Visualization of Cellular and Microvascular Relationships

David M. Mayerich, *Student Member, IEEE*, Louise Abbott, John Keyser, *Member, IEEE*

Abstract—Understanding the structure of microvasculature structures and their relationship to cells in biological tissue is an important and complex problem. Brain microvasculature in particular is known to play an important role in chronic diseases. However, these networks are only visible at the microscopic level and can span large volumes of tissue. Due to recent advances in microscopy, large volumes of data can be imaged at the resolution necessary to reconstruct these structures.
Due to the dense and complex nature of microscopy data sets, it is important to limit the amount of information displayed. In this paper, we describe methods for encoding the unique structure of microvascular data, allowing researchers to selectively explore microvascular anatomy. We also identify the queries most useful to researchers studying microvascular and cellular relationships. By associating cellular structures with our microvascular framework, we allow researchers to explore interesting anatomical relationships in dense and complex data sets.

Index Terms—microscopy, biomedical, medical, blood vessels, cells.

1 Introduction

The vertebrate vascular system allows transport of oxygen and other small molecules to cells throughout an organism. Several techniques exist for constructing volumetric images of the vascular system at the macroscopic scale. Magnetic Resonance Imaging (MRI) and Computed Tomography (CT) are often used to image vasculature in living tissue. However, these imaging modalities are incapable of resolving *microvasculature*. Microvasculature is the network of capillaries that complete the loop between arteries and veins. In contrast to the tree-like topology of larger-scale vascular structures seen in traditional biomedical data, microvasculature tends to have a network structure. Also, while large-scale structures serve primarily a transport function, microvasculature more directly exchanges molecules with surrounding tissues.

New developments in microscope technology allow imaging of these complex microvascular networks. A major difficulty with visualizing complex volumetric network information is that the density of the network is prohibitive in understanding its structure. For example, visualizing a small region of a microvascular network reveals little about its structure. In addition, features outside of the microvascular network, such as cell bodies, add to the complexity (Figure 1). Our goal is to selectively visualize the network and associated tissue data. We do this by constructing a graph describing the structure and connectivity of the network. We then encode the volumetric data describing the network into the graph, which can then be queried for interesting anatomical structures. These structures are then selectively displayed to the user.

In this paper, we make two major contributions. First, we develop a framework for visualizing dense microvascular networks. This framework allows biologists to selectively visualize filaments based on anatomical queries, such as connectivity and distance. We then extend our framework to incorporate the visualization of features in the surrounding tissue, such as cell positions and diameters. These tools provide a novel way of exploring densely packed data sets with large numbers of anatomical features and relationships.

- *David M. Mayerich is with the Department of Computer Science, Texas A&M University, E-mail: mayerichd@neo.tamu.edu.*
- *Louise C. Abbott is with the Department of Veterinary Integrative Biosciences, Texas A&M University, E-mail: labbott@cvm.tamu.edu*
- *John Keyser is with the Department of Computer Science, Texas A&M University, E-mail: keyser@cs.tamu.edu*

Manuscript received 31 March 2008; accepted 1 August 2008; posted online 19 October 2008; mailed on 13 October 2008.
For information on obtaining reprints of this article, please send e-mailto:tvcg@computer.org.

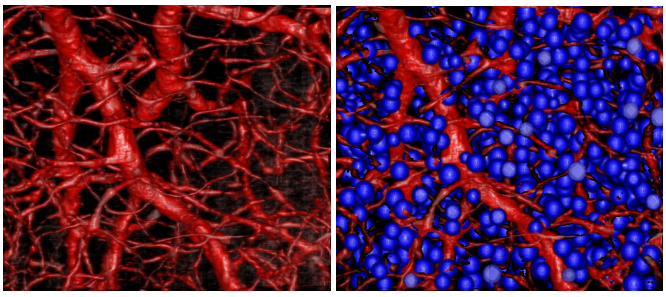


Fig. 1. Despite the density and complexity of the vascular isosurface, the volume occupied by the network is less than 3% of the total volume. Much of the remaining volume is occupied by cell bodies (right) and extracellular tissue.

1.1 Motivation

The vertebrate vascular system is formed by a cyclical network of arteries and veins, carrying oxygen rich and oxygen poor blood respectively. Starting from the heart, they branch into smaller and smaller vessels. At the microscopic scale, the smallest blood vessels, known as *capillaries* (or microvasculature), form a meshwork that connects veins and arteries and delivers nutrients to nearby cells.

The structure and function of microvasculature is important, particularly for neurological tissue. Changes in brain microvasculature have been linked to several chronic conditions such as Alzheimer's Disease, Parkinson's Disease, and Multiple Sclerosis [35]. The relationship between cellular and vascular tissue is also of interest in tumors and other chronic tissue.

The primary interest in all of these cases is the degree of access tissue has to blood-borne molecules. In the brain, the diffusion of these molecules is limited by the Blood-Brain Barrier (BBB), a layer of cells that tightly control molecular diffusion in the brain. Although there are means of active transport through these cells, the primary means through which tissue receives chemical input from the vascular system is through diffusion of molecules from capillaries (Figure 2). Therefore the distance between the capillary surface and surrounding tissue is an important factor describing the effect a capillary has on tissue.

