Blockwise Processing Applied to Brain Microvascular Network Study

Céline Fouard*, Grégoire Malandain, Steffen Prohaska, and Malte Westerhoff

***Abstract*—The study of cerebral microvascular networks requires high-resolution images. However, to obtain statistically relevant results, a large area of the brain (several square millimeters) must be analyzed. This leads us to consider huge images, too large to be loaded and processed at once in the memory of a standard computer. To consider a large area, a compact representation of the vessels is required. The medial axis is the preferred tool for this application. To extract it, a dedicated skeletonization algorithm is proposed. Numerous approaches already exist which focus on computational efficiency. However, they all implicitly assume that the image can be completely processed in the computer memory, which is not realistic with the large images considered here. We present in this paper a skeletonization algorithm that processes data locally (in subimages) while preserving global properties (i.e., homotopy). We then show some results obtained on a mosaic of three-dimensional images acquired by confocal microscopy.**

***Index Terms*—Chamfer map, digital topology, image mosaic, medial axis, skeleton, topological thinning.**

I. Introduction

THE STUDY of the brain microvascular network is crucial to understanding brain behavior. Indeed, microvascular blood flow affects not only macrocirculation [1], but also neuronal nutrition and development. A topological and morphometric study of the brain vascular network is thus necessary to develop models for a better understanding of functional imagery such as positron emission tomography (PET) or functional magnetic resonance imaging (fMRI). These imaging techniques, promising great progress in knowledge of brain cognitive function, are actually based on a relationship between microcirculation and neural activity [2]. Some authors showed that fMRI signal intensity strongly depends on microvascular density [3], [4]. To facilitate understanding of the underlying mechanisms, a quantification of microvascular features, as for example the number or the diameter of vessels, is needed. This could provide geometrical models to represent and/or simulate functional imaging modalities. The study of microvascularization can also characterize some brain tissues, and determine whether or not they are healthy [5], [6].

Manuscript received April 11, 2006; revised June 12, 2006. *Asterisk indicates corresponding author*.

*C. Fouard is with the French National Institute for Research in Computer Science and Control (INRIA), 06902 Sophia Antipolis, France, and with TGS Europe, 33708 Merignac Cedex, France, and also with INSERM U455, 31059 Toulouse Cedex, France (e-mail: celine.fouard@imag.fr).

G. Malandain is with French National Institute for Research in Computer Science and Control (INRIA), 06902 Sophia Antipolis, France.

S. Prohaska and M. Westerhoff are with ZIB, D-14195 Berlin-Dahlem, Germany.

Digital Object Identifier 10.1109/TMI.2006.880670

Our project aims to provide tools for anatomists and neuroanatomists to better study brain microvascular networks [7]. It turns out that extracting the vessel centerlines is an efficient method to achieve this goal. Indeed, centerlines are compact representations of data and allow one to compute vessel lengths and junctions. With an additional distance map, they also give vessel diameters and densities.

The above mentioned tools (centerline detection, distance map computation) have been widely studied in the literature. However, for this particular application, we have a practical problem, namely the size of the data to be processed, that prevents us to use existing methods, and that requires us to design dedicated methods.

Indeed, the study of a single microscopic image provides useful *qualitative* results that can hardly be extrapolated to the whole brain because of the small size of the imaged area. For statistically relevant *quantitative* results, a sufficiently large area (several square millimeters) of the brain has to be analyzed. Moreover, the small diameter of microvessels (about 3 μm) requires high resolution images. To deal with this issue, we subdivide the area to be studied creating an image mosaic: several images (to cover a large part of the brain) are acquired with a confocal microscope (for its high three-dimensional (3-D) resolution capacity). We obtain a large amount of data, up to 4 GB in size, which cannot be loaded and processed at once in the memory of a standard computer. Datasets have to be stored *out-of-core* and can only be partially loaded into main memory for processing.

External memory algorithms and data structures [8] aim at redesigning algorithms to run with minimal performance loss due to out-of-core data storage. The portions of an image loaded for processing will be called *blocks* in the following.

Some image processing algorithms might easily be applied block-wise without further difficulties (for example algebraic operations, morphological operators, filtering, etc.). This is not the case for skeletonization, because we must ensure that both *global* (i.e., homotopy) and *regional* (i.e., being located at the center of the global object) properties of a skeleton are preserved while applying *local* operators.

This paper is an extended version of one presented at a conference [9]. In Section II, we present our data, the imaging protocol, and the preprocessing steps that result in a binary image. Next, a distance map based skeletonization algorithm is described. Section IV presents how to efficiently compute vessel centerlines on image mosaics, which is the main methodological contribution of this paper. Section V presents samples of result obtained on synthetic data, as well as real data. Finally, we discuss the validity of our results.

0278-0062/$20.00 © 2006 IEEE

Authorized licensed use limited to: IEEE Xplore. Downloaded on February 20, 2009 at 11:36 from IEEE Xplore. Restrictions apply.

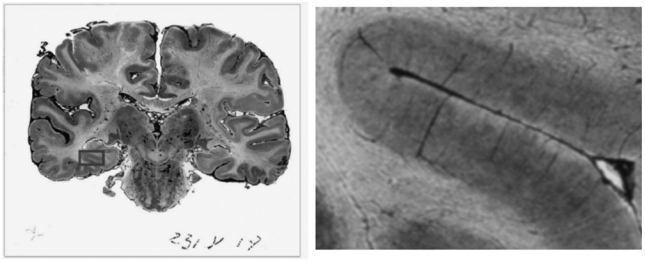


Fig. 1. Left: Brain section captured with an optical microscope (about 10×10 cm^2 wide). Right: Enlargement showing a sulcus (about 1×1 cm^2 wide).

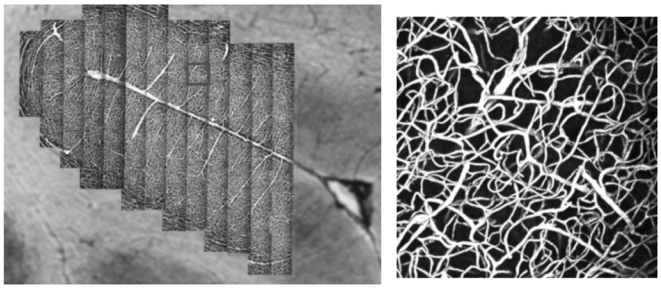


Fig. 2. Left: Image mosaic corresponding to the sulcus area of Fig. 1 (about $7 \times 7 \times 0.2$ mm^3 wide). Right: MIP view of one confocal image ($600 \times 600 \times 180$ μm^3 wide).

II. MicroVascular Data

The imaged material comes from Duvernoy's collection of samples [10]. Briefly, a human brain has been injected with Indian ink and then cut in thin sections to be observed with a traditional microscope (see Fig. 1). The visual (and tedious) inspection of several slices yields useful *qualitative* observations about the cerebral microvasculature, but the extraction of *quantitative* measures on a large area is unrealistic.

