Carving: Scalable Interactive Segmentation of Neural Volume Electron Microscopy Images

C. N. Straehle[1], U. Köthe[1], G. Knott[2], and F. A. Hamprecht[1⋆]

[1] University of Heidelberg, Heidelberg, Germany
[2] Ecole Polytechnique Fédérale, Lausanne, Switzerland

Abstract. Interactive segmentation algorithms should respond within seconds and require minimal user guidance. This is a challenge on 3D neural electron microscopy images. We propose a supervoxel-based energy function with a novel background prior that achieves these goals. This is verified by extensive experiments with a robot mimicking human interactions. A graphical user interface offering access to an open source implementation of these algorithms is made available.

Keywords: electron microscopy, seeded segmentation, interactive segmentation, graph cut, watershed, supervoxel

1 Introduction

Electron microscopy has provided images with revealing resolution in *two* dimensions since the mid-20th century. More recent volume imaging techniques such as Focused Ion Beam Scanning Electron Microscopy (FIBSEM), however, yield images with an isotropic resolution of a few nm in all *three* dimensions [11], see Fig. 1. Full automation is required for the analysis of very large scale experiments, e.g. in connectivity studies. Several recent methods [1, 16, 15, 10, 6, 9] already take advantage of the isotropy in new datasets. However, the error rates of these methods still fall short of human performance. In contrast, interactive methods allow live user corrections until the desired quality is achieved. This is useful during rapid exploration of new data, but even more important for generating the ground truth needed for training and validation of automated algorithms.

Interactive or "seeded" segmentation methods come in two main flavors: deformable models ("snakes" e.g. [7, 8, 17]) and random fields (e.g. [18, 19]). The former are particularly useful when objects are characterized by recurring shape properties that constrain the segmentation. Slice-oriented versions of this approach (using 3D tracking of 2D models) were successfully used on anisotropic neuron data [7], but truly 3-dimensional modeling is challenged by the intricate geometry of neurons, which bend and branch out in complex ways. Random fields naturally lend themselves to 3D modeling and thus seem a natural choice

⋆ HCI, Speyerer Strasse 6, D-69115 Heidelberg, fred.hamprecht@iwr.uni-heidelberg.de

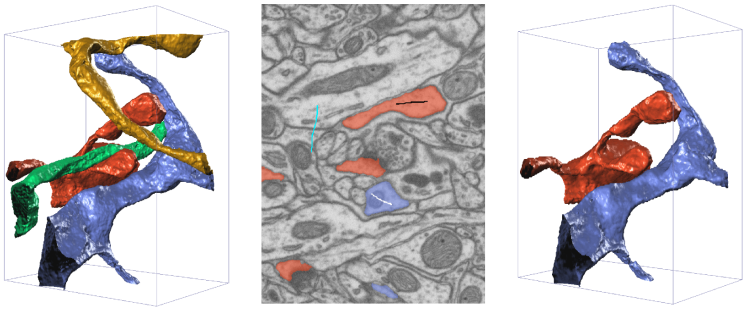

Fig. 1: (left) Small subset of the ground truth segmentation. (center) Slice from FIBSEM raw data, with seeds provided by the user (background: cyan, foreground: black,white) and the resulting segmentation (orange, blue). (right) 3D rendering of the segmentation shown in (b). (Rendering done with V3D [13])

for isotropic neuron data. However, different neurons of the same type are indistinguishable by their local appearance (intensity and texture), so that local class probabilities are hard to define. This also causes strong shrinking bias [18, 19]. Our main insights and contributions are a graph cut formulation and a watershed with:

- a problem-specific objective function whose potentials are biased toward the background region, so that uncertain voxels are preferably assigned to the background. We show that this simple idea works surprisingly well on neural data.
- a *supervoxel* approximation [14] of the original problem defined on a voxel grid. We achieve 100-fold speed-ups with virtually identical results.
- an edge weight parametrization which overcomes the shrinking bias. This is complementary to, and arguably simpler than, gradient fluxes [18, 19].
- an extensive quantitative comparison of graph cut and watershed results with ground truth on 20 dendrites by means of a *robot user* that mimics human seed selection.

Satisfactory segmentations of individual 3D objects from a 500^3 FIBSEM volume can typically be achieved with 2-4 seeds with a response time of ca. 1 second per click (after ≈ 3 minutes of preprocessing at program startup).

