Automatic 3D-Graph Construction of Nerve Cells from Confocal Microscope Scans

Anca Dima, Michael Scholz, Klaus Obermayer

Technische Universität Berlin, Fakultät IV - Elektrotechnik & Informatik
Sekretariat FR 2-1, Franklinstr. 28/29, D-10587 Berlin, Germany

Abstract We present a method of automatic graph construction for the description of the geometric structure of nerve cells from 3D confocal scans. The method is tracing the branch mid points, in the branch axial direction using as hints the location of difficult regions inside the neuronal branches. The axes were obtained in previous work by computing pairwise vector products of intersecting gradients associated to across-scales validated boundary edge points of the neuronal branches. The axes' anchor points are the branch mid points which are estimated as the "center of mass" of all intersecting gradient end points. The difficult regions are the axes' anchor points having a high directional variance of vector products contributing to the associated ax. The presented algorithm which uses all the information obtained from preprocessing, is robust to variable contrast, little sensitive to boundary irregularities, adaptive to variability of branch geometry, and produces a sparse, topology preserving graph of the neuron under investigation. A subsequent surface reconstruction based on this graph [21] accompanied by the labeling of the graph with geometric measurements, would be feasible.

Keywords: automatic graph construction, confocal microscopy, neuronal data, multiscale image decomposition, wavelets.

1 Introduction

In order to exploit the relationship between morphological and functional properties of neurons an exact geometrical reconstruction is needed. Based on this representation, detailed compartment models [6] could be constructed, to for example estimate those physiological parameters which could not easily be measured. The analysis of such a model allows to characterize the functional role of neurons in great detail and helps to understand neuronal circuitry and their malfunction in neuro-degenerative diseases. For the comparison and the classification of a large number of neuronal cells and for compartment model construction, an automatic method for the accurate geometric description of the neurons is needed. The current available software (Eutectic, Neurolucida, Imaris, Imagespace) facilitate the manual drawing of microscope images and allow only a $2\frac{1}{2}$ D reconstruction. These techniques demand several weeks of a specialist's time for one neuronal reconstruction and cannot offer the objectivity of an automatic tool.

The geometrically labeled graph description of the medial lines of the neuronal branches is a suitable data structure for a description of neuronal cells envisioning the derivation of compartment models from it. Other useful descriptions regarding the geometrical complexity of cells were derived in [25]. The problem of medial line construction is discussed very broadly in literature, starting with abrasive procedures like grass fire [1] and thinning algorithms [16], continuing with analytical solutions of the center line construction of difficult parameterized 2D shapes [24] and not the least with 2D wavelet edge detection, followed by boundary reconstruction and high curvature detection [4].

Thinning algorithms [16] rely on the deletion of border points, as long as their deletion is not disrupting the topological structure of the underlying data. Unfortunately, these procedures are sensitive to noise and to border irregularities of the objects. The expansion to 3D is not straightforward since the search space explodes exponentially (there are 3^{26} possible configurations of the $3 \times 3 \times 3$ neighborhood of a point in 3D, since each voxel can have the values black/white/don't care) [14] and a definition of minimal skeletons for some elementary 3D shapes cannot be formulated uniquely [15,19].

The analytic solution of medial line reconstruction [24] is a geometrically precise procedure, but it demands an analytical, parameterized form of the input data rather than the common discrete pixelized form. Whereas a parametric description of the object borders is feasible in 2D, the approximation of parametric surfaces in 3D is computationally very expensive.

3D center line graph construction is mostly done on cylindrically shaped biological data, like bronchial arterioles, colon or sulci of the brain [27]. This kind of data is relatively simple to process, since the images are of high contrast with relatively smooth contours. The analysis of 3D neuronal microscope images is more

difficult, since different neuronal scans can have very different spatial branch distributions and differing signal to noise ratio (Figure 9, Section 4.1). The signal can be locally very low, such that the neuronal processes are interrupted. Contrast can vary significantly inside one single branch, whereas the width of branches can differ by an order of magnitude (Figure 20.a). Some specific problems of neuronal data are discussed in more detail in [13,23,11,2,3,22].

Currently, there are several different approaches to extract 3D center line graphs, like: 1) thinning [5,26], or 2) continuous skeletonization by the construction of Voronoi Diagrams [18,17], 3) distance maps containing the minimal distance to the object's boundary for each discrete point [8,9], 4) seed fill [5] or seed fill followed by clustering of equally distanced points labeled by two different distance measures [27], 5) tracing the highest gray values of the original gray value image along a smooth path [12], 6) surface shrinking in the gradient direction of a distance field [20], which is related to the analytical 2D algorithm of [24] and to the algorithms based on distance transforms.

Some of these approaches [26,27,8,9,20] work on data, which is segmented with a global threshold and which will be either incomplete or will contain too much noise. Both can impede the skeletonization process. Others [12] assume non-fluctuating gray values of the foreground data and therefore may lose parts of the neuron which have low contrast. Approaches based on distance maps [18,17,8,9,20] or thinning [26] are sensitive to boundary irregularities.

In the current work, we present a method which constructs center line graphs from the skeleton trace points, their associated axial directions, and information about high curvature regions which are all calculated in one step from discrete multiscale edges and their gradient directions as described in previous work [7].

The graph construction algorithm should have following properties:

- noise robustness;
- robustness to object boundary irregularities;
- independence to branch thickness;
- independence to fluctuating branch contrast;
- topology preservation;

We will show in the following how the first three properties can be achieved. The fourth property is very hard to achieve, since neuronal data contains noisy regions, where not even the human observer can decide if the voxels belong to foreground or background.

Consequently we describe briefly in Section 2 the preprocessing steps which are necessary for the construction of the skeleton trace points and the longitudinal axes of neuronal branches. This information is then used in Section 3 by the skeletonization algorithm. Section 4 presents the analyzed data and shows intermediate results of the different processing steps. A discussion and comparison with other skeletonization procedures is done in Section 5 and conclusions are drawn in Section 6.

2 Preprocessing of the Confocal Microscope Data

3D image data is preprocessed as described in [7]. Edges are extracted using a 3D multiscale wavelet transform, which is followed by a validation of edges (noise vs. object boundaries) by comparing neighboring scales. Validation of edges across scales strongly enhances edges which belong to an object's boundary, while noise can be almost completely eliminated. The direction of the gradient of gray values at each edge point is then used to compute the “**medial trace points**”. These are the centers of mass of the edge points of intersecting gradients and therefore form a marrow-like, raw, few voxel wide skeleton. The vector associated to each “medial trace point” is perpendicular to the mean plane determined by the above participating gradient directions and therefore represents an estimate of the “**axial direction**” of the branch. The variance of the directions of the participating planes measures the reliability of the respective axial direction and it is proportional to the length of the axial vectors (i.e. long vectors have low variance of direction and short vectors denote axial directions with high variance). The points of highest variance typically correspond to varicosities (strongly stained spots inside the neuronal branches), bends, and branch points, and are points of high curvature. A flow diagram of the above processing steps is shown in Figure 1.

After preprocessing, the following information is available for further analysis:

- original gray value data,
- low pass filtered gray value data on each scale,
- edges of objects on several scales,
- the medial trace points which form a raw few voxels wide skeleton,
- estimates of the direction of the neuronal branches at each medial trace point,

Figure1. The Preprocessing Steps Needed for the Graph Construction. Thick rectangles denote final results, and thin rectangles are intermediate steps

- points of high variance of the direction estimate.

This information is now to be used to construct a *single, smooth, central* skeleton line for each neuronal branch, with branch points being accurately detected.

Because confocal microscope data is noisy and characterized by high variations in contrast, detected edges will be noisy even after the validation step and result in an incomplete “raw skeleton”, and in erroneous axial directions, i.e. errors due to noise are propagated. Therefore a fault tolerant and noise robust graph construction algorithm is needed to connect skeleton points using the information available after preprocessing.