A. Confocal Microscope Observation

These sections can also be observed with a confocal microscope. The mean size of a confocal microscope image is about $600 \times 600 \times 180$ μm stored in a $512 \times 512 \times 128$ voxels image (see Fig. 2, right). Each voxel is stored using 8 bits so the image memory size is 32 Mb.

The resolution of such images is $1.2 \times 1.2 \times 1.4$ μm^3 and this allows us to study small veins and arteries, as well as the capillary bed (indeed, the smallest vessels have a diameter of about 3 μm, so the Nyquist criterion is satisfied). Quantitative parameters can be computed, but should carefully be extrapolated to larger portion of the section. Indeed, a large number of vessels are only partially imaged.

To image wider areas, an *image mosaic* has to be built (see Fig. 2, left). The section is located on a table which can be translated with a micrometric screw. Once an image is acquired, the section is translated, the acquisition of the next image is performed, and so on. The size of the area that can be so imaged is virtually unlimited. However, since the acquisition time is rather long (between 10 and 20 min per image), we limit ourselves to mosaics of about 100 images that represent several square millimeters of the section and are sufficient to cover a whole sulcus. Fig. 2 (left) shows a mosaic of 118 images covering about $7 \times 7 \times 0.2$ mm^3 of the brain section and corresponding to an image of approximately $6000 \times 6000 \times 128$ voxels and 4 GB.

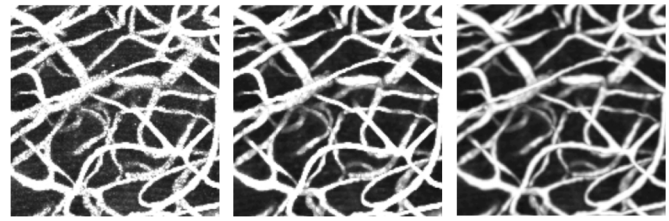


Fig. 3. MIP views of a part of a confocal image (left), after median filtering (center), and after Gaussian filtering (right).

To obtain a workable binary large image containing all the vessels in white and the background in black, we first perform some filtering on each small image to improve their quality (Section II-B). We then find the correct alignment between each image and build a single large image (Section II-C). We finally perform a segmentation to separate vessels from the background (Section II-D).

B. Filtering

We first perform a median filtering $(3 \times 3 \times 3$ kernel$)$ step to remove salt and pepper noise, then a Gaussian filtering ($3 \times 3 \times 3$ kernel, $\sigma = 1$ voxel) step to smooth borders (see Fig. 3). Filter parameters have been set empirically by experts.

C. Mosaic Creation

Building a mosaic from several images requires the knowledge of their relative position from each other. This position is given by a micrometric screw when displacing the imaged section under the microscope. However, some imprecision occurs between the real displacement and the value indicated by the screw, and moreover, the induced errors are cumulative. Thus, an automated in-plane repositioning is required.

To achieve a precise relative positioning of the mosaic images, we design the acquisition protocol so that there is an overlap of about 50 voxels along the overlap direction (i.e., $50 \times 512 \times 128$ voxels in total) between two adjacent mosaic images. This overlap allows to use standard registration methods [11].

The in-plane (along X and Y directions) translation is computed by optimizing a similarity measure on the overlapping area. In our case, a simple sum of squared differences (SSD) appears to be enough. We perform this step by considering two-dimensional (2-D) images, i.e., the maximum intensity projection (MIP) views of the images, for computational purposes. In addition to this in-plane translation (16 voxels in average), an in-depth (along Z direction) mislocation (three voxels in average) may occur when the imaged section is not perfectly orthogonal to the optical axis of the microscope. This is subsequently corrected by maximizing the SSD criterion on overlapping 3-D parts of the images. Decoupling both in-plane and in-depth corrections allows to reduce the time needed to compute them.

The mosaic of confocal image stacks is then merged into one large dataset comprising the whole examination volume. The

Authorized licensed use limited to: IEEE Xplore. Downloaded on February 20, 2009 at 11:36 from IEEE Xplore. Restrictions apply.

value of a voxel in the overlapping area is defined as a linear combination of the values of the corresponding voxels in the original images: weights depend on the distance to the image borders to ensure smooth transitions. We obtain a huge image, stored on hard disk, the value of each voxel being known and unique. This allows us to load any subvolume for inspection or for processing.

D. Segmentation

Finally, we perform a global segmentation with a user defined threshold value. Although not optimal, we chose this segmentation method because it is extremely fast and does not require any initialization or any modification to a mosaic. Moreover, the threshold can be efficiently chosen by considering the MIP view of the mosaic (see Fig. 2). Other better segmentation techniques exist, but are beyond the scope of this paper. As discussed in Section VI-B, the use of a single threshold in our confocal images is an acceptable solution.

III. Centerline Representation

We chose to represent blood vessels by their centerlines with local estimates of their radii. That way, the complete network can be efficiently visualized and processed, while quantitative information (vessel length, diameter, etc.) is still accessible. First, we present different centerline extraction methods, and then we present distance map calculation methods. By associating with each centerline point the smallest distance to the background, we obtain an estimate of the vessel radius at this point.

A. Centerlines

With respect to the original object, i.e., the thresholded image, we want the following properties to be verified.

Homotopy: the centerline set is topologically equivalent to the original image.

Thickness: the centerline set is thin i.e., one voxel wide. For discrete objects, however, it may happen that connectivity requires a two-voxel thickness at junction points.

Medialness: the centerline set is centrally located within the object. However, in case of an even number of voxels, a centerline one pixel wide cannot be exactly centered within the object.

Centerlines can first be obtained with methods derived from the *continuous* world, for instance the Voronoï diagram [12], or partial differential equations (PDE) [13]. However, such approaches cannot be easily adapted for an efficient computation in our context. We, thus, prefer to consider methods directly designed for discrete spaces. There exist methods that directly process the raw data and do not require any segmentation (e.g., [14]). However, our raw data are almost binary, so that the extra computational cost required by such methods is not justified. Hence, we mostly discuss approaches designed for binary objects in a discrete lattice.

On one hand, medialness is ensured by the extraction of the medial axis. This notion was introduced by Blum [15] with an analogy to grass fire: he defined the local axis as the locus of points at which the propagation fronts meet and extinguish each other. Calabi and Harnett [16] defined a medial axis as the set of centers of maximal disks of the object. However, the medial axis can be disconnected, and a postprocessing step is required to ensure that the homotopy property holds. In the same way, the medial axis can be defined as the locus of the ridges of the distance map [16]–[18]. The principle of these latter methods is to calculate the distance map of the object, to extract directional maxima, and to then reconnect these maxima.

On the other hand, homotopy is ensured by iterative thinning approaches or skeletonization. This notion was introduced by Hilditch [19], [20]. The skeleton is computed by iteratively peeling off the boundary of the object, layer-by-layer. Thinning methods are iterative and remove *deletable* points at each iteration, either sequentially [21], or in parallel [22]–[24]. In that case, removal strategies are designed to preserve at most the medialness property. For instance, a directional strategy [24] consists in dividing each iteration into subiterations, each subiteration reducing the thickness in one direction (e.g., top, down, north, south, east, west).

