

3D Segmentation of Glial Cells Using Fully Convolutional Networks and k -Terminal Cut

Lin Yang^{1(✉)}, Yizhe Zhang¹, Ian H. Guldner², Siyuan Zhang²,
and Danny Z. Chen¹

¹ Department of Computer Science and Engineering,
University of Notre Dame, Notre Dame, IN 46556, USA
lyang5@nd.edu

² Department of Biological Sciences, Harper Cancer Research Institute,
University of Notre Dame, Notre Dame, IN 46556, USA

Abstract. Glial cells play an important role in regulating synaptogenesis, development of blood-brain barrier, and brain tumor metastasis. Quantitative analysis of glial cells can offer new insights to many studies. However, the complicated morphology of the protrusions of glial cells and the entangled cell-to-cell network cause significant difficulties to extracting quantitative information in images. In this paper, we present a new method for instance-level segmentation of glial cells in 3D images. First, we obtain accurate voxel-level segmentation by leveraging the recent advances of fully convolutional networks (FCN). Then we develop a k -terminal cut algorithm to disentangle the complex cell-to-cell connections. During the cell cutting process, to better capture the nature of glial cells, a shape prior computed based on a multiplicative Voronoi diagram is exploited. Extensive experiments using real 3D images show that our method has superior performance over the state-of-the-art methods.

1 Introduction

Glial cells are heterogeneous central nervous system-specific cells. Recent studies have shown that glial cells are critical to various homeostatic biological processes (e.g., metabolic support of neurons and synapse formation and function), and are largely implicated in neuroinflammatory events such as traumatic brain injury and multiple sclerosis [3]. Quantitative analysis of glial cells, such as the number of cells, cell volume, number of protrusions, and length of each protrusion, can offer new insights to many studies. Although the current imaging modalities are able to image glial cells in 3D in their native environment, the complicated morphology of their protrusions and the entangled cell-to-cell network cause significant difficulties to the extraction of quantitative information (Fig. 1(a)). In this paper, we present a new method for instance-level segmentation of glial cells in 3D images based on fully convolutional networks (FCN) and k -terminal cut.

Many methods were developed to visualize and quantify glial cells in 3D microscopy images, such as angular variance [2], random walk [10], and local

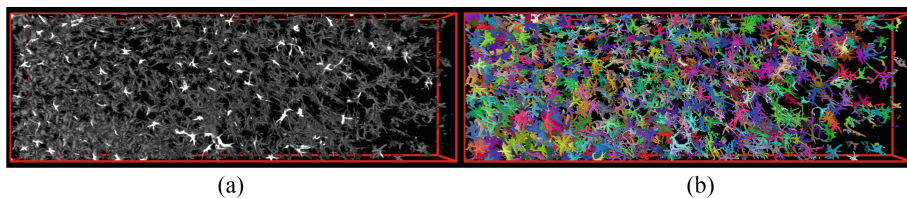


Fig. 1. (a) 3D visualization of a raw image of the glial cell network; (b) 3D visualization of the instance-level segmentation of (a) computed by our method. Random colors are assigned to every segmented instance (glial cell).

priority-based parallel tracing (LPP) [6]. Due to the difficulties in attaining accurate voxel-level segmentation, none of these methods can automatically compute accurate instance-level segmentation of glial cells in 3D. By leveraging the recent advances of fully convolutional networks (FCN) [7], we obtain accurate voxel-level segmentation (Fig. 2(d)). Based on that, we develop a new algorithm for computing instance-level segmentation of glial cells in 3D microscopy images.

Utilizing deep learning techniques greatly relieves the burden of image segmentation and detection. However, it is still non-trivial to attain accurate 3D instance-level segmentation of glial cells, for the following reasons. First, although some recent FCNs are capable of computing instance-level segmentation in 2D images [8], due to the memory limitation of the current computing systems, so far no FCN can directly yield good quality instance-level segmentation in 3D. Second, in the native environment of glial cells, they form a large network in which their protrusions entangle together (Fig. 1(a)). Hence, in the voxel-level segmentation, thousands of glial cells form a large-size connected network. In order to generate accurate 3D instance-level segmentation, we need to carefully cut this connected network into thousands of individual 3D glial cells.