3 The Graph Construction Algorithm

The graph construction algorithm builds skeleton lines by following the axial directions associated with the raw skeleton points. It searches for the optimal connection points, such that certain smoothness and confidence criteria are met.

As shown in Figure 2 the graph construction algorithm consists of several steps, which will be described in the following:

1. The choice of a “well suited” starting point (Section 3.1).
2. The construction of the skeleton line by stepwise following the axial directions of the medial trace points (Sections 3.2, and 3.3).
3. The merging of skeleton branches at branch points (Section 3.4).
4. The filling of remaining gaps in the skeleton (Section 3.5).

3.1 Determination of Starting Points

A good starting point should fulfill the following two requirements: (i) it should be a trace point with an associated axial direction which reflects with high probability the real direction of the neuronal branch, and (ii) it should not be part of an already constructed skeleton line nor of its neighborhood.

Since axial directions are noisy, special care must be taken in the selection of trace points. Otherwise, different sequences of trace points being considered may lead to different results. It is advisable to introduce

Figure2. Summary of the Graph Construction Algorithm. For details see text

processing steps which rank the trace points according to the reliability of their associated axial direction, i.e. according to their axial length. Trace points with longer axial directions (i.e. which lie in relatively smooth branch segments) are processed first. As we will see below this helps to avoid false tracing directions in critical regions (for a definition see Section 3.3) due to noise.

Here we propose the following method to chose an optimal starting point and the corresponding trace direction:

1. Rank all trace points by the length of their associated “axial direction” vector.
2. Choose the trace point with the highest axial length, which was not processed yet.
3. Choose the plane which is perpendicular to this axial direction and which is laterally limited by the branch edges.
4. BEGIN LOOP:
 - Calculate the most frequent *discrete* axial directions associated with trace points situated in this plane. In cases where more than one direction have the same highest frequency of appearance, the direction of the longer axes is chosen.
 - The next starting plane is perpendicular to the new found direction, centered in the center of mass of the points having this same direction. The plane is laterally limited by the branch edges.
 UNTIL the plane does not change its direction any more or a maximal number of iterations is reached.
5. The chosen start point is then the center of mass of all points with axial direction perpendicular to the last chosen plane.

A maximal number of iterations needs to be set in the above algorithm, to avoid infinite loops when axial directions are pointing equally in two or more directions. Here we have chosen to oscillate at most ten times, to allow the plane oscillation to converge (convergence is usually achieved after 2 – 3 iterations).

3.2 Construction of the Skeleton Line

Having found a start point, our goal is to construct a smooth skeleton line which follows the center of the neuronal branches. This is done in an iterative fashion by selecting “successor” graph nodes along the direction

of the neuronal branch (as indicated by the associated axial direction vectors). As soon as a medial trace point is visited by the line construction algorithm, it is marked, such that multiple processing of the same points is avoided. If a medial trace point is marked but it is not chosen to be a “successor” graph node, it registers as its parents the two nearest connected graph nodes between which it lies. This medial point becomes then the “child” of its parents. A “good” successor graph node fulfills the following conditions ¹:

C1. *Direction reliability:*

The successor Q_0 should have a reliable axial direction (low variance of direction), which is equivalent with a large norm $|\vec{A}_{Q_0}|$ of its associated axial direction vector $\vec{A}_{Q_0}$.

C2. *Axial direction consistency:*

The angle α between the axial direction $\vec{A}_{Q_0}$ of the successor Q_0 and the axial direction $\vec{A}_{P_0}$ of the predecessor P_0 should be small (Figure 3.a);

C3. *No abrupt direction changes:*

The angle β between the axial direction $\vec{A}_{P_0}$ of the predecessor P_0 and the next search direction (vector addition of $\vec{r}_{P_0 Q_0}$ between points Q_0 and P_0 and the axial direction $\vec{A}_{Q_0}$) should be small (Figure 3.b);

Figure 3. Criteria for the choice of successor points: a) Axial direction consistency (cf. Condition 2); b) Smooth directional changes (cf. Condition 3).

C4. *Preferred neighborhood size:*

Depending on the size and smoothness of the neuron, smaller or larger distances between graph nodes are more appropriate. Typically, for neurons with smooth branches skeleton points should have large distances between each other and therefore successors which are farther away should be preferred. For neurons with bended branches a skeleton with dense nodes is needed, in order to follow the more abundant changes in direction.

We now propose the following method to find the next skeleton point:

1. Construct a set of equally spaced (at one voxel from each other) planes which are perpendicular to the axial direction of the last graph node P_0 and which are laterally limited by the edges of the branch currently under consideration. All planes lie axially within a local neighborhood whose size $V_\delta \simeq \delta \|\vec{A}_{P_0}\|$ is proportional to the length of the current axial direction (i.e. in regions of low axial direction variance, wider neighborhoods are considered). The proportionality constant δ is set by the user and should roughly be equal to the overall mean diameter of the neuronal branches (measured in voxels) in the image.
2. To comply with the four conditions stated above, the next graph node Q_0 is chosen such that (i) it is not yet marked (see step 4) and (ii) the validation function T is maximal, where T is given by:

$$T_{n,m,l,\hat{\alpha}_0}(P_0, Q_i) = (\hat{\alpha}_{\vec{A}_{P_0}, \vec{A}_{Q_i}} \leq \hat{\alpha}_0) \cdot \min \left(\text{sgn}(\cos \hat{\alpha}_{\vec{A}_{P_0}, (\vec{r}_{P_0 Q_i} + \vec{A}_{Q_i})}), \text{sgn}(\cos \hat{\alpha}_{\vec{A}_{P_0}, \vec{A}_{Q_i}}) \right) \quad (1)$$

$$\cdot \left| \cos \hat{\alpha}_{\vec{A}_{P_0}, \vec{A}_{Q_i}} \|\vec{A}_{Q_i}\|^m \Omega_{n,l,\hat{\alpha}_0}(\vec{A}_{P_0}, (\vec{r}_{P_0 Q_i} + \vec{A}_{Q_i}), \vec{r}_{P_0 Q_i}) \right|,$$

$$\Omega_{n,l,\hat{\alpha}_0}(\vec{C}, \vec{D}, \vec{r}) = (\hat{\alpha}_{\vec{C}, \vec{D}} \leq \hat{\alpha}_0) \text{sgn}(\cos \hat{\alpha}_{\vec{C}, \vec{D}}) \left| \cos^n \hat{\alpha}_{\vec{C}, \vec{D}} \|\vec{r}\|^l \right| \quad (2)$$

$$Q_0 = \arg \max_{Q_i} T_{n,m,l,\hat{\alpha}_0}(P_0, Q_i). \quad (3)$$

¹ **Notation:** In the current work $P_i, Q_i \in \mathbb{R}^3$ represent points in 3D space, i.e. they are three dimensional vectors. For simplification of the notation we omit the arrows on top of the symbols. Arrows are only used for symbols denoting a direction in 3D space (for example axial directions, or displacement vectors).

Q_i are all trace points in the set of transversal planes, $\vec{r}_{P_0 Q_i} = Q_i - P_0$ is the displacement vector of the point Q_i to the current graph node P_0 , $\vec{A}_{Q_i}$, and $\vec{A}_{P_0}$ are their associated axes, and $\hat{\alpha}_{\vec{C}, \vec{D}}$ is the angle between two vectors $\vec{C}$ and $\vec{D}$. In Equation 1 the first term allows deviations of axes associated to potential “successor” graph nodes from the initial axial direction to lie only inside a cone ($\alpha_0 \in (0, \frac{\pi}{2})$), whereas the second term prohibits backward connections. The cosine in the third term implements condition C2, since it rewards smaller angles, and the length of the axial direction raised to the power of m ($m \geq 1$) implements condition C1 by amplifying the influence of longer axes. In Equation 2 the third term amplifies through n ($n \geq 1$) the influence of smaller angles between the current node’s axial direction $\vec{A}_{P_0}$ and the overall change of direction $\vec{r}_{P_0 Q_i} + \vec{A}_{Q_i}$, implementing thus condition C3. The last term amplifies or lowers (for $l \geq 1$ resp. for $l \leq -1$) the influence of distance between two skeleton points and therefore implements condition C4.