To preserve the axis from overshrinking, i.e., complete thinning (an object without hole nor cavity will be shrunk to a single point), a point is considered as deletable only if it is both *simple* and not an *end point*. A point is said to be simple if its deletion preserves the object topology; this is a local property [25] that only depends on the point's neighborhood. Conditions for *end points* are defined for points located on the border of a line or surface to keep it a line or a surface: e.g., for lines, end points are simple points that have only one neighbor [26].

Nevertheless, although preserved, medialness is not ensured. To this end, hybrid methods have been recently introduced to take advantage of both approaches [27], [28]. these are called distance ordered homotopic thinning (DOHT). They use a distance map to guide the process of iteratively removing simple points (homotopic thinning) towards the center of the object. They thus lead to a skeleton which is *homotopic* to the original object (only simple points are deleted), *thin* (points are deleted until no deletable point is found), and as centered as possible (point deletion follows the distance to the background). As discussed in Section IV, the adaptation of DOHT methods to our context allows us to control their computational cost.

B. Distance Map

DOHT algorithms require the computation of a distance map of the object to skeletonize. A distance map is a grey level image where the value of each object point corresponds to its shortest distance to the background. Numerous ways have been investigated to compute distance maps. A Euclidean distance map can be obtained through a particular PDE ($|\nabla f| = 1$), or by Euclidean distance mapping [29] where a vector is propagated, or by computing square distances [30]. Such methods cannot easily be adapted to our context (see Section IV), and, therefore, we consider approximations of the Euclidean distance.

Chamfer distance transforms, popularized by [31], achieve a good trade-off between precision and computational cost. They propagate local integer distances using chamfer masks. Briefly, a chamfer mask is a set of legal displacements weighted by local distances. Fig. 4 displays a $2-D$ 3×3 chamfer mask.

We exemplify the 2-D case in the following, but this algorithm can be easily adapted to higher dimensions. For a 2-D

Authorized licensed use limited to: IEEE Xplore. Downloaded on February 20, 2009 at 11:36 from IEEE Xplore. Restrictions apply.

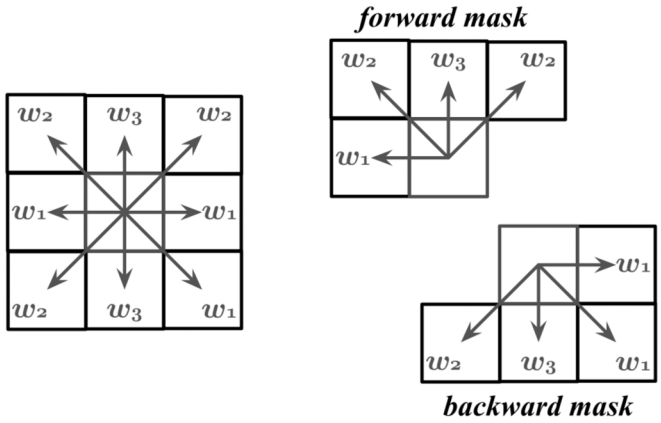


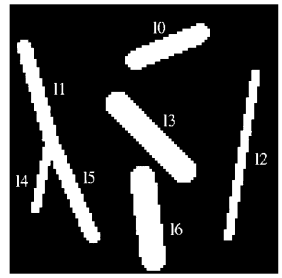
Fig. 4. Samples of 2-D 3 × 3 chamfer masks. Left: Global mask. Right: Two half-masks.

image, we use I_i, I_j to represent the image dimensions in the $\vec{x}$, $\vec{y}$ directions. Rosenfeld and Pfaltz [32] have shown that a distance map can be computed with two scans on the image: a so-called *forward* scan from the top left corner of the image [point with coordinates (1,1)] to the bottom right corner of the image [point with coordinates (I_i, I_j)], and a *backward* one, from the bottom right corner of the image [point (I_i, I_j)] to the top left corner [point (1,1)] of the image. To do so, the image is first initialized to 0 for the background and ∞ (practically, a very high value) for the object points. Then, they use half masks (cf. Fig. 4 right) and assign to each object point the minimum value between its previous value, and the value of the points located in its half neighborhood added with the corresponding weight of the half-mask. Thus, during the forward scan, the value of the current point depends on its neighbors located on its left and above (right and left) of this point. In the same way, during the backward scan, the value of the current point depends on its neighbors located on its right and under.

A distance between two points is generally defined as the length of the shortest path between these points. In the case of chamfer distance, we reduce the path choice to linear combinations of the legal displacements allowed by the mask. To obtain a chamfer distance as close as possible to the Euclidean one, one has to choose the mask coefficients leading to the smallest error with respect to the Euclidean distance. This computation is generally done on isotropic grids. In our case, however, slice thickness is larger than pixel size. Dealing with an isotropic grid would lead to having to interpolate data and to drastically increasing the amount of data. This is undesirable for the targeted application. In order to prevent such problem, we take into account the anisotropy of the lattice and consider the use of adapted coefficient computation methods [33].

IV. Blockwise Processing

The algorithms presented in the previous section implicitly assume that the image can be loaded and processed at once in the memory of the computer. Since this is not possible here, we have to process the image by subimages or blocks. Thus, this favors methods that can be more easily adapted to such a constraint



Line	Length	diameter
$l0$	26.93	6.21
$l1$	36.40	4.24
$l2$	60.83	2.46
$l3$	35.36	8.17
$l4$	25.50	2.46
$l5$	38.08	4.24
$l6$	30.27	8.16

Fig. 5. Synthetic image (left) and original dimensions (right).

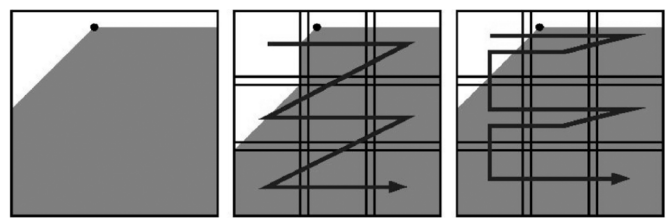


Fig. 6. Example of forward distance propagation. Black point represents a background point. Gray color represents the forward mask updated values. From left to right: propagation resulting from a 2-D *forward* scan; propagation resulting from a naive blockwise 2-D *forward* scan (each block is considered once); propagation resulting from an adequate blockwise 2-D *forward* scan (each block is considered twice, except the blocks at the end of block rows that are considered once).

(e.g., Chamfer distance map, see Section IV-B), and eliminate the ones that cannot (e.g., Voronoï diagram). Moreover, we are also concerned by the computational cost and want to restrict the number of sub-images to be loaded. Thinning or PDE methods have to be iterated until convergence and may not offer such a control. On the other hand, the DOHT can be adapted so that the number of subimages to be loaded is fixed (see Section IV-C). It is then the method of choice in this context.