We model this cutting task as a k -terminal cut problem, in which the k terminals correspond to the root points of glial cells (the root point of a glial cell is the convergence point of its protrusions). Much work was done on the general k -terminal cut problem [5], which is NP-hard and was solved only by approximation. Interestingly, by exploiting the morphological properties of glial cells in our 3D images, we are able to develop an effective k -terminal cut algorithm to disentangle the complex cell-to-cell connections. First, since glial cells touch one another with their protrusions, we design a shape prior computed by a multiplicative Voronoi diagram (MVD) [1]. Then, we develop an iterative k -terminal cut algorithm to incorporate this shape prior. Finally, we apply our k -terminal cut algorithm to the voxel-level segmentation results computed by deep learning. Extensive experiments using real 3D images show that our method has much better performance over the state-of-the-art glial cell analysis methods [6, 10].

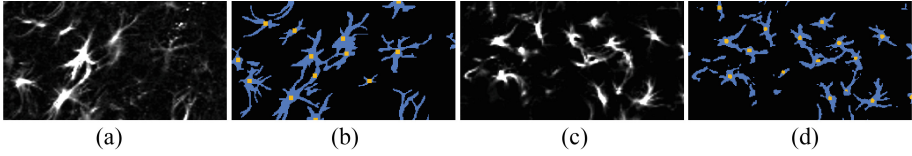


Fig. 2. (a) A training image; (b) the training labels for (a); (c) a testing image; (d) the testing result of (c).

2 Method

Our method consists of three major components: (1) voxel-level segmentation and root point detection using FCN; (2) computing shape prior based on a multiplicative Voronoi diagram (MVD) in a network model; (3) generating 3D instance-level segmentation using a k -terminal cut algorithm.

2.1 Voxel-Level Segmentation and Root Point Detection

Accurate voxel-level segmentation and root point detection are basic foundations of our method. As shown in [7], FCNs are a state-of-the-art method for segmentation in 2D images. Thus, we chose U-net [9] (a state-of-the-art FCN for biomedical image segmentation) to solve this problem. Note that due to the memory limitation, U-net is so far not effective for direct 3D segmentation. This section shows how we can still apply U-net to attain our goals on 3D images.

We first process an input 3D image slice by slice. The procedure for the pixel-level segmentation in a 2D slice remains the same as in [9]. For the root point detection, we mark each root point by a disk of a 5-pixel diameter in the training images. Figure 2(b) gives an example of the labeled training images for segmentation and root point detection. After processing the 2D slices, because we use optical sectioning when we image the tissue samples, there is no shift nor distortion between the slices. Thus the voxel-level segmentation can be obtained by simply stacking the 2D segmentation results together. However, it is more complicated for the root point detection. Figure 3(a) shows that after stacking the 2D root point detection slices, the detected root points form some columns rather than spheres and the columns belonging to different cells may be connected together. These are caused by two main factors: (1) The axial resolution for multi-photon microscopy is 2–3 times lower than the lateral resolution; (2) U-net is very sensitive in detecting root points (it can still detect the true center root points even when it is out of the focus plane). To discriminate different cells that have the same x - y positions but different z positions, we adopt the noise-tolerance non-maximum suppression method [4] (see Fig. 3(b) for an example).

2.2 Computing Shape Prior

After obtaining the voxel-level segmentation and root point detection results, we compute 3D instance-level segmentation using a k -terminal cut algorithm. But,

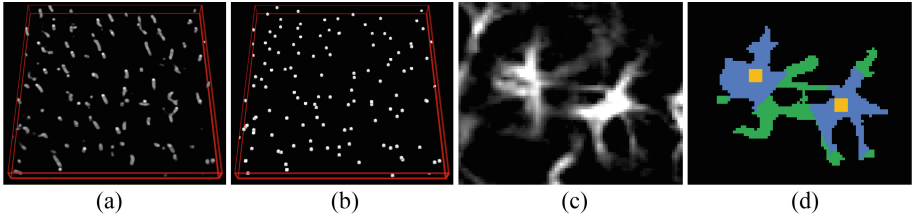


Fig. 3. (a) 3D visualization of a stacked root point detection probability map. (b) 3D visualization of the non-maximum suppressed root point detection map (each root point is dilated for illustration). (c) Some touching glial cells. (d) The cell cutting areas computed by our method. The root points of glial cells are in yellow, the segmentation is in blue, and the computed cell cutting areas are in green.

common minimum cut algorithms are sensitive to the number of cut edges and are likely to give some trivial solutions (e.g., cutting only some edges close to the terminals). Thus, in this section, we address this issue by computing a shape prior based on a multiplicative Voronoi diagram (MVD) in a network model.

