

Learning Context Cues for Synapse Segmentation

Carlos Becker, Karim Ali, Graham Knott, and Pascal Fua

Abstract—We present a new approach for the automated segmentation of synapses in image stacks acquired by Electron Microscopy (EM) that relies on image features specifically designed to take spatial context into account. These features are used to train a classifier that can effectively learn cues such as the presence of a nearby post-synaptic region. As a result, our algorithm successfully distinguishes synapses from the numerous other organelles that appear within an EM volume, including those whose local textural properties are relatively similar. Furthermore, as a by-product of the segmentation, our method flawlessly determines synaptic orientation, a crucial element in the interpretation of brain circuits. We evaluate our approach on three different datasets, compare it against the state-of-the-art in synapse segmentation and demonstrate our ability to reliably collect shape, density, and orientation statistics over hundreds of synapses.

Index Terms—Synapse Segmentation, AdaBoost, Pose-indexing, Electron Microscopy, Connectomics.

I. INTRODUCTION

New imaging technologies have been a key driver of recent advances in neuroscience. In particular, Focus Ion Beam Scanning Electron Microscopy (FIBSEM) can now deliver a $5nm$ nearly isotropic sampling and produce image stacks that reveal very fine neuronal structures. Stacks such as the one depicted in Fig. 1(a) can be used to analyse a variety of components that are critical to understanding brain function. For example, evidence obtained from EM images suggests that the size, shape and distribution of synapses vary during the course of normal life but also under specific pathological conditions [27]. Similarly, EM imaging has provided new insights into synaptic signaling [21] and its relationship to mitochondrial activity [25] as well as to some neuro-degenerative diseases [19], [32].

Currently, analysis is carried out by manually segmenting the various structures of interest using tools such as Fiji [33]. This is not only a tedious and time consuming process but also an error-prone one. Thus, while the growing number of EM datasets offers a unique opportunity to unlock new concepts of neuronal function, the required amount of human effort remains a major bottleneck. There has therefore been great interest in automating the process.

EM data poses unique challenges for automatic segmentation algorithms in part because the volumes are heavily

cluttered with structures that exhibit similar textures and are therefore difficult to distinguish based solely on local image statistics. The synapse segmentation task is well illustrative of this difficulty. As shown in Fig. 1(b), a synapse can only be distinguished from other structures by relying on contextual clues such as the presence of a nearby cluster of vesicles. Well-established criteria enable human experts to identify synapses: densities on the pre-synaptic and post-synaptic membranes, vesicles in the pre-synaptic axon terminal and finally a synaptic cleft, as shown in Fig. 1(c). It is therefore essential for an automatic segmentation method to proceed in a similar fashion.

Current methods for automated synapse detection either require first finding cell membranes [28], or operate on individual slices [14], thus failing to leverage the 3D structure of the data. By contrast, the recent method of [20] operates entirely in 3D. However, the latter does not exploit the contextual clues that allow human experts to distinguish synaptic clefts from other membranes exhibiting similar textures such as myelin sheaths. More generally, though progress has been made towards the segmentation of various organelles from EM stacks, context has yet to be exploited in a meaningful way.

In this work, we propose an approach designed to take such contextual cues into account and emulate the human ability to distinguish synapses from regions that merely share a similar texture. Our method is fully automated, processes the data directly in 3D and is specifically designed to leverage context cues. We run various filters over the EM stack and compute our features over arbitrarily sized cubes placed at arbitrary locations inside an extended neighborhood of the voxel to be classified. As this generates a feature representation for each voxel in the order of a hundred thousand, we rely on Boosting to select the relevant filter channels as well as the relevant cube locations and sizes. The resulting classifier is thus highly flexible, able to utilize context from a high variety of regions in the neighborhood of the voxel of interest.

We apply our classifier to the synapse segmentation task and compare our results with the state-of-the-art synapse segmentation method of Kreshuk et al. [20], a fully automated 3D approach which does not utilize context. By honing in on the presence of pre-synaptic vesicles and post-synaptic regions in addition to local texture, our method significantly outperforms the approach of [20]. As an added benefit, our method also flawlessly identifies synaptic orientation, a key and hitherto unexplored task.

We validate our method on three datasets obtained from three different regions of the adult mammalian brain: the Somatosensory cortex, the Hippocampus, and the Cerebellum. We demonstrate our ability to automatically process large EM stacks, reliably collect density, shape and orientation statistics from hundreds of synapses.

C. Becker and P. Fua are in the Computer, Communication, and Information Sciences Department, EPFL; K. Ali is with the Electrical Engineering and Computer Sciences Department at the University of California, Berkeley and with the Computer Science Department at the University of Massachusetts, Lowell. G. Knott is with the Interdisciplinary Center for Electron Microscopy, EPFL, Lausanne CH-1015 Switzerland. E-mail: firstname.lastname@epfl.ch.

Manuscript received December 21, 2012. Revised Apr 21, 2013.

This work was supported in part by the EU ERC grant MicroNano.

Copyright (c) 2012 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

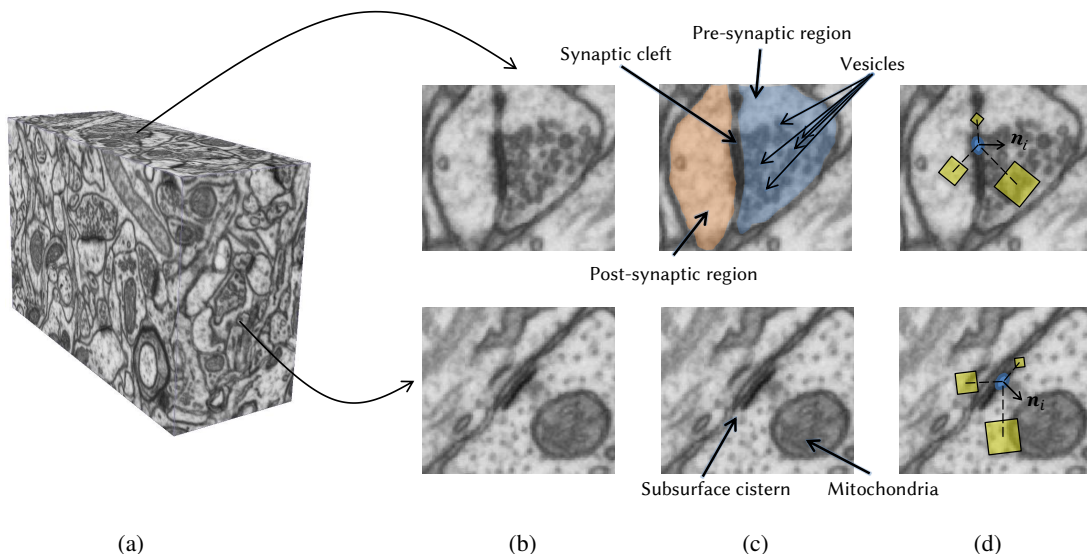


Fig. 1. Importance of context for synapse segmentation. (a) A FIBSEM stack with $5nm$ resolution in all three directions. (b) Two close-ups on regions containing wide dark structures that could potentially be synaptic clefts. However, only the one at the top really is one, as evidenced by the small spheres on its right, known as vesicles. These denote the presynaptic region and are missing from the bottom image. The diagram on top of (c) depicts the three elements that evidence the existence of a synapse, namely the synaptic cleft and the pre-synaptic and post-synaptic regions. The latter are labeled in blue and red, respectively. (d) The features we use are designed to capture this fact. To classify a voxel (blue), we consider sums over image cubes (shown as yellow squares) whose respective positions are defined relative to an estimated normal vector \mathbf{n}_i .

II. RELATED WORK

Prior work on segmenting neuronal structures from EM volumes has covered a range of approaches from early attempts at full manual tracing [26], [11], [10] to semi-automatic methods requiring user initialization [7], [22], and lately fully-automated methods [23], [16].

Manual segmentation has clear and well understood limitations for the analysis of EM stacks. One notable example can be found in [37], where the successful segmentation of the nervous system of a nematode worm, containing only 302 neurons, necessitated a sustained effort over a ten year period. The need for expert knowledge and the growing size of EM datasets render manual segmentation impractical and highlight the need for automation.

Semi-automated methods based on active contours and level sets [4], [22], [7], [31], [17], [24] as well as graphcuts [29] have achieved some measure of success on EM images. However, these methods require careful manual initialization of each object to be segmented, which is done by supplying seed points and tuning various parameters. Though active interactions and feedback may in the long term prove essential to the successful large-scale segmentation of EM stacks, the amount of user input required by these methods remains prohibitively high. Ultimately, when applied to large EM data sets containing millions or even billions of structures, these semi-automatic segmentation methods suffer from the same intractability issue as their manual counterparts.

Recent research has focused on methods relying on machine learning, requiring little to no user interaction. Among those, several follow the same methodology by performing a segmentation in individual 2D EM slices before linking the segmented regions across slices in 3D. For instance, in [28], a perceptron operating on Hessian ridge was shown to provide promising

results in segmenting membranes. However, in addition to the post-processing required to link membranes across the various 2D slices, this method also suffers from the need to remove internal sub-cellular structures from the segmentation result. In [36] a Boosted classifier operating on Gabor filter-based features is used to segment mitochondria in 2D slices while a connected component analysis generates the desired 3D segmentation. In [30], 2D mitochondria segmentation followed by simple 3D interpolation is obtained from a number of classifiers including Adaboost, Support Vector Machines and Nearest Neighbor trained on Texton features. Finally, in [18], a random forest classifier trained with Haar-like features is used to detect membranes in individual EM slices, while a graph cut optimization is used to enforce perceptual grouping and 3D continuity constraints.

