Efficient Automatic 3D-Reconstruction of Branching Neurons from EM Data

Jan Funke[1] Bjoern Andres[2] Fred A. Hamprecht[2] Albert Cardona[1] Matthew Cook[1]

[1]Institute of Neuroinformatics
University Zurich/ETH Zurich
http://www.ini.uzh.ch

[2]HCI/IWR
University of Heidelberg
http://hci.iwr.uni-heidelberg.de

Abstract

We present an approach for the automatic reconstruction of neurons from 3D stacks of electron microscopy sections. The core of our system is a set of possible assignments, each of which proposes with some cost a link between neuron regions in consecutive sections. These can model the continuation, branching, and end of neurons. The costs are trainable on positive assignment samples. An optimal and consistent set of assignments is found for the whole volume at once by solving an integer linear program. This set of assignments determines both the segmentation into neuron regions and the correspondence between such regions in neighboring slices. For each picked assignment, a confidence value helps to prioritize decisions to be reviewed by a human expert.

We evaluate the performance of our method on an annotated volume of neural tissue and compare to the current state of the art [26]. *Our method is superior in accuracy and can be trained using a small number of samples. The observed inference times are linear with about 2 milliseconds per neuron and section.*

1. Introduction

Neuroscientists are currently imaging multi-terabyte volumes of neural tissue for the purpose of reconstructing neuronal circuitry with synaptic resolution [2, 5, 6, 8], but this unfortunately requires an impractical amount of time to be spent on manual labeling. This young field, often termed "connectomics" [20], challenges the computer vision and machine learning community to develop accurate and efficient techniques for automatic neuronal reconstruction. The accuracy requirement is extremely high: a fraction of a percent of error results in unusable neuronal circuitry wiring diagrams [7, 11].

In this paper we focus on the problem of segmenting neural tissue in anisotropic volumes with high x- and y-resolution but low z-resolution, as obtained by serial section electron microscopy (EM) imaging procedures [8]. Due to this anisotropy, often also accompanied by registration errors, we prefer to treat the image data as a collection of 2D slices instead of a continuous 3D volume.

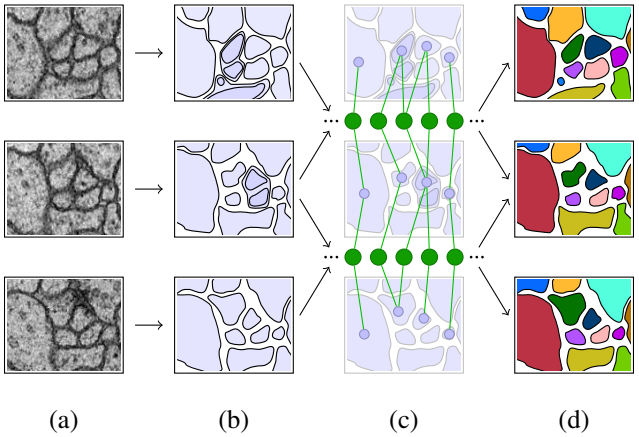


Figure 1. **Overview of our approach.** For all slices in the volume (a), different segmentation hypotheses are created (b). All possible assignments between these hypotheses are enumerated (c) and, according to assignment specific costs (obtained by a trainable classifier), a global optimum of both segmentations and assignments is found (d).

To begin, we generate different segmentation hypotheses for each slice of the volume individually, *i.e.*, without consideration of spatial context in the z-direction (Fig. 1). This is achieved by a sequence of parameterized graph-cut segmentations (see Sec. 2). Then, all possible assignments of these segmentation hypotheses, between each pair of consecutive slices, are enumerated and represented by binary assignment variables. Each possible continuation, branching, and end of a neuron is explicitly represented by an assignment variable of its own (see Sec. 3). To keep the number of variables low we discard assignments between candidates with an x-y-displacement that is above a certain threshold, assuming a coarse registration of the volume [24]. An assignment cost function that is pre-trained on positive samples is giving the costs for selecting an assignment variable. The final segmentations of the slices and

assignments between them are jointly found as the optimal solution to an *integer linear program* (ILP).

Related Work Several authors have contributed to the problem of neural reconstruction from electron microscopy (EM) data for isotropic volumes [1, 9, 10, 12, 21] as well as for anisotropic volumes [14–16, 25–29]. Although the majority of these publications focus on the detection of neuron boundaries for direct reconstruction [1, 9–11, 14–16, 25, 26, 28, 29], there have also been successful efforts in the detection of cell organelles like mitochondria [21], the increase of depth resolution in anisotropic volumes [27], and the invention of meaningful low-level feature detectors [26], all aimed to facilitate the detection of neurons.