Since glial cells touch one another with their protrusions, in the 3D instance-level segmentation, the cutting boundary between two touching cells should fall inside the area of their touching protrusions. We call this area the *touching protrusion area* (TPA). We observed two properties of the TPAs: (1) While two neighboring cells may have multiple touching protrusions, their TPA always lies between the two root points; (2) the TPA is at least $8\mu\text{m}$ (can be adjusted to accommodate different scenarios) away from their root point. Thus, we define the common region of two neighboring cells C_a and C_b that satisfies both these properties as the *cell cutting area*, denoted by $\mathcal{A}_{ab}$ (Fig. 3(d)). Due to the morphology of glial cells, we use the concept of MVD in a network model to define $\mathcal{A}_{ab}$. The MVD is a variant of Voronoi diagram that uses weight parameters for the sites (root points) to control the volumes of their Voronoi regions (i.e., Voronoi cells) [1]. Since cell touching is a pairwise relation between two neighboring root points, we apply the MVD in a pairwise manner to determine $\mathcal{A}_{ab}$.

The MVD is defined as: Given an undirected graph $G = (V, E)$ and m sites $v_{s_i} \in V$ with weight $w_i > 0$, divide V into m Voronoi regions R_i , one for each site v_{s_i} , such that for every $v_x \in V$, $v_x \in R_i$ if and only if $\mathcal{D}(v_x, v_{s_i})/w_i = \min_j \{\mathcal{D}(v_x, v_{s_j})/w_j\}$, where $\mathcal{D}(v_p, v_q)$ is the length of the shortest v_p -to- v_q path in G . We define the *Voronoi region border* of the MVD as a set V_{bd} of vertices: For each $v_x \in V$, $v_x \in V_{bd}$ if and only if either v_x is shared by different Voronoi regions or $\exists (v_x, v_y) \in E$ such that v_x and v_y belong to different Voronoi regions. In our model $G = (V, E)$ for two cells C_a and C_b , V corresponds to all the voxels in $C_a \cup C_b$, E is the 6-neighborhood system of V whose edge lengths are all 1, $m = 2$, v_{s_i} corresponds to a root point with $i \in \{a, b\}$, and w_i is the weight of v_{s_i} . Since our MVD has only two sites, different weight values w_a and w_b with the same ratio of w_a/w_b result in the same partition of V . Let $w_{ab} = w_a/w_b$ denote the relative weight. Note that different w_{ab} values yield different V_{bd} .

Intuitively, we consider all possible Voronoi region borders of the MVD in G that satisfy property (2) as lying between the two root points and as possible cell cutting area. Thus, we define $\mathcal{A}_{ab}$ as follows: $v_x \in V$ is in $\mathcal{A}_{ab}$ if and only if $\exists w_{ab} > 0$ such that $v_x \in V_{bd}$ (for this w_{ab} value) and all $v_y \in V_{bd}$ satisfy property (2) of the TPAs (i.e., for each $v_y \in V_{bd}$, $\mathcal{D}(v_y, v_{s_a}) \geq 8\mu\text{m}$ and $\mathcal{D}(v_y, v_{s_b}) \geq 8\mu\text{m}$). Based on the definition of MVD, V_{bd} satisfies property (2) if and only if $1/\theta_{ab} \leq w_{ab} \leq \theta_{ab}$, where $\theta_{ab} = (D(v_{s_a}, v_{s_b}) - d_{\min})/d_{\min}$ and $d_{\min}$ corresponds to $8\mu\text{m}$ in the image space. Thus, we compute $\mathcal{A}_{ab}$ as follows: For each $v_x \in V$, v_x is in $\mathcal{A}_{ab}$ if and only if either $1/\theta_{ab} \leq \frac{\mathcal{D}(v_x, v_{s_a})}{\mathcal{D}(v_x, v_{s_b})} \leq \theta_{ab}$ or $\exists (v_x, v_y) \in E$ such that the interval $[1/\theta_{ab}, \theta_{ab}] \cap (\min\{\frac{\mathcal{D}(v_x, v_{s_a})}{\mathcal{D}(v_x, v_{s_b})}, \frac{\mathcal{D}(v_y, v_{s_a})}{\mathcal{D}(v_y, v_{s_b})}\}, \max\{\frac{\mathcal{D}(v_x, v_{s_a})}{\mathcal{D}(v_x, v_{s_b})}, \frac{\mathcal{D}(v_y, v_{s_a})}{\mathcal{D}(v_y, v_{s_b})}\}) \neq \emptyset$.

