SEGMENTING NEURONS IN ELECTRONIC MICROSCOPY VIA GEOMETRIC TRACING

Luis Vazquez
Institute of Electrical Engineering
Universidad de la Republica
Montevideo, Uruguay
Guillermo Sapiro
Electrical and Computer Engineering
University of Minnesota
Minneapolis, MN 55455
guille@ece.umn.edu
Gregory Randall
Institute of Electrical Engineering
Universidad de la Republica
Montevideo, Uruguay

ABSTRACT

In this paper we describe a system that is being used for the segmentation of neurons in images obtained from electronic microscopy. These images are extremely noisy, and ordinary active contours techniques detect spurious objects and fail to detect the neuron boundaries. The algorithm here described is based on combining robust anisotropic diffusion with minimal weighted-path computations. After the image is regularized via anisotropic diffusion, the user clicks points on the boundary of the desired object, and the algorithm completes the boundary between those points. This tracing is based on computing paths of minimal weighted distance, where the weight is given by the image edge content. Thanks to advanced numerical algorithms, the algorithm is very fast and accurate. We compare our results with those obtained with *PictureIt*, a commercially available general purpose image processing package developed by Microsoft.

Key words: Electronic microscopy, neurons, anisotropic diffusion, weighted distances, geometric tracing, segmentation, curve evolution.

1. INTRODUCTION

Figure 1 shows an image of a neuron from the central nervous system. This image was obtained via electronic microscopy (EM). After the neuron is identified, it is marked via the injection of a color fluid. Then, a portion of the tissue is extracted, and after some processing, it is cut into thin slices and observed and captured via the EM system. The figure shows the output of the EM after some simple post-processing, mainly composed by contrast enhancement. The goal of the biologist is to obtain a three dimensional reconstruction of this neuron. As we observe from the example in Figure 1, the image is very noisy, and the boundaries of the neuron are difficult to identify. Segmenting the neuron is then a difficult task. In this paper we describe a semi-automatic system to obtain fast and accurate segmentations of the 2D slices. From these, the 3D neuron can be reconstructed.

One of the most commonly used approaches to segment objects as the neuron in Figure 1 are *active contours* or *snakes* [7, 18]. This technique is based on deforming a curve toward the minimization of a given energy. This energy is mainly composed by two terms, one attracting the curve to the objects boundaries, and the other one addressing regularization properties of the deforming curve. In [4, 5], it was shown that a re-interpretation of the classical snakes model leads to the formulation of the segmentation problem as the minimization of a weighted length given by

$$\int_{\mathcal{C}} (g(\| \nabla(\mathcal{I}) \|) + \alpha) ds, \qquad (1)$$

where $\alpha \in [0,1]$, $\mathcal{C} : I\!R \to I\!R^2$ is the deforming curve, $\mathcal{I} : I\!R^2 \to I\!R$ the image, ds stands for the curve arc-length ($\| \partial\mathcal{C}/\partial s \| = 1$), $\nabla(\cdot)$ stands for the gradient, and $g(\cdot)$ is such that $g(r) \to 0$ while $r \to \infty$ (the "edge detector"). This model means that finding the object boundaries is equivalent to computing a path of minimal weighted distance, a *geodesic curve*, with weight given by $g(\cdot)$ (see also [9, 17, 20]). This model not only improves classical snakes, but also provides a formal mathematical framework that connects between previous models (e.g., between [7] and [10]); see [5] for details.

THIS WORK WAS PARTIALLY SUPPORTED BY NSF-LIS, BY THE MATH, COMPUTER, AND INFORMATION SCIENCES DIVISION AT ONR, AND BY AN ONR YOUNG INVESTIGATOR AWARD.