A. Synthetic Images

To illustrate the problems raised by a block-wise process, we produced a synthetic image. Consider a $100 \times 100 \times 50$ voxel binary image. To simplify comparisons, we choose to create an isotropic image ($1 \times 1 \times 1$ voxel size). This image is small enough to be entirely processed at once, but we will also process it in four blocks to compare the result we obtain with the result obtained when processed as a single image.

We draw lines of known endpoints in this image with Bresenham's algorithm [34], and dilate the lines with spherical structuring elements of several known diameters.

Fig. 5 (left) shows the obtained image, while Fig. 5 (right) sums up the different lengths and diameters of lines. Notice that the diameters are not integers. This is due to the fact that the structuring elements we use are not real spheres, but combinations of discrete elements. This results in the fact that the shortest distance to the background is located at the corner of a voxel.

B. Blockwise Distance Map

If we compute distance maps independently on each image block, we obtain wrong values at block borders. Indeed, Fig. 6

Authorized licensed use limited to: IEEE Xplore. Downloaded on February 20, 2009 at 11:36 from IEEE Xplore. Restrictions apply.

(left) shows the result of the *forward* scan computed by processing the image as a whole and Fig. 6 (center) shows the result of the *forward* scan computed by sequentially considering the blocks of the mosaic.

Two reasons explain this behavior.

- First, local distances should be propagated from a block to another through chamfer masks. To do so, blocks must overlap each other. The overlap size depends on mask size. For example, for a $3 \times 3 \times 3$ chamfer mask, blocks must have at least one overlapping voxel in each direction (two voxels for a $5 \times 5 \times 5$ mask and so on). Fig. 6 (center) shows one computed with such an overlap.
- Second, to update a point value, for example in the forward scan, the process must know the values of its neighbors located on the left, above (on the left *and* on the right), and in the previous plane (above, under, on the left and on the right). In the case of a point located on the right border of the first block, points located on its right (above and on the previous plane) are not updated yet. A simple *forward* scan on each block from upper left to bottom right is thus not enough. This is depicted by Fig. 6 (center). It is the same for points located on the left border block for the backward scan. A simple *backward* scan on each block is not enough either.

To overcome these problems, we divide the image into overlapping blocks, and we perform several scans on rows of blocks, columns of blocks, and planes of blocks. Indeed, in the forward scan, once a row of blocks has been forward processed from left to right, we perform another forward process on this row of blocks, but from right to left [as depicted by Fig. 6 (right)], to update right border points of each block (except for the block located on the right border of the row which does not need to be updated). In the same way, once a plane of blocks has been processed from top to bottom, we perform the forward process from bottom to top on this plane of blocks to update points located on the bottom border of blocks (to update a point located on a bottom border of a block, values of points located under this point, in the previous plane, are needed and not yet updated). This way, we ensure that the result of our forward scan (with subimages) is strictly equivalent to a forward scan on the whole image, and we then have the same convergence properties as in Rosenfeld and Pfaltz's algorithm [32]. We perform for the backward scan in a similar fashion.

C. Blockwise Skeletonization

The blockwise skeletonization process also raises problems. Fig. 7 (left) shows the skeleton we would obtain by processing the whole image at once. Fig. 7 (right) shows the skeleton we obtain by independently processing each block. We can observe that disconnections appear at each block border.

Few authors propose tools to compute blockwise centerlines extraction. For example, Vossepoel *et al.* [35] process independently on partially overlapping blocks, and then they reconnect skeleton parts within the overlapping areas. However, to be able to know which point has to be reconnected with another, a previous labeling of each object to be skeletonized is required. In our application, labeling each vessel is simply not realistic. Pakura *et al.* [36] use mask driven skeletonization to determine nervous fibers on partially overlapping subblocks. Unfortunately, as skeleton location is not mandatory for their application, no precautions are taken to center skeletons within fibers located at block borders.

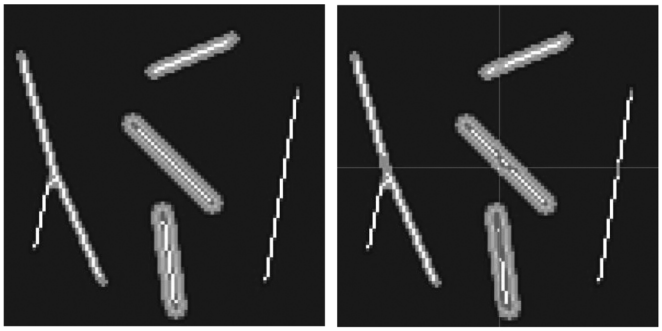


Fig. 7. If we process independently each block (right), disconnections appear on each block border with respect to the expect result (left).

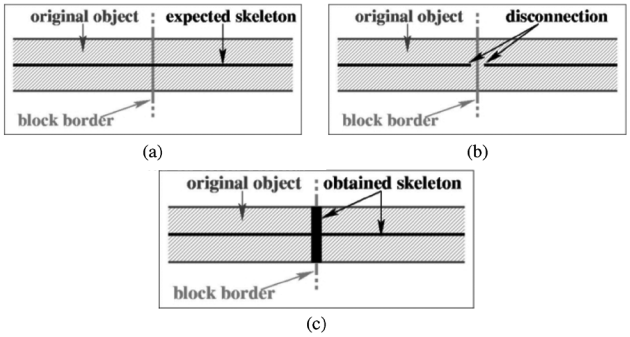


Fig. 8. Example of homotopy problems on block borders. Vessel crosses two blocks. If we process each block independently, its skeleton will be disconnected. (a) Expected skeleton and (b) independent computation of each block: the skeleton is disconnected on block borders. (c): Border points "freezing" guarantees the skeleton connectivity across borders.

We propose to adapt DOHT to a block-wise process. This adaptation is driven by the skeleton properties we want to preserve: homotopy, medialness, and thinness.

1) Homotopy: Homotopy can be guaranteed by deleting only simple points of the object. Problems of homotopy (vessel disconnections) may appear on block borders. Indeed, neighborhoods of points located on block borders are partially unknown. On one hand, if we assume that the unknown neighbors belong to the background, the object to be thinned can be disconnected from the border, and disconnections will appear on block borders. On the other hand, if we assume that these unknown neighbors belong to the foreground, the object to be thinned cannot be disconnected from the border, but the resulting thinned object (in the whole mosaic) can be disconnected as well since the continuity of the skeleton from block to block is not ensured.

Fig. 8(a) shows an example of a vessel located across two blocks and the skeleton expected for this vessel. Fig. 8(b) shows the skeleton obtained when the two blocks are processed independently.

To solve this problem, we *freeze* points located on block borders, i.e., we consider points as *deletable* only if the whole neighborhood is included within the block.

Authorized licensed use limited to: IEEE Xplore. Downloaded on February 20, 2009 at 11:36 from IEEE Xplore. Restrictions apply.