Most approaches for the processing of anisotropic volumes are based on an oversegmentation that is carried out for each slice individually and then merged within and between the slices [25, 26, 28, 29]. On the other hand, it has also been shown that the direct segmentation of neuron membranes is a possible approach: either by using a series of neural networks [14] or by extension of the graph-cut formulation with a "good-continuation" term that is also incorporating context from adjacent slices [16]. However, the joint segmentation of several images is a promising technique to increase the overall accuracy [23]. Merging of segments found between (and possibly within) slices is a special case of this technique and a common problem for neuron reconstruction that has mainly been solved approximately, for instance, by problem relaxations [28] or greedy assignment strategies [15, 29]. A noteworthy exception to these solutions, which is also the closest one to our approach, solves the merging problem optimally [26]. As in that work, we enumerate possibly contradictory segmentation hypotheses in individual slices, similar to the region selection approach in [19]. The hypotheses in [26] are fused in pairs between two slices, while our work also allows split and merge assignments. A simple linear function of a few weighted features of the involved image patches is used in [26] to compute the cost of accepting a pair, while we use a Random Forest classifier on many features. The resulting optimization problem of which assignments to accept is solved efficiently by an ILP that ensures consistency of the resulting segmentation. Our work goes beyond [26] by handling branching of neurons and providing confidence feedback. The approach in [26] is also restricted to a small number of features for computing the costs of an assignment: since there is no obvious way to learn the weight of each feature, the difficulty of tuning the weights limits the number of features. In contrast, we use Random Forest classifiers, which are well known to be able to handle a large number of features.

Contributions We provide a method for the reconstruction of branching neurons, based on assignments between slices. The cost function for assignments is trainable on a few samples. We give a confidence measure that can be used for directing an expert's attention during proof-reading and semi-automated segmentation correction. Training and inference are fast enough to respond to corrections interactively. We implemented our approach as an open-source Fiji[1] plug-in, which is available under http://ini.ch/~funke/research/sipnet.

2. Segmentation Hypotheses

In this section we describe our method to extract segmentation hypotheses from individual slices of the volume using a sequence of graph-cuts. We also show how the number of considered segmentation hypotheses can be reduced without altering the topological properties of the possible results.

Practically, in order to identify neurons in slices of a volume, we face a binary segmentation problem: it suffices to detect *membrane* pixels versus inner *neuron* pixels. Every connected component labeled as *neuron* would be the cross-section of one neuron. To the best of our knowledge, there exists no reliable method to discriminate between *neuron* and *membrane* pixels in a single slice. In situations with high ambiguity, even human experts cannot unequivocally draw boundaries without inspection of adjacent slices.

However, direct incorporation of spatial context in the segmentation task is impractical. The presence of registration errors – even in the scale of the width of a membrane – makes it very hard to model the influence of one pixel to its partners in adjacent slices [16]. Therefore, we propose to extract a set of *possible* connected components that might represent neuron sections. These connected components, that we will call segmentation hypotheses in the following, are allowed to overlap and thereby contradict each other. Thus, we increase the scale of ambiguity from pixels to larger regions. It remains to find consistent subsets of segmentation hypotheses. In Sec. 3 we show how this can be done with consideration of all slices at once.

2.1. Hypotheses Extraction

The segmentation hypotheses are extracted for each slice of the volume independently by performing a series of graph-cuts to optimize a parameterized energy term [3]. Given a field of per-pixel local features x, a binary segmentation y is obtained by minimizing

$$E(y,x) = \sum_{i\in\Omega} (D(x_i,y_i) + \lambda_N y_i) + \sum_{i,j\in\mathcal{N}} \lambda_S (1-\delta_{y_i=y_j}). \tag{1}$$

Here, $\Omega \subset \mathbb{R}^2$ is the image domain and $D(x_i,y_i) = -log\, p(y_i|x_i)$ is the log-likelihood of pixel $i \in \Omega$ belonging

[1] Fiji (Fiji Is Just ImageJ), http://fiji.sc

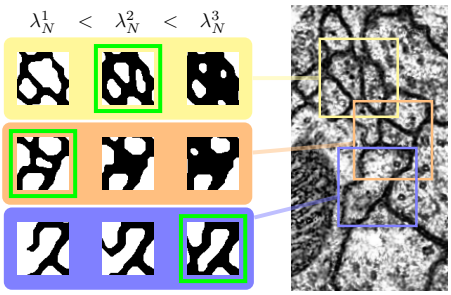


Figure 2. Segmentation results of nearby image regions for different values of λ_N. This parameter determines how many of the pixels are being labeled as *neuron* (white). The desired segmentation result for each case is highlighted in green.

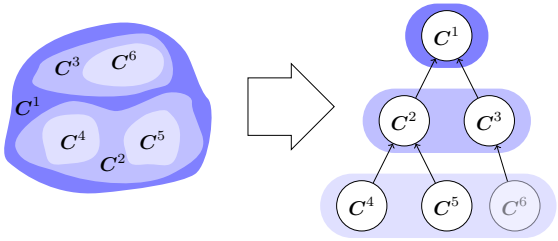


Figure 3. Visualisation of the segmentation hypotheses extraction. For different values of the prior parameter λ_N (shades of blue) connected components of the segmentation are found (left side). The subset relation of these connected components define the component tree (right side). The candidate $\boldsymbol{C}^6$ gets removed since it is the only child of its parent.

to *membrane* or *neuron*, as given by a pre-trained Random Forest classifier [4]. The set $\mathcal{N} \subset \Omega \times \Omega$ contains all pairs of 8-connected neighboring pixels. The second term in the exponent ensures smoothness of the segmentation.

