Semantic Volume Segmentation with Iterative Context Integration

Sven Sickert, Erik Rodner and Joachim Denzler
Computer Vision Group
Friedrich Schiller University Jena
{sven.sickert, erik.rodner, joachim.denzler}@uni-jena.de

***Abstract*—Automatic recognition of biological structures like membranes or synapses is important to analyze organic processes and to understand their functional behavior. To achieve this, volumetric images taken by electron microscopy or computed tomography have to be segmented into meaningful semantic regions. We are extending iterative context forests which were developed for 2D image data for image stack segmentation. In particular, our method is able to learn high order dependencies and import contextual information, which often can not be learned by conventional Markov random field approaches usually used for this task. Our method is tested for very different and challenging medical and biological segmentation tasks.**

I. Introduction

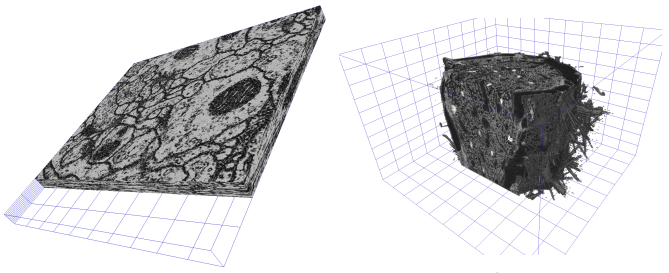
Extracting specific regions in volume images is an important task in medical image processing and a prerequisite for quantitative analysis of biological data (Fig. 1). For example, X-ray computed tomography (CT) is a common tool in medical imaging and the electron microscopy (EM) is used to investigate biological processes or structures in organic and inorganic specimens. However, manual annotation of 3D data is challenging and often requires a huge amount of time and expertise. Therefore, the aim of the paper is to present a method, which is easy to implement, to use, and can be applied to generic volume data by learning from a small number of previously annotated volumes. Being able to automatically segment semantic regions in volume data does not only save time, but it also allows for quantitatively analyzing a large number of volumes, important for providing applied researchers with robust statistics.

Our proposed method labels every single voxel in the volume and is able to capture context information along every axis of the volume. We show that the use of relatively simple feature extraction methods on channels for color and gradient values is sufficient to obtain a decent segmentation of volume images.

The outline of the paper is as follows: In Section II, we give an overview of related work. Section III and IV continue with a description and discussion of the proposed method. In Section V, we present segmentation results for different data sets. We conclude with a discussion of the results in Section VI.

II. Related Work

The task of semantic volume segmentation for image data provided by CT imagery or EM microscopy is of great interest in computer vision [1], [2], [3]. Related works most often propose segmentation techniques to find and reconstruct objects of a specific class, *e.g.*, synapses or membranes in neural tissue. A popular tool for segmentation is the graph-cut algorithm. The method in [3] uses a gradient flux term in the energy function and incorporates information from different sections by applying the SIFT flow algorithm [4]. In [1] a modified energy function is presented which omits the gradient flux term. Instead, perceptual grouping constraints for contour completion are introduced. It can be useful for the segmentation of thin elongated structures. However, the authors argue that this term may also lead to false positive membrane segmentations due to textures. Instead the final energy function consists of a data term, a directional energy term for smoothness, a penalty for discontinuities, and a term that incorporates information from adjacent sections.



(a) (b)

Fig. 1: Volume images used in our experiments: (a) an electron microscopy stack of neural tissue and (b) a CT scan of a sponge

In [2], the correspondence between nearby areas is transformed into a fusion problem. First, an RDF classifier is trained to obtain the probability of each pixel belonging to the cell boundary. Watershed transformation is applied to obtain segmentations for each section. After that, 3D links between these sections should connect segments of different slices. Finally, a fusion problem of 2D segments and 3D links is defined that identifies each neuron.

CT images typically show objects of a larger scale. As the authors of [5] show, semantic segmentation can also help with localizing organs in a human body. The authors are using so-called entangled-decision forests to model context between nearby organs in CT scans which usually have a fixed composition.