3. The successor point Q_0 from eq. 3 and its axial direction $\vec{A}_{Q_0}$ will become the current graph node and its ax of the next iteration.
4. Mark all points which belong to the transversal planes lying between Q_0 and P_0 and register $\vec{Q_0 P_0}$ as their “parent” segment, to avoid the multiple processing of a branch segment and the construction of spurious parallel skeleton lines (see Figure 4).

Figure4. *Plane Clustering:* Trace points (empty circles) lying in the transversal planes between two graph nodes P_0 and Q_0 (filled circles) are marked as “children” of the two nodes.

5. Repeat steps 1 to 4 until:
 - (a) All axial directions associated with the potential successor graph nodes lie outside the cone specified by the angle α_0 relatively to the current axial direction.
 - (b) All potential successor graph nodes Q_i have already been marked.
 - (c) No potential successor graph node is found in the current neighborhood.

The algorithm’s behavior can be adapted to the data at hand by changing the function Ω and its parameters. For example, the choice of farther or nearer skeleton points specified in condition C4 is implemented by changing the parameter l of Ω . Negative values of l imply the preference of nearer skeleton points, whereas positive values of l imply the preference of farther skeleton points. Similarly $n = 3$ in Equation 2 imparts the highest influence to the overall change of direction ($\vec{r}_{P_0 Q_i} + \vec{A}_{Q_i}$). Here we used $\alpha_0 = \frac{\pi}{5}$, such that the cone’s angle is less than $\pi/2$, $n = 3$, which confers more importance to the deviation between current axial direction and the overall direction change more than to the new axial length, which is stressed by $m = 2$. Finally $l = -1$ makes the validation function to prefer nearer neighbor points. The parameters’ influence on the behavior of the validation function is summarized in Table 1 below.

$\alpha_0 = (0, \frac{\pi}{2})$	$m = [1, \infty)$	$n = [1, \infty)$	$l \in (-\infty, -1] \cup [1, \infty)$	
			$l \in (-\infty, -1]$	$l \in (1, \infty]$
Increasing allowed deviations between the two node axes.	Increasing confidence into the new axial direction.	Decreasing allowed deviations between the current axial direction and the new search direction.	$l = -\infty$ - maximum preference of nearer located successors, which is decreasing with lower absolute values of l .	Increasing preference of farther located successors.

Table1. The Parameters’ Influence on the Behavior of the Validation Function.

The stopping condition 5a is fulfilled at varicosities, sharp bends, and branch points and avoids the generation of false skeleton lines, due to tracing in wrong axial directions (see Section 3.3). The stopping condition 5b is

typically fulfilled when an other skeleton branch is met. This case is discussed in detail in Section 3.4. Condition 5c appears due to missing trace points in edge gaps (at varicosities or regions of low image contrast) and leads to a premature end of the skeleton line. A solution to this problem is given in Section 3.5.

3.3 Critical Regions

Critical regions are regions of trace points which have high variance of their associated axial directions (as shown in [7]). Typically these kind of regions are found at branch points, sharp bends, and varicosities, i.e. wherever axial directions are unreliable or are changing abruptly (see Figure 5). The axial information in these regions should be ignored, since at varicosities it may lead to the construction of false skeleton lines. The above skeleton line construction algorithm has thus to be modified in the following way:

1. If the transversal planes constructed in the last step have passed (this happens if by chance a medial trace point inside the critical region has an axial direction which follows the previous axial direction) or has reached critical regions, then the best suited critical region's center Q_c is chosen by the maximum of C ,

$$C_{n,l,\hat{\beta}}(P_0, Q_i) = \Omega_{n,l,\hat{\beta}}(\vec{A}_{P_0}, \vec{r}_{P_0 Q_i}, \vec{r}_{P_0 Q_i}) \quad (4)$$

w.r.t Q_i , where the Q_i 's are here the centers of mass of each passed critical region. This leads to the choice of a critical region center which lies in the direction of the current node's P_0 associated ax $\vec{A}_{P_0}$ and is located at the preferred distance of the algorithm.

Since we want to bridge critical regions, the successor point Q_c is chosen to be the farthest trace point located in axial direction inside the chosen region (cf. Eq. 4).

2. A common fitness measure $\tilde{T}$ is evaluated for the previously found potential successor graph nodes Q_0 and the critical region point Q_c in order to choose the one which is better suited to the adopted point choice strategy of the algorithm (i.e. lies nearer to the axial direction of P_0 and is located at a preferred distance from it):

$$\tilde{T}_{n,l}(P_0, Q_0) = \Omega_{n,l,\pi}(\vec{A}_{P_0}, \vec{r}_{P_0 Q_0}, \vec{r}_{P_0 Q_0}), \quad Q_0 = \arg \max_{Q_i} T_{n,m,l,\hat{\alpha}_0}(P_0, Q_i), \quad (5)$$

$$\tilde{T}_{n,l}(P_0, Q_c) = \Omega_{n,l,\pi}(\vec{A}_{P_0}, \vec{r}_{P_0 Q_c}, \vec{r}_{P_0 Q_c}), \quad (6)$$

where T is taken from eq. 1.

In Equation 4 an other realization of the function Ω implements the selection of the critical region center. As for Eq. 2, $\hat{\beta} = \frac{\pi}{3}$ determines the maximal allowed direction deviation from the axial direction of the current skeleton node. This cone is wider than the one taken for the search of successor nodes, since in critical regions direction deviations are larger. As in Eq. 2 and Table 1 $n = 3$ imparts high influence to long axial vectors, and $l = -1$ chooses nearer located successor points. In Equations 5, and 6 the cone angle does not matter (and therefore it is set to π), since the points Q_0 and Q_c were chosen previously (Equations 3 and 4) to lie in a narrow cone. At varicosities, where axial information is disturbed, the algorithm will try to follow the direction it had before entering the region and will eventually find the continuation of the branch. At branch points or sharp bends where the high change of axial directions is correct, no further trace points will be found in the current branch direction and the line construction algorithm will terminate. The continuation will be found by a next skeleton line construction which starts at a point inside the other branch or branch half. This construction strategy is more reliable than unconditioned tracing of sharp directional changes.

Figure 5. Axial directions (arrows) of selected raw skeleton points (dots) in critical regions a) a varicosity; b) a sharp bend; c) a branch point.

3.4 Connection of Two Graph Segments

If marked points are reached during the line construction algorithm (i.e. the stopping condition 5b from Section 3.2 is encountered), then two skeleton lines will merge.

The newly constructed graph line has to be connected with an existing nearby graph segment. Each of the marked points which belong to the farthest located transversal plane considered for the line construction is associated to (is the “child” of) a graph segment (as described in Section 3.2, step 4). Among these segments, the best suited segment $\vec{S}$ (i.e. for which: i) the distance from the last added graph node P_0 to it is small, and ii) the angle between the projecting line onto the segment $\vec{B}$ and the point’s ax $\vec{A}_{P_0}$ is small), to which the new graph line can connect, has to be found.

Therefore, the nearest available skeleton line segment is found by maximizing the validation function M ,

$$M_{n,l}(P_0, \vec{S}) = \Omega_{n,l,\frac{\pi}{2}}(\vec{A}_{P_0}, \vec{B}, \vec{B}), \quad (7)$$

w.r.t all nearby segments $\vec{S}$. The parameters $n = 3$, and $l = -1$ are the same as in all previous equations, in order to assure at merge points the same behavior of the algorithm. The angle is allowed to be maximal (i.e. $\frac{\pi}{2}$), since directions are changing abruptly at branch points or bending points. Here $\vec{B}$ is the orthogonal projection vector of point P_0 onto a skeleton line segment $\vec{S}$, as it is shown in Figure 6.b.