The parameter λ_N is a prior on the expected number of pixels assigned to *neuron* and λ_S controls the influence of the smoothness term. Finding an optimal set of parameters is a non-trivial task and one cannot expect a fixed set of parameters to perform well on all images [18]. We claim that a fixed set of parameters cannot even be expected to perform well on all areas of one image. Therefore, we enumerate several local segmentation hypotheses by variation of λ_N (Fig. 2). This can be done efficiently by several warm-started graph-cuts [17], from which we obtain a series of segmentations. Each segmentation consists of a set of connected components $\boldsymbol{C}^i \subset \Omega$ that are labelled as *neuron*. Each of these components is considered as one segmentation hypothesis. As λ_N decreases, these components can grow and merge, thus establishing a tree-shaped subset hierarchy, the so-called *component tree* [13] (Fig. 3). Let $\mathcal{C}^z$ denote the set of all segmentation hypotheses of slice z. Any consistent subset $\mathcal{S}^z \subset \mathcal{C}^z$ of these hypotheses yields a valid segmentation of this slice. A subset is consistent if none of the containing components overlap, *i.e.*, $\boldsymbol{C}^i \cap \boldsymbol{C}^j = \varnothing$ for all $\boldsymbol{C}^i, \boldsymbol{C}^j \in \mathcal{S}^z$ with $i \neq j$.

2.2. Downsampling of Component Trees

To reduce the number of segmentation hypotheses, we propose to discard segmentation hypotheses that are already well represented by others and do not introduce a new interpretation of the image. In particular, we are not interested in *only children* of the component trees, *i.e.*, components that are the only child of their parent (Fig. 3). These segmentation hypotheses are the only subset of their parent and therefore carry the same information (there is a neuron) on a smaller set of pixels. In other words, if there are different conflicting segmentation hypotheses with the same topological properties, we choose to consider the biggest one only.

The effect of this downsampling is that the average distance between neighboring segmentation hypotheses is reduced – a fact that makes sense, considering that membranes are fairly thin compared to the diameter of neurons.

3. Assignment Model

In this section we describe our model for possible assignments of the segmentation hypotheses across slices. We introduce binary assignment variables and costs for selecting them. We show how we ensure the consistency of any solution with linear constraints and how to find the optimal solution with an ILP. In addition, we present our training method for the assignment costs, as well as our confidence measure that can be used to evaluate the solution.

3.1. Assignment Variables

For each possible assignment of a segmentation hypothesis in one slice to a hypothesis in the previous or next slice, we introduce one binary assignment variable. This variable is set to 1 if the involved hypotheses and their mutual assignment are accepted.

Let m be the number of all possible assignments. A binary vector $\mathbf{a} \in \{0,1\}^m$ of assignment variables is created similarly to the method proposed in [22]. Each possible continuation of a segmentation hypothesis $\boldsymbol{C}^i$ in slice z to $\boldsymbol{C}^j$ in slice $z+1$ is represented by a variable $a^{i \to j}$. A split of $\boldsymbol{C}^i$ in slice z to $\boldsymbol{C}^j$ and $\boldsymbol{C}^k$ in slice $z+1$ is represented as $a^{i \to j,k}$. Similarly, each possible merge is encoded as $a^{i,j \to k}$. Appearances and disappearances of neurons are encoded as assignments to a special end node $\boldsymbol{E}$, i.e., for each hypothesis $\boldsymbol{C}^i$ we introduce two variables $a^{i \to \boldsymbol{E}}$ and $a^{\boldsymbol{E} \to i}$. For the possible assignments, only components within a threshold distance θ_D to each other are considered. Thus, the number of assignment variables is linear in the number of segmentation hypotheses. See Fig. 4 for examples of assignments of a single segmentation hypothesis.

For each assignment variable we define costs representing the compatibility of the involved segmentation hypotheses. For that, a vector $\mathbf{c} \in \mathbb{R}^m$ is constructed. The costs for

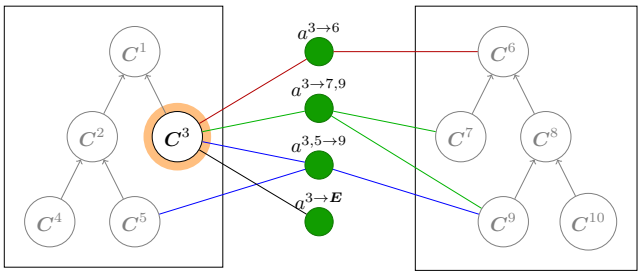
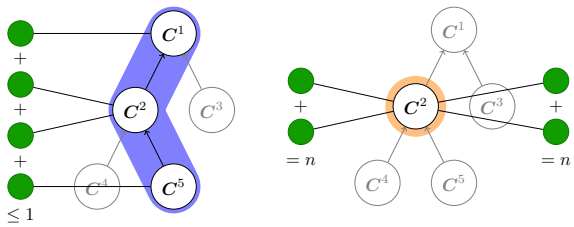


Figure 4. Examples of the four outgoing assignment categories for a single segmentation hypothesis (orange): A continuation (red) is modelled for each hypothesis in the next slice that is within a threshold distance. Possible splits (green) and merges (blue) are enumerated for neighboring hypotheses in the respective slices. The possible sources or targets of splits and merges are segmentation hypotheses within a threshold distance. The disappearance of a neuron is represented by a single assignment (black).



