

Parallelized Sampling We use Gibbs sampling [11] to draw the samples from $p(\mathbf{y}|\mathbf{x})$ that are needed for the training and generation of 2D neuron candidates. To tackle the slow convergence properties of Gibbs sampling, we restrict the interactions in the CRF to form a bipartite graph. For that, we divide the image domain Ω into "odd" and "even" locations, following a checkerboard pattern on the pixel grid. By modifying $\mathcal{N}$ to find the closest neighbor *of opposite parity* in the given direction and distance, we obtain a bipartite CRF. We write $\mathbf{y}^{(O)}$ and $\mathbf{y}^{(E)}$ to refer to the labeling of the pixels in Ω_O and Ω_E, respectively. It follows that $p(\mathbf{y}^{(E)}|\mathbf{y}^{(O)}, \mathbf{x}) = \prod_{i\in\Omega_E} p(y^{(i)}|\mathbf{y}^{(O)}, \mathbf{x})$, and $p(\mathbf{y}^{(O)}|\mathbf{y}^{(E)}, \mathbf{x}) = \prod_{i\in\Omega_O} p(y^{(i)}|\mathbf{y}^{(E)}, \mathbf{x})$, *i.e.*, labels in one partition are conditionally independent given the labels in the other half [13]. Samples $y^{(i)} \sim p(y^{(i)}|\mathbf{y}^{(E)})$ for $i \in \Omega_O$ and $y^{(i)} \sim p(y^{(i)}|\mathbf{y}^{(O)})$ for $i \in \Omega_E$ can be drawn independently and in parallel. We exploit this property by sampling a whole section in two half-steps, one for $\mathbf{y}^{(E)}$ and one for $\mathbf{y}^{(O)}$, each of which is carried out with our parallelized GPU implementation (source code available at http://github.com/funkey/prim).

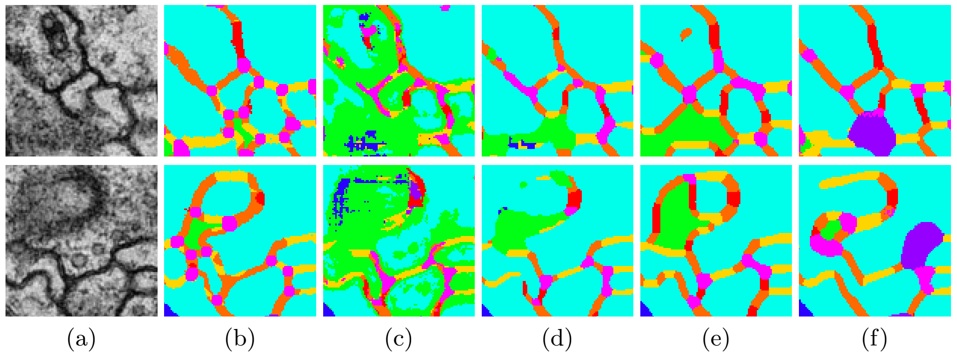


Fig. 3: Examples of samples drawn from our CRF for ambiguous cases. The used labels are membrane (four orientations |,/,—,\ and junctions ●), cell interior (●), mitochondria (●), glia (●), and synapse (●). (a) shows the raw images, (b) the ground-truth labelings, (c) the plain DNN prediction, (d-f) a representative sample from CRFs D_1, D_4, and D_8, respectively. See Section 3 for details.

3 Results

Setup We evaluate the performance of our 2D neuron candidate generation method on two stacks of 20 EM images of size 1024 × 1024 from *Drosophila melanogaster* larva neuropil with 4.6nm xy-resolution and 50nm section thickness [14]. These stacks have been selected to capture well the typical variations found in EM images. The first stack was used to train the DNN architecture proposed in [9] to predict the pixel labels (shown in Fig. 3) that are used as features $\mathbf{x}$ in the CRF. The second stack, for which we manually generated ground truth, is used to compare different candidate generation methods. To investigate the effect of different neighborhood sizes, we created three instances of our model with different distance sets: model D_1 (a baseline of our CRF) uses $\Delta = \{1\}$, D_4 uses $\Delta = \{1, 5, 9, 15\}$, and D_8 uses $\Delta = \{1, 3, 5, 7, 9, 15, 21, 31\}$. All models use the same labels (shown in Fig. 3) and the same set of rotations $\Phi = \{0°, 45°, 90°, 135°\}$. From each model, we drew 20 samples per section to generate the 2D neuron candidates. Each sample is the last labeling after 100 Gibbs iterations on the whole section, starting from a random initialization.