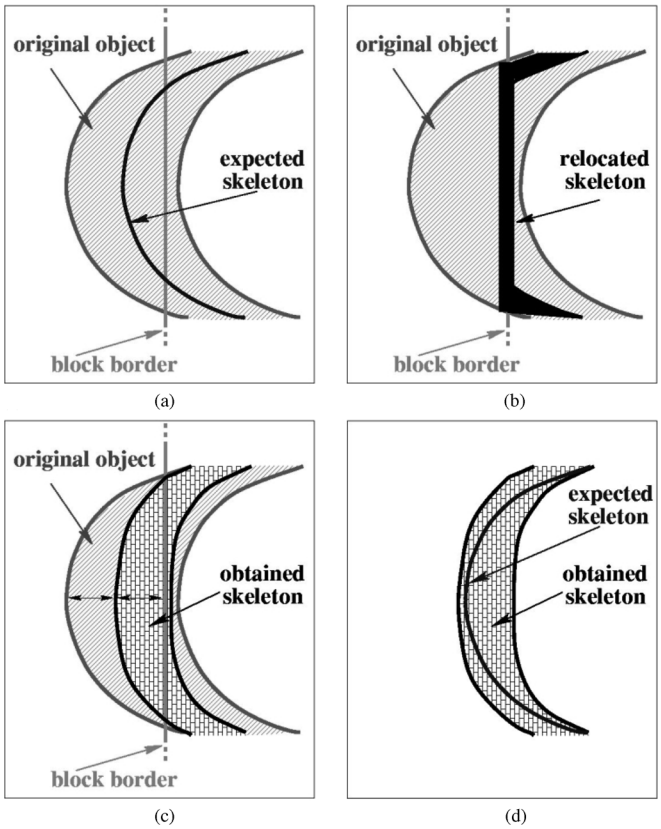


Fig. 9. Example of skeleton relocation problem. If we freeze only border points, the skeleton is "stuck" to the block border. (a) Expected skeleton and (b) relocated skeleton. (c) *Frozen* points and (d) expected skeleton inside *frozen* points.

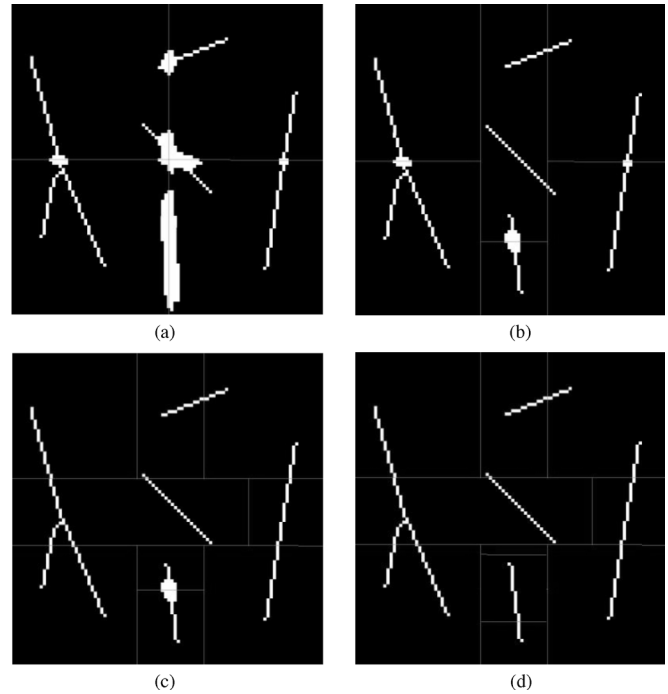


Fig. 10. Steps of Fig. 5(a) skeletonization using our method. In the first step of blockwise skeletonization, contiguous blocks are processed (border points are freezed according to previous conditions). Then border blocks are processed in x, y (and z) direction. In this synthetical example, we consider only one block in z direction. (a) Process on independent contiguous blocks and (b) process borders on vertical direction. (c) Process borders on horizontal direction and (d) result.

This condition guarantees a homotopic skeleton. Indeed, changes of homotopy (e.g., disconnections) appear when considering each block independently because some points located on block borders are not simple points in the whole mosaic, but appear to be simple with respect to one block as their neighborhood is not entirely known. Freezing border points ensures that each deleted point is simple with respect to the whole mosaic (because the neighborhood of each point *within one block* is entirely known). By definition of simple point, deleting simple points does not alter the object topology.

However, one may notice that thick portions of the object appear at the borders. They will be later deleted by further considering other blocks that straddle the borders.

2) Medialness: As opposed to properties that can be ensured locally, e.g., homotopy, medialness is a regional property which is more difficult to guarantee, and that is no more verified if we only freeze points on the one-voxel border as proposed above. Indeed, if we delete all simple points except for the border of a block, the skeleton may be mislocated.

As shown in Fig. 9(a) and (b), some points expected to be in the skeleton can be deleted. The connected component kept from the object becomes the "frozen" points of the border.

The skeleton is then "stuck" to the border of the first thinned block and not located at the object center. To overcome this problem, we consider points to be *deletable* only if their distance to the block border is larger than their distance to the object border [see Fig. 9(c)]. This means that a point can be deleted only if its associated maximal ball is entirely included within the block, or equivalently that points that do not verify this property have to be "frozen" to prevent them to be deleted. This last property locally adapts the shape of the "frozen" part of the object on borders to ensure that points of the expected skeleton will not be deleted at this phase. Fig. 9(d) shows the object component which is kept after this skeletonization phase. Fig. 10(a) shows the result of the skeletonization process applied to the synthetic lines (cf. Fig. 5), taking into account these two conditions. The expected skeleton is located within this component, but it is not really centered.

3) Thinness: The two previous conditions lead to a *homotopic* (because we remove only simple points) and potentially *medial* (because points located on maximal ball centers are not deleted) skeleton. But it may still be thick. Indeed, object areas located on block borders have not entirely been thinned. To obtain a thin skeleton, we reapply the skeletonization algorithm with the same conditions, on block border areas. To do so, blocks are redesigned as thick strips centered on previous block borders. The strip thickness depends on the largest distance found in the mosaic. As we are dealing with vessels, we are sure that the largest vessel diameter is far smaller than a block width. Fig. 10(b) and (c) shows different scans in y- and x-directions. For 3-D images, we handle the z-direction in the same way.

This blockwise DOHT methodology ensures that most of the mosaic will only be accessed once, while only a small part of the mosaic (the points inside the additional strip blocks on the

Authorized licensed use limited to: IEEE Xplore. Downloaded on February 20, 2009 at 11:36 from IEEE Xplore. Restrictions apply.

TABLE I
LENGTHS AND DIAMETERS OBTAINED WITH OUR METHOD FOR SYNTHETIC LINES (CF. FIG. 5)

Line	Length	error (%)	diameter	error (%)
$l0$	23.31	13.4	6.059	2.4
$l1$	38.73	6.4	4.195	1.1
$l2$	52.73	13.3	2.338	5.1
$l3$	29.70	16.0	7.700	5.7
$l4$	17.8	29.8	2.401	2.6
$l5$	35.80	5.9	4.167	1.8
$l6$	25.66	15.2	8.130	0.4

border) will be accessed several times. This way, the additional cost due to the blockwise processing is minimized.

