

# SEGMENT3D: A WEB-BASED APPLICATION FOR COLLABORATIVE SEGMENTATION OF 3D IMAGES USED IN THE SHOOT APICAL MERISTEM

Thiago V. Spina<sup>1,2,3,\*</sup>

Johannes Stegmaier<sup>2,4,5</sup>

Alexandre X. Falcão<sup>3</sup>

Elliot Meyerowitz<sup>6</sup>

Alexandre Cunha<sup>2,\*</sup>

<sup>1</sup>Brazilian Synchrotron Light Laboratory, Brazilian Center for Research in Energy and Materials, Campinas, SP, Brazil

<sup>2</sup>Center for Advanced Methods in Biological Image Analysis, California Institute of Technology, Pasadena, CA, USA

<sup>3</sup>Institute of Computing, University of Campinas, Campinas, SP, Brazil

<sup>4</sup>Institute for Applied Computer Science, Karlsruhe Institute of Technology, Karlsruhe, Germany

<sup>5</sup>Institute of Imaging and Computer Vision, RWTH Aachen University, Aachen, Germany

<sup>6</sup>Howard Hughes Medical Institute and Division of Biology and Biological Engineering, Caltech, Pasadena, CA, USA

## ABSTRACT

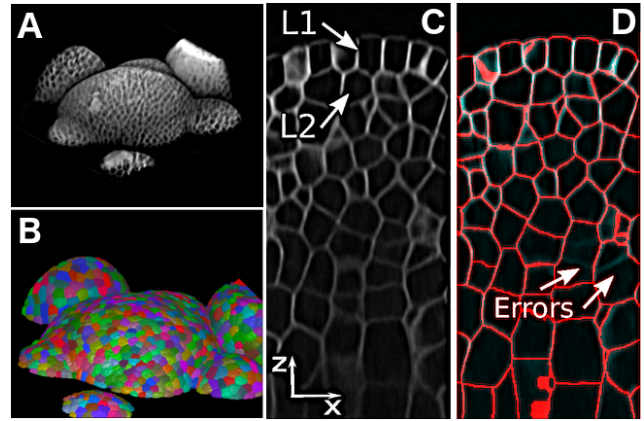
The quantitative analysis of 3D confocal microscopy images of the *shoot apical meristem* helps understanding the growth process of some plants. Cell segmentation in these images is crucial for computational plant analysis and many automated methods have been proposed. However, variations in signal intensity across the image mitigate the effectiveness of those approaches with no easy way for user correction. We propose a web-based collaborative 3D image segmentation application, SEGMENT3D, to leverage automatic segmentation results. The image is divided into 3D tiles that can be either segmented interactively from scratch or corrected from a pre-existing segmentation. Individual segmentation results per tile are then automatically merged via consensus analysis and then stitched to complete the segmentation for the entire image stack. SEGMENT3D is a comprehensive application that can be applied to other 3D imaging modalities and general objects. It also provides an easy way to create supervised data to advance segmentation using machine learning models.

**Index Terms**— 3D Image Segmentation, Shoot Apical Meristem, Web-based Tool, Collaborative Segmentation, Confocal Microscopy

## 1. INTRODUCTION

The *shoot apical meristem* is a collection of stem cells of flowering plants responsible for all above ground plant growth (Figure 1). Since most of what humans eat, such as grains and fruits, is derived from the shoot, it is important to investigate

We are grateful for funding by the São Paulo Research Foundation in projects 2016/11853-2, 2015/09446-7, and 2014/12236-1 (TS, AF), CNPq (AF), the Center for Advanced Methods in Biological Image Analysis, Beckman Institute at Caltech (TS, JS, EM, AC), the German Research Foundation DFG in the project MI1315/4-1 (JS), the Howard Hughes Medical Institute (EM) and the Gordon and Betty Moore Foundation through grant GBMF3406 (EM, AC), and the Serrapilheira Institute in the project Serra-1708-16161 (TS). Correspondence should be addressed to thiago.spina@lnls.br, cunha@caltech.edu.