2 The Optimization Problem: Basic Definition, Enhancements and Efficient Solution via Supervoxels

Reference [3] is the culmination of a series of important papers relating graph cut, random walk, shortest paths and watershed in a unified seeded segmentation framework, the "Power watershed". We will start from these insights, and represent an image as a weighted graph $G = (V, E)$ of pixels $v \in V$ and their neighborhood relations $e \in E$. A segmentation is represented by a set of labels $\mathbf{x}$

associated with the vertices V. A label x_i is coupled to the underlying observations at node v_i by a node weight w_i, and to its neighbors by weights $w_{ij} \in [0,1]$ assigned to the edges e_{ij}. Large edge weights express a pronounced penchant for the incident vertices to share the same label. Finally, user input is added in the form of seeds $y_i \in \{0,\dots,|C|-1\}$ in an interactive fashion, where C is the set of distinct regions. Graph cut can handle only $|C|=2$ distinct regions (foreground vs. background), while the watershed can account for an arbitrary number of regions. A segmentation is then given by the minimizer

$$\operatorname{argmin}_{\mathbf{x}} \sum_{e_{ij}\in E} w_{ij}^p |x_i - x_j|^q + \sum_{v_i \in V} w_i^p |x_i - y_i|^q \tag{1}$$

with graph cut ($p=q=1, y_i, x_i \in \{0,1\}$), random walk ($p=1, q=2$, $y_i \in \{0,1\}$, $x_i \in [0,1]$) and watershed ($p\to\infty, q=1, y_i, x_i \in \{0,\dots,|C|-1\}$) emerging as special cases.

2.1 The Crucial Choice of Weight Functions

Given the above framework, the developer has to choose p, q and, importantly, the weights w. Node weights are often used to couple the node label to the appearance of the underlying region, as in GrabCut. This is not viable in the data studied here: for instance, the texture, color, etc. of one dendrite are indistinguishable from that of others in the vicinity. The modeling effort must hence concentrate on region boundaries. This can happen indirectly, through nonlocal geometric terms which can be cleverly encoded in node potentials [12, 18], or directly through boundary detectors. These are often implemented through $w_{ij} = \exp\left(-\beta |\nabla I|^2_{ij}\right)$ where $|\nabla I|^2_{ij}$ is the squared image gradient between nodes i and j. As noted in [3], the parameter β plays a crucial role. It not only indicates what gradients are deemed significant, but is directly related to the parameter p from (1): increasing β is tantamount to increasing p!

Since the staining in the neural EM images studied here can directly be interpreted as an edge map, we instead define

$$w_{ij} = \exp\left(-\beta \left(I_i + I_j\right)/2\right) \tag{2}$$

where I is an estimate for membrane probability. Besides the (inverted) EM image itself, $I \in [0,1]$ can be any membrane indicator, for instance a discriminative classifier [1, 15] or the first eigenvalue of the Hessian matrix of the membrane image. We tried both, but concentrate on the latter here because it needs no training and allows us to focus on the improvements described below.

The impact of β. Changing β in (2) has important implications for graph cut: When using $\beta = 1$ and unary weights $w_i = 0$ (except at locations with user seeds), severe shrinking bias [12] occurs: the cheapest segmentation boundary tightly encloses either the foreground or background seeds, assigning nearly the complete volume to the other class. Increasing β to around 100 (and thus raising p 100-fold) implies that erroneous cuts within regions of low membrane strength

become very expensive, while label transitions on membranes remain cheap. In our data, this is sufficient to obviate more complex modeling, e.g. in terms of gradient flux unary potentials [18].

In contrast, changing β in the watershed amounts to a monotonous height transformation of the landscape that this algorithm implicitly sees, without however changing its salient features: the locations and ordering of the minima, maxima, saddle points and, most importantly, the watersheds do not change. The same conclusion – invariance of the watershed algorithm to the choice of β – can be reached by a study of the limit $p \to \infty$ in (2).

Background bias. Assume that an appropriate value of β has been found. If a single foreground and a single background seed of similar size are given, one would expect around half of all neurons to be assigned to either class. This is borne out in practice. To achieve something closer to the desired result, namely a segmentation of one neuron versus all others, we propose to use a small bias favoring the background. For the graph cut, this bias is easily incorporated by assuming an implicit background label $y_i = 0$ for all unseeded points, and adding a small unary weight $w_i = \alpha$ to these points. The quality of the resulting segmentation is robust w.r.t. α across three orders of magnitude: choosing $\alpha \in [10^{-5}, 10^{-3}]$ results in much better segmentations than $\alpha = 0$ (results not shown).