(a) hypothesis consistency (b) explanation consistency

Figure 5. Visualization of the two types of consistency constraints. The hypothesis consistency (a) ensures that no pixel is assigned to more than one segmentation hypothesis: For each path of the component tree (blue), the sum of all *incoming* assignment variables (gray) has to be at most one. The explanation consistency (b) ensures a continuous sequence of assignments: For each segmentation hypothesis (orange), the sum of all *incoming* assignment variables (from the previous image) has to be equal to the sum of all *outgoing* assignment variables (to the next image). The incoming or outgoing assignment variables for a component are all assignment variables that have the component as target or source, respectively.

one-to-one assignments are modelled as:

$$c^{i\to j} = A_C(\boldsymbol{C}^i, \boldsymbol{C}^j) + \theta_S S(\boldsymbol{C}^{ij}). \tag{2}$$

Here, we write $\boldsymbol{C}^{ij}$ as a shorthand for $\boldsymbol{C}^i \cup \boldsymbol{C}^j$. The term $A_C(\boldsymbol{C}^i, \boldsymbol{C}^j)$ is the negative log-probability of a continuation, as obtained by a Random Forest classifier (see Sec. 3.3 for details). The term $S(\boldsymbol{C})$ gives the cost of assigning all pixels of $\boldsymbol{C}$ to *neuron* as given by Eq. 1, *i.e.*,

$$L(\boldsymbol{C}) = \sum_{j\in\boldsymbol{C}} (D(x_j,0) - D(x_j,1)) + \sum_{\substack{j,k\in\mathcal{N};\\ j\in\boldsymbol{C},\ k\notin\boldsymbol{C}}} \lambda_S(1-\delta_{y_i=y_j}). \tag{3}$$

In a similar way, we define the costs for splits:

$$c^{i\to j,k} = A_B(\boldsymbol{C}^i, \boldsymbol{C}^j, \boldsymbol{C}^k) + \theta_S S(\boldsymbol{C}^{ijk}), \tag{4}$$

where $A_B(\boldsymbol{C}^i, \boldsymbol{C}^j, \boldsymbol{C}^k)$ is the negative log-probability of a branching. The merge cases are defined analogously and the appearance or disappearance of neurons cause the following costs:

$$c^{i\to \boldsymbol{E}} = c^{\boldsymbol{E}\to i} = A_E(\boldsymbol{C}^i) + \theta_S S(\boldsymbol{C}^i). \tag{5}$$

The segmentation likelihood weight θ_S is a parameter of the model.

3.2. Consistency Constraints

A solution to both the segmentation of slices and the assignments between the segments can be found by finding the subset of assignment variables with minimal costs: every segmentation hypotheses that is involved in a selected assignment is taken to be a valid segmentation of a neuron.

However, overlapping segmentation hypotheses are contradictory and thus impose constraints on the assignment variables that can be picked. In particular, we have to ensure that for every path $\boldsymbol{P}$ in every component tree (*i.e.*, for every set of overlapping segmentation hypotheses) the number of assignments that connect them in any direction (to the previous or next slice) is at most 1. We call this the *hypotheses consistency* constraint. Furthermore, we have to ensure that a segmentation hypotheses that was picked from an assignment to a previous slice will also be picked by an assignment to the next slice. We call this the *explanation consistency* constraint, since it ensures a continuous sequence of assignments, one continuing where the previous one ended. See Fig. 5 for a visualisation of these constraints. Both types of constraints can be expressed by the following linear (in)equalities:

$$\sum_{i\in\boldsymbol{P}} \sum_{a\in\mathbf{a}^{\to i}} a \leq 1 \qquad \forall \boldsymbol{P}\in\mathcal{P} \tag{6}$$

$$\sum_{a\in\mathbf{a}^{\to i}} a - \sum_{a'\in\mathbf{a}^{i\to}} a' = 0 \qquad 1\leq i\leq n \tag{7}$$

where $\mathcal{P}$ is the set of all paths in every component tree and n the number of all segmentation hypotheses. The sets $\mathbf{a}^{\to i}$ and $\mathbf{a}^{i\to}$ denote all assignment variables that involve segmentation hypotheses $\boldsymbol{C}^i$ to the previous or next slice, respectively.

The optimal solution can now be found by solving an ILP that minimizes $\mathbf{c}^\mathsf{T}\mathbf{a}$ subject to the linear constraints 6 and 7.

However, with this formulation, the optimal solution would favour segmentation hypotheses that are high in the component tree, since this would minimize the number of

assignments. Therefore, we scale the costs for each possible assignment by the sum of leaves that are under the involved segmentation hypotheses. The intuitive meaning of this strategy is the following: for an assignment involving a *high* segmentation hypotheses to be favoured, its cost has to be less than the *average* cost of the assignments involving *lower* segmentation hypotheses.

3.3. Training

The training of our assignment model consists of learning a Random Forest classifier on possible assignments between regions. The idea to use a classifier to help merging regions is not entirely new [15, 28]. The biggest difference here is that we use the same classifier for continuation, branching, and end assignments. For that, we create a feature vector for each assignment variable. This vector has the same components regardless of which assignment case it refers to. For features that do not exist in all cases (for example, there is no center distance measurable in end assignments) the respective components are set to a dummy value.

To train the classifier, we need both positive and negative samples of assignments. However, only positive samples need to be provided by a user. Due to the hypotheses consistency constraints on the assignment variables, every positive assignment has a number of conflicting assignments that we take as negative samples.