**Fig. 1:** A) 3D rendition of a *shoot apical meristem* of *Arabidopsis thaliana* and B) its 3D segmentation showing epithelial cells. C) An  $xz$ -slice of an enhanced (denoised and sharpened) meristem reveals the weak signal in lower parts. Panel D) shows missing cell boundaries, indicated by arrows, in the automatic 3D segmentation, in red, created by the method in [2].

the regulation mechanisms that govern meristem growth and consequently the crops that depend on it. Computer simulations done on phantom models with idealized and uniform cell shapes, sizes, and connectivity have recently helped to elucidate cell regulation [1]. However, it is desirable to obtain more realistic models derived directly from real data, which require segmentation of thousands of cells in many 3D confocal microscopy images (Figure 1B).

The cells in the meristem are clustered together, forming top layers L1 and L2 (Figure 1C) approximately uniform in cell size and depth. The thickness of the tissue makes it difficult to obtain via confocal microscopy a uniform intensity signal across image layers. This causes the cell membranes at the bottom of the meristem, where the signal is weaker and noisier, to be less evident than those in or near L1 and L2. Many methods have been proposed to segment

meristematic cells [2, 3, 4, 5, 6], but while they work fairly well for L1 and L2, they tend to fail in deeper layers of the meristem (Figure 1D). Deep learning approaches have recently produced superior results in bioimage segmentation [7, 8, 9], albeit also incomplete and limited by training data. For the meristem, we lack significant segmented and validated data, specially in deeper layers. We are not aware of a single validated segmentation for a full meristem.

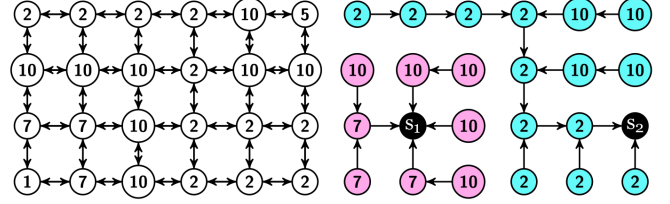
User interaction is thus often required to ensure high quality segmentation results, which are particularly essential when used as training data for deep learning models. However, the development of interactive methods for 3D multi-cell segmentation is challenging. From one aspect, the cognitive overload for a typical user is high when assessing a large number of 3D objects simultaneously. A divide-and-conquer strategy, where small portions of data are annotated one at a time, has shown to reduce such difficulty [10, 11]. Another difficulty is the interpretation of the 3D image content as projected on a flat screen. It is challenging to communicate visual information and make sound inferences of cell boundaries when only a small portion of the 3D object is displayed. Another challenge is to provide an efficient interactive segmentation algorithm with real-time feedback such that users can quickly assess the results of their editing actions and make sure they are correct. This is crucial to recruit and retain users and guarantee quality. Although tools such as Fiji [12], EpiTools [13], and Cell Profiler [14] allow interactive cell segmentation, they are hindered by the aforementioned problems, particularly to the limitation that only a single user has to tackle the segmentation task. We thus propose a collaborative web-based application, named SEGMENT3D, to the 3D image segmentation problem.

## 2. WEB-BASED 3D COLLABORATIVE IMAGE SEGMENTATION

SEGMENT3D is a collaborative 3D image segmentation tool designed to create good quality segmentation results and generate training data. It is not necessarily a replacement for automatic segmentation algorithms and tools but rather a complement to leverage automatic solutions by providing ways to assess and interactively correct erroneous results. Our application is easily accessible, running on most browsers, aiming to facilitate its dissemination and use by experts and non-experts alike. SEGMENT3D borrows ideas from and extends a Collaborative Segmentation application developed earlier to assist segmentation of 2D images of plant cells and other biological images [10].