While slice-by-slice methods have been shown to provide both reasonable segmentation results and computational savings, they fail to leverage the consistency of the structures in all three dimensions. This situation arises in part from the fact these approaches were designed for anisotropic EM modalities, such as transmission electron microscopy (TEM). Though the reduced resolution in the z-direction makes slice-by-slice approaches a reasonable choice, recent works [16], [15], [3], [34] have demonstrated the benefits of processing the data directly in 3D even in highly anisotropic image stacks. More generally, the appearance of objects in 2D slices can be significantly altered depending on the 3D orientation of the object with respect to the stack axes. Given that such variability is far less pronounced when observing the objects in 3D, processing EM stacks slice by slice significantly complexifies the segmentation task and can prove exceedingly detrimental when compared to direct 3D processing. Such a strategy is clearly foolhardy in the case of 2D images, where

the analogue would consist of a column by column or a row by row processing.

For these reasons, a number of works have addressed the segmentation of various neuronal structures directly in 3D. For example, [16] uses a multilayer convolutional artificial neural network (ANN) to segment neuronal membranes. By employing a convolutional ANN, [16] removes the need to hand design features and instead learns the necessary filters directly from the data. Andres et al. [3] propose a bottom-up hierarchical segmentation framework that uses a Random Forest classifier and watersheds to segment neural tissue. Though both of these methods produce excellent membrane segmentation results, they are designed for datasets prepared with an extra-cellular stain which highlights cell membranes while suppressing the various intracellular structures. In [34], an affinity graph that can be paired with standard partitioning algorithms is generated using a convolutional ANN. Much as in [16], this method learns both the features as well as the decision function directly from the data.

Even though progress has been made towards the automatic segmentation of neural structures, none of the aforementioned methods, whether operating in 2D or in 3D, can reliably segment objects such as synapses, which are characterized by specific arrangements of structures in addition to local textural cues. Though current algorithms generally compute features in a neighborhood around the voxel of interest, they do not exploit context in a meaningful way: features are either pooled into global histograms [23], [30], are computed in regions centered around the voxel of interest [36], [16], [3], [18], or operate on a limited neighborhood around the voxel of interest [16], [34]. The resulting classifiers are therefore unable to hone in on arbitrary localized context cues.

The importance of context for the purposes of segmentation has been highlighted by a few attempts to leveraging ad-hoc and heuristic contextual cues to improve segmentation. For instance, [36] uses vesicle detection cues to suppress false alarms on vesicle clusters that can interfere with mitochondria segmentation, while [17], [35] propose to sample features in a 2D stencil neighborhood around the pixel of interest. By allowing the classifier to measure features computed at various locations in addition to the pixel of interest, [17], [35] are able to identify membranes at regions of minor discontinuities. However, by relying on a pre-determined set of locations from which features can be sampled, these approaches strongly restrict the use of context. By contrast, our approach learns the relevant context automatically, overcoming these limitations.

Closest to our work is the state-of-the-art method of Kreshuk et al. [20], specifically targeted to synapse segmentation in isotropic image stacks. This approach relies on voxel-wise classification, training a Random Forest classifier that employs a set of pre-defined features such as smoothed gradient magnitudes, Laplacian of Gaussians and Hessian and Structure Tensor eigenvalues, evaluated at the voxel of interest. Therefore, context can only be captured through the isotropic Gaussian filters applied to the image stack, ignoring the presence of the asymmetric and localized context information generated by the pre-synaptic and post-synaptic regions.

III. PROPOSED APPROACH

We describe here our approach, which was first introduced in [5]. Let $x \in \mathcal{X} = [0, 1]^{W \times H \times D}$ be an EM volume of width W , height H and depth D . Voxels are indexed by $i \in \{1, \dots, W \times H \times D\}$, and the location of each voxel is designated $\ell_i \in \mathbb{N}^3$. Our goal is to find a function $\varphi(x, \ell_i) \in \mathbb{R}$ that yields high scores at locations ℓ_i in the volume that are part of synaptic tissue, and lower score values at those that are not.

As shown in Fig. 1(b), it can be difficult to distinguish synapses from other structures based solely on local texture. Human experts confirm their presence by looking nearby for post-synaptic densities and vesicles. This protocol cannot be emulated simply by measuring filter responses at the target voxel [20], pooling features into a global histogram [23], [30] or relying on hand-determined locations for feature extraction [17], [35].

To emulate the human ability to identify synapses, we design features, termed context cues, that can be extracted in any cube contained within a large volume centered on the voxel to be classified at ℓ_i , as depicted in Fig. 2(b). They are computed in several image channels using a number of Gaussian kernels, as shown in Fig. 3. As will be discussed in Sec. IV-B this yields more than 100,000 potential features. We therefore rely on AdaBoost [13] to select the most discriminative ones.

Given that synapses have arbitrary 3D orientations, we ensure that our context cues are computed at consistent locations across differently oriented synapses. We rely on the pose-indexing framework of [12], [2] to enforce this consistency.

In the remainder of this section, we describe briefly the main structure of our context features. Their implementation is discussed in more detail later in § IV.

A. Context Cue Location

Let us consider a voxel located at ℓ_i and an associated unit vector $\mathbf{n}_i \in \mathbb{R}^3$, as in Fig. 2(a). This unit vector is computed so that it is normal to the synaptic cleft. Let

$$\mathbf{c}_p \in \mathbb{R}^3, \quad p = 1, \dots, P \quad (1)$$

denote a set of P locations expressed in the common x_o, y_o, z_o reference frame shown at the center of Fig. 2(a). These locations are translated and rotated to occur at consistent locations relative to a target voxel by defining,

$$\mathbf{c}_p^{\ell_i} = \ell_i + \mathbf{R}(\ell_i)\mathbf{c}_p \quad (2)$$

where $\mathbf{R}(\ell_i)$ is a 3×3 rotation matrix such that $\mathbf{R}(\ell_i)(0, 0, 1)^T = \mathbf{n}_i$.

B. Context Cue Features

Given the $\mathbf{c}_p^{\ell_i}$ locations of Eq. 2, our goal now is to compute image statistics inside cubic neighborhoods $\mathcal{N}_r(\mathbf{c}_p^{\ell_i})$ of edge length $2r$ centered around these locations, such as those depicted in Fig. 2(b).

To this end, we process the original EM volume by convolving it with a number of different filters as depicted in Fig. 3. Each of the resulting data cubes, in addition to the

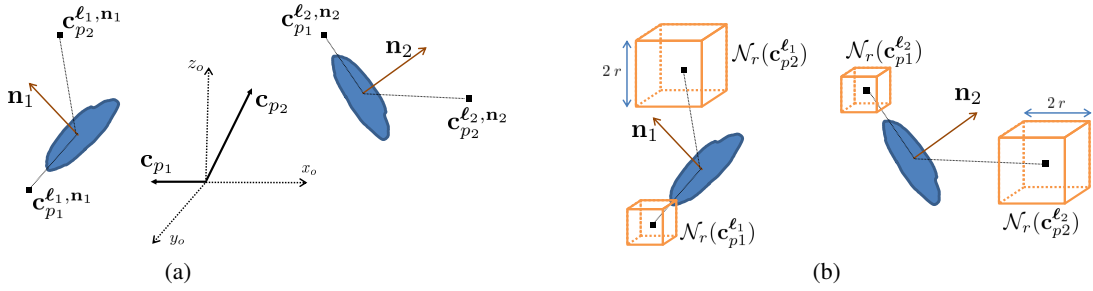


Fig. 2. Context cues representation: (a) context cue locations \mathbf{c}_p in the global coordinate system x_o, y_o, z_o are rotated according to the orientation estimate of the voxel of interest \mathbf{n}_i to yield locations $\mathbf{c}_p^{\ell_i}$ that are consistent. (b) At each of these locations, image channels are summed over cubes of radius r around their center. Our approach employs AdaBoost to select the most discriminative features for synapse segmentation.

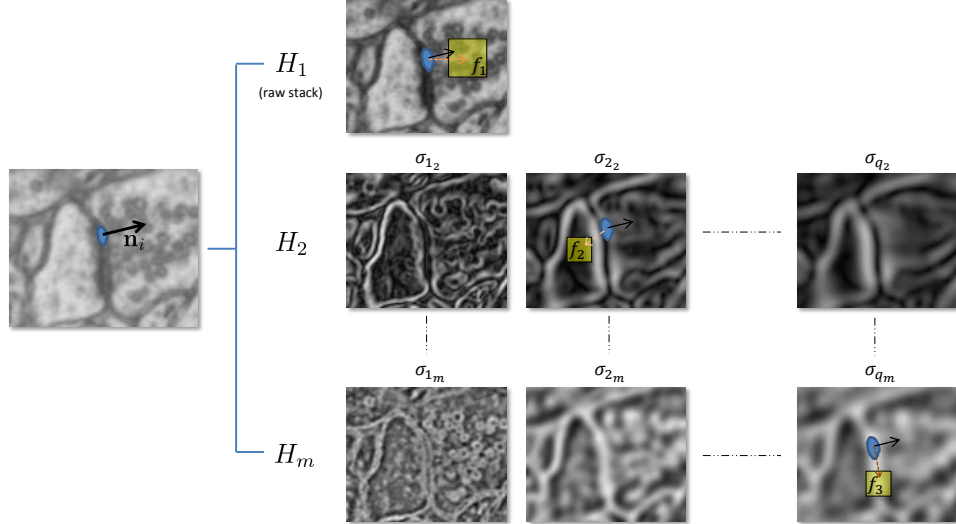


Fig. 3. Image channels. The image is convolved with different filters and features are computed within the yellow rectangles whose coordinates are expressed with respect to the location of the voxel to be classified and the local orientation vector \mathbf{n} . Each H_i line depicts a specific channel designed to capture different statistical characteristics.

original one, is treated as a data channel m , and is smoothed using several isotropic Gaussian kernels with variance σ_{n_m} . We denote the gray levels in the resulting data volumes as

$$H_{m,\sigma_{n_m}}(x, \mathbf{z}) \in \mathcal{X}, \quad (3)$$

where x is the original EM volume and \mathbf{z} represents the 3D location. We take context cue features to be

$$f_{\mathbf{c}_p, m, \sigma_{n_m}, r}(x, \ell_i) = \sum_{\mathbf{z} \in \mathcal{N}_r(\mathbf{c}_p^{\ell_i})} H_{m,\sigma_{n_m}}(x, \mathbf{z}). \quad (4)$$

In other words, we sum the smoothed channel output over the cubic boxes centered at all $\mathbf{c}_p^{\ell_i}$ for all possible values of m , σ_n , and r . This yields a set of K features, which we will denote for simplicity

$$f_k(x, \ell_i), \quad k = 1, \dots, K, \quad (5)$$

and which we use for classification purposes as explained next.