Moreover, deletion of simple points is ordered by the original object distance map. The correct skeleton is located on this distance maps maxima, and the skeletonization algorithm deletes every point located around these maxima before reaching the expected skeleton points. This leads to a medial skeleton.

We thus obtain a skeleton which is *homotopic* to the original object, is *centered* and is as *thin* as possible with respect to the definition given in the previous section.

V. RESULTS

Once the distance map and skeleton are computed, we perform the following:

- convert the binary image obtained into a network of line segments by considering point connectivity.[1] This allows us to avoid storing a huge binary image, and only store connected lines with the coordinates of each of their points.
- attach to each line point the corresponding distance found in the distance map, which gives us the vessel radius on each point of its centerline (and allow us to compute its mean diameter).

In this way, as a line models a vessel, we have direct access each to vessel's number, length, and mean diameter.

A. Synthetic Data

Table I presents the lengths and mean diameters we obtained on synthetic data presented in Fig. 5.

Except for lines $l1$, $l4$ and $l5$, we can notice that lengths are always underestimated (by about 15%). This is due to the fact that during the skeletonization process, several points are deleted before a line end-point is recognized. In the case of lines $l1$, $l4$ and $l5$, values differ because the junction point between these three lines has been relocated when we performed a dilation on the Bresenham lines to give them a diameter. This, added to the relocation of line borders, gives higher errors.

Concerning diameters, they are systematically underestimated, too. But the error never exceeds 6% which corresponds to the error between the weighted and the Euclidean distance [33]. If the skeleton would have been misplaced or not correctly centered, this error would combine with the previous one and would then be higher. This agrees with the visual observation that the skeleton is correctly centered with respect to the original object. Moreover, the skeleton obtained with this method is exactly the same (same binary image) as the one obtained by processing the image as a whole (Fig. 7).

[1]End points have one neighbor, curve points separates the foreground into two components and have two neighbors, others points are junctions [26]. If the junction consists of several points, we use the barycenter of these points.

TABLE II
COMPUTATIONAL TIMES FOR THE SYNTHETIC IMAGE

Process on	image as a whole $100^2 \times 50$ voxels	4 sub-images 50^3 voxels
Independent contiguous blocks	/	0.11s
Borders on x direction	/	0.03s
Borders on y direction	/	0.04s
Total time	0.09s	0.18s

TABLE III
COMPUTATION TIMES OF THE BLOCK-WISE CALCULATIONS

Steps	Number of blocks	comput. time	comput. time / block
Distance map			
Initialization	121	2min23s	1.19s
Forward scan	341	5min13s	0.92s
Backward scan	242	5min03s	1.25s
Total	704	12min39s	1.08s
Skeletonization			
Contiguous blocks	288	1h00min26s	12.6s
Block borders (x direction)	132	6min16s	2.9s
Block borders (y direction)	264	12min07s	2.8s
Block borders (z direction)	144	13min37s	5.68s
Total	828	1h32min29s	6.7s

Table II shows the execution times of the different skeletonization steps for the synthetic image processed as a whole and in subimages. In this case, the slicing in subimages is performed within the XY plane (no subimage is created in the depth direction). However, it should be pointed out that in realistic situations, the border strips are very narrow with respect to the contiguous blocks.

B. Real Data

We integrated these algorithms into the visualization system Amira [37], [38]. The experiments were conducted on a Pentium 4 1.7 GHz laptop with 512 Mb RAM. The amount of data (4 GB for the original image, 4 GB for the segmented image, 8 GB for the distance map stored on short integers, as bytes are not enough to store the chamfer map, and 4 Gb for the skeleton creation image, i.e., 20 Gb for the whole process) leads us to store result images on a distant hard disk, which slowed down the image I/O process, and considerably increased the total computational time. I/O procedures were implemented using the HDF5 library [39].

Computing chamfer distances took 13 min of actual calculation, and 14 h 17 min of total process due to image I/O procedures. The skeletonization process took 1 h 32 min of actual calculation for a total duration of 8 h 8 min for the whole mosaic. During these two processing steps, the mosaic was cut into about $256 \times 256 \times 128$ voxel subimages (the number of voxels may slightly change considering block location, e.g., blocks located on the border of the mosaic, and overlap needed between blocks). Table III summarizes the computational times of the different process.

The whole process takes then several hours on a standard PC, due to the I/O procedures. This remains comparable to the total acquisition time of the mosaic. Contrary to the acquisition, this

Authorized licensed use limited to: IEEE Xplore. Downloaded on February 20, 2009 at 11:36 from IEEE Xplore. Restrictions apply.

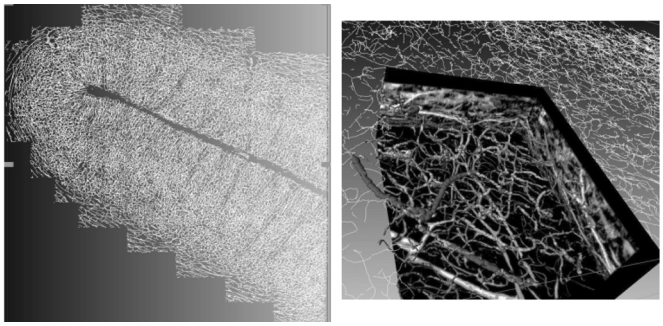


Fig. 11. Left: Vessel centerline network of the mosaic presented in Fig. 2. Right: Enlargement with vessels modelled as cylinders.

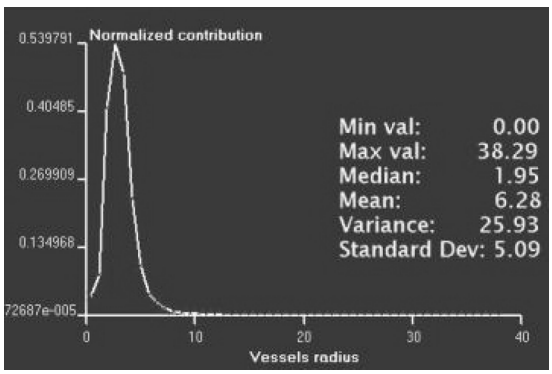


Fig. 12. Vessel diameters obtained from the mosaic of Fig. 2.

pipeline is fully automatic and does not require any human intervention after the choice of the threshold.

After converting the voxel representation of the skeleton to a geometric representation of lines (Fig. 11 shows vessel center lines obtained from the image shown in Fig. 2: the whole line-set contains more than 2 000 000 points and about 74 000 lines), the result can be interactively explored. Displaying the original image data together with the generated skeleton in one visualization is possible (see Fig. 11, right). This helps to verify results as well as to document them.

From these data, we can also extract statistical data, as for example the histogram of vessel diameters (see Fig. 12). We can notice that most vessel diameters are lower than 10 μm, with a peak at about 5 μm.

VI. Discussion

A. Validation

Validation is a critical issue in any medical imaging process. We performed a *qualitative* validation on real data and a *quantitative* validation on simulated images.