Figure 6 shows the three situations which occur when merging:

- If the skeleton segment $\vec{S}$ is a line end, connect the new skeleton line to the skeleton line end (Figure 6.a). This situation appears at sharp bends.
- If the skeleton segment $\vec{S}$ is not an end of the respective skeleton line, then project P orthogonally onto $\vec{S}$. The corresponding projection point will be added as a node to the newly constructed graph line and to the segment of the already existing skeleton line, becoming thus a branch point (Figure 6.b).
- At connections to sharp bended skeleton lines, the projection onto all nearby segments lie outside the segments. Then the nearest vertex will be chosen as the connection point of the two merging skeleton lines (Figure 6.c).

Figure6. Possible connection configurations between the currently constructed graph segment and an already existing segment. a) Linear connection: if the nearest skeleton point is a line end, connect to this end; b) Project orthogonally to the nearest segment of the met skeleton line; c) At sharp bends connect to the nearest skeleton node if all projections onto segments lie outside the segments.

The condition of topology preservation is fulfilled by not allowing the algorithm to cross existing object boundaries when two lines are merging.

3.5 Skeleton Completion by Gray Value Tracing

Large edge gaps may arise around strong varicosities which cannot be filled due to the limited size of the “look ahead” neighborhood. Therefore we complete open ends of the skeleton lines in a second pass. Since no reliable

edge and medial trace point information is available at these points, alternative tracing methods have to be applied. Since varicosities have high gray values, skeleton lines can be completed by following voxels in the original gray value data which have higher gray values than the gray value of the considered skeleton line end.

The strategy is, first to perform an exhaustive search for potential skeleton completion points (successor graph nodes) and subsequently construct a smooth line through the obtained point cloud.

The Gray Value Completion Algorithm (Figure 7):

- *Sort existing skeleton lines* by number of nodes in order to complete first the lines with highest probability of correctness (i.e. lines with a high number of nodes had axial directions with low variance and therefore denote smooth continuous branches).
- *FOR EACH skeleton line end DO:*
 1. *Gather all Start/“Seed” Points:* Start/“Seed” points around a free skeleton line end are **medial trace points** which are the “children” of the current skeleton segment (cf. Section 3.2, step 4). We consider all points which are:
 - (a) the skeleton line end itself,
 - (b) in a plane, found by plane oscillation around the skeleton line end, as in Section 3.1,
 - (c) in several planes orthogonal to the starting ax found in the previous step and constructed as described in Section 3.2, and
 - (d) and additionally found by a seed fill [10] (seed points are those found by 1a,1b,1c) which gathers all connected **medial trace points** from a local neighborhood.
 2. *Construct the longest possible line:*
 - (a) Perform a seed fill (with seed points obtained in 1a,1b,1c, and 1d) to find all connected voxels which are **no trace points**, and which have higher gray values than g_0 :

$$g_0 = \max_{P_i} (g(P_i)) - \sigma_g, \quad (8)$$

where P_i are all start points, $g(P_i)$ their gray values, and σ_g the standard deviation of all $g(P_i)$'s.

- (b) Group all points (obtained from 1a,1b,1c, 1d, and 2a) into planes, which are orthogonal to the axial direction associated with the skeleton line end as described in Section 3.2, step 1 (Figure 7).
- (c) Construct the gray point line by choosing a point Q_i in each plane, for which G ,

$$G_{n,l}(P_0, Q_i) = \min \left(\text{sgn}(\cos \hat{\alpha}_{\vec{A}_{P_0}, \vec{r}_{P_0 Q_i}}), \text{sgn}(g(Q_i) - g_0) \right) \left| \frac{g(Q_i) - g_0}{255} \Omega_{n,l,\hat{\gamma}}(\vec{A}_{P_0}, \vec{r}_{P_0 Q_i}, \vec{r}_{P_0 Q_i}) \right|, \quad (9)$$

is maximal w.r.t. Q_i . P_0 is the skeleton line end point. Parameters are $n = 3, l = -1, \hat{\gamma} = \frac{\pi}{3}$. A narrower search cone is used here in order to keep smoothly the direction of the skeleton line end, whereas n and l have the same values as in the previous formulas, in order to ensure the same behavior of the algorithm.

END LOOP

Figure7. Gray Value Tracing: The skeleton end segment (thick continuous line with nodes as filled circles) is completed (thin dashed lines and dashed circles) after: i) gathering all potential successor points (empty circles denote trace points, crosses denote voxels with high gray values which are not trace points), and ii) clustering them into planes perpendicular to the skeleton end segment.

The steps of the gray value tracing algorithm are shown schematically in Figure 8.

During this chapter we have presented an algorithm which constructs a graph from medial trace points (Section 3.2). We have shown that noisy regions or regions of high gray value fluctuations can mislead the algorithm. After detecting these “critical” regions automatically [7] we apply at these locations a slightly modified paradigm (as described in Section 3.3), which is more noise robust. A post-processing of the yet available preliminary graph by tracing on original gray values (Section 3.5) is used to complete graph gaps at varicosities and in regions of low contrast of the foreground voxels.

Figure8. *Skeleton Completion by Gray Value Tracing.*

3.6 Computational Complexity

The different algorithmic steps presented in this chapter are implemented efficiently. The initial sorting of the medial trace points has the highest computational burden ($O(n \log n)$, with n being the number of medial trace points, i.e. which are much fewer than the total data size). The line construction algorithm has linear complexity, since it processes only once each medial trace point. The gray seed fill algorithm first sorts the graph lines by their length, which is again of complexity $O(m \log m)$, where m is the number of constructed skeleton lines, i.e. it is almost negligible relatively to initial data size. The seed fill algorithm is in its worst case of complexity $O(p)$, where p is the number of points visited by the algorithm. Since it is operating only on areas of limited spatial extent around the open ends of the skeleton lines, i.e. it is called $2m$ times, its computational burden is again negligible.

4 Results

In this section we will present an artificial dataset and two real datasets (Section 4.1). First we will show the medial trace points obtained from the preprocessing steps described in [7] in Section 4.2. Then we will compare the partial results obtained from the graph construction steps mostly on a zoom-in of the “ISO” neuron, since on this dataset the influence of different parameter choices can be shown the clearest. At steps where the other two datasets give good examples too we show them as well. Therefore we will exemplify following steps:

- The influence of the distance choice during line construction (from Section 3.2) on the resulting skeleton is shown in Section 4.3.
- The influence of critical region consideration during line construction (from Section 3.3) is commented in Section 4.4;
- The skeleton line completion by gray value tracing (from Section 3.5) is shown in Section 4.5;

4.1 Datasets

The datasets are one artificial dataset and two 3D-confocal microscope scans of locust neurons with different spatial and noise characteristics. They were kindly provided by Dr. D. Bucher (Institute for Zoology, University of Cologne, Germany).

Artificial Data: The dataset shown in Figure 9.a is an artificially drawn binary neuron ($64 \times 64 \times 64$ in size), which has branches with diameter between 1 and 16 voxels. Two benchmark datasets are created from it by adding noise of two different kinds to the artificial dataset. For the first benchmark (Figure 10.a) the binary dataset is blurred with a Gauss low-pass filter and Gaussian noise is added. The final image has high background noise (SNR 7.3) and blurring leads to a merge of the branches at the tip of the tree. For the second benchmark (Figure 10.b) the binary image is perturbed with Poisson noise, which creates higher gray value variations inside the neuronal branches, but less background noise and blurring. As can be seen, the small pin-like branch at the root of the tree has completely disappeared in the “Gaussian” benchmark, due to the blurring of the data, whereas in the “Poisson” benchmark the connecting neck of the pin is lost. Therefore, this branch cannot be recovered by the algorithm.