The above recipe does not work for the watershed. Instead, we build the bias directly into the priority queue that controls the region assignment order of the algorithm: In every iteration, there is a set of regions (initially the seeds), and all edges connecting these regions to unlabeled nodes are ordered by weight w_{ij}. The end node of the most expensive edge is then assigned to the region where the edge starts. In this way, watersheds cut cheap edges, minimizing (1). Clearly, manipulations of the unary weights w_i and monotonous transformations of the binary weights w_{ij} have no influence on the outcome. Therefore, we change (2) so that assignments to the background class are preferred: $w_{ij}(x_i) = \exp\left(-\gamma(x_i)\,\beta\,(I_i + I_j)\,/2\right)$ with $0 < \gamma(x_i = 0) < \gamma(x_i \neq 0) \leq 1$. Qualitatively speaking, the dams in the flooding metaphor of the watershed algorithm appear lower to the background class.

2.2 Speedup by Coarse-Graining: Supervoxels

All algorithms studied here are too slow, in their native implementations, for a truly interactive experience when working with 3D volumes of the order of 512^3 voxels on a standard desktop PC. We hence suggest a coarse graining which is a heuristic approximation for graph cut and random walker, but does *not* affect the solution of the watershed. The simplification is best explained for random walk segmentation, but applies to the other cases as well: the labels computed by the random walker change little (cf. [4]) over homogeneous regions (where the binary weights are large), and abruptly near boundaries (where the binary weights are small). One natural simplification of the problem is to reduce the number of unknowns in the linear system of equations by grouping such pixels as are expected to have very similar labels in the solution. Conceptually, this amounts to setting some of the binary weights to very large values, and hence constraining

all pixels within a group or "superpixel" to have the same label. More explicitly, we propose to augment (1) with a set of constraints $x_i = x_j | i, j \in s_k$ for a suitable partitioning $\bigcup_k s_k = V$, $s_k \cap s_l = \emptyset$ of the original graph G.

The quality of this approximation crucially depends on the partitioning used, and finding a good partitioning looks like a daunting task: it should on the one hand be cheap to compute, and on the other hand anticipate the solution of the segmentation algorithm even before that is executed. However, based on the analysis of the powerwatershed energy (1), we argue that the catchment basins of the watershed are a useful proxy: remember that both graph cut and random walk require very pronounced edge weights (large β, or large p) to yield sensible solutions on the type of data used here. Such large powers $p \approx 100$ are an approximation to the special case $p \to \infty$ which is solved by the watershed. Conversely, the catchment basins $\{s_k\}$ of an (unseeded) watershed based on the same edge weights are regions in which the labels of graph cut or random walk are relatively homogeneous.

Making this simplifying assumption, we obtain a supervoxel graph $G'(E', V')$ in which the vertices $s_k \in V'$ represent the individual catchment basins of the original graph G, and the weight $w'_{k,l}$ of an edge in E' is given by the sum over the edge weights connecting the two catchment basins s_k, s_l from the original graph. This approximation makes for large savings, especially for the segmentation of multiple objects in the same data set for which other strategies such as reusing the residual flow of a graph cut computation cannot be employed.

3 Experiments

Data. The $600 \times 800 \times 409$ dataset shows neural tissue imaged with a FIBSEM instrument [11]. To evaluate the algorithms objectively, we designed an **Interactive Segmentation Robot.** [5]. The automaton emulates the human seeding strategy for different parameters and objects. Given ground truth (Fig. 1), the robot seeks to interactively segment a single object of interest using the following strategy:

1. Calculate the set differences between ground truth and current segmentation. Place a correcting single voxel seed in the center (maximum of the Euclidean distance transform) of the largest false region.
2. Re-run the segmentation algorithm with the new set of seeds.
3. Iterate until convergence to ground truth.

The convergence criterion is that the detected boundaries are within three pixels of the true boundaries (which roughly corresponds to the accuracy of the ground truth). Note that the estimated number of seeds required to reach a good segmentation is conservative because the robot only labels a single voxel at a time, while most humans would provide extended brush strokes which are potentially more informative.