C. Contextual Classifier

Given the context features f_k , we create decision stumps by simple thresholding and combine these stumps via a standard

AdaBoost procedure [13] into a strong learner of the form

$$\varphi(x, \ell_i) = \sum_{t=1}^T \alpha_t \mathbf{1}_{\{f_t(x, \ell_i) > \rho_t\}}. \quad (6)$$

AdaBoost [13] solves for Eq. (6) in a stage-wise manner, building it one term at a time by greedy minimization of an empirical exponential loss. Our resulting classifier is pose-indexed as its constituent features translate and rotate according to ℓ_i and $\mathbf{R}(\ell_i)$ respectively.

IV. IMPLEMENTATION DETAILS

In what follows, the specifics of our implementation are provided. We follow the same notation as in §III and summarize all algorithm parameters in Table I.

A. Image Channels

We broadly follow the methodology employed by [20] and process each EM volume with several different filters, resulting in different data channels such as those of Fig. 3. More specifically, we use:

- The identity (original stack)

TABLE I
ALGORITHM PARAMETERS AND DEFAULT VALUES.

Parameter	Symbol	Default value
AdaBoost iterations	T	2000
Weighting-by-resampling ratio	$M = \frac{N_{\text{neg}}}{N_{\text{pos}}}$	2
Number of weak learners explored per iteration	Ω	4000
Context cue maximum distance	$\ \mathbf{c}_p\ _{\text{max}}$	40 voxels
Context cue distance quantization steps	$Q_{\ \mathbf{c}\ }$	6
Context cue maximum box size	r_{max}	20 voxels
Context cue box size quantization steps	Q_r	11
Context cue φ quantization steps	Q_φ	9
Context cue θ quantization steps	Q_θ	9
Supervoxel seed size	SV_n	2 voxels
Supervoxel cubeness	SV_m	16
Hessian scale for orientation estimation	$\sigma_{\text{Ho}} = \frac{w_s}{2\sqrt{2}}$	18 nm

- Gradient Magnitude,
- Structure Tensor Eigenvalues

The Gradient Magnitude channels are computed by first smoothing the image with isotropic Gaussian filters of $\sigma_{\text{GM}} = \{1.0, 1.6, 3.5, 5.0\}$. On the other hand, structure tensor eigenvalues are computed at $\rho_{\text{ST}} = \{1.0, 1.6, 3.5, 5.0\}$ with $\sigma_{\text{ST}} = \frac{\rho_{\text{ST}}}{2}$. No smoothing is applied to the identity channel. Given that there are three eigenvalues per structure tensor, this results in a total of 17 different filtered versions of the original EM volume, which are available to construct the weak learners.

Note that, in contrast with our first approach [5] and with the work of [20], we opted for a reduced set of channels. An important observation regarding our framework lies in the fact that our features sum the response of a filter inside a box: In the case of *linear* filters and boxes larger than a single voxel, these sums can be directly computed in the original image channel, up to a scale factor and additional negligible filter border effects. We were therefore able to eliminate the Gaussian smoothing over the original image as well as the Laplacian of Gaussian channels. Several experiments confirmed this observation, showing no performance loss when using the reduced set of channels w.r.t. the full set used in [5]. Further experiments allowed us to eliminate the Hessian Eigenvalue channel, which was found to be uninformative for our framework as well.

B. Context Cue Parametrization

Context cue locations $\mathbf{c}_p, p = 1, \dots, P$ in the common reference frame are parametrized in spherical coordinates as

$$\mathbf{c}_p = \left(\|\mathbf{c}_p\| \cos \varphi_p \sin \theta_p, \|\mathbf{c}_p\| \sin \varphi_p \sin \theta_p, \|\mathbf{c}_p\| \cos \theta_p \right)$$

with $0 \leq \|\mathbf{c}_p\| \leq \|\mathbf{c}_p\|_{\text{max}}$, $0 \leq \theta_p \leq \pi$ and $0 \leq \varphi_p \leq 2\pi$. The parameter space is quantized uniformly in $Q_{\|\mathbf{c}\|}$, Q_θ and Q_φ bins respectively. This is also applied to the cube edge length $2r$, which is quantized in Q_r steps with $\frac{1}{2} \leq r \leq r_{\text{max}}$.

To compute our context cue features of Eq. 4 efficiently, we employ 3D integral images for each channel H_{m,σ_n} . This

allows us to compute the sum of any channel inside an arbitrary cube in constant time. Note that to allow for maximum consistency across the differently oriented synapses, the cubes over which sums of image channel values are computed should also be pose-indexed and hence rotate according to $\mathbf{R}(\ell_i)$. However, this would either impose a heavy memory burden if rotated integral volumes were used or a large computational cost otherwise. For this reason, we do not pose-index the cubes and restrict the boxes over which channel voxels are summed to be axis-aligned, as shown in Fig. 2(b). Note that, in practice, the axis-aligned cubes overlap significantly with their rotated counterparts and therefore provide a fairly good approximation.

C. Estimating Synaptic Cleft Orientation

The context cues defined above are located relative to a normal estimate \mathbf{n}_i , which induces the rotation matrix $\mathbf{R}(\ell_i)$ of Eq. 2, such that

$$\mathbf{R}(\ell_i)(0, 0, 1)^T = \mathbf{n}_i \quad (7)$$

Let $\{\xi_1(\ell_i), \xi_2(\ell_i), \xi_3(\ell_i)\}$ be the Eigenvectors of the Hessian matrix at the voxel of interest, ordered by increasing magnitude of their respective eigenvalues. Hence, $\xi_3(\ell_i) = \mathbf{n}_i$ corresponds to the eigenvector with the highest-magnitude eigenvalue, and is perpendicular to the synaptic cleft, assuming there is one at ℓ_i . We write

$$\mathbf{R}(\ell_i) = \begin{pmatrix} \omega_1(\ell_i) & \omega_2(\ell_i) & \omega_3(\ell_i) \end{pmatrix} \quad (8)$$

with

$$\omega_3(\ell_i) = \xi_3(\ell_i) \quad (9)$$

$$\omega_2(\ell_i) = \text{sign} [D_{\xi_2(\ell_i)}(\ell_i)] \xi_2(\ell_i) \quad (10)$$

$$\omega_1(\ell_i) = \omega_3(\ell_i) \times \omega_2(\ell_i) \quad (11)$$

where $D_{\mathbf{u}}(\mathbf{x})$ is the directional derivative of the image along \mathbf{u} at location \mathbf{x} . As shown in Fig. 4, introducing Eq. 10 makes $\omega_2(\ell_i)$ point towards the outside of the synaptic cleft when ℓ_i is close to the border of the synapse. The motivation behind

the latter is that $\mathbf{R}(\ell_i)$ in Eq. 7 is only defined up to a rotation in the $\omega_1(\ell_i)/\omega_2(\ell_i)$ plane. However, it is preferable to define a rotation matrix that is consistent, particularly at ℓ_i close to the edge of the synapses. Towards the center of the synaptic cleft, there exists a rotational ambiguity in the $\omega_1(\ell_i)/\omega_2(\ell_i)$ plane which can in fact be ignored due to the symmetry of synaptic structures there. Both the Hessian and the derivatives $D_{\mathbf{u}}(\mathbf{x})$ are computed at scale $\sigma_{\text{Ho}} = \frac{w_s}{2\sqrt{2}}$ where w_s is the average synaptic cleft width, estimated once per EM volume from a single slice.

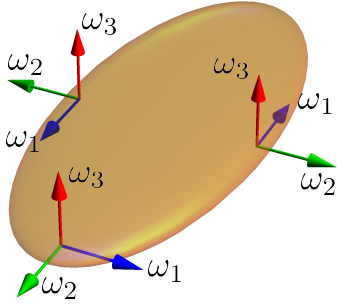


Fig. 4. Coordinate system versors (unit vectors) computed for $\mathbf{R}(\ell_i)$ at different locations of a simulated synapse. $\omega_1(\ell_i)$, $\omega_2(\ell_i)$, $\omega_3(\ell_i)$ shown in blue, green and red respectively. Note that $\omega_2(\ell_i)$ (green) always points towards the outside of the synapse, which comes as a consequence of Eq. 10.

D. Learning Method

To make training computationally tractable, we specialize the AdaBoost learning procedure [13] as follows:

1) *Weighting-by-sampling*: Denote $W = \{w_1 \dots w_N\}$ and $Y = \{y_1 \dots y_N\}$ the weights and labels for each training sample, with $1 \leq i \leq N$. Assume that the weights have been normalized such that $\sum_{i=1}^N w_i = 1$. At each AdaBoost iteration, instead of searching for the weak learner that minimizes the weighted error over all N training samples, we employ the weighting-by-sampling scheme [12], [2] to approximate the distribution W . This is done by finding the weak learner that minimizes a weighted error computed on a subset S of the training data, formally

$$S = \{S_P, S_N\} \quad (12)$$

where S_P comprises all the positive samples and S_N is a subset of the negative samples, obtained by weighted sampling with replacement according to the weights W . The weight of each sample in S_P is its respective weight in W , while the samples in S_N are assigned a constant weight $\frac{\sum_{y_i=-1} w_i}{\|S_N\|}$.