2.3 The Iterative k -Terminal Cut Algorithm

Given the voxel-level segmentation and k detected root points, to compute the 3D instance-level segmentation, we aim to assign to each segmented cell voxel a label for one of the k root points. We use the k -terminal cut [5] to model this task. The definition of the k -terminal cut problem is: Given an undirected graph $G = (V, E)$ with edge costs and k terminals $v_{t_1}, \dots, v_{t_k} \in V$, assign to each $v_x \in V$ a label $L(v_x) \in \{t_1, \dots, t_k\}$, such that $L(v_{t_i}) = t_i$, all vertices with label t_i form a connected subgraph in G using edges whose end vertices are both labeled by t_i , for each $i = 1, 2, \dots, k$, and the total cost $T_{\text{cost}} = \sum_{(v_p, v_q) \in E} c(v_p, v_q) * \mathbb{I}(L(v_p) \neq L(v_q))$ is minimized, where $c(v_p, v_q) > 0$ is the cost of edge (v_p, v_q) , and $\mathbb{I}(b)$ is the indicator function that is 1 when b is true and 0 otherwise. In our k -terminal cut model, all the segmented cell voxels are vertices in G , the k root points are the k terminals, the labels $L(v)$ specify the instance-level segmentation, E is the 6-neighborhood system of V , all edge lengths are 1, and the edge cost $c(v_p, v_q) = \min\{I(v_p), I(v_q)\} + 1$, where $I(v_x)$ is the intensity of voxel v_x in the raw image.

In Sect. 2.2, we introduce a shape prior to guide our cell cutting. This shape prior is determined between the pairs of neighboring cells. Thus, we solve the k -terminal cut problem as a collection of minimum st -cut problems: We iteratively refine the cut between each pair of neighboring cells. We use the ordinary Voronoi diagram [1] in our graph G to define the initial neighboring relations (two glial cells are neighbors of each other if their initial Voronoi regions touch in their boundaries), and update the neighboring relations as the iterative algorithm refines the cuts. The details of the algorithm are given below.

Step 1: We first compute the ordinary Voronoi diagram in G , whose k sites correspond to the k root points. The definition of the ordinary Voronoi diagram is the same as that of MVD in Sect. 2.2, but with all $w_i = 1$. To avoid conflict in the labeling, for each $v_x \in V$ that belongs to multiple Voronoi regions, we assign v_x to the Voronoi region with the largest index. As initialization, for each $v_x \in V$, we let $L(v_x) = t_i$ if and only if $v_x \in R_i$. Let CC_{t_i} denote the set of all the vertices with label t_i . By the definition of minimum st -cuts and the ordinary Voronoi diagram algorithm in G , every CC_{t_i} forms a connected subgraph in G by using edges whose end vertices are both labeled by t_i . Intuitively, each CC_{t_i}

(roughly) corresponds to a cell in the image. Two cells C_a and C_b are taken as a neighboring pair if and only if $\exists v_p \in CC_{t_a}$ and $\exists v_q \in CC_{t_b}$ such that $(v_p, v_q) \in E$. We construct the set $\mathcal{N}$ of neighboring pairs, with $(CC_{t_i}, CC_{t_j}) \in \mathcal{N}$ if and only if CC_{t_i} and CC_{t_j} are neighbors in G .

Step 2: For each $(CC_{t_i}, CC_{t_j}) \in \mathcal{N}$, we construct a subgraph $G' = (V', E')$ of G to compute (or refine) the cut between this pair, where V' is the set of vertices in $CC_{t_i} \cup CC_{t_j}$ and $E' = \{(v_p, v_q) \mid (v_p, v_q) \in E \text{ and both } v_p, v_q \in V'\}$. Since we allow cutting only in the TPAs, we compute $\mathcal{A}_{ij}$ as in Sect. 2.2 (with G' as the input graph, and v_{t_i} and v_{t_j} as the two sites). Note that even for the same neighboring pair (CC_{t_i}, CC_{t_j}) , G' may change iteration by iteration. Thus, $\mathcal{A}_{ij}$ needs to be recomputed in each iteration. To restrict the minimum st -cut in G' , for each $(v_p, v_q) \in E'$, we set its cutting cost $c'(v_p, v_q) = c(v_p, v_q)$ if both $v_p, v_q \in \mathcal{A}_{ij}$, and $c'(v_p, v_q) = \infty$ otherwise. Then we refine the cut between CC_{t_i} and CC_{t_j} by computing the minimum st -cut in G' (with $s = v_{t_i}$ and $t = v_{t_j}$). After the st -cut, for each v_x in G' belonging to the component H_s containing s (resp., H_t containing t), we assign $L(v_x) = t_i$ (resp., $L(v_x) = t_j$).