The “PLAN” Dataset: The first real dataset, which we will call “PLAN”, is a very clean high resolution scan (50x oil immersion lens) of a sensory neuron of the locust. The dataset is $512 \times 512 \times 130$ voxels in size, each voxel covering $0.38 \times 0.38 \times 0.76 \mu m$. The image (Figure 9.b) shows the mainly planar oriented structure of the neuron with branches having a diameter between 3 and 16 voxels. The dataset has many varicosities (i.e. black spots inside the branches), it shows high gray value fluctuations, and exhibits regions of very low contrast.

The “ISO” Dataset: is a low resolution scan (20x air lens), showing an A4I1 neuron from the Meso1 ganglion of the locust and is characterized by a higher level of noise. It is $512 \times 512 \times 135$ voxels in size, each voxel covering $0.98 \times 0.98 \times 0.98 \mu m$. Branch diameters range between 3 and 20 voxels in (x,y) directions, and between 6 and 40 voxels in z direction. The noise has locally the same gray values as the neuronal branches and is therefore very difficult to distinguish from the neuronal processes (Figure 9.c). In contrast to the planar oriented “PLAN” neuron, this neuron has spatially isotropic oriented branches.

Figure9. *The Data:* a) The binary artificial data ($64 \times 64 \times 64$ voxels, thinnest branch 1 voxel, thickest branch 16 voxel diameter); b) The “PLAN” Neuron ($512 \times 512 \times 130$ voxels in size, corresponding to $0.38 \times 0.38 \times 0.76 \mu m$ per voxel, thinnest branch 3 voxel, thickest branch 16 voxel diameter), with enhanced contrast (gray values above 100 are mapped to black); c) The “ISO” Neuron ($512 \times 512 \times 135$ voxels in size, corresponding to $0.98 \times 0.98 \times 0.98 \mu m$ per voxel, thinnest branch 3 voxels, thickest branch 40 voxels diameter) with enhanced contrast (gray values above 50 are mapped to black); The box marks a part of the neuron which will be investigated in more detail later on in this chapter; Arrows mark zones of low contrast. The images are maximum intensity projections.

4.2 Preprocessing

Edge detection and the across-scales edge validation are first applied on the two artificial benchmarks and on the two real datasets. Since edge validation enhances object boundaries and lowers edge values corresponding to noise the choice of the cut-off threshold is not critical any more and may be applied globally to obtain clean boundary edges [7]. For all three datasets presented here, the threshold was chosen visually. Thereafter we compute the medial trace points, their axes and directional variances, as described in [7]. The resulting medial trace points are shown in Figure 11.

Figure10. *Two Benchmarks Created from the Artificial Dataset:* We show the maximum intensity projections onto the Cartesian planes and slices 17 and 28 with enhanced contrast (maximal gray value 105 is mapped to white): in a) the artificial neuron was blurred with a Gaussian low-pass filter and perturbed with Gaussian noise (SNR is 7.3); the maximum intensity projection shows, that the branches at the tip of the tree merged due to the blurring; in b) the neuron was perturbed by Poisson noise; branches contain higher gray value variations but are less blurred than in a).

The medial trace points of the artificial datasets (Figure 11 a) and b)) are somewhat irregular, but otherwise reflect correctly the topology of the neuron. In the “Gaussian” benchmark (Figure 11.a) the medial trace points at the tip of the tree have merged too, since the corresponding branches have merged. The detected “critical” regions of the artificial dataset are shown in Figure 12. They correspond to the branch points of the neuron and are detected in both noisy benchmarks similarly. Therefore only one of them is shown. We omit to show the critical regions in the real datasets, since they are too complex and therefore difficult to inspect on 2D images.

We applied a cut-off threshold on the edges of the “PLAN” dataset, such that all edges corresponding to noise are eliminated. Therefore the medial trace points computed on the “PLAN” dataset are very clean but contain some gaps mainly in the critical regions (i.e. at low contrasted regions and at varicosities, where circularly shaped edges around the varicosities arise, which interrupt the linear course of the branches).

The “ISO” dataset has a much lower resolution in the z direction than in the x and y directions. The projection onto the y-z plane shows high variations in contrast along branches. Even though, our medial trace points have homogeneous thickness in all three spatial directions and contain relatively few gaps. This shows, that our algorithm is not sensitive to the resolution differences in the lateral and axial directions.

4.3 The Influence of Distance Choice

The density of graph nodes is influenced by two parameters of the graph construction algorithm of Section 3.2: i) the neighborhood parameter δ (Section 3.2 step 1), which determines the maximal lookahead distance for finding next potential successor graph nodes, and ii) the parameter l of the function Ω (Equation 2), which determines if larger or smaller distances between skeleton nodes are preferred. A larger lookahead distance δ combined with “farther” next point selection creates a sparser and more connected skeleton.

These parameters have to be set depending on the shape and size of the neuronal processes in the underlying data. For smooth neurons, with long linear branches, large δ ’s and “farther” point selection is suitable, whereas neurons with highly bended branches or high density of thin processes, small δ ’s and “nearest” point selection is recommended.

The influence of the different choices of nearer or farther point selection by the function Ω on constructed skeletons is shown on the “ISO” neuron in Figure 13. Here we show the generated skeletons overlaid on the volume rendered image of the “ISO” neuron. The graph generated with nearest point selection and small neighborhood has skeleton lines which are smoothly following the neuronal branches, but which have gaps at varicosities and low contrasted regions. The skeleton generated with farthest point selection and larger neighborhoods is more sparse (i.e. contains less graph nodes) and is more connected, but at edge gaps false connections between trace points lying in different neighboring branches may also arise.

Since the “PLAN” neuron is a smooth, not noisy dataset, distance choice does not influence significantly the shape of the skeletons. Therefore we do not show figures for the “PLAN” neuron for the comparison of different distance choice parameters.

4.4 Critical Regions

Since axial directions are noisy at critical regions (like varicosities, branches, or sharp bends), we modified in Section 3.3 the initial graph construction algorithm, such that it doesn’t consider the axial information of the medial trace points in these regions.

In Figure 14 we apply the modified algorithm on the “ISO” neuron. We show the generated graph overlaid onto the volume rendered original image. The balls denote branch points and critical points. It can be observed that some branches are traced more correctly (i.e. false connections at some edge gaps are corrected) and branch points are identified more exactly. An additional improvement brought by the consideration of critical regions is the bridging of some of the skeleton gaps which appear in real images around varicosities.

We show the improvement brought by the critical region consideration in the skeleton construction algorithm also on the “Gaussian” benchmark in Figure 15. First a skeleton with too low a neighborhood ($\delta = 3$), nearest point selection, and without critical region consideration is generated. The skeleton looks jagged and has gaps due to the small look-ahead distance (Figure 15.a). At branches the skeleton line connections are noisy and irregular (arrows point to these regions). The stars denote falsely connected branches which appear due to the merging of the branches (as shown in Figure 10.a). In Figure 15.b the skeleton is generated with nearest point selection and the same neighborhood ($\delta = 3$), but critical regions are taken into account during skeleton construction. It can be observed that clean merge points are generated at the branch points and that skeleton lines are therefore smoother.

In Figure 16, we applied the skeleton construction with critical region consideration on the “PLAN” dataset. We computed the skeleton with farthest point selection and a much too high neighborhood parameter $\delta = 15$. Observe that some completely false connections arise at edge gaps (arrows in Figure 16.a). We used the augmented algorithm which considers critical regions with farthest point selection and the same neighborhood. Figure 16.b shows a clean graph, which still contains gaps (at low contrast regions and at varicosities), but for which skeleton lines are following closely the neuron’s branches, showing that this algorithm is much more insensitive to non optimal parameter settings.

4.5 Skeleton Completion by Gray Value Tracing

The gray value tracing algorithm is needed to fill in the gaps of the skeleton lines left by the simple graph-construction algorithm around critical regions, due to the limited size of the “lookahead” neighborhood.