### 2.1. Interactive segmentation method

SEGMENT3D provides two main modes of operation: interactive image segmentation and interactive correction of pre-existing results. Either way, the Image Foresting Transform by Seed Competition [15] (IFT-SC) is used for delineation. The IFT-SC interprets the 3D tile image  $I$  as a graph



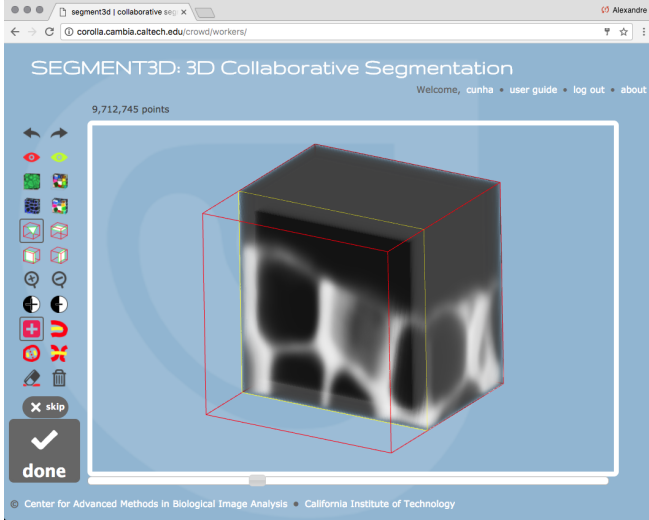
**Fig. 2:** The figure shows on the left a 4-connectivity image graph with nodes showing the gradient magnitude at the corresponding pixel in the image. In 3D we would have a 6-connectivity graph,  $\mathcal{A}_6$ . The gradient ultimately drives the segmentation by seed competition. Two seeds  $s_1$  and  $s_2$  are added in sequence, shown in black on the right panel. The IFT-SC then partitions the graph into an optimum-path forest by forcing the seeds to compete among themselves to conquer the nodes. This partition is the seeded watershed transform [15].

$G = (\mathcal{N}, \mathcal{A}_6)$ , in which  $\mathcal{N}$  is the set of voxels of  $I$  and  $\mathcal{A}_6$  is an adjacency relation connecting 6-adjacent neighbors. Let  $\pi_t = \langle t_1, t_2, \dots, t \rangle$  represent a simple path in  $G$  with terminus at node  $t$ , such that every  $(t_i, t_{i+1})$  is an arc of  $\mathcal{A}_6$ . A function  $f(\pi_t) = \max_{i=1,2,\dots,|\pi_t|-1} \{w(t_i, t_{i+1})\}$  attributes to every path  $\pi_t$  in  $G$  a cost given by the maximum weight (gradient)  $w(t_i, t_{i+1}) = I(t_{i+1})$  for arc  $(t_i, t_{i+1}) \in \mathcal{A}_6$  along the path, expressing the strength of connectivity that  $t$  has to the source of the path. By forcing the paths to be originated from a set  $\mathcal{S}$  of seed nodes, and minimizing an optimum connectivity map  $V(t) = \min_{\pi_t \in \Pi_t(G)} \{f(\pi_t)\}$  for every node  $t$  in the graph, the IFT-SC partitions  $G$  into an *optimum-path forest*, thereby computing the *seeded watershed transform* of the tile. Cells are composed of the optimum-path trees rooted at their seed voxels. The challenge lies in how to properly estimate the seed set, including its optimal location, to render a desired segmentation. Automatic methods rarely produce a faithful seed set.

### 2.2. User interaction

Once a user logs into the system, SEGMENT3D searches on the server and feeds an available tile that can be either segmented interactively from scratch or interactively corrected if a pre-segmentation exists. A tile is displayed inside a wireframe box for orientation, and is initially sliced by a clipping plane to show inner portions of the 3D data (Figure 3). The user can then perform segmentation by adding and removing scribbles. The result may be accepted by clicking on **(done)**. If judged too difficult or confusing to segment, the user can opt to **(skip)** and proceed to next tile.