In our approach we do not model the connection between different sections explicitly, but use *auto-context* features [6]

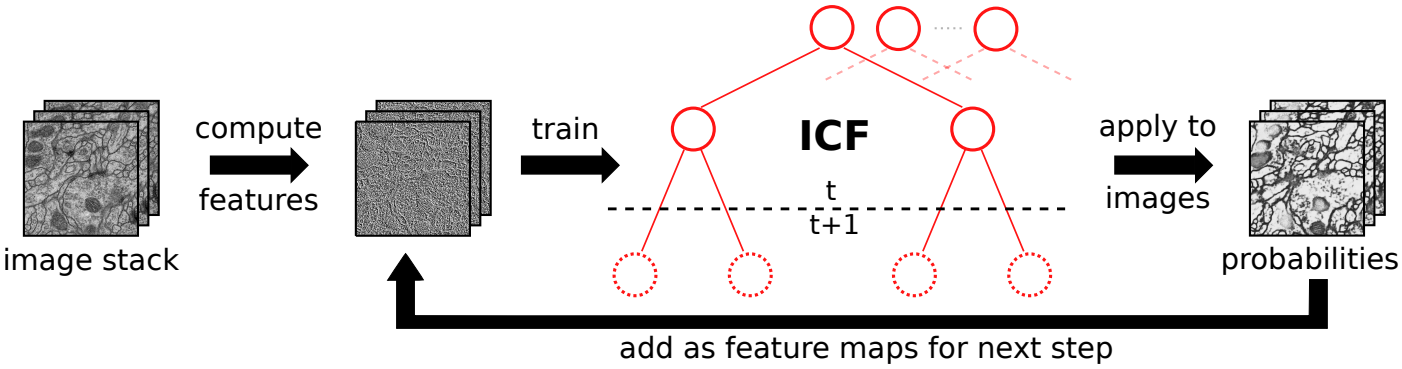


Fig. 2: Framework of our method: First, slices of a image stack are processed in order to compute feature maps. Feature extraction methods applied to these maps are used to train the ICF by finding *good* splits at the current level t. The ICF learned so far is applied to the training slices in order to create class probabilities for each pixel. Then, these probability maps are add to the pool of feature maps in the training step $t + 1$.

within an iterative pixelwise labeling approach. Moreover, we are using easy to compute features like gray-values and gradients in a cubic neighborhood. Therefore, we extend the iterative context forests (ICF) of [7] to 3D data which was previously limited to 2D images and multi-channel images occurring in remote sensing applications.

More common applications for semantic segmentation are urban scenes and images of landscape, persons, or animals. In these settings typical constellations like *car on street*, *sky above building* or *sheep on grass* are observable as well. Many methods in this field of application are using either random decision forests (RDF) [8], [9], [10], [7] or conditional random fields (CRFs) [11], [12], [13] which is a common technique to model context. As we show in this paper, in contrast to CRFs, our approach is not restricted to pairwise potentials and is able to learn high order dependencies.

III. Iterative Context Forests for Image Stacks

The proposed approach for semantic segmentation consists of two fundamental parts: feature extraction and pixel classification. In this section, we focus on the classification framework which is based on the popular random decision forests. Details on the efficient computation of features will follow in the next section.

A. Random Decision Forests

The core of ICF [7] is a random decision forest (RDF) [14], which has been specifically adapted to semantic segmentation. The concept of decision trees is well known, so we will not elaborate on how they work and refer the interested reader to [14]. In order to explain iterative context forests, we give a short introduction to random decision forests. The RDF approach in general aims at overcoming drawbacks of original decision trees, like over-fitting and long training times, with two concepts of massive randomization during training.

The first concept is known as bagging and trains several decision trees individually with a random subset of the training data. Furthermore, a second randomization concept is applied determining binary splits in inner nodes. Instead of computing all available features in each inner node only a random subset is drawn from the set of all possible features, which we will refer to as *feature pool*. Among them the best feature and split is determined by maximizing the impurity criterion, which in our case is the information gain.