The features we are using belong to two groups: geometry features and texture features. For the geometry features, we measure the distance between the centers of the segmentation hypotheses, the symmetric set difference of their pixels and the plain size of them. For the texture features we perform a cross-correlation between image patches surrounding the segmentation hypotheses in question and use the position and value of the maximum in two different scales. In addition, we use the difference of normalized intensity histograms between the segmentation hypotheses that are supposed to be assigned to each other. A complete list of features with more details can be found in the supplemental material.

3.4. Confidence Measure

To provide a confidence measure, we exploit again the hypotheses consistency constraints on the assignment variables: for each assignment variable a^i that is part of the solution, we determine the minimal cost $\overline{c}^i$ of any of its directly conflicting assignment variables:

$$\overline{c}^i = \min(\mathbf{c}^{\lightning i}), \tag{8}$$

where, $\mathbf{c}^{\lightning i}$ denotes the set of costs of all assignment variables that are in direct conflict with a^i. According to the hypotheses consistency constraint, these are all assignments that link to any segmentation hypothesis that shares a path with the segmentation hypotheses that are involved in a^i.

The confidence of an assignment is now given as

$$\text{conf}(a^i) = \frac{c^i}{\overline{c}^i}. \tag{9}$$

4. Experiments

We evaluated the performance of our approach on an annotated sample of *Drosophila* larva neural tissue [6]. This publicly available dataset consists of 30 serial sections, imaged with transmission electron microscopy at a resolution of 4x4x50 nm/pixel. The dataset covers a 2x2x1.5 micron cube of neural tissue and provides labels of cellular membranes, cytoplasms and mitochondria of 170 neural processes.

Hypotheses Generation We used the same segmentation hypotheses for all experiments. To that end, we took 100 equidistant samples of the neuron prior λ_N in an interval that ranged from obvious over- to undersegmentation. The limits of this interval and the weight of the Potts term in the segmentation energy (1) have been found by visual inspection on the first slice of the training dataset using an interactive graph-cut implementation.

After the downsampling of the component trees (Sec. 2.2) we were left with 5800 segmentation hypotheses, distributed in 3633 trees with a mean depth of 0.252 ($\pm$0.491).

Training Data We tested our approach by splitting the dataset into two parts: the first 5 slices have been used for the training, while the remaining 25 slices have been used for evaluation. From the gold standard of the first 5 slices we extracted 510 positive assignment samples, for which our assignment model found 10340 negative assignment samples that have been made impossible (details on the selection of negative training samples can be found in Sec. 3.3). Fig. 6 shows qualitative results of our approach on three subsequent slices of the evaluation dataset. In Fig. 7 we show examples of successfully found branchings.

Error Measure In order to evaluate the performance of our approach, we use an error measure that reflects the number of steps a human expert would have to take at least to restore the gold standard from the result [12]. For that, we distinguish between errors that have been made between slices (additional or missing assignments) and within slices (additional or missing segments). We will call these errors *inter-* and *intra-slice errors*, respectively.

For the inter-slice errors, we say that two segments are *linked*, whenever there was an assignment selected that involves both of them. Hence, a selected continuation assignments introduces one link between two segments, a branching assignment introduces two links between three segments, and an end assignment has no effect. We count

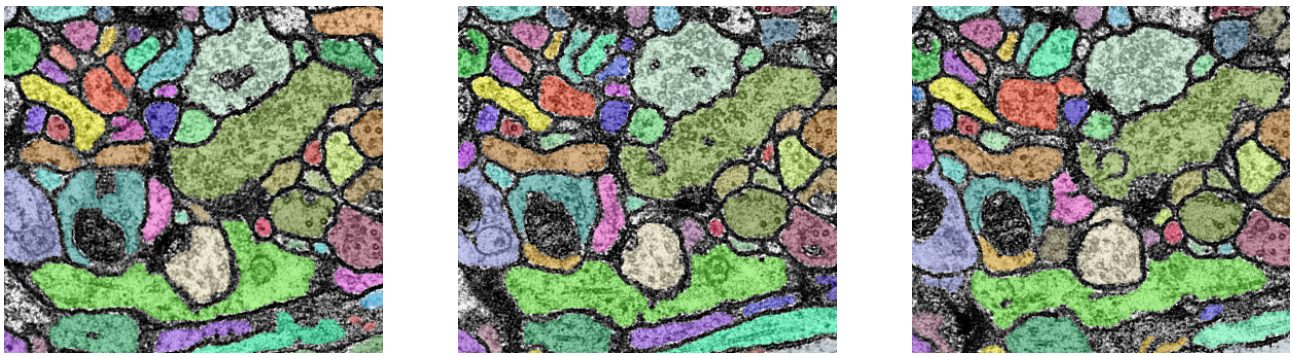


Figure 6. Segmentation result of three subsequent slices. Assignments between slices are represented by the use of the same color.