Since the skeleton shown in Figure 15.a) contained gaps, we apply the gray value tracing on the artificial benchmark (Figure 17). Skeletons are complete and do approximate the topology of the neurons well. In the “Gaussian” benchmark four gaps are left by the algorithm due to its built-in condition which prohibits the creation of circularly closed skeleton lines.

The significantly improved skeletons of the “ISO” and “PLAN” neurons are shown in Figures 18, and 19 overlaid on the volume rendering of the original image. Many of the low contrasted regions of the neurons are traced, such that some skeleton gaps are closed or branches are continued until no more reliable gray value information is found. The skeletons are more complete and more connected. The points added by the gray value tracing algorithm are shown as balls.

5 Discussion

The methods presented in the current work and in [7] form a system which provides the complete path of solutions for segmentation, for the extraction of critical points, for tracing and for skeletonization. Previous work in the area (see Section 1) usually discusses only one of these steps. Therefore we will compare intermediate results of our algorithms with commonly used preprocessing methods (Section 5.1), and our skeletonization with currently available skeletonization methods (Section 5.2).

5.1 Premises for Correct Skeleton Construction:

Most of the previously described methods for skeleton construction (distance maps [8,9], thinning [15,14,19], surface shrinking [20], clustering [27]) require the segmentation and binarization of the data. Segmentation is usually performed by applying a global threshold, which is prone to noise (for low thresholds), or which results

in the loss of foreground voxels (for high thresholds), in particular, if there is a wide range of contrasts in the data. Our approach on the other hand [7] relies on edge information which is obtained by a multiscale wavelet transform followed by an across-scales boundary edge validation. This segmentation procedure is more robust against noise and creates better conditions for raw skeleton point extraction.

Finally, this section discusses the relation between our raw skeleton points construction algorithm and a 2D analytical method [24].

Thinned Medial Points vs. Trace Points: Skeleton construction by thinning is based on the repeated deletion of foreground voxels of the object’s boundary, as long as their deletion does not disrupt the topological structure (i.e. by disconnection of parts or change of the basic shape of the object). [16] provided a set of rules for distinguishing simple points (i.e. the points which can be deleted safely) from safe points (i.e. points for which deletion would cause topology disruption). The extension of this so-called SPTA algorithm to three dimensions, however, is not straight forward due to the exponential explosion of possible neighborhood configurations (3^{26}) and the lack of a unique theoretical definition of minimal skeletons for some of the elementary 3D shapes [15]. The ESPTA algorithm of [15] applies the SPTA algorithm consecutively on the Cartesian planes of a $3 \times 3 \times 3$ cube. The authors tested the algorithm on objects which contained only boundaries parallel to the coordinate axes and showed, that box-shaped objects are reduced to thin planes, which cannot be reduced further due to the safe point conditions.

We have tested the ESPTA algorithm on an artificial neuron and noticed, that the algorithm creates holes in oblique object parts (figure not shown).

Applied on real data (like the “ISO” neuron, binary segmented with a threshold of 14 from 255 gray values), the ESPTA algorithm is misled by the inherent (oblique) irregularities of the neuronal boundary, which cause topologically wrong voxel deletions (the z direction has very irregular structure, Figure 20.b). These thinning results are not suited for further processing, since they have significant topology disruptions.

In contrast, our medial trace points have homogeneous thickness, are thinner, and preserve the neuron’s topology (Figure 20.c). The trace point gaps arise due to the varicosities inside the branches. Nevertheless, together with the associated axial directions and the variance values, they represent richer and less noisy initial information for skeleton construction than the thinned medial points can give.

Comparison with Analytic Medial Line Construction: The algorithm presented in [24] constructs the skeleton lines of analytically expressed curves in 2D space. There, the medial trace points are the center points of the smallest circle which is tangent to two opposite boundary points, i.e. they are the intersection of the curve’s normals equally distanced from it. The method is applied on topologically complex, but smooth and unnoisy artificial curves. For real, noisy, pixelized data, first a curve fitting and an eventual smoothing would have to be performed, in order to apply the method.

We calculate medial trace points from gray value gradients of boundary points. Since gradients are orthogonal to the object boundary, our algorithm can be considered as a computationally cheap 3D approximation method of the 2D algorithm [24], which does not need a parametric surface fitting of the pixelized data and therefore avoids its implicit errors. The consideration of mean quantities instead of exact values for the calculation of gradient intersection points makes our algorithm more robust against boundary irregularities.

5.2 Comparison between Skeletonization Algorithms:

There are two main classes of skeletonization algorithms: i) algorithms which use distance transforms [8,9,27], [20] and ii) algorithms which use thinned data [5,26] for the construction of skeleton lines. We have shown above, that thinning results are completely unreliable if the original data has irregular object boundaries. Therefore we will comment below on algorithms based on distance maps.

The Surface Shrinking Algorithm [20]: In the following we will show, that algorithms based on distance transforms are - like thinning algorithms - also sensitive to boundary irregularities and prone to noise.

The 3D surface shrinking algorithm [20] is a good prototype example, since it first has to segment the data - as all other algorithms based on distance transforms - to approximate a surface and then it performs shrinkage steps in the gradient direction of the distance map [8,9], i.e. orthogonally to the surface. This resumes in finding the points which lie on the symmetry ax of the surface, and therefore the shrinking algorithm can be regarded as well as a 3D extension of the 2D method [24] (see Section 5.1). Since surface shrinking unifies several ideas and gives very good results for sharply defined objects, we have chosen to analyze its behavior also on noisy data.

We apply the surface shrinking algorithm on the two real datasets presented in Section 4.1.

The “PLAN” Neuron: Since the “PLAN” neuron is an almost noiseless scan, a very low global segmentation threshold (i.e. 5 of 255 total gray values) is chosen, such that the generated iso-surface does not contain gaps. The skeleton obtained by the surface shrinking method is highly connected but has problems at branch points, where the surface cannot be shrunk to a single connection point. At varicosities, due to noisy gradients, stellate structures arise, which disturb the topology of the neuron (Figure 21.b).

Our algorithm creates a skeleton which reflects the overall structure of the neuron, but it encounters problems at difficult points like varicosities or low contrast regions too, leaving sometimes disconnected skeleton lines (Figure 21.c). We consider, that these gap artifacts are no impediment for further automatic analysis of the neuronal data, since a subsequent surface reconstruction (relying on the given skeleton and original gray value data) [21] would be able to bridge these areas. In contrast, the stellate surface artifacts produced by the surface shrinking method could mislead a surface reconstruction algorithm.

The “ISO” Neuron: The iso-surface for the “ISO” neuron is generated at threshold 13 from 255 gray values. A lower threshold would include too much noise, such that the neuron’s surface would be completely disrupted. Unfortunately this threshold is too high for the segmentation of the finest branches, thus providing incomplete initial object data to the surface shrinking algorithm. We emphasize again, that all other algorithms which rely on threshold segmentation of the data would suffer the same disruptions.

In contrast to the “PLAN” neuron, where the surface shrinking method gave good results, here the surface shrinking method fails (Figure 22.c), since thick branches could not be shrunk to a single line, due to their boundary irregularities (a common problem of all algorithms using distance transforms).

Due to its lower sensitivity to contrast variations and boundary irregularities, our algorithm is able to find low contrasted branches, reflecting thus the underlying data (see Figure 22.d). The thick branches as well as the thin branches are single lined without any spurious branches.

6 Conclusions

Throughout this paper we have shown that the multiscale edge detection algorithm presented in [7] is effective in object boundary enhancement, such that the crucial segmentation step needed by other algorithms becomes rather uncritical. This permitted a topology preserving medial trace point generation along with their associated axial directions and directional variance estimate, which provide richer information than other algorithms do have, such that the necessary premises for a correct graph construction are given.