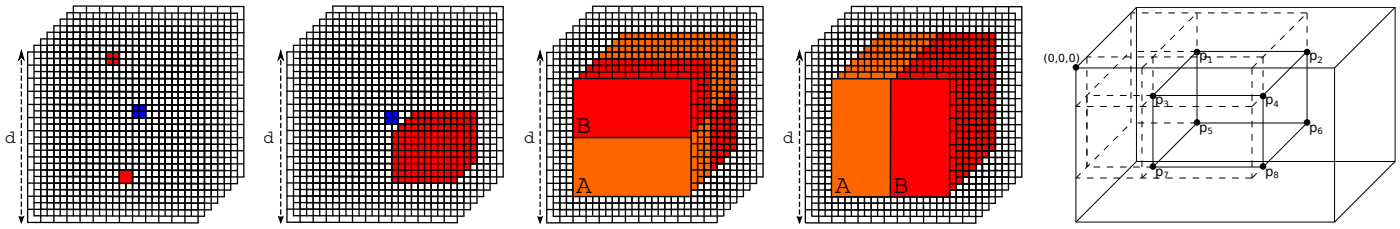
In our case, we are confronted with different types of features (see Sec. IV for more details) with each of them having a number of parameters (*e.g.*, position, used feature map, *etc.*). To sample a specific feature, we first sample the feature type and we then sample the parameters in a second step. This guarantees that the learning is not biased towards feature types with a larger parameter space. The whole random selection process is exactly what renders learning in our case with millions of features tractable.

Although finding a good split in a single tree node makes it necessary to test various randomly chosen splits, the training of random decision forests is still fast. Learning an RDF is a matter of minutes, while convolutional neural networks for instance need several days on a GPU for such a task [15]. The classification of test images is even faster. Each pixel of an example is traversing the trees of the trained RDF until it reaches the leaf nodes. The empirical distributions in all leaves are then combined in order to estimate class-wise probabilities.

B. Incorporating Context Knowledge

Applying RDFs for pixel- or voxelwise classification directly has two disadvantages: first, for each pixel the tree has to be traversed down to the leafs, and second, feature extraction is limited to a local neighborhood and is unable to integrate high-order dependencies.

To address the second issue, [7] proposed to sequentially traverse the tree level by level for all pixels in each image. This allows for using outputs of the previous level as an additional source for features. At each level of the random forest class probability maps for the current image (or volume) are computed. On a lower level these information allow the extraction of contextual features in order to model relational dependencies like one class is above another. This concept was introduced in [6] and is called *auto-context*. See Fig. 2 for an overview of our our pixel-wise classification framework.



(a) (b) (c) (d) (e)

Fig. 3: Feature extraction methods for a 3D neighborhood of side length d around a voxel (blue): (a) pairs of voxels, (b) a smaller cuboid of arbitrary size or (c-d) different types of 3D Haar-like features where the feature value is the difference of the voxel intensity sums of red and orange regions. (e) Eight references are necessary to compute the sum of an arbitrary cuboid within the integral volume image.

Furthermore, during testing time the structure of ICFs also addresses a trade-off between accuracy and time needed for inference. In some applications it might be sufficient to have a rough approximation of the best obtainable result after a short time. Since ICFs are built in breadth-first manner, a prediction on each level of the trees can be done by returning the empirical class distribution stored at inner nodes.

As observed in our experiments, each depth-level gives a more accurate classification result than the one before until saturation is reached. Stopping at an earlier level will give only rough results depending on fewer features but will also save time. The ability of an algorithm to allow for iterative refinements during testing is usually referred to as *anytime classification* [16].

IV. Efficient Features in Volumetric Images with ICF3D

In this section, we describe how computation of specific features is achieved. In the first part we broach the issue of creating a large feature pool with millions of possible features. After that, a fast method for computing such features is presented.

A. Creating a large feature pool

The ICF classifier allows inputs to have an arbitrary number of channels. Depending on the application, the algorithm can make use not only of raw color channels but additional layers like gradient images, resulting probabilities of other pre-learned classifiers, or unsupervised segmentation outputs. The relevance of these *feature maps* is automatically determined by the RDF classifier, since all binary splits are evaluated by an impurity measure and automatically selected.