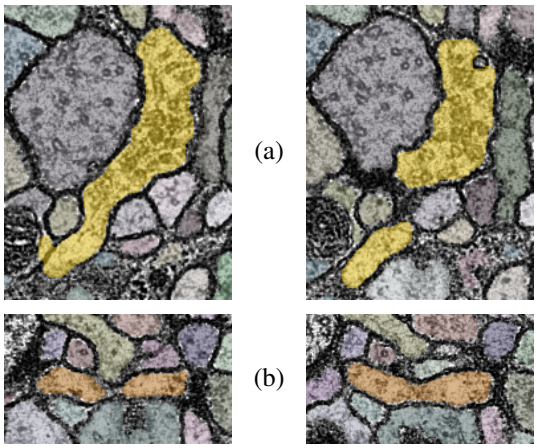


Figure 7. Examples of successfully found neuron branchings. In (a), the central neuron (left) bisects into two parts (right) and is correctly segmented (yellow color). In (b), two smaller processes (left) merge into one neuron (right). Again, the segmentation is correct (brown color).

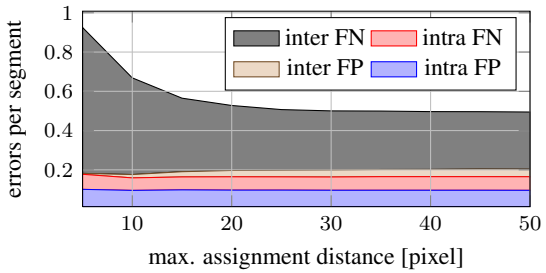


Figure 8. Influence of the maximal assignment distance on the errors of the result. After about 30 pixels there is no mentionable improvement.

every link that is in the gold standard but not in the result as a *false negative inter-slice* (inter FN) error and every link in the result that is not in the gold standard as a *false positive inter-slice* (inter FP) error.

The intra-slice errors are being computed similarly: Every segment of the gold standard that has no corresponding segment in the result is a *false negative intra-slice* (intra FN) error, and vice versa for the *false positive intra-slice* (intra FP) error.

We normalized these errors by the number of segments in the test dataset, *i.e.*, by the sum of all neuron sections over all slices.

Pipeline Parameters Given a trained assignment classifier, only two parameters of our pipeline are open: the distance threshold θ_D for the generation of the assignment variables and the weight θ_S for the influence of the segmentation energy in the assignment costs. Errors for different values of θ_D can be found in Fig. 8. Beyond a distance of about 30 pixels there is no mentionable improvement on the accuracy anymore. Changes due to perturbations of θ_S were very little. We found a value of 1.0 to work well.

Comparison We compared our method to the segmentation fusion (SF) approach presented in [26], which was shown to be superior to other existing methods. There, the cost function for assignments is a linear function of weighted features of the involved segmentation hypotheses (x-y-displacement and cross-correlation). Since their approach does not provide a way to learn the feature weights from a training set, we performed a simple grid-search to pick weights that provided good results. However, what constitutes a good result is an application dependent question: a false positive might be harder to fix than a false negative. Therefore, instead of just taking the result with the best F1-score (SF-1), we decided to take two more results that reflect the range of the trade-off between false positives and negatives: one that had the smallest intra FN error (SF-2) and one that had the smallest inter FN error (SF-3). To keep the comparison fair, we performed the grid-search on the same five slices that we used for training and not on the whole dataset. We also used the same segmentation hypotheses for both approaches to obtain an objective view on the assignment model performances. The results on the evaluation dataset can be seen in Fig. 9. In the same Figure we also give a lower bound on the errors based on the extracted segmentation hypotheses, *i.e.*, errors that stem from

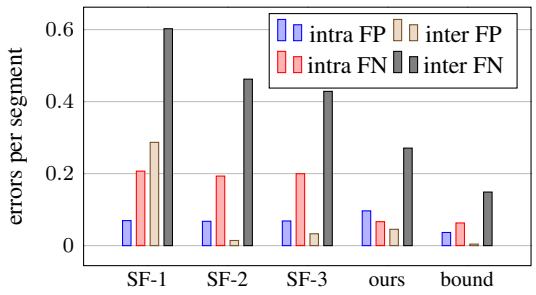


Figure 9. Comparative results to three different weight sets of the segmentation fusion approach [26] (SF-1 to SF-3) that have been found via grid-search on the test dataset. In total our results are closer to the lower error bound that is given by the extracted hypotheses.

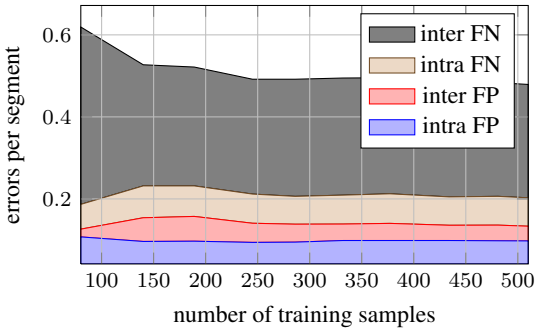


Figure 10. Influence of the number of training samples on the accuracy. After about 300 training samples there is almost no improvement.

missing or wrong segmentation hypotheses that cannot be compensated for by the assignment search.

4.1. Training

The training has been carried out with different numbers of (positive) training samples and classifier parameters, *i.e.*, the number of trees in the Random Forest and the number of features per node in the trees. The results with respect to the number of training samples can be seen in Fig. 10. The performance converges after about 300 training samples. Regarding the training parameters, we found an optimum at 100 trees and 20 features per node, while further increases did not have a substantial effect.