Even though our preprocessing creates better starting conditions for skeletonization, the medial trace points still contain difficult or faulty areas. Therefore the skeletonization algorithm tries to avoid the inherent problems of the underlying data, by:

- *Adapting the generated skeleton line lengths* to the local noise level and smoothness of the data.
- *Adapting to the local contrast*, such that even the finest branches are detected.
- *Global generation of skeleton lines.* Since all detected medial trace points are considered for skeleton line construction, the algorithm cannot get stuck in local difficult regions, losing the data behind those regions.
- *Natural branch point detection* by the merging of meeting skeleton lines.
- *Avoiding the generation of spurious/multiple skeleton lines*, by clustering of “related” medial trace points.
- *Being computationally cheap.* The highest complexity of the algorithms is given by the sorting algorithm, which is typically $O(n \log n)$, where n is the number of medial trace points in the dataset (i.e. it is much sparser than the original gray value data). The skeleton construction algorithm depends linearly on the number of medial trace points.

The algorithm achieves a single lined, sparse, and smooth skeleton, which is following even the finest and lowest contrasted neuronal processes despite the fact that structures come in widely different sizes and contrast levels, that noise is strong and that the staining is not homogeneous. The remaining small problems of the algorithm, like gaps or false branches, will be solved by a subsequent surface reconstruction algorithm [21] which relies on the skeleton and on original gray values. Thereafter neuronal data will be reduced to an accurate skeleton augmented with geometrical measurements, which represents a complete description of the neuron.

Acknowledgements

We are grateful to D. Bucher (Institute for Zoology, University of Cologne, Germany) for providing the datasets and to L. Mucha for the implementation of the 3D Thinning algorithm. This work was supported by BMBF (Grant No. 0310962) and a “Stipendium des Frauenbeirats und der Kommission für Forschung und wissenschaftlichen Nachwuchs der TU Berlin” to A. Dima.

References

1. H. Blum. A transformation for extracting new descriptors of shape. In *Models for the Perception of Speech and Visual Form*. MIT Press, 1967.
2. D. Bucher, M. Scholz, K. Obermayer, and H.-J. Pflüger. Correction methods for 3D reconstruction of multi-channel confocal images of neurons in wholemount preparations. In U. Eysel N. Elsner, editor, *From Molecular Neurobiology to Clinical Neuroscience (Proceedings of the 27th Göttingen Neurobiology Conference)*, volume II, page 907. Georg Thieme Verlag Stuttgart, New York, May 1999.
3. D. Bucher, M. Scholz, K. Obermayer, and H.-J. Pflüger. Correction methods for three-dimensional reconstructions from confocal images: I. tissue shrinking and axial scaling. *Journal of Neuroscience Methods*, 100:135–143, 2000.
4. R.M. Cesar Jr. and L. da F. Costa. Neural cell classification by wavelets and multiscale curvature. *Biological Cybernetics*, 79(4):347–360, 1998.
5. A. Cohen, B. Roysam, and J. Turner. Automated tracing and volume measurements of neurons from 3-D confocal fluorescence microscopy data. *Journal of Microscopy*, 173(2):103–114, February 1994.
6. E. de Schutter and J.M. Bower. An active membrane model of the cerebellar Purkinje cell. i. simulation of current clamps in slice. *J. Neurophysiol.*, 71:375–400, 1994.
7. Anca Dima, Michael Scholz, and Klaus Obermayer. Automatic segmentation and skeletonization of neurons from confocal microscope images based on the 3-D wavelet transform. Technical report, Technische Universität Berlin, 2001. <http://edocs.tu-berlin.de/einzeldokument/index.html>.
8. N. Gagvani and D. Silver. Parameter controlled skeletonization of three dimensional objects. Technical report caip-tr-216, Rutgers State University of New Jersey, June 1997.
9. Y. Ge and J. M. Fitzpatrick. On the generation of skeletons from discrete euclidian distance maps. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(11):1055–1066, November 1996.
10. Paul Heckbert. A seed fill algorithm. In Andrew Glassner, editor, *Graphics Gems*, pages 275–277 721–722. Academic Press, Boston, 1990.
11. S. Hell, G. Reiner, C. Cremer, and E. H. K. Stelzer. Aberrations in confocal fluorescence microscopy induced by mismatches in refractive index. *J. Microscopy*, 169:391–405, 1992.
12. A. Herzog, G. Krell, B. Michaelis, and W. Zuschratter. Tracking on tree-like structures in 3-D confocal images. In Abraham Katzir, editor, *Proceedings of Three Dimensional and Multidimensional Microscopy: Image Acquisition and Processing V*, volume 3261 of *SPIE*, pages 165–176, 1998.
13. P. R. Hiesinger, M. Scholz, I. A. Meinertzhagen, K.-F. Fischbach, and K. Obermayer. Visualization of synaptic markers in the drosophila optic neuropils using a new constrained deconvolution method. *The Journal of Comparative Neurology*, 429:277–288, 2001.
14. L. Latecki and C. M. Ma. An algorithm for a 3D simplicity test. *Computer Vision and Image Understanding*, 63(2):368–393, 1996.
15. J. Mukherjee, P.P. Das, and B.N. Chatterji. Thinning of 3-D images using the safe point thinning algorithm (SPTA). *Pattern Recognition Letters*, 10:167–173, 1989.
16. N. J. Naccache and R. Shinghal. Spta: A proposed algorithm for thinning binary patterns. *IEEE: Trans. Syst. Man Cybernet*, 14:409–418, 1984.
17. M. Näf, O. Kübler, R. Kikinis, M. E. Shenton, and G. Székely. Characterization and recognition of 3D organ shape in medical image analysis using skeletonization. In *IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, San Francisco, June 1996.
18. R. L. Ogniewicz. *Discrete Voronoi Skeletons*. Dissertation no. 9876, Swiss Federal Institute of Technology, Zurich, 1992.
19. V. Ranwez and P. Soille. Order independent homotopic thinning. In G. Bertrand, M. Couprie, and L. Perroton, editors, *Proc. of Discrete Geometry for Computer Imagery'99*, volume 1568 of *Lecture Notes in Computer Science*, pages 337–346. Springer Verlag, 1999.
20. H. Schirmacher, M. Zöckler, D. Stalling, and H.-C. Hege. Boundary surface-shrinking - a continuous approach to 3D center line extraction. In Hans-Peter Seidel Bernd Girod, Heinrich Niemann, editor, *Proceedings of IMDSP '98, Alpbach Austria*, pages 25–28, July 1998.
21. S. Schmitt, A. Dima, M. Scholz, and K. Obermayer. Automatic three dimensional reconstruction of neurons from confocal images. In Norbert Elsner and Georg W. Kreutzberg, editors, *The Neuroscience at the Turn of the Century (Proceedings of 28th Göttingen Neurobiology Conference)*, volume 2, page 1042. German Neuroscience Society, Georg Thieme Verlag, 2001.
22. M. Scholz, D. Bucher, H.-J. Pflüger, and K. Obermayer. Correction schemes for geometrically exact 3D reconstructions of neurons from confocal images. In *Soc. of Neuroscience Abstr.*, volume 24, page 1508, 1999.
23. M. Scholz, M. Stetter, and K. Obermayer. Deblurring of confocal microscope images: Tissue properties, optics and simulations. In *Soc. Neurosci. Abstr.* 24, volume 1, page 1060, 1998.
24. D. Shaked and A. M. Bruckstein. The curve axis. *Computer Vision and Image Understanding*, 63(2):367–379, 1996.
25. T.G. Smith and G.D. Lange. Biological cellular morphometry-fractal dimensions, lacunarity and multifractals. In G.A. Losa, D. Merlini, T.F. Nonnenmacher, and E.R. Weibel, editors, *Fractals in Biology and Medicine*, volume II, pages 30–49. Birkhäuser Verlag, 1998.
26. F. Xu, P.H. Lewis, J.E. Chad, and H.V. Wheal. Extracting generalised cylinder models of dendritic trees from 3-D image stacks. In Abraham Katzir, editor, *Proceedings of 3-D and Multidimensional Microscopy: Image Acquisition and Processing V*, volume 3261 of *SPIE: Biomedical Optics*, pages 149–158, 1998.