Step 3: After obtaining the cut between every neighboring pair in $\mathcal{N}$, the neighboring relations in G may change. Hence, we update $\mathcal{N}$ accordingly. If T_{cost} for the k -terminal cut in G thus computed decreases after step 2, we go back to step 2 with the updated $\mathcal{N}$. Otherwise, we output the results and stop.

When applying the above algorithm to our real 3D images, it usually stops after 2–3 iterations. Once the labels $L(v)$ are assigned, to generate the 3D instance-level segmentation, we assign each cell voxel in the image a color for its label. All the cell voxels sharing the same label have the same unique color. The time complexity of the k -terminal cut algorithm is $O(nkmv^3)$, where n is the number of iterations, k is the number of terminals, m is the maximum number of neighbors of a glia cell, and v is the number of voxels in the largest glial cells.

3 Experiments and Results

To perform quantitative performance analysis, we collected four sample 3D images of mouse brains using two-photon microscopy. The sizes of the 3D images are $640 \times 640 \times 25$ voxels and the resolution of each voxel is $1 \times 1 \times 2 \mu\text{m}$. Fluorescence background in all images is corrected by the algorithm in [11]. To evaluate the performance of the root point detection, human experts manually marked all the root points in these images. Due to the difficulties in labeling ground truth in 3D, to evaluate the performance of the 3D instance-level segmentation, three 2D representative slices were selected from each of these four 3D images. To capture the changes of glial cells in different 3D depths, we use the 7th, 13th, and 19th slices. Human experts then labeled all the glial cells in these 2D representative slices. After that, glial cells are assigned unique IDs and these IDs are used to evaluate the 3D instance-level segmentation. The 1st and 25th slices of each 3D image are also labeled and used as training data for U-net.

Two most recent methods are selected for comparison with our method. The first one is an interactive method based on random walk [10]. The second method

detects the root points using a logistic classifier based on hand-craft features and detects the center lines of the protrusions using the LPP algorithm [6].

To measure the accuracy of the root point detection, we compute the maximum cardinality bipartite matching between the true root points and the detected root points based on the following criterion. A true root point r_i can be matched with a detected root point r'_j only if the Euclidean distance between r_i and r'_j is smaller than $10\mu\text{m}$. All the matched detected root points are counted as true positive, all the unmatched true root points are counted as false negative, and all the unmatched detected root points are counted as false positive. Then F1 scores are calculated, and the results are shown in Table 1. Note that since method one [10] requires that the positions of all root points be given as input, it is not applicable to this comparison.

For a fair comparison on the 3D instance-level segmentation, we give method one [10] our root detection results and method two [6] our voxel-level segmentation results. To generate segmentation by method two [6], each cell voxel in our voxel-level segmentation is assigned to the nearest protrusion center line.

Table 1. F1 scores of the root point detection results.

	Sample 1	Sample 2	Sample 3	Sample 4	Average F1 score
Our method	0.8999	0.9208	0.9148	0.8783	0.9035
LPP [6]	0.8222	0.7960	0.8336	0.7697	0.8054

Table 2. F1 scores of the 3D instance-level segmentation results.

	Sample 1	Sample 2	Sample 3	Sample 4	Average F1 score
Our method	0.8709	0.9087	0.9032	0.8931	0.8940
Our w/o MVD	0.8423	0.8930	0.8852	0.8937	0.8786
Random walk [10]	0.7781	0.7855	0.7620	0.7856	0.7778
LPP [6]	0.8017	0.7943	0.8150	0.7960	0.8018
LPP w/ our roots	0.8203	0.8320	0.8308	0.8157	0.8247

To measure the accuracy of the 3D instance-level segmentation, we also compute the maximum cardinality bipartite matching between the true instances and the segmented instances. A true instance I_i and a segmented instance I'_j can be matched if and only if $|I_i \cap I'_j|/|I_i \cup I'_j| > 0.5$, where $|I|$ denotes the total number of voxels of an instance I . All the voxels of the unmatched true instances are counted as false negative, and all the voxels of the unmatched segmented instances are counted as false positive. Among the matched instances, the numbers of true positives, false negatives, and false positives are calculated based on their definitions. F1 scores are computed, and Table 2 summaries the results.