The extraction of features from these maps is done in our case with simple operations performed in a neighborhood of the current center pixel: (a) Single values extracted from neighboring voxels, (b) sum or difference of two neighboring voxels, (c) sum of values within a cuboid and (d) 3D Haar-like features given by the difference of two or more cuboids. Visual examples can be found in Fig. 3.

With the size of the neighborhood the feature pool grows exponentially and it is not possible to compute all of the possible features in the training step. As a consequence, we only draw a fixed number of features. From this subset of features, the best split in each level of a decision tree is chosen by the impurity criterion.

B. Fast computation of 3D features

In order to compute 3D features in a fast manner, it is possible to use an intermediate representation. In the case of 2D images the so called *integral image* (or *summed area table*) is used [17], [18]. It contains at location (x, y) the sum of the pixel values above and to the left of the same location in the original image . The computation of rectangle features as for instance Haar-like features benefit from this representation. The sum of a rectangle area of arbitrary size can be computed by only four points in the integral image.

This idea can be easily extended to *integral volumes*. In consequence, an integral volume image $\mathcal{V}$ contains at location (x, y, z) the sum of gray values from the voxels above, to the left and in front of its location in the input image $\mathcal{I}$ as well as the gray value of the voxel (x, y, z) itself:

$$\mathcal{V}(x, y, z) = \sum_{x' \leq x,\ y' \leq y,\ z' \leq z} \mathcal{I}(x', y', z') \quad . \tag{1}$$

With the integral volume of an image, any sum of values in a cuboid can be be computed by eight array references. Analogously to [18], the values of the cuboids to the left, on top and in front of the currently processed volume have to be removed. Otherwise these region would be incorporated twice in the final computation. Consider a cuboid $\mathcal{C}$ with an arbitrary position inside a larger cuboid, *i.e.,* the whole integral volume image V. The eight corners of $\mathcal{C}$ are $p_1, ..., p_8$ with $p_i = (x_i, y_i, z_i)$ being the locations in V. Location p_6 marks the lower right back corner and p_2 the upper right front corner of $\mathcal{C}$. Then, the integral volume $\mathcal{V}_\mathcal{C}$ of $\mathcal{C}$ can be computed as:

$$\mathcal{V}_\mathcal{C} = p_1 + p_4 + p_6 + p_7 - p_2 - p_3 - p_5 - p_8 \quad . \tag{2}$$

A corresponding visualization showing the eight reference points can be found in Fig. 3(e). With this method cuboid features and especially the 3D Haar-like features can be computed very efficiently. Now that we have extended the ICF framework with 3D features and added the possibility to analyze volume images, we will henceforth call it ICF3D.