27. Y. Zhou and A. W. Toga. Efficient skeletonization of volumetric objects. *IEEE Transactions on Visualization and Computer Graphics*, 5(3):196–209, July–September 1999.

Anca Dima, was born in Bucharest, Romania in 1970; 1993 MSc. degree in computer science engineering, “Politehnica” University of Bucharest, Romania; 1993 to 1996 Software Systems Engineer at “Omnis Group” Ltd., Bucharest, Romania, and 1994 to 1996 Research Assistant, electrical engineering Department, “Politehnica” University of Bucharest, Romania; 1996 may to october Research Assistant at the Universität Trier, Germany; 1996 to 1997 DAAD scholarship for a research stay at the Technical University of Berlin, Germany; since 1997 Ph.D. studies in biological image processing at the Neural Information Processing Group, Computer Science Department, Technical University of Berlin, Germany. Her research interests are in signal processing, wavelet analysis, 3D image processing of microscope data, artificial neural networks.

Michael Scholz, was born in Berlin, Germany in 1963; 1989 Diploma Degree in electrical engineering at the Technical University of Berlin, Germany; 1989-1994 teaching assistant position at the Technical University of Berlin, Germany; 1995 Dr. Ing., Computer Science Department, Technical University of Berlin, Germany; since 1996 Research Assistant, Neuronal Information Processing Group, Department of Computer Science, Technical University of Berlin. Areas of research: Computational neuroscience (models of information processing in the visual system), applications in image processing (confocal microscopy, warping, neuronal reconstructions).

Klaus Obermayer, was born in Ludwigsburg, Germany in 1961; 1980 to 1987 member of the Studienstiftung des deutschen Volkes; 1987 Diploma degree in physics, University of Stuttgart, Germany; 1990 to 1992 research stay at the University of Illinois at Urbana-Champaign and the National Center for Supercomputing Applications; 1992 Dr. rer. nat., Department of Physics, Technical University of Munich, Germany; 1992 to 1993 research stay at the Rockefeller University, New York, and the Salk Institute for Biological Studies, La Jolla; 1994 to 1995 research stay at the Technische Fakultät, University of Bielefeld, Germany; since 1995 Associate Professor, Neural Information Processing Group, Department of Computer Science, Technical University of Berlin. Areas of research: Computational neuroscience (models of neural development, models of information processing in the visual system), theory of artificial neural networks (statistical data analysis), applications in image and signal processing (3D image processing in confocal microscopy, optical recording of brain activity, laser-based flow field estimation).

Figure11. The Medial Trace Points: a) The trace points of the “Gaussian” benchmark; due to the performed blurring the branches at the tip of the tree merge (see Figure 10.a) and therefore the medial trace points meet as well; b) The trace points of the “Poisson” benchmark; c) Volume rendering of the “PLAN” dataset (with enhanced contrast). Arrows point to critical regions; d) Projection view of the binarized medial points of the “PLAN” dataset. At critical regions gaps are present in the medial trace points (arrows); Noisy edges were left on purpose in the image. These generate noisy trace points; e) Volume rendering (with enhanced contrast) of the original gray values in the boxed region from Figure 9.c; f) The projection view of the binarized medial trace points computed in the region shown in Figure 9.c.

Figure12. *The Critical Regions* of the artificial dataset correspond to the branching points of the neuron and the endpoints of the branches. They are similar in both noisy benchmarks.

Figure13. *The Influence of Distance Choice on the “ISO” Neuron* a) Nearest point selection at neighborhood 3; b) Farthest point selection at neighborhood 5; the connectivity is improved (arrows), but false connections due to edge gaps may also arise (stars). The skeletons are overlaid on the volume rendering of the volume boxed in 9.c.

Figure14. *Critical Region Consideration on the “ISO” Neuron* a) Skeleton with neighborhood $\delta = 3$ and nearest point selection; b) The skeleton with $\delta = 3$, nearest point selection, and critical zone consideration; Balls denote critical points and branch points; some skeleton gaps are bridged and false connections are avoided (arrows). The skeletons are overlaid on the volume rendering of the volume boxed in 9.c.

Figure15. *Critical Region Consideration on the “Gaussian” benchmark:* a) The skeleton constructed with too low a neighborhood ($\delta = 3$), nearest point selection, and without critical region consideration. Gaps in the skeleton lines arise due to the small look-ahead distance. At branches the connections are noisy and irregular (arrows). The stars denote falsely connected skeleton lines which appear due to the merging of the branches in the noisy dataset (Figure 10.a); b) The skeleton constructed considering the critical regions, with nearest point selection and the same neighborhood. Clean merge points arise at the branch points and skeleton lines are smoother.

Figure16. *Critical Region Consideration on the “PLAN” Neuron:* Skeletons constructed with farthest point selection at a much too high neighborhood $\delta = 15$: a) With the simple algorithm. Faulty cross connections arise at edge gaps (arrows); b) with the algorithm considering critical regions. The skeleton lines are following closely the neuronal branches. The skeleton is overlaid on the volume rendered neuron image.

Figure17. *Gray Value Completed Skeleton Lines of the Artificial Benchmarks:* The upper two rows show the projections of the skeletons onto the Cartesian planes (x-z) and (y-z) overlaid with the projections of the maximal intensity values of the original data. The lowest row shows the 3D views of the two skeletons: a) on the “Gaussian” benchmark (neighborhood $\delta = 3$ as for Figure 15.b); gaps are closed, except at points where loops would have been created (see Section 4.5 for explanation). b) on the “Poisson” benchmark. Both skeletons were constructed with nearest point selection and critical region consideration.

Figure18. *Gray value Tracing on the "ISO" Neuron* a) Nearest point selection at neighborhood 3 with critical regions; b) Same as a) with gray value tracing; the balls (arrows indicate the zones) are the nodes added by gray value tracing. The skeletons are overlaid on the volume rendering of the volume boxed in 9.c.

Figure19. *Gray Value Tracing on the "PLAN" Neuron:* a) Farthest point selection at neighborhood 3 with critical regions; b) Same as a) with gray value tracing; the balls (arrows indicate the zones) are the nodes added by gray value tracing. The skeletons are overlaid on the volume rendering of the neuron's left half.

Figure 20. *Thinned Medial Points vs. Trace Points of the “ISO” Neuron* a) Original data: zoom in of the boxed region in Figure 9.c (gray values between 0 and 255); b) Points obtained by the ESPTA thinning procedure on binary segmented data with threshold 14; The branches are not thinned homogeneously in z direction due to boundary irregularities; Our medial trace points are more robust to boundary irregularities, have homogeneous width, are smoother and follow the center of the branches; the trace gaps arise at edge gaps around critical regions, but they can be filled due to the associated axes of the trace points (see Figure 18); All figures are maximum intensity projections.

Figure21. *Surface Shrinking vs. Our Skeletonization on the “PLAN” Neuron:* a) A volume rendering of the planar oriented “PLAN” neuron with black mapped to 150 (from 255 gray values); arrows point to low contrasted regions; b) The skeleton obtained by surface shrinking; branch points cannot be reduced to a single point and varicosities generate stellate structures (arrows) c) The skeleton obtained by axial tracing is less connected than b) but is single lined everywhere; gaps are present at the very low contrasted regions or at varicosities (arrows);

Figure22. *Surface Shrinking vs. Our Skeletonization on “ISO” Neuron* a) Maximum intensity projection (values above 50 are mapped to black); arrows point to very low contrasted regions; branches are spatially isotropic; b) Surface Shrinking on an iso-surface with a threshold of 13 (from 255 gray values); the branches in the low contrasted regions are completely lost (arrow) and thick branches could not be linearized due to boundary irregularities; c) Our skeletonization reproduces smoothly even the lowest contrasted branches; thick as well as thin branches are single lined without any spurious skeleton lines;