There are two main techniques to find the geodesic curve, that is, the minimizer of (1):

1. Compute the gradient descent of (1), and starting from a closed curve either inside or outside the object, deform it toward the (possibly local) minima, finding a geodesic curve. This approach gives a curve evolution flow, based on curvature motion, leading to very efficient solutions for a large number of applications. This was the approach followed in [5], where the model was first introduced. This model gives a completely automatic segmentation procedure (modulo initialization). When tested with images like the one in Figure 1, we found two major drawbacks [2]. First, due to the large amount of noise, spurious objects are detected, and is left to the user to manually eliminate them. Second, due to the fact that the boundary of the real neuron is very weak, this is not always detected. An initialization very close to the goal is then required.
2. Connect between a few points marked by the user on the neuron's boundary, while keeping the weighted length (1) to a minimum. This was developed in [6], and it is the approach we follow in this work. In contrast with the technique described above, this approach always needs user intervention to mark the initial points. On the other hand, for images like the one in Figure 1, it permits a better handling of the noise. In the rest of this paper we will briefly describe this technique and the additions incorporated to address our specific problem.

2. COMPUTING THE MINIMAL GEODESIC

We now describe the algorithm used to compute the minimal weighted path between points on the objects boundary. That is, given a set of boundary points $\{\mathcal{P}\}_{i=1}^{N+1}$, and following (1), we have to find the N curves that minimize ($\mathcal{P}_{N+1} \equiv \mathcal{P}_1$)

$$d(I(\mathcal{P}_i), I(\mathcal{P}_{i+1})) := \int_{\mathcal{P}_i}^{\mathcal{P}_{i+1}} (g(\| \nabla \mathcal{I} \| + \alpha) ds. \qquad (2)$$

The algorithm is composed of three main steps: 1- Image regularization, 2- Computation of equal distance contours, 3- Back propagation. We briefly describe each one of these steps now. For details on the first step, see [3]. For details on the other steps, see [6].

2.1. Image regularization

In order to reduce the noise on the images obtained from EM, we perform the following two steps:

1. Subsampling.

   We use a 7×7, symmetric and separable filter, approximating a Gaussian function, to smooth the image before a 2×2 subsampling is performed. This not only removes noise and regularizes the neuron border, but also gives a smaller image to work with, thereby accelerating the algorithm by a factor of 4. That is, we will work on the sub-sampled image (although the user marks the end points on the original image), and only after the segmentation is computed, the result is extrapolated to the original size image. Further subsampling was found to already produce not very accurate results. The result from this step is then an image $\mathcal{I}_{2\times2}$ which is one quarter of the original image $\mathcal{I}$.
2. Robust anisotropic diffusion.

   In order to further reduce noise in the image, we smooth it with an anisotropic diffusion technique, hereby preserving the edges. The denoising flow is obtained from the gradient descent of

   $$\int_\Omega \rho(\| \nabla \mathcal{I}_{2\times2} \|) d\Omega,$$

   which is given by

   $$\frac{\partial \mathcal{I}_{2\times2}}{\partial t} = \text{div}\left(\rho'(\| \nabla \mathcal{I}_{2\times2} \|) \frac{\nabla \mathcal{I}_{2\times2}}{\| \nabla \mathcal{I}_{2\times2} \|}\right),$$

   where ρ is the Tukey's biweight robust function. This flow is an extension of the anisotropic diffusion technique introduced by Perona and Malik [12]. We also found that the algorithm in [1] can be used here.

At the end of the pre-processing stage we then obtain an image $\hat{\mathcal{I}}_{2\times2}$ which is the result of the subsampling of $\mathcal{I}$ followed by edge-preserving noise removal. Although the user marks the points $\{\mathcal{P}\}_{i=1}^{N}$ on the original image $\mathcal{I}$, the algorithm makes all the computations on $\hat{\mathcal{I}}_{2\times2}$ and then extrapolates and displays them on $\mathcal{I}$.