In situations where the amount of negative samples outnumbers the number of positives, approaches such as weighting-by-sampling can reduce training time significantly. We call $M = \frac{\|S_N\|}{\|S_P\|}$ the ratio between the number of negative and positive samples selected by weighting-by-sampling, which is a parameter for our algorithm. In our experiments we have observed that segmentation performance is robust against the value of M . We set $M = 2$, which yields faster training without decreasing the performance of the final classifier.

2) *Random Weak Learner Search*: Due to the large number of possible weak learners $f_k(x, \ell_i)$, $k = 1, \dots, K$, it is impractical to explore them all at each AdaBoost iteration. Instead, we only explore a subset of size Ω , obtained by randomly sampling, at each Boosting iteration, from the pool of K possible weak learners. This also speeds up training significantly. We have experimented with different values of Ω and observed that segmentation performance is fairly independent of its value. For all the results presented here we used $\Omega = 4000$.

E. Pose Indexing

1) *Pose Annotations*: Learning our pose-indexed classifier requires annotated training data. Since our contextual features are computed both for a given location and orientation, our training data must include both. While the location of synaptic voxels is manually specified by user annotation, synaptic orientation $\omega_3(\ell_i)$ is automatically extracted as explained in §IV-C, using eigen analysis on the Hessian matrix computed at ℓ_i . Unfortunately, the obtained $\omega_3(\ell_i)$ vector is only defined up to a polarity, which is insufficient given that the pre-synaptic and post-synaptic regions are starkly different in appearance. We therefore follow the standard pose-indexing methodology [12] by labeling the polarity of the $\omega_3(\ell_i)$ vector during training. Note that the orientation labeling procedure in training is extremely efficient, requiring a single user click per synapse to consistently direct all $\omega_3(\ell_i)$ vectors to the pre-synaptic region. During testing, our learned contextual classifier $\varphi(\cdot)$ is evaluated for both polarities of the extracted $\omega_3(\ell_i)$ vector and the maximum response retained.

2) *Sampling Negatives*: Under the pose-indexing framework [12], [2], samples that do not exhibit the same pose (location and orientation) as positive samples should be considered as negative during training. However, in practice, samples that are too close in pose-space to the positives should be excluded from training, as their appearance can be similar to that of positives and their inclusion can therefore deteriorate performance.

For example, following [12], any voxel lying on the synaptic cleft with an incorrect orientation should be treated as a negative sample. Likewise, a voxel that is immediately next to a synaptic cleft should also be considered a negative sample. However, given the overlap in appearance, it is difficult for the learning method to disambiguate such voxels from the positive set. Thus, as is commonly done in object detection [2], we setup conservative training exclusion zones in pose-space around our positive examples and sample negative examples outside these exclusion zones. In particular, we do not use positive voxels with the wrong orientation as negatives, as discussed above. Moreover, we also exclude voxels that are outside the synaptic cleft and less than 10 voxels away from a positive-labeled voxel.

F. Supervoxels

Our entire algorithm including feature extraction, training and testing is designed and implemented to operate on individual voxels of the EM volumes. However, significant

TABLE II
DATASET DESCRIPTION.

Dataset	Voxel Size	Train		Test	
		Size (voxels)	Labeled synapses	Size (voxels)	Labeled synapses
(A) Somatosensory cortex	6.8 nm	$750 \times 564 \times 750$	9 (some)	$655 \times 429 \times 250$	28 (all)
(B) Hippocampus	5 nm	$1024 \times 653 \times 165$	20 (all)	$1024 \times 883 \times 165$ $1024 \times 1536 \times 200$	79 (all)
(C) Cerebellum	5 nm	$1398 \times 1343 \times 299$	7 (some)	$1966 \times 1343 \times 200$	56 (all)

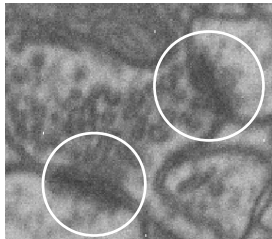


Fig. 5. Slice cut of a region from the hippocampus dataset. Synapses indicated with circles. Labeling voxels close to the synaptic boundary is an ill-posed problems.

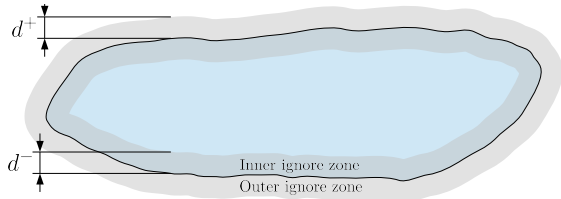


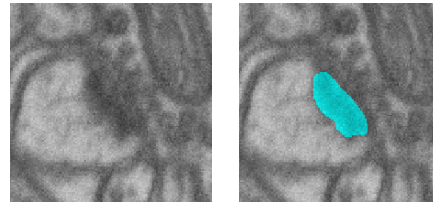
Fig. 6. Diagram showing inner (d^-) and outer (d^+) exclusion zones on synthetic synapse ground truth (blue). Voxels within the exclusion zone are ignored during evaluation.

computational savings can be achieved by grouping voxels into supervoxels [1] for specific operations for training and testing. Thus, during training, instead of using every voxel as a positive or negative data sample, we restrict our method to training only on voxels corresponding to centers of super-voxels. In effect, this amounts to a spatially-driven sampling of the training data which significantly speeds up training while maintaining performance. Likewise, during testing, instead of evaluating our learned contextual classifier $\varphi(\cdot)$ on every voxel in the EM test volume, we only evaluate $\varphi(\cdot)$ on voxels corresponding to super-voxel centers while off-center voxels are assigned a response equal to that of the center.

V. EXPERIMENTS

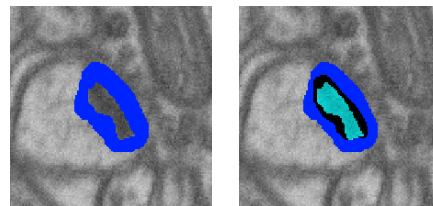
We evaluated our method on three different EM stacks acquired from different regions of the adult rat brain¹. We assessed performance both in terms of voxel-wise segmentation and synapse detection.

In this section, we first describe these datasets and our training and evaluation methodology. We then use our datasets to evaluate both the voxel-wise precision of our method and its accuracy in terms of how many entire synapses are correctly detected. We use the method of [20] as a baseline against



(a) Raw slice

(b) Ground Truth



(c) Exclusion zone

(d) Overlapped voxels in black

Fig. 7. Example of exclusion zone with $d^+ = 4$ and $d^- = 1.6$ voxels. Overlap between ground truth and exclusion zone shown in black in (d). Note the labeling ambiguity in (b) for the voxels close to the vesicles.

which we compare our results. Finally, we show that our method can be used to compute biologically relevant statistics and discuss computational complexity issues.

A. Datasets and Evaluation Methodology

We used three different datasets from (A) the Somatosensory Cortex, (B) Hippocampus, and (C) Cerebellum of an adult rat. Example slice cuts of each dataset are shown in Fig. 8.

The amounts of training and test data for each dataset are summarized in Table II. The volumes were annotated in a voxel-wise fashion using Fiji [33]. Note that testing volumes were fully annotated, each voxel being assigned a synapse or a non synapse label, in order to generate as large as possible a test set and report meaningful results. Training volumes on the other hand, in particular for the large datasets A and C, were only partially annotated in order to reduce labeling cost. In those cases, an approach similar to what was used in [20] was followed, labeling a fraction of the voxels inside the volume as positive or negative, leaving most of the voxels un-annotated and therefore not used for training. In the case of Dataset B, two test subvolumes were extracted from different regions of the Hippocampus.

1) *Data Annotation*: Synapse labeling is an ill-posed problem because boundaries are generally blurry, as shown in Fig. 5.