Error Measure We report the accuracy of the reconstruction in terms of an edit distance proposed in [8], since this directly reflects the amount of time needed to fix it. The errors are reported in four categories: "FP" (false positives) are spuriously detected neurons in a section, "FN" (false negatives) are missed neurons in a section, "FS" (false splits) missed links between neurons across sections, and "FM" (false merges) are spurious links across sections. Details about this error measure are given in the supplemental material.

Comparison We compare our model instances D_1, D_4, and D_8 to the current state of the art in neuron candidate generation: A series of graphcuts (GraphCuts), applied on the *membrane* predictions of the DNN, as proposed in [8], and

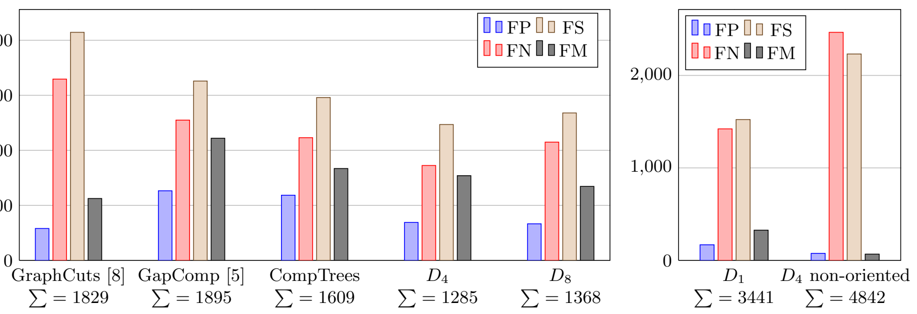


Fig. 4: Reconstruction errors for different 2D neuron candidate generation methods (a). Results are given as false positives (FP), false negatives (FN), false splits (FS), and false merges (FM), see text for details. Our method D_4 produces 30% less errors compared to the best competing approach (GraphCuts). The importance of a rich neighborhood and oriented membranes in the proposed model is shown in (b): Decreasing the neighborhood radius to one (D_1) or ignoring membrane orientations (D_4 non-oriented) dramatically sacrifices reconstruction accuracy.

gap completion (GapComp) [5] as proposed in [10]. As a baseline, we also generate candidates from component trees (CompTrees) [15], extracted from the same predictions. We reconstructed neurons in the test stack for each candidate generation method using the publicly available ILP tracker Sopnet [8]. The trainable parts of this tracker have been trained and validated for each method individually on the first 10 sections of the test stack. The test results on the last 10 sections are shown in Fig. 4. Model D_4 provides by far the best result, improving the error by 30% compared to the best competing approach (GraphCuts), which is already outperformed by our baseline model (CompTrees). Surprisingly, the larger neighborhood of model D_8 does not lead to better results.

Model Properties To show the importance of oriented membrane labels, we trained and evaluated a version of D_4 with non-oriented membrane labels (right bars in Fig. 4b). Due to undersegmentation, the model proposed only a few large candidates and thus missed a lot of neurons (high FN and FS). To investigate the effect of the bipartite restriction of our CRF, we also implemented a non-bipartite version of model D_4. The reconstruction errors differ by 1% in favour of the bipartite version and may well be attributed to sampling noise.

Inference Time Tested on a NVIDIA Quadro 4000, we were able to draw 100 complete samples of size 1024×1024 from D_4 using our GPU implementation in $4.4s$. A single core CPU Gibbs sampler implementation took $174s$ on a Intel Xeon CPU at 3.47 GHz to achieve the same, resulting in a speed-up factor of 39.

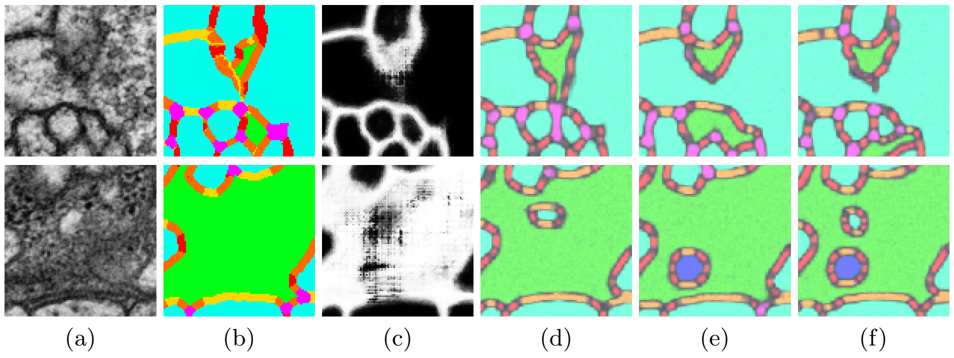


Fig. 5: Demonstration of the candidate generation capabilities of model D_4 (for label legend see caption of Fig. 3). (a) shows the raw image, (b) the ground-truth labeling and (c) the prediction for the class *membrane*. Images (d-f) show three different samples drawn from D_4.