2.2. Equal distance contours computation

After the image $\hat{\mathcal{I}}_{2\times2}$ is computed, we have to compute, for every point $\hat{\mathcal{P}}_i$, where $\hat{\mathcal{P}}_i$ is the point in $\hat{\mathcal{I}}_{2\times2}$ corresponding to the point $\mathcal{P}_i$ in $\mathcal{I}$ (coordinates divided by two), the weighted distance map, according to the weighted distance d. That is, we have to compute the function

$$\mathcal{D}_i(x,y) := d(\hat{\mathcal{I}}_{2\times2}(\hat{\mathcal{P}}_i), \hat{\mathcal{I}}_{2\times2}(x,y)),$$

or in words, the weighted distance between the pair of image points $\hat{\mathcal{P}}_i$ and (x,y).

There are basically two ways of making this computation, computing equal distance contours, or directly computing $\mathcal{D}_i$. We briefly describe each one of these now.

Equal distance contours $\mathcal{C}_i$ are curves such that all the points on the contour have the same distance d to $\hat{\mathcal{P}}_i$. That is, the curves $\mathcal{C}_i$ are the level-sets or isophotes of $\mathcal{D}_i$. It is easy to see, [6], that following the definition of d, these contours are obtained as the solution of the curve evolution flow

$$\frac{\partial \mathcal{C}_i(x,y,t)}{\partial t} = \frac{1}{g(\| \nabla \hat{\mathcal{I}}_{2\times2} \|)} \vec{\mathcal{N}},$$

where $\vec{\mathcal{N}}$ is the outer unit normal to $\mathcal{C}_i(x,y,t)$. This type of flow should be implemented using the standard level-sets method [11].

A different approach is based on the fact that the distance function $\mathcal{D}_i$ holds the following Hamilton-Jacobi equation [8, 16, 19]:

$$\frac{1}{g(\| \nabla \hat{\mathcal{I}}_{2\times 2} \|)} \| \nabla \mathcal{D}_i \| = 1.$$

Optimal numerical techniques have been proposed to solve this static Hamilton-Jacobi equation [8, 16, 19]. Due to this optimality, this is the approach we follow in our algorithm. At the end of this step, we have $\mathcal{D}_i$ for each point $\hat{\mathcal{P}}_i$. We should note that we do not need to compute $\mathcal{D}_i$ for all the image plane. It is actually enough to stop the computations when the value at $\hat{\mathcal{P}}_{i+1}$ is obtained.

2.3. Back propagation

After the distance functions $\mathcal{D}_i$ are computed, we have to trace the actual minimal path between $\hat{\mathcal{P}}_i$ and $\hat{\mathcal{P}}_{i+1}$ that minimizes d. Once again it is easy to show (see for example [8, 16]), that this path should be perpendicular to the level-curves $\mathcal{C}_i$ of $\mathcal{D}_i$, and therefore tangent to $\nabla\mathcal{D}_i$. The path is then computed backing from $\hat{\mathcal{P}}_{i+1}$, in the gradient direction, until we return to the point $\hat{\mathcal{P}}_i$. This back propagation is of course guaranteed to converge to the point $\hat{\mathcal{P}}_i$, and then gives the path of minimal weighted distance.

3. EXAMPLES

We present now a number of examples of the algorithm described above. We compare our results with those obtained with *PictureIt*, a commercially available general purpose image processing package developed by Microsoft. [1] As in our algorithm, this software allows for the user to click a few points on the object's boundary, while the program automatically completes the rest of it. Three to five points are used for each one of the examples. The points are usually marked at extrema of curvature or at areas where the user, after some experience, predicts possible segmentation difficulties. The same points were marked in our algorithm and in *PictureIt*. The results are shown in figures 2, 3, and 4. We observe that our technique outperforms *PictureIt*. Moreover, we found *PictureIt* to be extremely sensible to the exact position of the marked points, a difference of one or two pixels can cause a very large difference in the segmentation results. Our algorithm is very robust to the exact position of the points marked by the user.

4. CONCLUDING REMARKS

We have described a system that is being used to segment images from electronic microscopy. The performance of this system was compared with that of *PictureIt*, a general purpose image processing package developed by Microsoft. Following the framework in [13, 14], we are now working on the extension of this technique to color and texture data, to deal with cropping and re-touching of natural images [15].