1) Qualitative Validation on Real Images: The first aim of our study was a proof of feasibility. We showed that extracting morphometric data on real data representing several square millimeters of the brain was feasible in reasonable time (at least in finite time). This study allowed us to recover several qualitative features identified by anatomists [10]:

- the shape of vessel junctions corresponds to their expectations: veins and arteries emerge from the cortex with right angles, and form sharp angles in deep layers;
- the distribution of vessel density corresponds to the observation on several cortex layers;
- the distribution of vessel radii also corresponds to their observation.

Unfortunately, this validation on real data cannot be quantitative as we are not aware of a ground truth set of morphometric values on such data.

2) Qualitative Validation on Synthetic Binary Image: We produced a synthetic binary image to test our block-wise processing methods. We showed that we obtained a skeleton which is homotopic, thin, and centered with respect to the binary object. Moreover, we showed that mean vessel diameters were estimated with an error smaller or equal to 6%. This validation on synthetic data did not take into account the segmentation step, which is beyond the scope of this paper.

B. Dependence on Segmentation

By construction, the obtained skeleton is topologically equivalent and centered with respect to the binary object. However, spurious branches may appear, that are not representative of the vessel network. These branches are bounded by an end point, that generally corresponds to an irregularity of the vessel to be thinned, and a bifurcation. When studying a vessel network, end points should not appear, except at image border, or for segmentation error (typically a vessel partially filled by indian ink), or for spurious branches. Characterizing the latter is facilitated by the distance information [17], [40], but requires an additional threshold, and may also remove branches corresponding to missing vessels. Thus, to allow the network correction (by manual editing), we keep all branches.

Nevertheless, by smoothing the original before thresholding it, we reduce the vessel irregularities, and consequently the number of spurious branches. Our experiments demonstrated that DOHT methods generally generated less of such branches than pure thinning methods (e.g., [24]), and that this number further decreases if a directional strategy (as in thinning method) is adopted, in addition to the distance ordering.

However, being topologically equivalent to the binary object leads the skeleton to highly depend on segmentation, and on the threshold choice. For example, if binarization creates a hole within a vessel, then the skeleton will get around this hole to be topologically equivalent to the object. Here again, the median and the Gaussian filtering help to avoid such problems.

Additionally, the measurements also depend on the threshold choice. Indeed, the higher the threshold value is, the less points are taken for each vessel, and the smallest their mean diameter is. Conversely, the lowest the threshold value is, the most points are taken for each vessel and the highest their mean diameter is.

By choosing different thresholds[2] on the same mosaic (40, 50, and 60), we obtain respectively as mean radius 3.31, 3.09, and 2.88, while the shape of the obtained histograms are identical, but shifted. This emphasizes that a particular attention must be paid for binarization. Ongoing work on a more accurate segmentation, better adapted to our specific data (e.g., histogram equalization, mathematical morphology, etc.), is currently conducted.

[2]Values of original images vary between 0 and 255. Inter and intra-operator variability for the choice of threshold does generally not exceed 5 or 6 value units.

Authorized licensed use limited to: IEEE Xplore. Downloaded on February 20, 2009 at 11:36 from IEEE Xplore. Restrictions apply.

We favor automatic segmentations techniques which limit inter and intraoperator variations.

However, it has to be pointed out that vessel shapes and diameters have been considerably altered by the preparation process. Indeed, vessels may have been inflated by Indian ink injection, distorted by the cutting process, and flattened between the slide and the cover-glass to be observed with a microscope. The imprecision brought by the choice of the binarization threshold is less important with respect to the deformations vessels were subjected to.

VII. Conclusion and Perspectives

The extraction of morphometric parameters from a mosaic of 3-D confocal microscopic images has been presented. This has required the design and development of dedicated software tools. Indeed, as huge images cannot be loaded at once in a standard computer memory, they need adapted algorithms. The proposed block-wise skeletonization method preserves global as well as local skeleton properties by avoiding border effects, and allows to control the number of subimages to be processed. In a first pass, we process 3-D blocks without overlapping. Next, we process subimages covering boundaries. The size of these subimages depends on the size of the object to be thinned. Doing so, inner block areas are processed only once. The algorithm overhead due to the block by block process appears only on boundaries. We have also shown that for our application, our tools allow to extract quantitative information precious for neuroanatomists and neurophysiologists to describe the brain microvascular network [7].

This application can be widened to other acquisition techniques (as long as it can lead to a binarized image with vessels outlined from the background), and to other research area (e.g., oil industry or plant roots study [41]). Indeed, the presented techniques can be applied to any binarized large data, as long as considered objects represent tubular structures or small shapes with respect to the size of a block to ensure that we can design strips at block borders for the skeletonization.

References

[1] D. Kleinfeld, "Cortical blood flow through individual capillaries in rat vibrissa S1 cortex: Stimulus induced changes in flow are comparable to the underlying fluctuations in flow," in *Proc. Brain Activation CBF Control*, Tomita, Ed., Jun. 2001.

[2] D. Heetger and D. Ress, "What does fMRI tell us about neuronal activity?," *Nature Rev. Neurosci.*, vol. 3, pp. 142–151, Feb. 2002.

[3] S. Lai, A. Hopkins, E. Haacke, D. Li, B. Wasserman, P. Buckley, L. Friedman, H. Meltzer, P. Hedera, and R. Friedland, "Identification of vascular structures as a major source of signal contrast in high resolution 2-D and 3-D functional activation imaging of the motor cortex at 1.5 T: Preliminary results," *Magn. Reson. Med.*, vol. 30, no. 3, pp. 387–392, September 1993.

[4] R. Turner, "How much cortex can a vein drain? Downstream dilution of activation-related cerebral blood oxygenation changes," *NeuroImage*, vol. 16, pp. 1062–1067, Oct. 2001.

[5] O. Craciunescu, S. Das, and D. M. K. , "Three-dimensional microvascular networks fractal structure: A potential for tissue characterization?," in *Adv. Heat Mass Transfer Biotechnol.,*. New York: ASME, 1999, vol. 363, pp. 9–13.

[6] E. Farkas and P. Luiten, "Cerebral microvascular pathology in aging and Alzheimer's disease," *Progress Neurobiol.*, vol. 64, pp. 571–611, 2001.

[7] F. Cassot, F. Lauwers, C. Fouard, S. Prohaska, and V. Lauwers-Cances, "A novel three-dimensional computer-assisted method for a quantitative study of microvascular networks of the human cerebral cortex," *Microcirculation*, vol. 13, no. 1, pp. 1–18, Jan.–Feb. 2006.

[8] J. S. Vitter, "External memory algorithms and data structures: Dealing with massive data," *ACM Comput. Surveys*, vol. 33, no. 2, pp. 209–271, 2001.

[9] C. Fouard, G. Malandain, S. Prohaska, M. Westerhoff, F. Cassot, C. Mazel, D. Asselot, and J.-P. Marc-Vergnes, "Skeletonization by blocks for large datasets: application to brain microcirculation," in *Proc. IEEE Int. Symp. Biomedical Imaging: From Nano to Macro (ISBI'04)*, Apr. 2004, vol. 1, pp. 89–92.