For interactive segmentation, the seeds are simply the set of voxels under the user-drawn scribbles (Figure 4G). When the user edits scribbles the result is updated locally, in time proportional to the number of voxels in the affected region, using the differential IFT-SC [16] (DIFT-SC). This can be significantly smaller than the total number of voxels in the tile, thus its superior efficiency. We have simpli-



**Fig. 3:** The figure shows our web-based 3D collaborative image segmentation application, SEGMENT3D, depicting a tile of a 3D image as a self-contained box that the user can interactively segment from multiple angles. The tile can be sliced with a clipping plane to view its inner portions at any direction.

fied user interaction by making each newly added scribble to become the source of a new candidate cell region with a different label. Hence, although all scribbles are displayed with the same color, they each represent a different cell. To correct an existing segmentation, the seed set is initialized with the geodesic centers of the pre-segmented labels (cell candidates) for IFT-SC (Figure 4H). More sophisticated procedures may also be considered for non-cell objects [17]. Since the user may desire to extend, merge, or split existing cells, the system provides for both interaction modes in total eight operations that can be used when editing scribbles: add, remove, extend, split, merge, undo, redo, and erase all.

### 2.3. Implementation

We implemented the interface and segmentation methods in JavaScript to run natively on the web browser, aiming to alleviate the server-side programming and slow network traffic. For efficiency, the image is displayed in 3D using the WebGL framework through the Three.js library [18]. The backend server code to create tiles, manage user sessions, and store images and segmentation results in a database was written in Python/Django, adapted from [10] to 3D.

Ray casting is used to render the tile volume by treating the image as a 3D texture, inspired by the open source implementation from [19] (Figure 4A). A challenge we had to consider was how to display many cells at once for segmentation, such that a user can easily decide when cells are properly delineated. We offer 2D and 3D solutions: we can overlay the result directly onto the clipping plane, in a 2D slice fashion (Figure 4B,C), or render the segmentation borders or labels as 3D triangular meshes (Figure 4D,E). In the

former, the borders of the segmentation labels and the label image itself are loaded on to WebGL as textures for rendering during ray casting, emulating 2D image view modes.

Since cells have a well-defined polyhedral shape, the 3D visualization modes facilitate the perception of when errors occur. Due to the confocal acquisition limitations, cell membranes are stronger in the  $xy$  focal plane than in the  $xz$  and  $yz$  orthogonal planes. This often causes leaking in the  $z$  direction, which are more easily visualized when viewing the 3D meshes generated from the segmentation label or using the 2D borders in the right orientation. We use the Marching Cubes algorithm [20] to create cell surface meshes, whose visual appeal is improved by using a few iterations of Laplacian. The border of the segmentation and of each individual cell are rendered this way (Figure 4D,E).

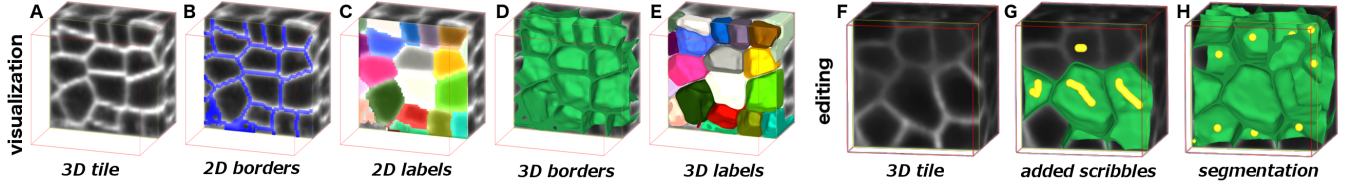
We keep track of all eight available operations to represent each label by a single scribble. For instance, the user may click on one label and drag the mouse onto another region to perform a label extension. The new scribble is then automatically connected to the scribble representing the first label. Regardless, each operation is always converted to a series of seed additions and removals for the DIFT-SC.

### 2.4. Consensus of tile results

We gather a few results for each tile offered for segmentation. The rationale is that users eventually make mistakes, so by having more than a single segmentation per tile we can potentially automatically eliminate errors by combining many results. We have adopted the STAPLE algorithm [21] to create a consensus segmentation for each tile, as it has shown to perform well in practical examples. The method reaches a consensus by using the expectation-maximization algorithm on the borders of the segmentation labels. The *FI-score* of the individual segmentation results are then assessed against the consensus to measure the user’s accuracy.