[1]To the best of our knowledge, the exact algorithm used by this product was not published.

Acknowledgments

We wish to thank Dr. R. Kimmel from the Technion for discussions regarding his algorithms.

5. REFERENCES

[1] L. Alvarez, P. L. Lions, and J. M. Morel, "Image selective smoothing and edge detection by nonlinear diffusion," *SIAM J. Numer. Anal.* **29**, pp. 845-866, 1992.

[2] M. Bertalmio, G. Sapiro, and G. Randall, "Morphing active contours: A geometric, topology-free, technique for image segmentation and tracking," *Proc. IEEE ICIP*, Chicago, October 1998.

[3] M. Black, G. Sapiro, D. Marimont, and D. Heeger, "Robust anisotropic diffusion," *IEEE Trans. Image Processing*, March 1998.

[4] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," *Proc. Int. Conf. Comp. Vision '95*, Cambridge, June 1995.

[5] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," *International Journal of Computer Vision* **22:1**, pp. 61-79, 1997.

[6] L. Cohen and R. Kimmel, "Global minimum for active contours models: A minimal path approach," *Int. J. of Computer Vision* **24**, pp. 57-78, 1997.

[7] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision* **1**, pp. 321-331, 1988.

[8] R. Kimmel and J. A. Sethian, "Fast marching method for computation of distance maps," *LBNL Report* **38451**, UC Berkeley, February, 1996

[9] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi, "Conformal curvature flows: from phase transitions to active vision," *Archive for Rational Mechanics and Analysis* **134**, pp. 275-301, 1996.

[10] R. Malladi, J. A. Sethian and B. C. Vemuri, "Shape modeling with front propagation: A level set approach," *IEEE-PAMI* **17**, pp. 158-175, 1995.

[11] S. J. Osher and J. A. Sethian, "Fronts propagation with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations," *Journal of Computational Physics* **79**, pp. 12-49, 1988.

[12] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE-PAMI* **12**, pp. 629-639, 1990.

[13] G. Sapiro, "Color snakes," *Computer Vision and Image Understanding* **68:2**, pp. 247-253, 1997.

[14] G. Sapiro and D. Ringach, "Anisotropic diffusion of multivalued images with applications to color filtering," *IEEE Trans. on Image Processing* **5**, pp. 1582-1586, 1996.

[15] G. Sapiro, "Color cutting edge: A geometric technique for image cropping," in preparation.

[16] J. Sethian, *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision and Materials Sciences,* Cambridge University Press, Cambridge-UK, 1996.

[17] J. Shah, "Recovery of shapes by evolution of zero-crossings," Technical Report, Math. Dept. Northeastern Univ. Boston MA, 1995.

[18] D. Terzopoulos, A. Witkin, and M. Kass, "Constraints on deformable models: Recovering 3D shape and non-rigid motions," *AI* **36**, 1988.

[19] J. N. Tsitsiklis, "Efficient algorithms for globally optimal trajectories," *IEEE Transactions on Automatic Control* **40** pp. 1528-1538, 1995.

[20] R. T. Whitaker, "Algorithms for implicit deformable models," *Proc. ICCV'95*, pp. 822-827, Cambridge, June 1995.