[10] H. Duvernoy, S. Selon, and J. Vannson, "Cortical blood vessels of the human brain," *Brain Res. Bull.*, vol. 7, no. 5, pp. 519–579, Nov. 1981.

[11] J. Maintz and M. Viergever, "A survey of medical image registration," *Med. Image Anal.*, vol. 2, no. 1, pp. 1–36, Mar. 1998.

[12] R. Ogniewicz, "Skeleton-space: A multiscale shape description combining region and boundary information," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, Jun. 1994, pp. 746–751.

[13] J. Gomes and O. Faugeras, Reconciling distance functions and level sets French National Institute for Research in Computer Science and Control (INRIA), Le Chesnay Cedex, France, Tech. Rep. RR-3666, 1999.

[14] K. Krissian, G. Malandain, N. Ayache, R. Vaillant, and Y. Trousset, "Model-based detection of tubular structures in 3-D images," *Comput. Vision Image Understand.*, vol. 80, no. 2, pp. 130–171, 2000.

[15] H. Blum, "A transformation for extracting new descriptors of shape," *Models Perception Speech Visual Form*, pp. 362–380, 1967.

[16] L. Calabi and W. Harnett, "Shape recognition, prairie fires, convex deficiences and skeletons," *Amer. Math. Monthly*, vol. 75, pp. 335–342, 1968.

[17] G. Malandain and S. Fernández-Vidal, "Euclidean skeletons," *Image Vision Comput.*, vol. 16, no. 5, pp. 317–327, Apr. 1998.

[18] Y. Zhou, A. Kaufman, and A. W. Toga, "3-D skeleton and centerline generation based on an approximate minimum distance field," *Int. J. Visual Comput.*, vol. 14, no. 7, pp. 303–314, 1998.

[19] C. Hilditch, *Machine Intelligence*. Edinburgh, U.K.: Edinburgh Univ. Press, 1969, vol. 4, ch. Linear skeletons from square cupboards, pp. 403–420.

[20] T. Kong and A. Rosenfeld, "Digital topology: Introduction and survey," *Comput. Vision Graph. Image Process.*, vol. 48, pp. 357–393, 1989.

[21] K. Palágyi, E. Sorantin, E. Balogh, and A. Kuba, "A sequential 3-D thinning algorithm and its medical applications," in *Proc. Inf. Process. Med. Imag. (IPMI 2001)*, Jun. 2001, vol. 2082, pp. 409–415.

[22] Y. Tsao and K. Fu, "A parallel thinning algorithm for 3-D pictures," *Comput. Graph. Image Process.*, vol. 17, pp. 315–331, 1981.

[23] C. Ma and S. Wan, "Parallel thinning algorithms on 3-D (18, 6) binary images," *Comput. Vision Image Understand.*, vol. 80, pp. 364–378, 2000.

[24] C. Lohou and G. Bertrand, "A new 3-D 6-subiteration thinning algorithm based on p-simple points," in *Discrete Geometry for Computer Imagery*. New York: Springer, 2002, vol. 2301, pp. 102–113.

[25] G. Bertrand and G. Malandain, "A new characterization of three-dimensional simple points," *Pattern Recognit. Lett.*, vol. 15, no. 2, pp. 169–175, Feb. 1994.

[26] G. Malandain, G. Bertrand, and N. Ayache, "Topological segmentation of discrete surfaces," *Int. J. Comput. Vision*, vol. 10, no. 2, pp. 183–197, 1993.

[27] T. Saito and J. Toriwaki, "A sequential thinning algorithm for three dimensional digital pictures using the Euclidean distance transformation," in *Proc. 9th Scandinavian Conf. Image Analysis (SCIA'95)*, 1995, pp. 507–516.

[28] C. Pudney, "Distance-ordered homotopic thinning: A skeletonization algorithm for 3-D digital images," *Comput. Vision Image Understand.*, vol. 72, no. 3, pp. 404–413, Dec. 1998.

[29] I. Ragnemalm, "The Euclidean distance transform in arbitrary dimensions," *Pattern Recognit. Lett.*, vol. 14, no. 11, pp. 883–888, November 1993.

[30] T. Saito and J. Toriwaki, "New algorithms for Euclidean distance transformation of an n-dimensional digitized picture with applications," *Pattern Recognit.*, vol. 27, no. 11, pp. 1551–1565, 1994.

[31] G. Borgefors, "Distance transformations in arbitrary dimensions," *Comput. Vision Graph. Image Process.*, vol. 27, pp. 321–345, Feb. 1984.

[32] A. Rosenfeld and J. Pfaltz, "Sequential operations in digital picture processing," *J. Assoc. Comput. Mach.*, vol. 13, no. 4, pp. 471–494, October 1966.

Authorized licensed use limited to: IEEE Xplore. Downloaded on February 20, 2009 at 11:36 from IEEE Xplore. Restrictions apply.

[33] C. Fouard and G. Malandain, "3-D chamfer distances and norms in anisotropic grids," *Image Vision Comput.*, vol. 28, no. 2, pp. 143–158, Feb. 2005.

[34] J. Bresenham, "Algorithm for computer control of digital plotter," *IBM Syst. J.*, vol. 4, no. 1, pp. 25–30, 1965.

[35] A. Vossepoel, K. Shutte, and C. Delanght, "Memory efficient skeletonization of utility maps," in *Proc. 4th Int. Conf. Document Anal. Recognit.*, Ulm, Germany, 1997, pp. 797–800.

[36] M. Pakura, O. Schmitt, and T. Aach, "Segmentation and analysis of nerve fibers in histologic sections of the cerebral human cortex," in *Proc. 5th IEEE Southwest Symp. Image Anal. Interpretation (SSIAI'02)*, 2002, pp. 62–66.

[37] Microvisu3d [Online]. Available: http://www.tgs.com/pro_div/solution/mv3d/mv3d.htm

[38] *"Amira 3.1-User's Guide and Reference Manual-Programmer's Guide,"* Zuse Institute Berlin (ZIB) and Indeed-Visual Concepts, Berlin, Germany, Oct. 2003 [Online]. Available: http://www.amira.zib.de

[39] S. Prohaska, A. Hutanu, R. Kähler, and H.-C. Hege, "Interactive exploration of large remote micro-CT scans," in *Proc. 15th IEEE Visual. 2004 Conf. (VIS 2004)*, 2004, pp. 345–352.

[40] D. Shaked and A. M. Brucktein, "Pruning medial axes," *Comput. Vision Image Understand.*, vol. 69, no. 2, pp. 156–169, Feb. 1998.

[41] P. Kolesik, C. Fouard, S. Prohaska, and A. McNeill, "Automated method for non-destructive 3-D visualisation of plant root architecture using X-ray tomography," in *Proc. 4th Int. Workshop Functional-Structural Plant Models*, Jun. 2004, p. 27.

Authorized licensed use limited to: IEEE Xplore. Downloaded on February 20, 2009 at 11:36 from IEEE Xplore. Restrictions apply.