## 3. EXPERIMENTAL RESULTS

For an initial evaluation of SEGMENT3D, four workers (authors TS,JS,AF,AC) used the system and carried out three types of experiments when segmenting a 3D meristem. Our goal was to assess the correctness and usability of the system, in terms of the amount of user effort spent on i) correcting an existing pre-segmentation and ii) interactively segmenting an image from scratch, the two operating modes of SEGMENT3D. The first two experiments involved a  $128 \times 128 \times 200$  voxel image region from deeper layers of the meristem, where signal is weaker and causes most automated methods to perform poorly. It was subdivided into a total of 96 tiles with size  $40 \times 40 \times 40$ , with 10% overlap between them and a contextual border of also 10% – this border facilitates the user’s understanding of how the cells are laid out. We segmented the entire region using the method proposed in [2] and concluded that only 72 required interactive correction, after a conservative visual inspection.



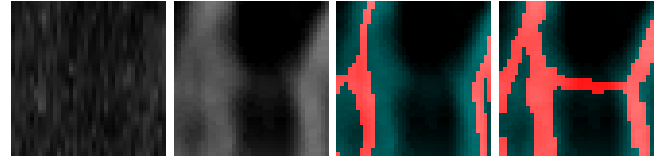
**Fig. 4:** Users can chose different visualization modes in 2D (panels B,C) and 3D (panels D,E) to evaluate the segmentation after editing. In the interactive segmentation of a 3D tile (F), a user draws scribbles (shown in yellow) over the cells on the clipping plane (G) and the current segmentation is updated and displayed in real time after each added scribble. Each scribble defines a single cell. Some seeds reconstructed at the geodesic centers of the labels for a given pre-segmentation are shown in (H).

**Table 1:** User effort and accuracy for all experiments.

User	Experiment 1: Pre-segmentation Correction		
	Time (s)	# interactions	F1-score
TS	198 ± 158	1.7 ± 1.7	0.93 ± 0.03
AC	204 ± 96	3.3 ± 3.7	0.93 ± 0.03
JS	105 ± 70	1.5 ± 1.5	0.93 ± 0.03
<b>Avg.</b>	<b>171 ± 72</b>	<b>2.2 ± 2.1</b>	<b>0.93 ± 0.03</b>
Experiment 1: Interactive Segmentation			
TS	185 ± 53	12.2 ± 4.0	0.93 ± 0.02
AC	335 ± 125	12.6 ± 4.2	0.93 ± 0.02
JS	138 ± 63	10.7 ± 3.8	0.92 ± 0.03
<b>Avg.</b>	<b>219 ± 68</b>	<b>11.8 ± 3.4</b>	<b>0.93 ± 0.02</b>
Experiment 2: Pre-segmentation Correction			
TS	165 ± 105	2.0 ± 2.0	0.95 ± 0.02
AC	203 ± 111	2.8 ± 2.8	0.95 ± 0.03
JS	115 ± 101	1.5 ± 1.5	0.95 ± 0.03
<b>Avg.</b>	<b>163 ± 65</b>	<b>2.3 ± 2.0</b>	<b>0.95 ± 0.02</b>
Experiment 3: Pre-segmentation Correction			
TS	357 ± 863	4.0 ± 3.7	0.97 ± 0.02
AC	319 ± 133	7.1 ± 4.2	0.97 ± 0.03
AF	341 ± 409	2.8 ± 6.1	0.95 ± 0.04
<b>Avg.</b>	<b>339 ± 733</b>	<b>4.1 ± 4.4</b>	<b>0.96 ± 0.02</b>