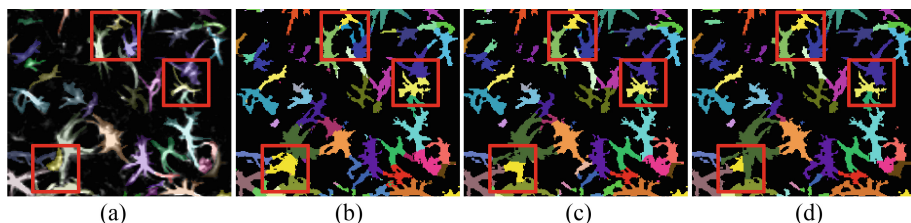


Fig. 4. A cropped window in a slice from Sample 3. (a) The original image (the ground truth is overlaid on the raw image for illustration); (b) the result of method one [10]; (c) the result of method two [6]; (d) our result. It shows that our method attains the best result when multiple cells are entangled together.

Some visual results are shown in Fig. 4. One can see that in all cases, our method achieves the most accurate results over the other two methods. Finally, the average computation time for each cell is ~ 3.5 s on a personal computer.

4 Conclusions

In this paper, we present a new method for instance-level segmentation of glial cells in 3D images based on fully convolutional networks and an iterative k -terminal cut algorithm. Further, we apply a shape prior of glial cells using a multiplicative Voronoi diagram. Extensive experiments on real 3D images show that our method has superior performance over the state-of-the-art methods.

Acknowledgment. This research was supported in part by NSF Grants CCF-1217906 and CCF-1617735, and by NIH Grant 5R01CA194697-02.

References

1. Aurenhammer, F.: Voronoi diagrams – a survey of a fundamental geometric data structure. *ACM Comput. Surv. (CSUR)* **23**(3), 345–405 (1991)
2. Bjornsson, C.S., Lin, G., Al-Kofahi, Y., Narayanaswamy, A., Smith, K.L., Shain, W., Roysam, B.: Associative image analysis: a method for automated quantification of 3D multi-parameter images of brain tissue. *J. Neurosci. Methods* **170**(1), 165–178 (2008)
3. Clarke, L.E., Barres, B.A.: Emerging roles of astrocytes in neural circuit development. *Nat. Rev. Neurosci.* **14**(5), 311–321 (2013)
4. Collins, T.J.: ImageJ for microscopy. *Biotechniques* **43**(1 Suppl.), 25–30 (2007)
5. Cunningham, W.H.: The optimal multiterminal cut problem. *DIMACS Ser. Discrete Math. Theor. Comput. Sci.* **5**, 105–120 (1991)
6. Kulkarni, P.M., Barton, E., Savelonas, M., Padmanabhan, R., Lu, Y., Trett, K., Shain, W., Leasure, J.L., Roysam, B.: Quantitative 3-D analysis of GFAP labeled astrocytes from fluorescence confocal images. *J. Neurosci. Methods* **246**, 38–51 (2015)

7. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: 2015 IEEE Conference on CVPR, pp. 3431–3440 (2015)
8. Romera-Paredes, B., Torr, P.H.: Recurrent instance segmentation. arXiv preprint [arXiv:1511.08250](https://arxiv.org/abs/1511.08250) (2015)
9. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Heidelberg (2015). doi:[10.1007/978-3-319-24574-4_28](https://doi.org/10.1007/978-3-319-24574-4_28)
10. Suwannatat, P., Luna, G., Ruttenberg, B., Raviv, R., Lewis, G., Fisher, S.K., Höllerer, T.: Interactive visualization of retinal astrocyte images. In: 2011 IEEE Conference on ISBI, pp. 242–245 (2011)
11. Yang, L., Zhang, Y., Guldner, I.H., Zhang, S., Chen, D.Z.: Fast background removal in 3D fluorescence microscopy images using one-class learning. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 292–299. Springer, Heidelberg (2015). doi:[10.1007/978-3-319-24574-4_35](https://doi.org/10.1007/978-3-319-24574-4_35)