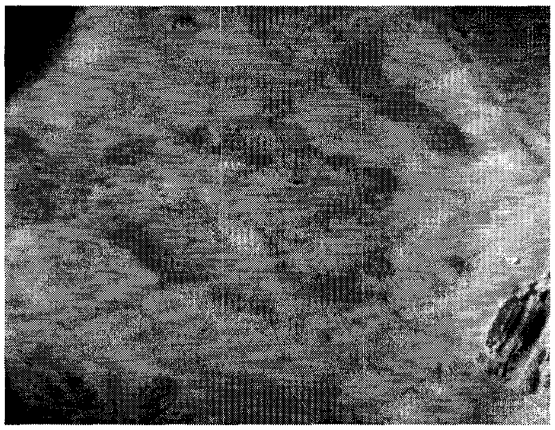

Figure 1: *Example of an EM image of a neuron (one slice).*

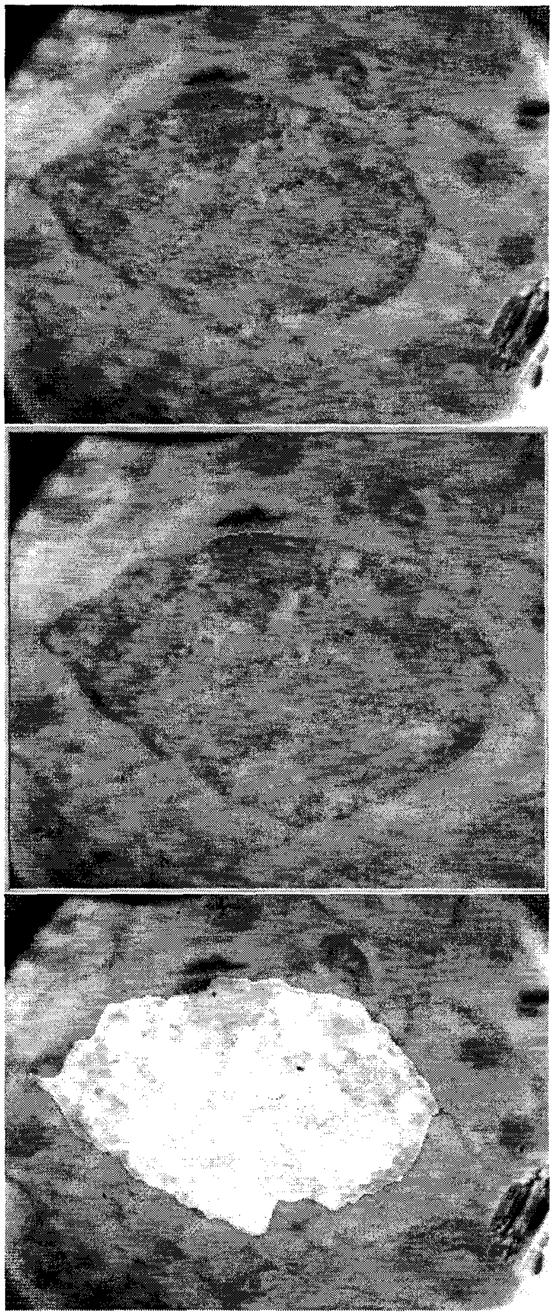

Figure 2: *Comparison of our results with those obtained with the commercial software PictureIt. The original image is shown on the top, the result of our algorithm (green line) on the bottom-left, and the result of PictureIt on the bottom-right. For this last, the area segmented by the algorithm is shown brighter. (This is a color figure.)*