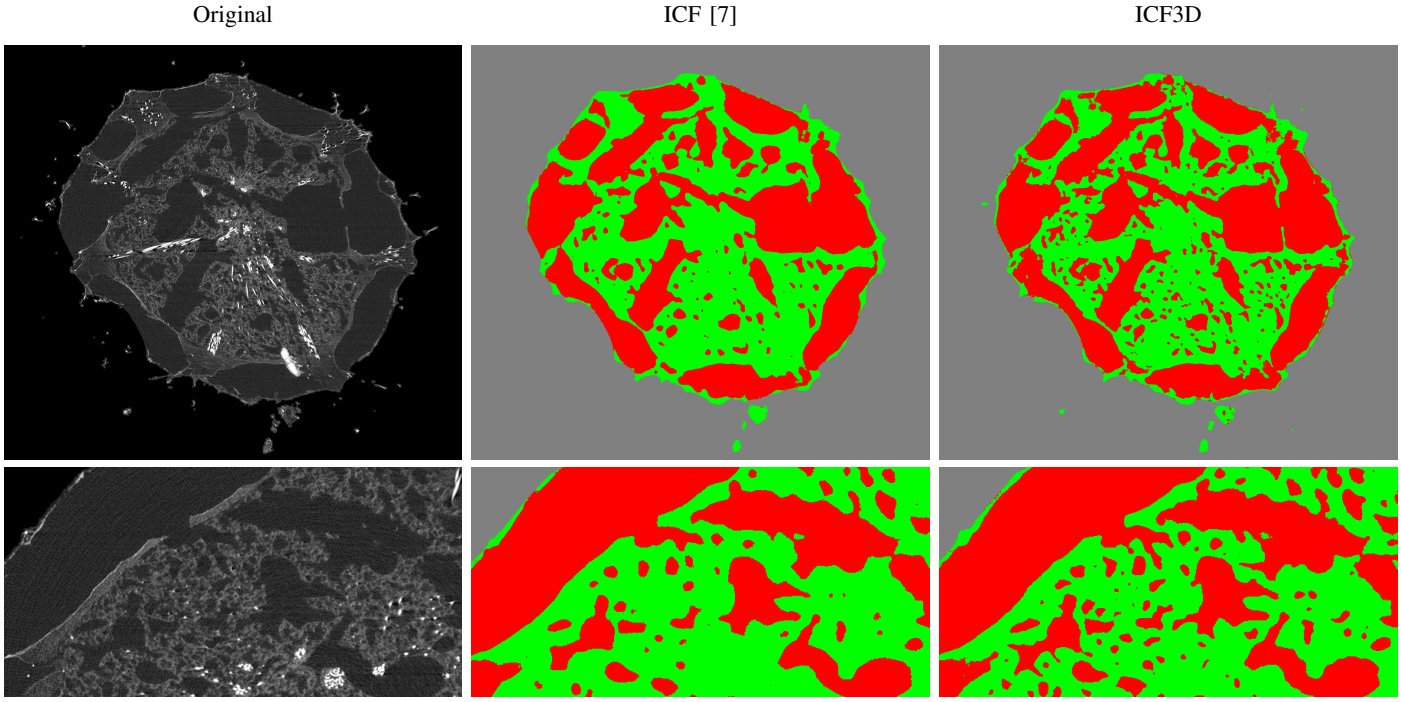


Fig. 4: Results for the sponge CT data: The first row depicts slice #85 of the testing stack and corresponding segmentation results by ICF [7] and ICF3D with labeled classes *canal* (red), *tissue* (green) and *background* (gray). Second row shows an enhanced comparison of slice #259. Our ICF3D approach improves segmentation results of subtle canal structures. Figures are best viewed in color.

V. Experiments

We evaluate our method on two different datasets. For a qualitative demonstration of ICF3D, we use data of a CT scan. A quantitative comparison with other approaches using an EM stack of neural tissue is done after that.

A. CT data of a sponge

We are using CT scanning data of a sponge with an isotropic resolution of 3.4 micrometers per pixel. The resolution of each slice is 1536×1536 pixels. The task is to distinguish between the classes *canals*, *tissue*, and *background*. There is a stack of eleven labeled slices and a stack of 351 unlabeled slices. We used the former stack for training and the second one for testing. All images feature noise and artifacts from the recording procedure. The training slices were accurately labeled by a biologist. In Fig. 4 (a) you can find a typical image. According to the expert, all dark gray structures belong to the class *canals*, even the tiny ones. A magnified part of another slice can be seen in Fig. 4 (d).

Because the resolution of this data is in all dimensions the same, a 2D window of the neighborhood for each pixel can be simply replaced by a cube with the same side length. In consequence, we can run the experiments with ICF [7] and ICF3D with comparable parameter configurations. We use feature maps originating from gray values and gradients. Color information is not available for this data.

Figure 4 demonstrates the potential of our volume segmentation method. Images (c-e) and (f-h) each show a slice of the test stack and the voxel classification results of [7] and ICF3D. As can be seen our approach shows improvements when it comes to subtle structures. Without using 3D information from nearby slices the segmentation is more coarse and fine structures get lost.

B. EM stack of neuronal tissue

We use the freely available *Drosophila* first instar larva ventral nerve cord (VNC) data [19] of the ISBI 2012 challenge. A stack with 30 slices is labeled to distinguish between the classes *membrane* and *interior* of neurons. The resolution of these stacks is $4 \times 4 \times 50$ nanometers, which is rather coarse in z-direction.