All tests were executed on an Intel iCore 7 machine with 2.4GHz. Unfortunately, the random walker was too slow for an interactive procedure even in the

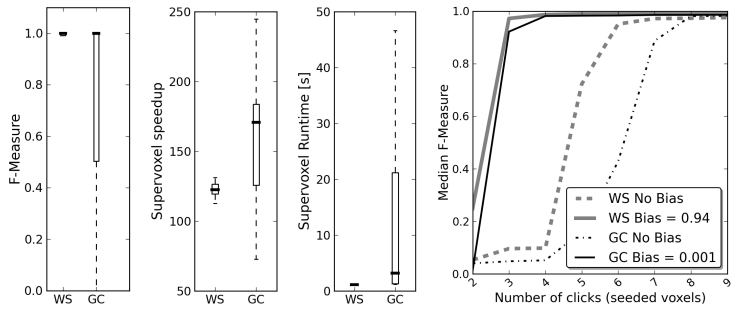

Fig. 2: (a) Agreement between voxel and supervoxel variants of the algorithms. (b) Speedup factor due to supervoxels. (c) Distribution of absolute response times to each click (after one-time preprocessing), on a $600 \times 800 \times 409$ volume. (d) Agreement between ground truth and robot segmentations, as a function of the number of interactions.

supervoxel implementation, so the corresponding results are omitted. The graph cut problem was solved using the public code of [2]. The free parameters of the algorithms (scale of the Hessian matrix feature, exponent p in (1), strength of background bias $\gamma(x_i)$, α) were optimized on a volume that does not overlap the test region. Interestingly, the best choice for p was $p \approx 100$, confirming our discussion about the impact of this parameter in Section 2.

Results. The first experiment (Fig. 2a) investigates whether supervoxel segmentation is a valid approximation of voxel-level segmentation. This can be affirmed for the watershed (the observed minor differences can be traced to different implementations of the priority queue). For the graph cut, the results are a little surprising: while there is a discrepancy between voxel and supervoxel implementations, it is actually the latter that performs better! The reason is the shrinking bias: since the robot labels individual voxels, graph cut sometimes ignores these seeds by turning them into tiny "islands". This phenomenon does not occur when an entire supervoxel is seeded. The second experiment compares the speed of the supervoxel algorithms relative to their voxel-level implementations (Fig. 2b) and the absolute run time (Fig. 2c). We observe a median speed-up of around 120 for the watershed and around 170 for graph cut. Once the supervoxel graph has been constructed – this is a matter of minutes – the turnaround time for a user interaction is of the order of (sometimes many) seconds for the graph cut, and consistently around one second for the watershed. The third experiment (Fig. 2d) shows the number of clicks (single voxel seeds) given by the robot following the segmentation strategy and the ground truth agreement of the resulting segmentations. The figure shows the median over the 20 largest objects in a 300^3 subregion of the volume. Using a background bias

clearly reduces the number of interactions required. For a qualitative impression, exemplary carving results are shown in Figs. 1 and 3.

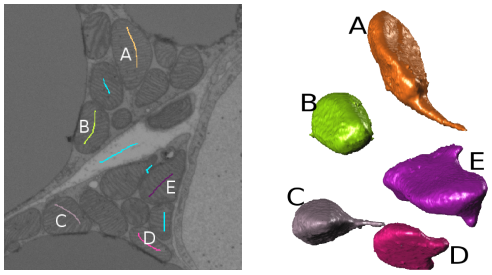

Fig. 3: The proposed approach works well on objects other than dendrites, here: mitochondria. Shown are (left): raw data, seeds (background: cyan) and (right): resulting segmentation.

4 Conclusions

We have systematically evaluated two well known algorithms (watersheds and graph cut) with regard to their applicability to interactive neuron segmentation in FIBSEM volume data. Our experiments have shown that the supervoxel approximation can be safely used as an approximation of the voxel-level energy function. This is in part due to the choice of a very large power $p \approx 100$ in (1). The main effect of this choice is to counter the shrinking bias of the graph cut, but it also serves to minimize the error arising from grouping voxels into supervoxels. The simple idea of preferring background over foreground assignments effectively helps ensuring that each foreground region consists of a single neuron only – the number of required user interactions is reduced considerably relative to unbiased assignments, resulting in correct segmentations after only a handful of mouse clicks. Moreover, no additional calculations are required to realize this behavior. All results were confirmed by extensive experiments using an objective segmentation robot. It applies a transparent seed placement strategy that emulates the actions of a human user.