In Experiment 1, we compared the two previously stated operating modes i) and ii). We selected 15 evenly distributed tiles from the 72 that were considered wrong to be interactively corrected and also interactively segmented from scratch. We measured user effort by evaluating the total amount of the user’s time that was required to segment a tile and also the required number of interactions (i.e., operations such as add and delete). Table 1 reveals that interactively segmenting the tiles from scratch took on average 219s vs. 171s for interactive correction, while the number of interactions was 11.8 for segmentation from scratch vs. 2.2 for correction. Hence, interactive correction took 22% less time and 81% less user interaction. However, since each cell required at least one scribble to be segmented, this may indicate that it takes less cognitive effort to segment from scratch than to correct, since the decrease in user interaction was not the same as in the user’s time. The extra time was probably making sure scribbles were leading to correct results. Regarding user accuracy, the average F1-score was 0.93 for all users, w.r.t. the merger of their results. This value may be explained by the fact that the tiles were selected from deeper layers of the meristem, in which it is harder to obtain a user consensus. Figure 5 depicts the final result obtained by correcting the pre-segmentation.

In Experiment 2, we asked the users to correct the remaining 57 tiles. Our goal was to verify if the patterns observed in the first experiment were the same. From Table 1, it took on average 163s to make corrections in each tile, consuming about 2.3 interactions, similarly to before. In contrast, user accuracy was higher, since cells from upper layers were present.



**Fig. 5:** *Left to right:* original tile seen on  $xz$  plane, enhanced version for visualization, incorrect pre-segmentation, and correction obtained after the STAPLE consensus for three individual user results.

In the final experiment, three workers assessed and corrected 200 automatically segmented tiles [2] with size  $60 \times 60 \times 60$ , from an image region with size  $256 \times 256 \times 508$ . This region was selected to encompass cells ranging from the upper layers L1 and L2 to the bottom of the meristem. The larger size of the tiles doubled the amount of time required for segmentation, on average 339 seconds, as well as the number of interactions, on average 4.1, w.r.t. Experiment 2. The accuracy also increased, once again due to the presence of cells from the upper layers with more clearly defined walls. Note that not all users segmented all 200 tiles, so the results are weighted averages.

## 4. CONCLUSIONS

We have presented SEGMENT3D, a web-based 3D collaborative image segmentation tool. SEGMENT3D was designed for cell segmentation, specifically for those of the shoot apical meristem, and can be used both to correct the result of automated methods and to segment an image from scratch interactively. Future works involve improving visualization of segmentation, testing the tool with more users, adapting the interface to support mobile device screens, and devising automatic methods for determining the regions of a pre-segmentation label that must be interactively corrected.