To account for this setting, we adapt the neighborhood in z-direction in an analogous way. The length of the neighborhood cuboid in z-direction is only 10% of the length in the other two directions, which are of higher resolution. For instance, a neighborhood of size $d = 50$ represents a $50 \times 50 \times 5$ cuboid in the image volume. In consequence, only two slices in front of and behind the current slice are used in the process.

We report results for the measures *pixel error* and *rand error* and compare our performance with selected methods of the challenge. The former one is a common measure in binary classification of pixels and is also referred to as accuracy or overall recognition rate. The second measure is related to the well known F-score, which is the harmonic mean of precision and recall. However, in this specific task the so-called *Rand index* is used for the computation. It is a measure of similarity

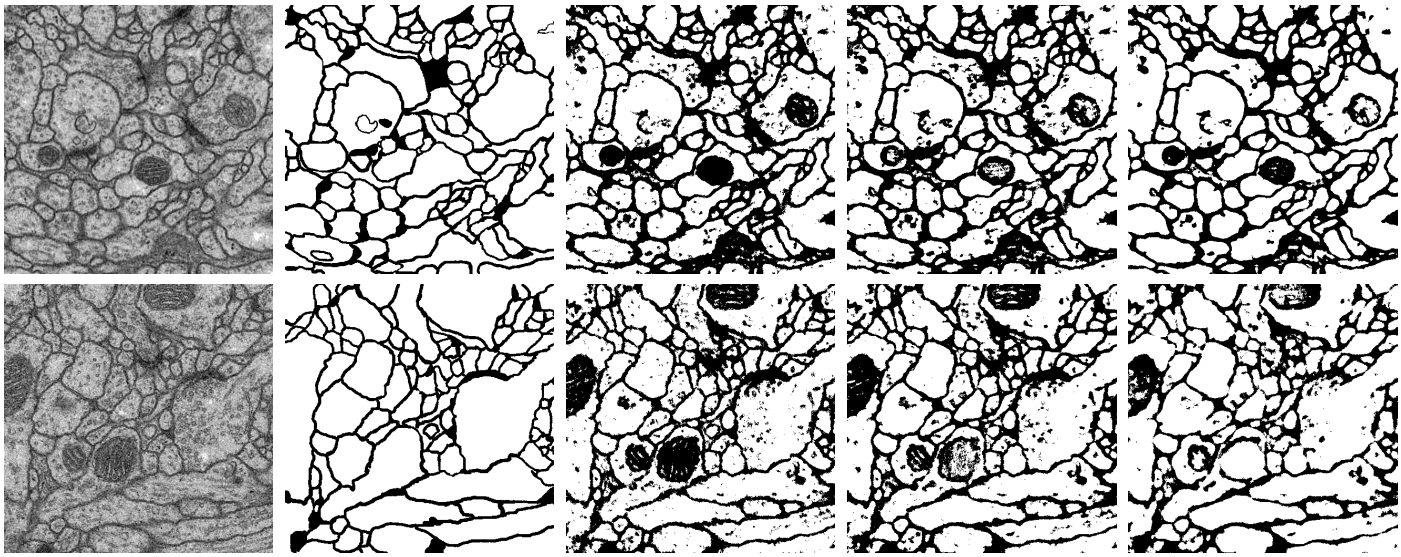


Fig. 5: Qualitative comparison of the neural tissue dataset using different types for the feature extraction. When only using pixel pair features the results appear to be cluttered. Especially in the larger interior areas the labels of nearby pixels are not consistent. Furthermore, synapses are labeled as cell membrane. These issues lessen or even vanish when Haar-like features and context features are added.

between two clusters or segmentations. The most important measure in the ISBI 2012 challenge is the rand error and we therefore also sort our results by this measure (see Table I).

In order to model the textural differences between membrane and mitochondria in the data not visible in the gradients maps, we incorporate local binary patterns (LBP) [20] as additional feature maps. They are a powerful tool in texture classification and increased the performance slightly in our case.