4.2. Confidence Measure

To show the usability of our proposed confidence measure, we sorted all selected assignment variables by their confidence values and evaluated the errors that are contributed by the upper $k\%$, *i.e.*, by the assignment variables our model is most confident about. For that, we limited our attention to the false positive errors (inter- and intra-slice), since the false negative errors lose their meaning if we deliberately remove assignments from the solution. The errors have additionally been normalized by the number of assignment variables in the respective fraction to compensate for the decrease of false positive errors due to selecting less assignments.

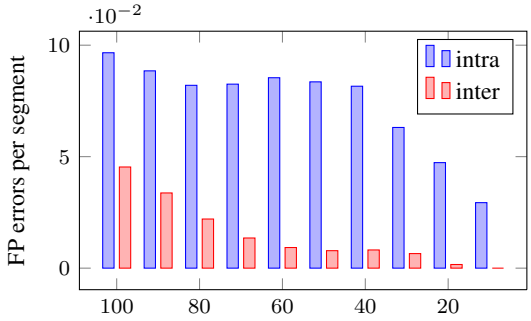


Figure 11. Relative FP errors contributed by the upper $k\%$ of the confidence sorted assignments. To account for the differences in the solution sizes, the values have been normalized by the number of assignments in the respective portion. Note that the upper 10% of the confidence sorted assignments do not contain inter-slice FPs.

The results can be seen in Fig. 11. The 20% most confident assignment variables contribute only 0.00163 inter-slice FPs per segment and 0.047 intra-slice FPs per segment. In the top 10% there were no inter-slice FPs at all and only 0.015 intra-slice FPs per segment. We also found that the assignment variables with the lowest confidence measure contributed the most FPs.

4.3. Inference Time

We solved the ILP to find the best set of assignments using the Gurobi solver[2], which is free for academic use. The average inference time for the whole stack was 6.2 seconds ($\pm$ 0.13), giving 2.02348 milliseconds ($\pm$ 0.04242) per segment on a 12 core Intel Xeon CPU at 3.47GHz.

5. Discussion

We have shown that the learning of assignment costs between multiple segmentation hypotheses can effectively be done by the use of Random Forests. Not only did this eliminate the need to grid-search feature weights, we also observed higher accuracy of the resulting solution compared to the current state of the art approach [26]. We further improve on the state of art by detecting branches – a key feature in the reconstruction of neurons.

Currently, the output of any neuron segmentation algorithm must in practice be proof-read by human experts. The confidence measure produced by our algorithm can be used to prioritize areas for human review.

Our method is trainable on a few hundred positive assignment samples and the inference is fast, providing the ability to retrain and resegment immediately in response to interactive error correction.

[2]Gurobi Optimizer Version 4.6, www.gurobi.com

6. Acknowledgments

This work was supported by the Swiss National Science Foundation under grant CRSII3 130470 and by the Institute of Neuroinformatics of the University of Zurich and ETH Zurich.

References

[1] Bjoern Andres, Ullrich Koethe, Thorben Kroeger, Moritz Helmstaedter, Kevin L. Briggman, Winfried Denk, and Fred A. Hamprecht, *3d segmentation of sbfsem images of neuropil by a graphical model over supervoxel boundaries*, Medical Image Analysis (2011). ↑2

[2] DD Bock, WC Lee, AM Kerlin, ML Andermann, G Hood, AW Wetzel, S Yurgenson, ER Soucy, HS Kim, and RC Reid, *Network anatomy and in vivo physiology of visual cortical neurons*, Nature **47** (2011), no. 7337, 177–82. ↑1

[3] Yuri Boykov and Vladimir Kolmogorov, *An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision*, IEEE Transactions on Pattern Analysis and Machine Intelligence **26** (2004sept.), no. 9, 1124 –1137. ↑2

[4] Leo Breiman, *Random Forests*, Machine Learning **45** (2001), 5–32. 10.1023/A:1010933404324. ↑3

[5] Kevin L Briggman and Winfried Denk, *Towards neural circuit reconstruction with volume electron microscopy techniques*, Current Opinion in Neurobiology **16** (2006), no. 5, 562 –570. ↑1

[6] Albert Cardona, Stephan Saalfeld, Stephan Preibisch, Benjamin Schmid, Anchi Cheng, Jim Pulokas, Pavel Tomancak, and Volker Hartenstein, *An integrated micro- and macroarchitectural analysis of the* Drosophila *brain by computer-assisted serial section electron microscopy*, PLoS Biology **8** (2010), no. 10, e100050. ↑1, 5

[7] Moritz Helmstaedter, Kevin Briggman, and Winfried Denk, *High-accuracy neurite reconstruction for high-throughput neuroanatomy*, Nat Neurosci **14** (2011), no. 8, 1081–8. ↑1

[8] Moritz Helmstaedter, Kevin L. Briggman, and Winfried Denk, *3D Structural Imaging of the Brain with Photons and Electrons.*, Current Opinion in Neurobiology **18** (2008), no. 6, 633–641. ↑1