¹Source code available at <http://cvlab.epfl.ch/software/synapse>

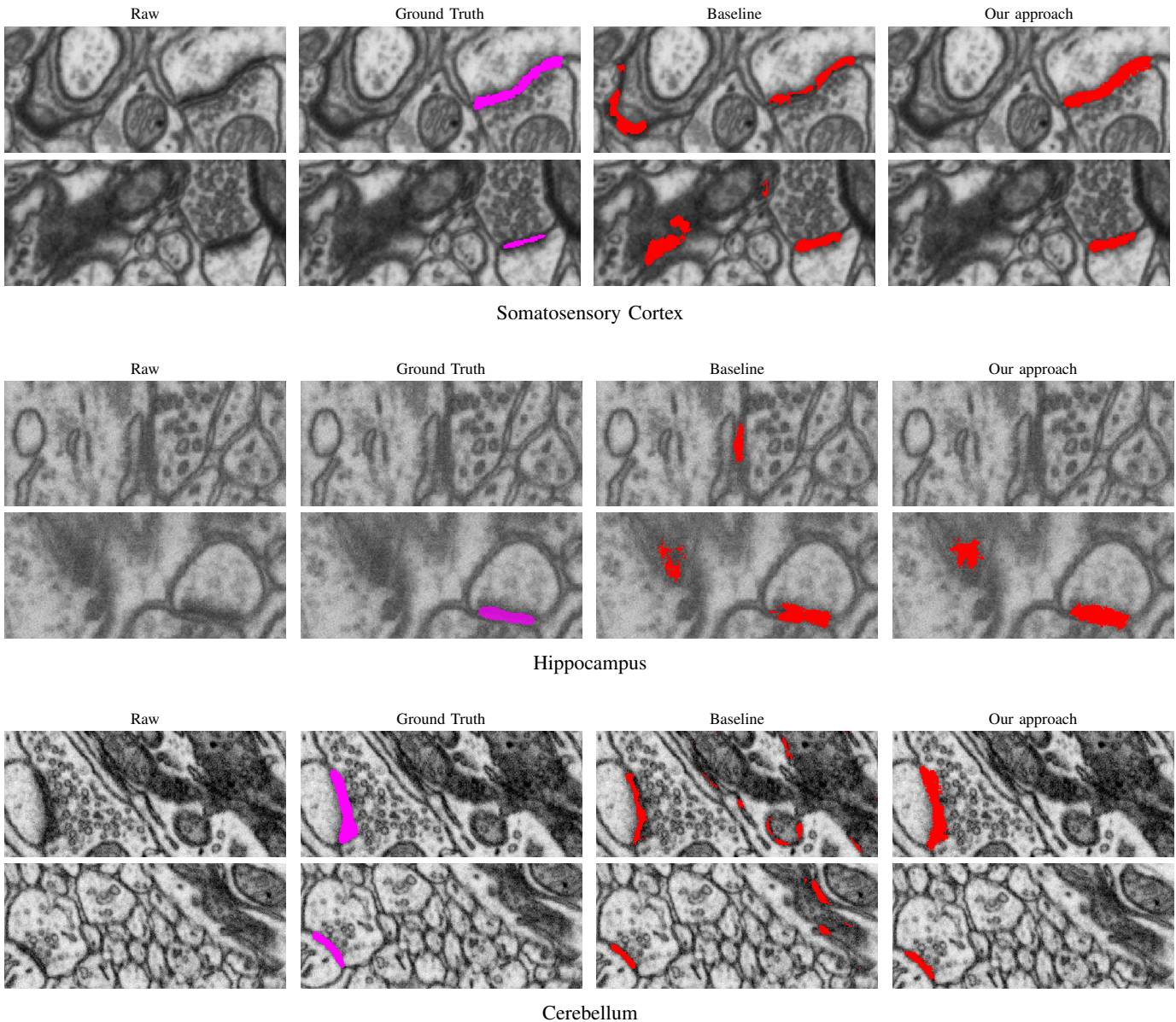


Fig. 8. Qualitative results (slice cuts) for the three different datasets after thresholding. Threshold set at best VOC. Note that our approach yields more accurate segmentation results as well as reducing the amount of false positives.

For training purposes, we adopted a conservative labeling policy whereby voxels are labeled positive only if experts are highly confident. This yields a ground truth volume whose positive samples are mainly located at the center of synapses.

For testing purposes, the behavior of a method on ambiguous voxels may be of particular interest to the practitioner. Thus, we adopted a different procedure for the annotation of test volumes. Experts were given more freedom in deciding whether or not a specific voxel lies on a synaptic cleft while our evaluation procedure, explained next, was designed to study the behavior of our detector at synaptic boundaries.

2) *Evaluation Methodology*: Voxel-wise evaluation is essential to assess and compare the performance of different segmentation methods. However, such an evaluation must take into account the aforementioned boundary issue to provide

meaningful performance measures.

Much as we defined a training exclusion zone, we define a testing exclusion zone about the labeled border of the synapse with an exterior radius of d^+ and an interior radius of d^- , as depicted in Fig. 6. Voxels within the exclusion zone, shown in blue in Fig. 7(c), are ignored during evaluation.

Rather than arbitrarily fixing the values of d^+ and d^- , we propose to assess performance as a function of their values. We plot precision-recall (PR) curves at different exclusion zone sizes, and the value of the Jaccard Index [9], [23], also known as the VOC Score, as a function of d^+ and d^- . The VOC score measures the segmentation quality when ground-truth data is available. It is computed as the ratio of the area of the intersection between what has been segmented and the ground truth, and of the area of the union of these two regions.

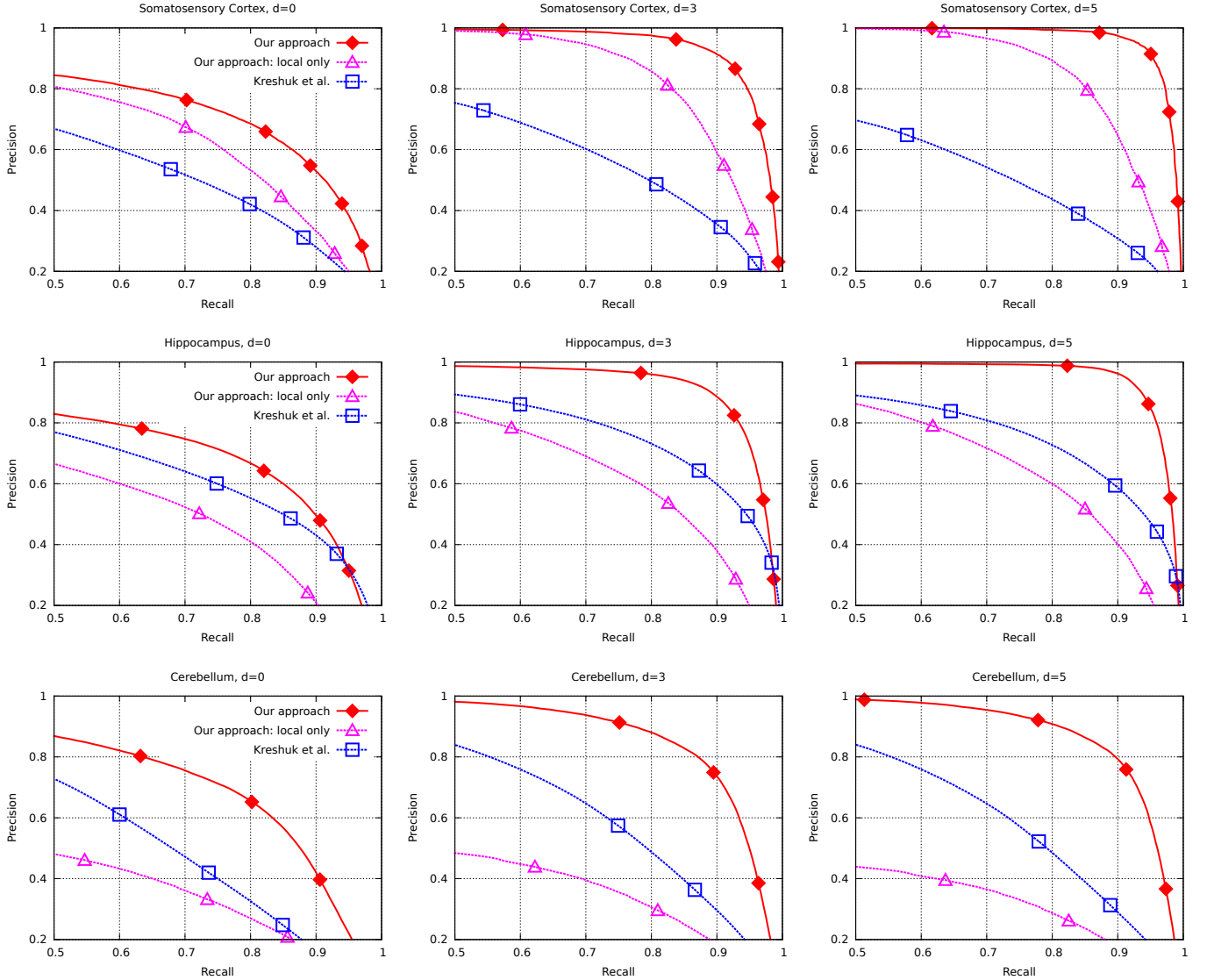


Fig. 9. Precision-recall curves for each dataset for different values of d . Our approach always yields better performance than the baseline of [20].

To facilitate interpretation, we fix the ratio $\eta = \frac{d^+}{d^-}$ and plot the performance measure as a function of $d = d^+$. In our experiments we considered $0 \leq d \leq 5$ voxels and fixed $\eta = 2.5$ so that the maximum d^- is 2 voxels. Limiting d^- is essential to confine the exclusion zone to boundary voxels only, thus preserving most of the labeled synaptic voxels such that the evaluation remains meaningful. In practice, the exclusion zone is found by pre-computing a chamfer distance volume w.r.t. the synaptic boundaries in the ground truth.

B. Voxel-wise Accuracy

We evaluate the voxel-wise segmentation performance of our approach and compare ourselves to [20] in terms of precision-recall (PR) curves and Jaccard index values, also known as VOC scores [23].

Precision-recall curves at different exclusion zone sizes d are shown in Fig. 9. Our approach clearly outperforms the baseline for all recall values as well as exclusion zone sizes

d . We also show the value of the highest Jaccard index obtained at different values of d in Fig. 10, which manifests the same behavior. Overall, our results indicate a significant improvement in segmentation performance for both border and center voxels.

Importance of contextual information: To demonstrate that improved performance comes from using context, we also evaluate the performance of a degenerate version of our approach that relies only on local information and ignores context. To this end, we set the parameter $\|c_p\|_{\max}$ of Sec. IV-B to 0. This means that post-synaptic and pre-synaptic context is not exploited and only local statistics are leveraged. We plot the corresponding curves in Figs. 9 and 10, labeled as *Our approach: local only*. It can be seen that our full approach utilizing context does systematically and significantly better than the degenerate version which ignores context, highlighting the importance of contextual information for synapse segmentation.

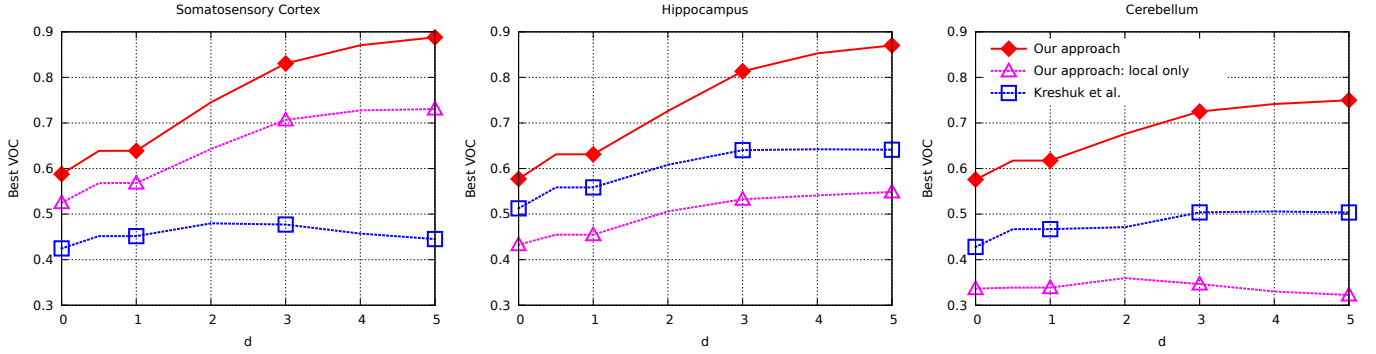


Fig. 10. Highest Jaccard index (VOC score) as a function of exclusion zone size d for the different datasets. Our approach outperforms [20] for all values of d .