4 Discussion

We showed that replacing fixed sets of 2D neuron candidates with samples drawn from a distribution increases the reconstruction accuracy. Going beyond neuron reconstruction, this scheme might also be applicable to other approaches that rely on the quality of initial candidates, such as super-pixel based algorithms for image segmentation.

In our case, an interesting aspect of our method is the rich labeling in the samples. Besides membrane locations, they also indicate the locations of mitochondria, synapses, and glia cells. These labels are not only of biological relevance, but could also be used to improve the reconstruction accuracy by, for instance, exploiting the fact that mitochondria are surrounded by neuron and synapses are separating neurons.

An open question in our method is how many samples need to be drawn to obtain good results with minimal computational overhead. An interesting solution might be to start with a few initial samples and draw more on demand for locations that are unlikely according to higher level priors, for example where there is a sudden change in direction or an unexpected end of a neural process.

Acknowledgements

We thank Verena Kaynig for training and creating candidates for GapComp. This work was funded by the SNF grant CRSII3 130470.

References

1. Cardona, A.: Towards Semi-Automatic Reconstruction of Neural Circuits. Neuroinformatics **11** (2012) 31–33

2. Yuriy, Mishchenko: Automation of 3D reconstruction of neural tissue from large volume of conventional serial section transmission electron micrographs. Journal of Neuroscience Methods **176**(2) (2009) 276 – 289
3. Jurrus, E., Paiva, A.R., Watanabe, S., Anderson, J.R., Jones, B.W., Whitaker, R.T., Jorgensen, E.M., Marc, R.E., Tasdizen, T.: Detection of neuron membranes in electron microscopy images using a serial neural network architecture. Medical Image Analysis **14**(6) (2010) 770 – 783
4. Kaynig, V., Fuchs, T., Buhmann, J.M.: Geometrical Consistent 3D Tracing of Neuronal Processes in ssTEM Data. In: Proceedings of the Conference on Medial Image Computing and Computer-Assisted Intervention. Volume 6362 of Lecture Notes in Computer Science., Springer Berlin / Heidelberg (2010) 209–216
5. Kaynig, V., Fuchs, T., Buhmann, J.M.: Neuron Geometry Extraction by Perceptual Grouping in ssTEM Images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Los Alamitos, CA, USA (2010) 2902–2909
6. Vitaladevuni, S.N.P., Basri, R.: Co-Clustering of Image Segments Using Convex Optimization Applied to EM Neuronal Reconstruction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2010) 2203 – 2210
7. Vazquez-Reina, A., Huang, D., Gelbart, M., Lichtman, J., Miller, E., Pfister, H.: Segmentation Fusion for Connectomics. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), Barcelona, Spain, IEEE (2011)
8. Funke, J., Andres, B., Hamprecht, F.A., Cardona, A., Cook, M.: Efficient Automatic 3D-Reconstruction of Branching Neurons from EM Data. In: CVPR. (2012)
9. Ciresan, D.C., Giusti, A., Gambardella, L.M., Schmidhuber, J.: Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images. In: NIPS. (2012)
10. Kaynig, V., Vazquez-Reina, A., Knowles-Barley, S., Roberts, M., Jones, T.R., Kasthuri, N., Miller, E., Lichtman, J., Pfister, H.: Large-Scale Automatic Reconstruction of Neuronal Processes from Electron Microscopy Images. ArXiv e-prints (March 2013)
11. Geman, S., Geman, D.: Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. IEEE Transactions on Pattern Analysis and Machine Intelligence **6 (6)** (1984) 721–741
12. Sutton, C., McCallum, A.: An Introduction to Conditional Random Fields for Relational Learning. Technical report, Department of Computer Science University of Massachusetts (2006)
13. Gonzalez, J., Low, Y., Gretton, A., Guestrin, C.: Parallel Gibbs Sampling: From Colored Fields to Thin Junction Trees. In: In Artificial Intelligence and Statistics (AISTATS), Ft. Lauderdale, FL (May 2011)
14. Gerhard, S., Funke, J., Martel, J., Cardona, A., Fetter, R.: Segmented anisotropic ssTEM dataset of neural tissue (2013) http://dx.doi.org/10.6084/m9.figshare.856713.
15. Jones, R.: Component Trees for Image Filtering and Segmentation. In: Proceedings of the 1997 IEEE Workshop on Nonlinear Signal and Image Processing. Mackinac Island. (1997)