[9] Viren Jain, Benjamin Bollmann, Mark Richardson, Daniel R. Berger, Moritz N. Helmstaedter, Kevin L. Briggman, Winfried Denk, Jared B. Bowden, John M. Mendenhall, Wickliffe C. Abraham, Kristen M. Harris, Narayanan Kasthuri, Ken J. Hayworth, Richard Schalek, Juan Carlos Tapia, Jeff W. Lichtman, and H. Sebastian Seung, *Boundary Learning by Optimization with Topological Constraints*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2010. ↑2

[10] Viren Jain, Joseph F. Murray, Fabian Roth, Srinivas Turaga, Valentin P. Zhigulin, Kevin L. Briggman, Moritz Helmstaedter, Winfried Denk, and H. Sebastian Seung, *Supervised Learning of Image Restoration with Convolutional Networks*, Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2007, pp. 1–8. ↑2

[11] Viren Jain, H. Sebastian Seung, and Srinivas C. Turaga, *Machines that learn to segment images: a crucial technology for connectomics*, Current Opinion in Neurobiology (2010). ↑1, 2

[12] Viren Jain, Srinivas C. Turaga, Kevin L. Briggman, Moritz N. Helmstaedter, Winfried Denk, and H. Sebastian Seung, *Learning to Agglomerate Superpixel Hierarchies*, Advances in Neural Information Processing Systems, 2011. ↑2, 5

[13] Component Trees for Image Filtering and Segmentation (1997) ↑3

[14] Elizabeth Jurrus, Antonio R.C. Paiva, Shigeki Watanabe, James R. Anderson, Bryan W. Jones, Ross T. Whitaker, Erik M. Jorgensen, Robert E. Marc, and Tolga Tasdizen, *Detection of neuron membranes in electron microscopy images using a serial neural network architecture*, Medical Image Analysis **14** (2010), no. 6, 770 –783. ↑2

[15] Verena Kaynig, Thomas Fuchs, and Joachim M. Buhmann, *Geometrical Consistent 3D Tracing of Neuronal Processes in ssTEM Data*, Proceedings of the Conference on Medial Image Computing and Computer-Assisted Intervention, 2010, pp. 209–216. ↑2, 5

[16] ———, *Neuron Geometry Extraction by Perceptual Grouping in ssTEM Images*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 2902–2909. ↑2

[17] Pushmeet Kohli and Philip Torr, *Dynamic Graph Cuts and Their Applications in Computer Vision*, Computer Vision, 2010, pp. 51–108. ↑3

[18] Vladimir Kolmogorov, Yuri Boykov, and Carsten Rother, *Applications of Parametric Maxflow in Computer Vision*, Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2007. ↑3

[19] M.P. Kumar and D. Koller, *Efficiently selecting regions for scene understanding*, Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010june, pp. 3217 –3224. ↑2

[20] Jeff W. Lichtman and Joshua R. Sanes, *Ome sweet ome: what can the genome tell us about the connectome?*, Current Opinion in Neurobiology **18** (2008), 346–353. ↑1

[21] Aurélien Lucchi, Kevin Smith, Radhakrishna Achanta, Graham Knott, and Pascal Fua, *Supervoxel-Based Segmentation of Mitochondria in EM Image Stacks with Learned Shape Features*, IEEE Transactions on Medical Imaging **30** (2011October), no. 11. ↑2

[22] Dirk Padfield, Jens Rittscher, and Badrinath Roysam, *Coupled minimum-cost flow cell tracking for high-throughput quantitative analysis*, Medical Image Analysis **In Press, Corrected Proof** (2010). ↑3

[23] C. Rother, T. Minka, A. Blake, and V. Kolmogorov, *Cosegmentation of Image Pairs by Histogram Matching - Incorporating a Global Constraint into MRFs*, Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, 2006june, pp. 993 –1000. ↑2

[24] Stephan Saalfeld, Albert Cardona, Volker Hartenstein, and Pavel Tomancák, *As-rigid-as-possible mosaicking and serial section registration of large ssTEM datasets.*, Bioinformatics **26** (2010), 57–63. ↑1

[25] Amelio Vazquez-Reina, Shai Avidan, Hanspeter Pfister, and Eric Miller, *Multiple Hypothesis Video Segmentation from Superpixel Flows*, Proceedings of the European Conference on Computer Vision (ECCV), 2010. ↑2

[26] Amelio Vazquez-Reina, Daniel Huang, Michael Gelbart, Jeff Lichtman, Eric Miller, and Hanspeter Pfister, *Segmentation Fusion for Connectomics*, Proceedings of the IEEE International Conference on Computer Vision (ICCV), 201106/11/2011. ↑1, 2, 6, 7

[27] Ashok Veeraraghavan, Alex V. Genkin, Shiv Vitaladevuni, Lou Scheffer, Shan Xu, Harald Hess, Richard Fetter, Marco Cantoni, Graham Knott, and Dmitri Chklovskii, *Increasing depth resolution of electron microscopy of neural circuits using sparse tomographic reconstruction*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2010june, pp. 1767 –1774. ↑2

[28] Shiv Naga Prasad Vitaladevuni and Ronen Basri, *Co-Clustering of Image Segments Using Convex Optimization Applied to EM Neuronal Reconstruction*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 2203 –2210. ↑2, 5

[29] Yuriy and Mishchenko, *Automation of 3D reconstruction of neural tissue from large volume of conventional serial section transmission electron micrographs*, Journal of Neuroscience Methods **176** (2009), no. 2, 276 –289. ↑2