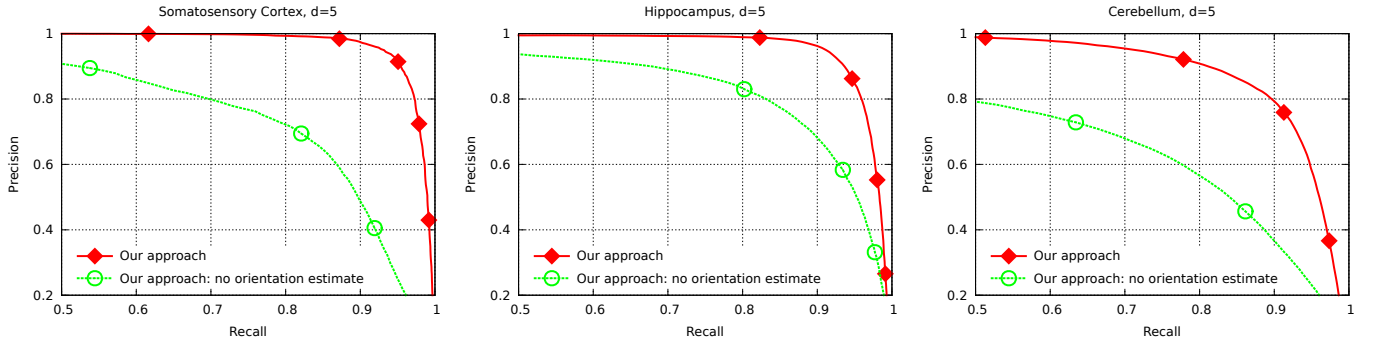


Fig. 11. Comparison of our approach (diamonds) with a degenerate version of our method (circles) that uses a fixed orientation estimate for all voxels, where $\mathbf{R}(\ell_i)$ is set to identity matrix for all voxels ℓ_i . The latter results in a considerable performance loss, motivating the use of contextual cues that rotate to be consistent with the orientation estimate of each voxel ℓ_i .

We note that the relative performance between [20] and the degenerate version of our algorithm is somewhat variable. Both these methods ignore context and rely solely on local image statistics. We believe the difference in performance between the two has to do with our degenerate version relying on decision stumps with a reduced feature set, while [20] relies on decision trees with a richer feature set.

Consistency of Contextual Cues: Context cue locations are rotated according to the orientation estimate at each voxel of interest ℓ_i , which allows us to exploit synaptic context in a meaningful way. Therefore, it is interesting to observe the effects of setting the orientation estimate of all voxels to a fixed value. To do so, we have fixed $\mathbf{R}(\ell_i)$ to the identity matrix for all locations ℓ_i , which makes $\omega_3(\ell_i)$ point in the z direction. This is equivalent to our approach without pose-indexing, similar to the method proposed in [16], [34]. We call this particular implementation *Our approach: fixed orientation estimate*, and a comparison with our full method is presented in Fig. 11. As expected, performance drops significantly since context is relative to the orientation of the synaptic cleft, and fixing $\mathbf{R}(\ell_i)$ to an identity transformation yields inconsistent contextual cues, highlighting once again the importance of contextual information for synapse segmentation.

C. Selected Features

Our approach lets AdaBoost pick the most discriminative features for synapse segmentation at every boosting step. To

observe how our method exploits context, we plot the pixel locations over which channel values $H_{m,\sigma_{n_m}}$ are summed in Fig. 12. To ease visualization, context cue locations \mathbf{c}_p are projected on the z_o axis (see Fig 2(a)).

When, according to AdaBoost, a higher value of $f_k(x, \ell_i)$ contributes to the voxel in the center being a synapse, its corresponding α_t (Eq. (6)) is added to the maps in Fig. 12. On the other hand, when the label is negatively correlated with $f_k(x, \ell_i)$, its α_t value is subtracted.

It can be seen that the raw image (Fig. 12(a)) provides important clues, particularly at the voxel of interest (center) but in its surroundings as well, especially at the post-synaptic region, where the classifier expects an average high image value to vote for a synapse. Another interesting channel is the lowest magnitude structure tensor, shown in Fig. 12(c), which signals the presence of vesicles in the pre-synaptic region, which is a strong clue used by experts to evidence the presence of a synapse.

D. Detection Accuracy

So far, we have evaluated the different approaches for voxel-wise segmentation. However, it is also interesting to evaluate synapse detection performance, that is if a synapse as a whole is detected by the algorithm or not. We measure detection performance by clustering thresholded score volumes. This can be summarized as:

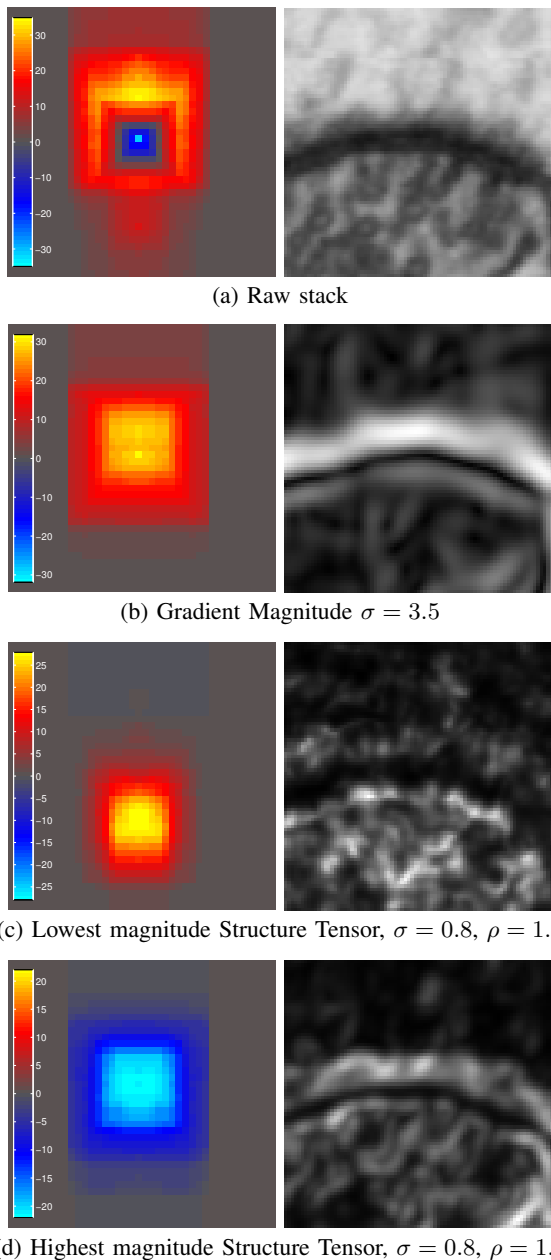


Fig. 12. Some of the features selected by AdaBoost on four different channels for the Somatosensory Cortex dataset. The left column shows the voting map (see text for detailed description) and the right column shows an example synapse and its context.

- 1) Threshold the score volume at the value that yields the best VOC score for all d .
- 2) Run connected component analysis on the resulting binary volume.
- 3) Remove detected clusters with less than 1000 voxels, as in [20].
- 4) Count the number of missed and false-positive clusters/detections.

False-positive detections are clusters of positive-predicted voxels which do not intersect with the ground truth, while missed (false-negative) detections are ground truth clusters that do not intersect with the predicted score volume after step 3 above.

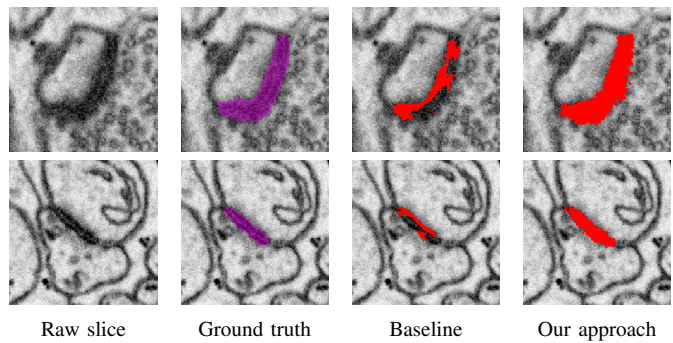


Fig. 13. Examples of detected synaptic voxels thresholding for the best Jaccard index. Synapses are detected by both methods, but the baseline method yields poor results from a segmentation perspective.

As shown in Fig. 14, our approach yields a significant reduction in the number of false positives when compared to the baseline, obtaining perfect detection on the first two datasets. Furthermore, because our method yields much better voxel-wise accuracy, the shape of the recovered synapses is much closer to the ground-truth, as depicted in Fig. 13. This is important when computing biologically relevant statistics such as synapse shape and size, as discussed below.

E. Biological Statistics

We applied the classifier trained on the Somatosensory Cortex data to a large volume consisting of $1500 \times 1125 \times 750$ voxels or $9.8 \times 7.4 \times 4.9 \mu\text{m}$. This is the original volume from which the train and test sub-volumes were extracted.

The resulting score volume was smoothed with a Gaussian filter with unit variance and thresholded at the value that corresponds to the maximum VOC score in the test volume. Afterwards, clusters of positive detections of less than 1000 voxels were discarded, as in [20] and §V-D. This resulted in a total of 405 clusters of voxels that our approach labeled as synapses.