As can be seen in the rand error column, we are not able to produce state-of-the-art results. When taking the actual pixel errors into consideration ICF3D performs decently. It is a typical binary segmentation task, where modeling of context is hardly possible. However, we achieve comparable results to [1] and the patch-based SVM of [21]. The convolutional network approach of [15] shows best performance for this data. Note, that we are not using task specific knowledge (*e.g.* shape information) or applying any post-processing for smoothing.

Method	**Rand error** $[\cdot 10^{-2}]$	**Pixel error** $[\cdot 10^{-2}]$
CNN [15]	4.8	6.0
Dense correspondence [3]	6.4	8,3
Watershed Tree [22]	8.4	13.4
Perceptual Grouping [1]	8.4	15.7
CellProfiler [23]	9.0	10.0
Segment features [24]	13.9	10.2
Two-step class. [25]	15.3	8.8
Contextual Grouping [26]	16.2	10.9
Patch-based SVM [21]	23.0	15.0
ICF [7]	28.1	13.5
Ours: ICF3D w/o LBP	24.1	12.4
Ours: ICF3D w/ LBP	22.9	12.4

Table I: Results of some competitors and our generic method on the ISBI 2012 challenge data [19].

C. Analysis of feature types

In a third set of experiments, we analyzed the influence of different feature types (see Sec. IV-A) for the segmentation of neural tissue. For this task, we only used the training data of the ISBI 2012 challenge data, because ground truth labels are available. We split the 30 slices of the training stack into six stacks of five slices each to do a 6-fold cross-validation. This allowed us to analyze performance without using the evaluation server.

In Fig. 5, we show a qualitative comparison of two slices taken from two different cross-validation runs. While the first two columns depict original input images and their corresponding ground truth annotation the other three columns show results with different feature extraction methods. It can be seen that the use of pixel pair differences is sufficient to segment cell membranes. However, synapses are also labeled as membrane and areas of cell interior are not very homogeneous. This is due to the limitation of single pixel values instead of average values in feature maps across whole regions.

When rectangle features and Haar-like features are added, some parts of the synapses are not labeled as membrane anymore. Furthermore, the inhomogeneity in the cell interiors is less. Both issues even improve when context features taken

from probability maps are incorporated and some wrong labeled synapses even vanish.

For a quantitative evaluation we used the provided script of the ISBI 2012 challenge. When only pixel pair differences are allowed the pixel error is 15.6% (average of 6-fold cross-validation). With incorporated rectangle and Haar-like features the error decreases to 14.2%. The best result of 13.5% with respect to the pixel error can be achieved when context features are used.

VI. Conclusions

In this paper, we presented a fast method for volume image segmentation. We extended an existing semantic segmentation method that incorporates context information. Our proposed algorithm is able to segment even subtle structures in CT data. The method is also applicable to EM stacks, as we have shown for the ISBI 2012 challenge data.

We also did some more experiments with data of EM stacks and CT scans and discovered that incorporating information from nearby slices can also be misleading. Whether segmentation performance improves with these information highly depends on the quality of the data and on the recording method itself.

While CT imagery usually creates volume images with isotropic resolution and similar illumination at each slice, EM stacks often have a coarse resolution in z-dimension as well as illumination and contrast changes. On the other hand, CT scans usually contain artifacts or noise. These things have to be considered when working with such stacks to attain improvements over sequential 2D slice segmentation.

Acknowledgment

The authors would like to thank Henry Jahn and Jörg U. Hammel of the *Porifera.net Lab* at the *Friedrich Schiller University Jena* for providing pixel-wise labeled CT imagery data of sponges.

References

[1] V. Kaynig, T. Fuchs, and J. Buhmann, "Neuron geometry extraction by perceptual grouping in sstem images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 2902–2909.

[2] A. Vazquez-Reina, M. Gelbart, D. Huang, J. Lichtman, E. Miller, and H. Pfister, "Segmentation fusion for connectomics," in *Proceedings of the International Conference on Computer Vision (ICPR)*, 2011, pp. 177–184.

[3] D. Laptev, A. Vezhnevets, S. Dwivedi, and J. Buhmann, "Anisotropic sstem image segmentation using dense correspondence across sections," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2012, pp. 323–330.