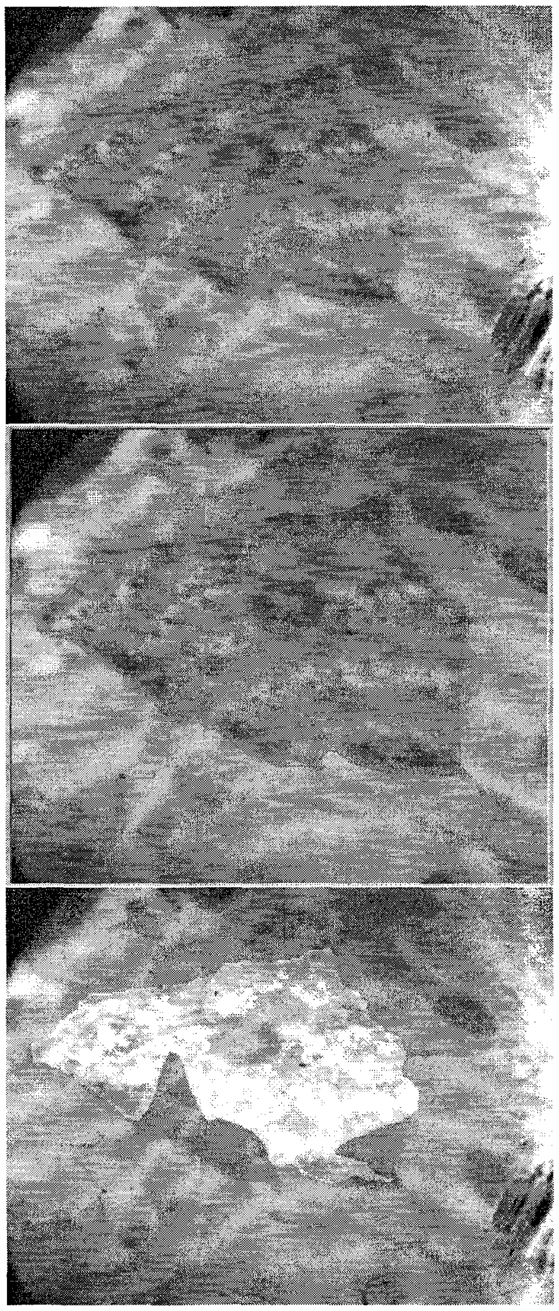

Figure 3: *Same as Figure 2 for an additional image. (This is a color figure.)*

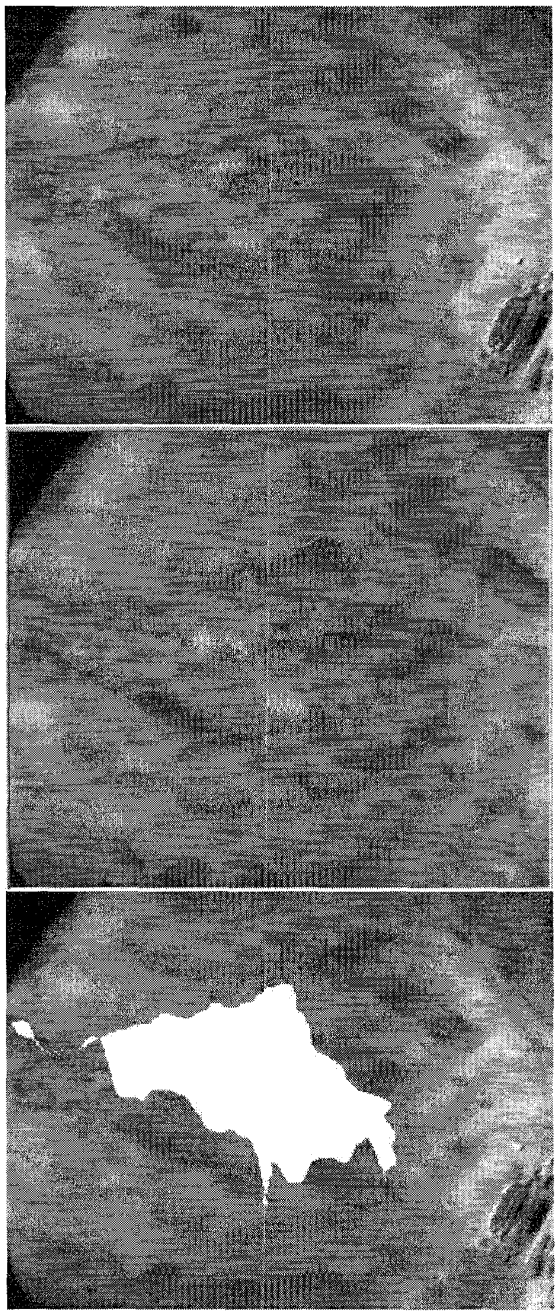

Figure 4: *Same as Figure 2 for an additional image. (This is a color figure.)*