## 5. REFERENCES

- [1] O. Hamant, M. G. Heisler, H. Jönsson, P. Krupinski, M. Uyttewaal, P. Bokov, F. Corson, P. Sahlin, A. Boudaoud, E. M. Meyerowitz, and Y. Couder, “Developmental patterning by mechanical signals in *Arabidopsis*,” *Science*, vol. 322, no. 5908, pp. 1650–1655, 2008.
- [2] J. Stegmaier, T. V. Spina, A. X. Falcão, A. Bartschat, R. Mikut, E. Meyerowitz, and A. Cunha, “Cell segmentation in 3D microscopy images using supervoxel merge-forests with CNN-based hypothesis selection,” in *ISBI*, Washington, USA, 2018.
- [3] R. Fernandez, P. Das, V. Mirabet, E. Moscardi, J. Traas, J.-L. Verdeil, G. Malandain, and C. Godin, “Imaging plant growth in 4D: Robust tissue reconstruction and lineaging at cell resolution,” *Nat. Methods.*, vol. 7, no. 7, pp. 547–553, 2010.
- [4] A. L. Cunha, A. H. Roeder, and E. M. Meyerowitz, “Segmenting the sepal and shoot apical meristem of *Arabidopsis thaliana*,” in *EMBC*, 2010, pp. 5338–5342.
- [5] F. Federici, L. Dupuy, L. Laplaze, M. Heisler, and J. Haseloff, “Integrated genetic and computation methods for in planta cytometry,” *Nat. Methods.*, vol. 9, no. 5, pp. 483–485, 2012.
- [6] P. B. de Reuille, A.-L. Routier-Kierzkowska, D. Kierzkowski, G. W. Bassel, T. Schüpbach, G. Tauriello, N. Bajpai, S. Strauss, A. Weber, A. Kiss, et al., “Morphographx: A platform for quantifying morphogenesis in 4D,” *eLife*, vol. 4, e05864, 2015.
- [7] V. Jain, H. S. Seung, and S. C. Turaga, “Machines that learn to segment images: a crucial technology for connectomics,” *Curr. Opin. Neurobiol.*, vol. 20, no. 5, pp. 653–666, 2010.
- [8] S. E. A. Raza, L. Cheung, D. Epstein, S. Pelengaris, M. Khan, and N. M. Rajpoot, “Mimo-net: A multi-input multi-output convolutional neural network for cell segmentation in fluorescence microscopy images,” in *ISBI*. IEEE, 2017, pp. 337–340.
- [9] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, “3D U-Net: learning dense volumetric segmentation from sparse annotation,” in *MICCAI*, 2016, pp. 424–432.
- [10] R. Barreto, P. Barbosa, W. Oliveira, I. Tsang, E. Meyerowitz, and A. Cunha, “Collaborative Segmentation: <http://cose.cacr.caltech.edu/>,” 2013.
- [11] J. S. Kim, M. J. Greene, A. Zlateski, K. Lee, M. Richardson, S. C. Turaga, M. Purcaro, M. Balkam, A. Robinson, B. F. Behabadi, et al., “Space-time wiring specificity supports direction selectivity in the retina,” *Nature*, vol. 509, no. 7500, pp. 331–336, 2014.
- [12] J. Schindelin, I. Arganda-Carreras, E. Frise, V. Kaynig, M. Longair, T. Pietzsch, S. Preibisch, C. Rueden, S. Saalfeld, B. Schmid, J.-Y. Tinevez, D. J. White, V. Hartenstein, K. Eliceiri, P. Tomancak, and A. Cardona, “Fiji: an open-source platform for biological image analysis,” *Nat. Methods.*, vol. 9, pp. 676–682, Jun 2012.
- [13] D. Heller, A. Hoppe, S. Restrepo, L. Gatti, A. L. Tournier, N. Tapon, K. Basler, and Y. Mao, “EpiTools: An open-source image analysis toolkit for quantifying epithelial growth dynamics,” *Dev. Cell*, vol. 36, no. 1, pp. 103–116, 2016.
- [14] A. E. Carpenter, T. R. Jones, M. R. Lamprecht, C. Clarke, I. H. Kang, O. Friman, D. A. Guertin, J. H. Chang, R. A. Lindquist, J. Moffat, P. Golland, and D. M. Sabatini, “CellProfiler: image analysis software for identifying and quantifying cell phenotypes,” *Genome Biology*, vol. 7, no. 10, R100, Oct 2006.
- [15] A. X. Falcão, J. Stolfi, and R. A. Lotufo, “The image foresting transform: theory, algorithms, and applications,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26(1), pp. 19–29, 2004.
- [16] A. X. Falcão and F. P. G. Bergo, “Interactive volume segmentation with differential image foresting transforms,” *IEEE Trans. Med. Imag.*, vol. 23, no. 9, pp. 1100–1108, sept. 2004.
- [17] A. Tavares, P. A. V. Miranda, T. V. Spina, and A. X. Falcão, “A supervoxel-based solution to resume segmentation for interactive correction by differential image-foresting transforms,” in *Mathematical Morphology and Its Applications to Signal and Image Processing: 13th International Symposium*. Springer, 2017.
- [18] R. Cabello, “Three.js: <https://threejs.org/>.”
- [19] L. Barbagallo, “WebGL volume rendering made simple: <http://www.lebarba.com/>.”
- [20] Stenkoski, “Marching cubes algorithm: <http://stenkoski.github.io/Three.js/Marching-Cubes.html>.”
- [21] S. K. Warfield, K. H. Zou, and W. M. Wells, “Simultaneous truth and performance level estimation (STAPLE): an algorithm for the validation of image segmentation,” *IEEE Trans. Med. Imag.*, vol. 23, no. 7, pp. 903–921, July 2004.