[4] L. Ce, , J. Yuen, and A. Torralba, "Sift flow: Dense correspondence across scenes and its applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 33, pp. 978–994, 2011.

[5] A. Montillo, J. Shotton, J. Winn, J. Iglesias, D. Metaxas, and A. Criminisi, "Entangled decision forests and their application for semantic segmentation of ct images," in *Proceedings of the International Conference on Information Processing in Medical Imaging (IPMI)*, 2011, pp. 184–196.

[6] Z. Tu and X. Bai, "Auto-context and its application to high-level vision tasks and 3d brain image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 32, no. 10, pp. 1744–1757, 2010.

[7] B. Fröhlich, E. Rodner, and J. Denzler, "Semantic segmentation with millions of features: Integrating multiple cues in a combined random forest approach," in *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2012, pp. 218–231.

[8] G. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds." in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2008, pp. 44–57.

[9] C. Zhang, L. Wang, and R. Yang, "Semantic segmentation of urban scenes using dense depth maps." in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2010, pp. 708–721.

[10] B. Fröhlich, E. Rodner, and J. Denzler, "A fast approach for pixelwise labeling of facade images," in *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2010, pp. 3029–3032.

[11] C. Galleguillos, A. Rabinovich, and S. Belongie, "Object categorization using co-occurrence, location and appearance." in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[12] L. Ladicky, C. Russell, P. Kohli, and P. Torr, "Graph cut based inference with co-occurrence statistics," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2010, pp. 239–253.

[13] X. Boix, J. Gonfaus, J. van de Weijer, A. Bagdanov, J. Serrat, and J. GonzÃ lez, "Harmony potentials - fusing global and local scale for semantic image segmentation," *International Journal of Computer Vision (IJCV)*, vol. 96, no. 1, pp. 83–102, 2012.

[14] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[15] D. C. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," in *Advances in Neural Information Processing Systems (NIPS)*, 2012, pp. 2852–2860.

[16] S. Esmeir and S. Markovitch, "Anytime learning of anycost classifiers," *Machine Learning*, vol. 82, no. 3, pp. 445–473, 2011.

[17] F. C. Crow, "Summed-area tables for texture mapping," in *International Conference and Exhibition on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1984, pp. 207–212.

[18] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision (IJCV)*, vol. 57, pp. 137–154, 2002.

[19] A. Cardona, S. Saalfeld, S. Preibisch, B. Schmid, A. Cheng, J. Pulokas, P. Tomancak, and V. Hartenstein, "An integrated micro- and macroarchitectural analysis of the drosophila brain by computer-assisted serial section electron microscopy," *PLoS Biology*, vol. 8, no. 10, 2010.

[20] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 24, no. 7, pp. 971–987, 2002.

[21] S. Iftikhar and A. Godil, "The detection of neuronal structure using patch-based multi-features and support vector machines algorithm," in *Proceedings of ISBI 2012 EM Segmentation Challenge*, 2012.

[22] T. Liu, M. Seyedhosseini, E. Jurrus, and T. Tasdizen, "Neuron segmentation in em images using series of classifiers and watershed tree," in *Proceedings of ISBI 2012 EM Segmentation Challenge*, 2012.

[23] L. Kamentsky, "Segmentation of em images of neuronal structures using cellprofiler," in *Proceedings of ISBI 2012 EM Segmentation Challenge*, 2012.

[24] R. Burget, V. Uher, and J. Masek, "Trainable segmentation based on local-level and segment-level feature extraction," in *Proceedings of ISBI 2012 EM Segmentation Challenge*, 2012.

[25] X. Tan and C. Sun, "Membrane extraction using two-step classification and post-processing," in *Proceedings of ISBI 2012 EM Segmentation Challenge*, 2012.

[26] E. Bas, M. G. Uzunbas, D. Metaxas, and E. Myers, "Contextual grouping in a concept: a multistage decision strategy for em segmentation," in *Proceedings of ISBI 2012 EM Segmentation Challenge*, 2012.