Finally, an expert went through the resulting segmentation volume, discarding 31 false positive synapses, obtaining a total of 374 verified synapses. This number is in agreement with the expected synapse density in the Somatosensory Cortex region (layer II) [8]

A 3D visualization of the detected synapses is shown in Fig. 16. It is interesting to observe the large variation in synapse shape and size, which is evidenced in the histograms of synapse size and synapse flatness Fig. 16(a) and Fig. 16(b).

Another interesting observation comes from Fig. 16(c), which is a scatter plot of synapse volume and flatness. There is a strong correlation between synapse volume and flatness. This occurs because synapses are membrane-like structures and the synaptic cleft width is constant across different synapses, independently of their size. Therefore, larger synapses are flatter than smaller ones, which is evidenced in Fig. 16(c).

In addition to generating segmentation results, our approach can also be used to determine the location of the post-synaptic and pre-synaptic regions. This can be highly relevant in practice to determine the location of the axons and dendrites relative to a given synapse, helping reveal neural circuit connectivity.

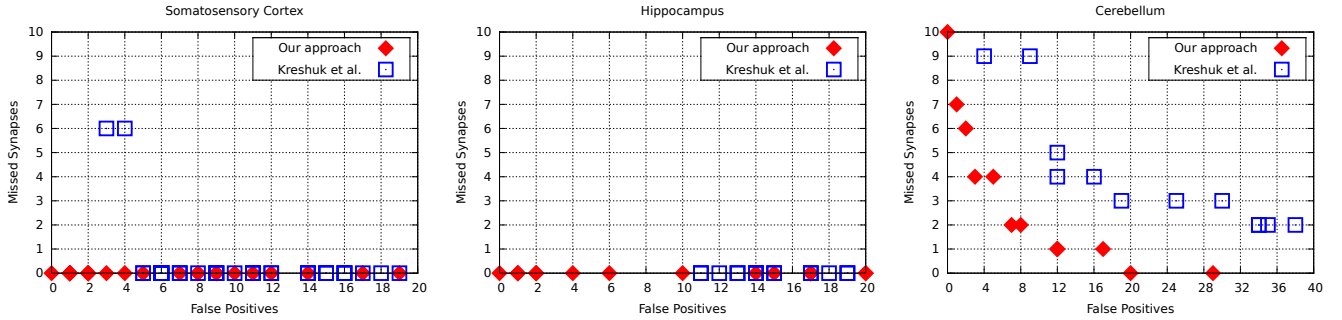


Fig. 14. Detection performance on the three datasets, after thresholding and clustering voxels labeled as synapses by our approach and [20], at different thresholds. Our method yields less false-positive detections than [20], obtaining perfect detection performance in the first two datasets.

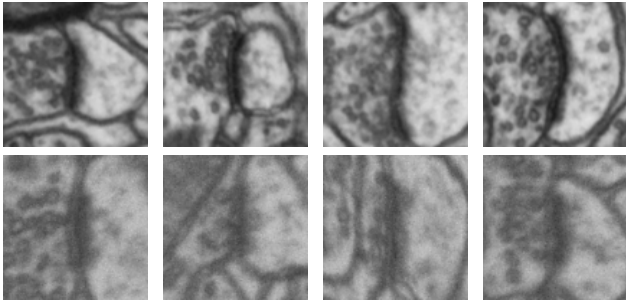


Fig. 15. Examples of polarity prediction. Images have been aligned such that the horizontal axis travels from the pre- to post-synaptic region from left to right. Our approach achieved 100% accuracy for this task.

As discussed Sec. IV-E, our technique evaluates two different scores for location ℓ_i at test time, one for each possible orientation polarity. The polarity with the highest score can be employed as a polarity estimate. Once the score image is thresholded, majority voting can be used to determine the most likely polarity.

We applied this technique to the predicted scores for the three datasets and observed 100% accuracy at predicting synapse polarity. Examples are shown in Fig. 15, where the synapse normal has been aligned to the horizontal axis. This is also an indication that our approach is exploiting context information.

F. Computational Complexity

In terms of computational cost, our approach is faster in both training and testing than [20]. Training time is reduced with the techniques mentioned in §IV-D. Using stumps as weak learners is also an important advantage at test time, in contrast to Random Forest’s trees which are built with as many splits as necessary to separate training data perfectly.

Another speed up is obtained by using supervoxels to oversegment the EM volumes. The chosen seed size of 2 voxels translates in an average supervoxel size of $2 \times 2 \times 2 = 2^3$ voxels [1], which yields a 8x speed factor since only supervoxel centers are classified.

Table III summarizes timings obtained for both methods. The number of trees in the Random Forest classifier was set to 500. To make comparison fair, we have modified the Random Forest implementation used by Kreshuk et al. [20]

to use multiple threads, speeding up training and testing substantially². Note that, in the case of using the default Vigna implementation, the timings for the baseline method in Table III would be an order of magnitude higher.

The parameters used for our approach are the default ones described in Table I. Note that the test timings for our approach already consider evaluating the two possible polarities for each location ℓ_i .

If further speed up was needed, soft cascades [6] could be employed to stop early during the evaluation of the boosted classifier. This is likely to provide a considerable speed up since most background voxels can be discarded with a simple intensity check, given that synapses appear as dark structures.

VI. CONCLUSION

We presented a novel approach to synapse segmentation. It relies on a large set of image features, specifically designed to take spatial context into account, which are selected, weighed and combined using AdaBoost. We used three different EM datasets to demonstrate that our algorithm effectively distinguishes true synapses from other organelles that exhibit the same local texture.

Even though our approach was designed for synapse segmentation, we believe that it can be applied to segment other types of tissue where context plays an important role, such as mitochondria membranes. Another potential extension is to handle anisotropic volumes by scaling our features differently in the isotropic and anisotropic axes.

REFERENCES

- [1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, Nov 2012.
- [2] K. Ali, F. Fleuret, D. Hasler, and P. Fua. A real-time deformable detector. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(2):225–239, Feb 2012.
- [3] B. Andres, U. Kthe, M. Helmstaedter, W. Denk, and F. A. Hamprecht. *Segmentation of SBFSEM Volume Data of Neural Tissue by Hierarchical Classification*, volume 5096 of *Lecture Notes in Computer Science*, chapter 15, pages 142–152. Springer Berlin Heidelberg, 2008.
- [4] A. Bartsch, G. Sapiro, and S. Subramaniam. An energy-based three-dimensional segmentation approach for the quantitative interpretation of electron tomograms. *Trans. Img. Proc.*, 14(9):1314–1323, Sept. 2005.

²The original version is found in the Vigna library (<http://hci.iwr.uni-heidelberg.de/vigna/>) and our version can be found online at <http://cvlab.epfl.ch/%7Eecjbecker>

TABLE III
 TRAIN AND TEST TIMES FOR OUR APPROACH AND THE METHOD OF KRESHUK ET AL. [20]. TIMINGS OBTAINED WITH MULTITHREADED IMPLEMENTATIONS, BOTH OF THEM USING 14 THREADS ON THE SAME 64-BIT ARCHITECTURE.

Dataset		Our approach		Kreshuk et al. [20]	
		Total ($T = 2000$)	Single stump (average)	Total (500 trees)	Single tree (average)
Somatosensory Cortex	Train	0:57 hs	1.69 sec	2:16 hs	16.4 sec
	Test	0:13 hs	6.2 msec	0:21 hs	2.5 sec
Hippocampus	Train	2:40 hs	4.8 sec	21:22 hs	154 sec
	Test	1:20 hs	2.4 sec	1:46 hs	12.8 sec
Cerebellum	Train	1:41 hs	3.05 sec	10:31 hs	75.7 sec
	Test	1:35 hs	2.8 sec	2:47 hs	20.1 sec