Minimizing user effort and algorithm response times allows to analyze large data sets interactively, i.e. with immediate feedback on segmentation quality. The described supervoxel algorithms can be downloaded, along with a graphical user interface, as an open source program from http://www.ilastik.org/carving.

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: Rigoll, G. (ed.) Pattern Recognition. LNCS, vol. 5096, pp. 142–152. Springer, Heidelberg (2008)
2. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE TPAMI 26, 1124–1137 (2004)

3. Couprie, C., Grady, L., Najman, L., Talbot, H.: Power watershed: A unifying graph-based optimization framework. IEEE TPAMI 33, 1384–1399 (2011)
4. Grady, L.: Random walks for image segmentation. IEEE TPAMI 28, 1768–1783 (2006)
5. Gulshan, V., Rother, C., Criminisi, A., Blake, A., Zisserman, A.: Geodesic star convexity for interactive image segmentation. In: IEEE CVPR'10. pp. 3129 –3136 (2010)
6. Jain, V., Bollmann, B., Richardson, M., Berger, D., Helmstaedter, M., Briggman, K., Denk, W., Bowden, J., Mendenhall, J., Abraham, W., Harris, K., Kasthuri, N., Hayworth, K., Schalek, R., Tapia, J., Lichtman, J., Seung, H.: Boundary learning by optimization with topological constraints. In: IEEE CVPR'10. pp. 2488 –2495 (2010)
7. Jeong, W., Beyer, J., Hadwiger, M., Blue, R., Law, C., Vazquez, A., Reid, C., Lichtman, J., Pfister, H.: Ssecrett and neurotrace: Interactive visualization and analysis tools for large-scale neuroscience datasets. IEEE COMPUT GRAPH 30, 58–70 (05/2010 2010)
8. Jurrus, E., Hardy, M., Tasdizen, T., Fletcher, P., Koshevoy, P., Chien, C., Denk, W., Whitaker, R.: Axon tracking in serial block-face scanning electron microscopy. MED IMAGE ANAL 13(1), 180–188 (2009)
9. Jurrus, E., Paiva, A., Watanabe, S., Anderson, J., Jones, B., Whitaker, R., Jorgensen, E., Marc, R., Tasdizen, T.: Detection of neuron membranes in electron microscopy images using a serial neural network architecture. MED IMAGE ANAL 14(6), 770–783 (2010)
10. Kaynig, V., Fuchs, T., Buhmann, J.: Neuron geometry extraction by perceptual grouping in sstem images. In: IEEE CVPR'10. pp. 2902 –2909 (2010)
11. Knott, G., Marchman, H., Wall, D., Lich, B.: Serial section scanning electron microscopy of adult brain tissue using focused ion beam milling. J NEUROSCI 28(12), 2959–2964 (2008)
12. Kolmogorov, V., Boykov, Y.: What metrics can be approximated by geo-cuts, or global optimization of length/area and flux. In: IEEE ICCV'05. vol. 1, pp. 564–571 (2005)
13. Peng, H., Ruan, Z., Long, F., Simpson, J., Myers, E.: V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets. Nature Biotechnology 28(4), 348–353 (2010)
14. Ren, X., Malik, J.: Learning a classification model for segmentation. In: IEEE ICCV'03. vol. 2, pp. 10–17 (2003)
15. Turaga, S.C., Murray, J.F., Jain, V., Roth, F., Helmstaedter, M., Briggman, K., Denk, W., Seung, H.S.: Convolutional networks can learn to generate affinity graphs for image segmentation. NEURAL COMPUT 22, 511–538 (2010)
16. Turaga, S., Briggman, K., Helmstaedter, M., Denk, W., Seung, H.: Maximin affinity learning of image segmentation. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) NIPS'09, pp. 1865–1873 (2009)
17. Vazquez-Reina, A., Miller, E., Pfister, H.: Multiphase geometric couplings for the segmentation of neural processes. IEEE CVPR'09 0, 2020–2027 (2009)
18. Vu, N., Manjunath, B.: Graph cut segmentation of neuronal structures from transmission electron micrographs. In: IEEE ICIP'08. pp. 725–728 (2008)
19. Yang, H.F., Choe, Y.: Electron microscopy image segmentation with graph cuts utilizing estimated symmetric three-dimensional shape prior. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Chung, R. (eds.) Proceedings of the 6th international conference on Advances in visual computing - Volume Part II. LNCS, vol. 6454, pp. 322–331. Springer, Heidelberg (2010)