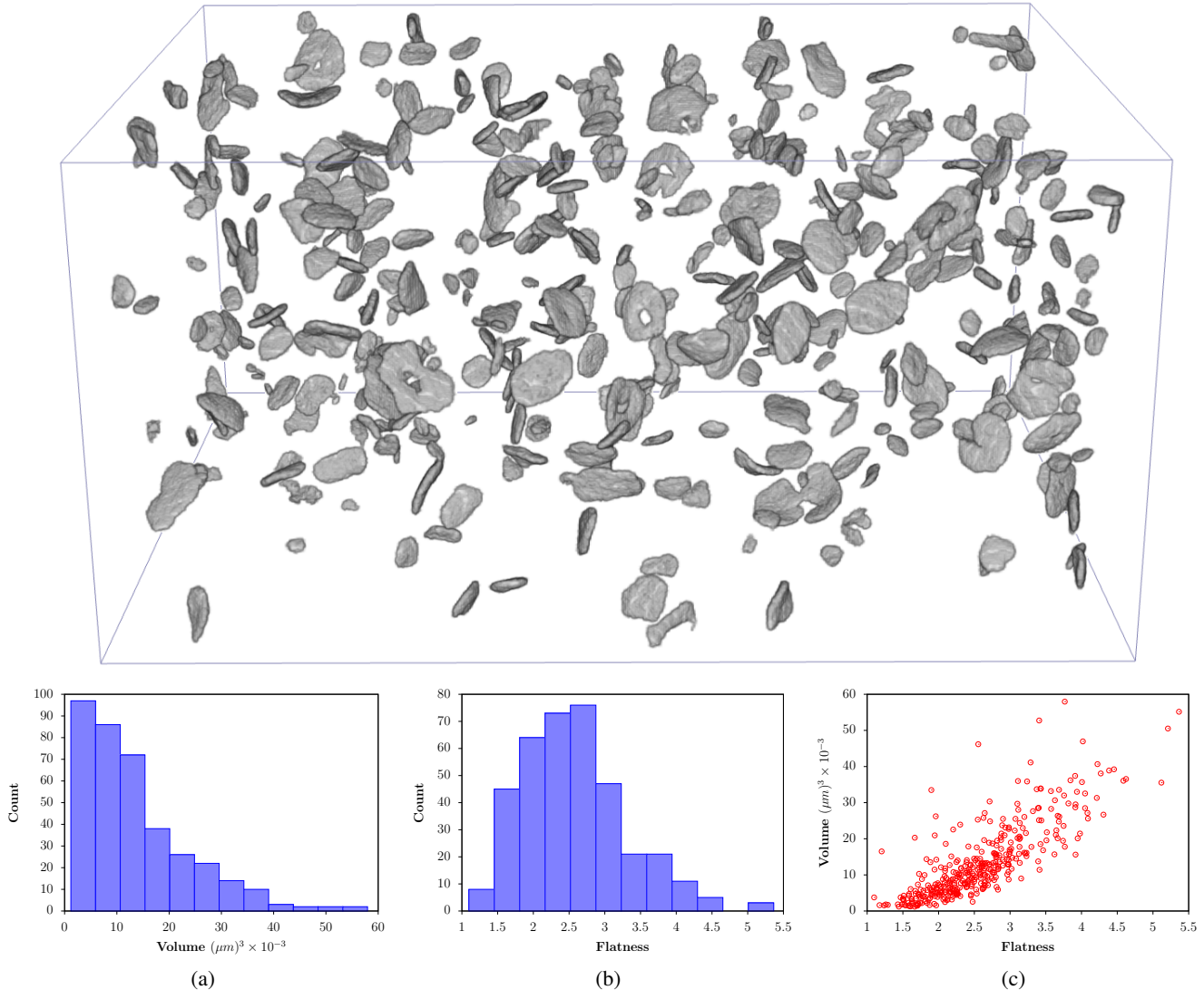


Fig. 16. Results of our approach on a large Somatosensory Cortex volume of 1500x1125x750 voxels: (top) 3D Visualization of the detected synapses and (bottom) some of the statistics that can be extracted from voxel-wise segmentation. The correlation between synapse volume and flatness in (c) evidences the fact that synapses are membrane-like objects.

- [5] C. Becker, K. Ali, G. Knott, and P. Fua. Learning Context Cues for Synapse Segmentation in EM Volumes. In *MICCAI*, September 2012.
- [6] L. Bourdev and J. Brandt. Robust Object Detection via Soft Cascade. In *Conference on Computer Vision and Pattern Recognition*, pages 236–243, 2005.
- [7] I. Carlom, D. Terzopoulos, and K. Harris. Computer-assisted registration, segmentation, and 3d reconstruction from images of neuronal tissue sections. *IEEE Transactions on Medical Imaging*, 13(2):351–362, Jun 1994.
- [8] J. DeFelipe, P. Marco, I. Busturia, and A. Merchán-Pérez. Estimation of the number of synapses in the cerebral cortex: Methodological considerations. *Cerebral Cortex*, 9(7):722–732, Oct 1999.
- [9] M. Everingham, C. W. L. Van Gool and, J. Winn, and A. Zisserman. The Pascal Visual Object Classes Challenge 2010 (VOC2010) Results.
- [10] J. C. Fiala. Reconstruct: a free editor for serial section microscopy. *Journal of microscopy*, 218:52–61, 2005.
- [11] J. C. Fiala and K. M. Harris. Extending Unbiased Stereology of Brain

- Ultrastructure to Three-dimensional Volumes. *Journal of the American Medical Informatics Association : JAMIA*, 8(1):1–16, 2001.
- [12] F. Fleuret and D. Geman. Stationary Features and Cat Detection. *Journal of Machine Learning Research*, 9:2549–2578, 2008.
- [13] Y. Freund and R. Schapire. Experiments with a New Boosting Algorithm. In *International Conference on Machine Learning*, pages 148–156, 1996.
- [14] J. Herold, W. Schubert, and T. W. Nattkemper. Automated detection and quantification of fluorescently labeled synapses in murine brain tissue sections for high throughput applications. *Journal of Biotechnology*, 149(4):299–309, Sep 2010.
- [15] Q. J. Hieu Nguyen. Shape-driven three-dimensional watersnake segmentation of biological membranes in electron tomography. *IEEE Transactions on Medical Imaging*, 27(5):616–628, May 2008.
- [16] V. Jain, J. Murray, F. Roth, S. Turaga, V. Zhigulin, K. Briggman, M. Helmstaedter, W. Denk, and H. Seung. Supervised Learning of Image Restoration with Convolutional Networks. In *International Conference on Computer Vision*, pages 1–8, 2007.
- [17] E. Jurrus, A. R. Paiva, S. Watanabe, J. R. Anderson, B. W. Jones, R. T. Whitaker, E. M. Jorgensen, R. E. Marc, and T. Tasdizen. Detection of neuron membranes in electron microscopy images using a serial neural network architecture. *Medical Image Analysis*, 14(6):770–783, Dec 2010.
- [18] V. Kaynig, T. Fuchs, and J. Buhmann. Neuron Geometry Extraction by Perceptual Grouping in Sstem Images. In *Conference on Computer Vision and Pattern Recognition*, pages 2902–09, 2010.
- [19] A. B. Knott, G. Perkins, R. Schwarzenbacher, and E. Bossy-Wetzel. Mitochondrial fragmentation in neurodegeneration. *Nature Reviews Neuroscience*, 9(7):505–518, Jul 2008.
- [20] A. Kreshuk, C. N. Strachle, C. Sommer, U. Koethe, G. Knott, and F. Hamprecht. Automated Segmentation of Synapses in 3D Em Data. In *ISBI*, pages 220–223, 2011.
- [21] P.-C. Lee, C.-C. Chuang, A.-S. Chiang, and Y.-T. Ching. High-throughput computer method for 3d neuronal structure reconstruction from the image stack of the drosophila brain and its applications. *PLoS Computational Biology*, 8(9):e1002658, Sep 2012.
- [22] J. Liang, T. McInerney, and D. Terzopoulos. United snakes. *Medical Image Analysis*, 10(2):215–233, Apr 2006.
- [23] A. Lucchi, K. Smith, R. Achanta, G. Knott, and P. Fua. Supervoxel-Based Segmentation of Mitochondria in EM Image Stacks with Learned Shape Features. *TMI*, 31(2):474–486, 2011.
- [24] J. H. Macke, N. Maack, R. Gupta, W. Denk, B. Schlkopf, and A. Borst. Contour-propagation algorithms for semi-automated reconstruction of neural processes. *Journal of Neuroscience Methods*, 167(2):349–357, Jan 2008.
- [25] C. A. Mannella, M. Marko, and K. Buttle. Reconsidering mitochondrial structure: New views of an old organelle. *Trends Biochem. Sci.*, 22:37–38, 1997.
- [26] B. Marsh, D. Mastronarde, K. Buttle, K. Howell, and J. McIntosh. Organellar Relationships in the Golgi Region of the Pancreatic Beta Cell Line, Hit-T15, Visualized by High Resolution Electron Tomography. *PNAS*, 98:2399–2406, 2001.
- [27] A. Merchán-Pérez, J.-R. Rodríguez, L. Alonso-Nanclares, A. Schertel, and J. Defelipe. Counting synapses using fib/sem microscopy: A true revolution for ultrastructural volume reconstruction. *Front Neuroanat*, 3:18, 2009.
- [28] Y. Mishchenko, T. Hu, J. Spacek, J. Mendenhall, K. M. Harris, and D. B. Chklovskii. Ultrastructural analysis of hippocampal neuropil from the connectomics perspective. *Neuron*, 67(6):1009–1020, Sep 2010.
- [29] N. Vu and B. Manjunath. Graph Cut Segmentation of Neuronal Structures from Transmission Electron Micrographs. In *International Conference on Image Processing*, pages 725–728, 2008.
- [30] R. Narasimha, H. Ouyang, A. Gray, S. W. McLaughlin, and S. Subramaniam. Automatic joint classification and segmentation of whole cell 3d images. *Pattern Recognition*, 42(6):1067–1079, Jun 2009.
- [31] D. Padfield, J. Rittscher, N. Thomas, and B. Roysam. Spatio-temporal cell cycle phase analysis using level sets and fast marching methods. *Medical Image Analysis*, 13(1):143–155, Feb 2009.
- [32] A. C. Poole, R. E. Thomas, L. A. Andrews, H. M. McBride, A. J. Whitworth, and L. J. Pallanck. The pink1/parkin pathway regulates mitochondrial morphology. *Proceedings of the National Academy of Sciences*, 105(5):1638–1643, Feb 2008.
- [33] B. Schmid, J. Schindelin, A. Cardona, M. Longair, and M. Heisenberg. A high-level 3d visualization api for java and imagej. *BMC Bioinformatics*, 11(1):274, 2010.
- [34] S. C. Turaga, J. F. Murray, V. Jain, F. Roth, M. Helmstaedter, K. Briggman, W. Denk, and H. S. Seung. Convolutional networks can learn to generate affinity graphs for image segmentation. *Neural Computation*, 22(2):511–538, Feb 2010.
- [35] K. Venkataraju, A. Paiva, E. Jurrus, and T. Tasdizen. Automatic Markup of Neural Cell Membranes Using Boosted Decision Stumps. In *IEEE Symposium on Biomedical Imaging: From Nano to Macro*, pages 1039–42, 2009.
- [36] S. Vitaladevuni, Y. Mishchenko, A. Genkin, D. Chklovskii, and K. Harris. Mitochondria Detection in Electron Microscopy Images. In *Workshop on Microscopic Image Analysis with Applications in Biology*, 2008.
- [37] J. G. White, E. Southgate, J. N. Thomson, and S. Brenner. The structure of the nervous system of the nematode *caenorhabditis elegans*. *Philosophical Transactions of the Royal Society B Biological Sciences*, 314(1165):1–340, Nov 1986.