Finally, high-resolution surface details are important for understanding microvascular structure and its relationship to cells. In particular, rendering the capillary surface allows us to explore regions of angiogenesis, where microvasculature is undergoing changes through the formation of new capillaries or the destruction of existing ones. There is also a close relationship between the capillary surface and certain types of cells. For example, endothelial cells play an impor-

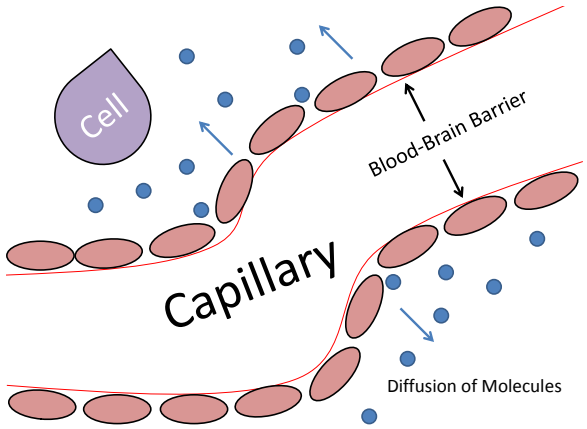


Fig. 2. Small molecules (such as oxygen) used to sustain cells are diffused through the capillary wall and into the surrounding tissue.

tant part in defining the capillary surface. Visualization of this mutual relationship is difficult using simplified representations of vasculature commonly used for macroscopic data [8, 17, 4].

1.2 Imaging

In this section, we briefly discuss the imaging methods used to create microvascular and cellular data sets. The advent of high-throughput microscopy allows researchers to quickly produce large volumetric data sets representing high-resolution biological tissue. Knife-Edge Scanning Microscopy (KESM) [14] is capable of imaging large specimens at microscopic resolution producing data at a rate in excess of 100MB/second. Imaging entire organs such as the mouse brain produces several terabytes of data. At this resolution biological tissue is highly complex, containing densely packed high-frequency structures.

In this paper, we use data acquired from KESM scans of rat brain. In order to image tissue using light microscopy, the tissue samples are stained and then imaged. In particular, cell bodies (soma) are stained (dark). During the staining process, all blood is flushed from the capillaries in the tissue. The microvasculature is therefore visible as unstained filaments (light) (Figure 3).

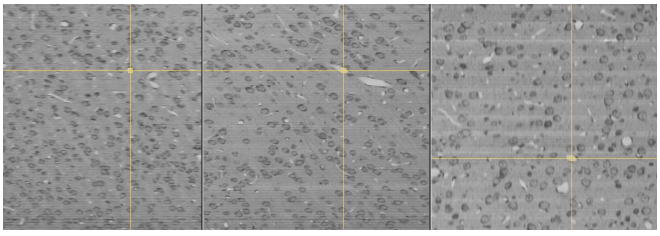


Fig. 3. Orthogonal cross-sections from a 512x512x512 voxel data set. Cross-hairs indicate the same position (within a capillary) in each section. Cell bodies and cell nuclei are dark while surrounding tissue is light gray. Capillaries are unstained.

2 Segmentation

Although our primary focus in this paper is the visualization of filament networks, we briefly discuss the segmentation methods used to extract the capillary network and cells. Filament structures such as microvasculature are inherently difficult to segment. This is due to low overall contrast, significant interfering (cellular) data, and image noise. In addition, thin filaments often drop below the resolving power of the microscope, causing gaps in the network. Naive methods such as isosurface reconstruction yield misclassifications, causing gaps in the network and erroneous surface details (Figure 4).

Many filament tracking algorithms have been developed for medical imaging methods such as X-Ray and MRI. An extensive review on the subject is done by Kirbas, et al. [10]. Methods that rely on the isosurface of the network [23, 33] work well for thick structures but frequent gaps and misclassifications make locating the medial axis difficult since there is no easy way to locally differentiate between noise and structure. Template matching [29] is effective in the presence of noise and gaps in the network, however this requires matching scaled and oriented cylinders in a three-dimensional space, which is computationally expensive for large data sets.

The methods that worked best for tracking these types of thin filaments are medial-axis (or *vector*) tracking algorithms such as those designed for tracking neurons in confocal [1] and serial [15] microscopy. These methods rely only on information local to the filament and can offer significant speedups over template matching. Fast algorithms are important since processing time often exceeds the amount of time required for imaging.

We first extract a skeleton of the microvascular network using a vector tracking algorithm. Full details can be found in Al-Kofahi et al. [1], while techniques for accelerating these algorithms can be found in our previous work [15]. This allows us to establish the positions of capillary centerlines and connectivity. We initialize vector tracking by placing seed points throughout the data set based on a conservative threshold. A template is then placed at each seed point and rotated and scaled to minimize a heuristic

$$h(T) = \int\int\int |\Phi(T\bar{x}) - \gamma(\bar{x})| d\bar{x} \tag{1}$$

where Φ is the data set, γ is a template function, the vector $\bar{x}$ is a point on the template. The transformation matrix T is constructed from the position, orientation, and size of the template:

$$\mathrm{T} = \mathrm{Tr} \times \mathrm{R} \times \mathrm{S} \tag{2}$$

where Tr transforms the template to the initial position and R and S define the orientation and scaling of the template respectively.

The minimum value of h is found by sampling discrete sets of transformations. We construct Tr based on our initial position. We then sample a series of orientations, searching for a minimum value of h. Finally, we update the size of the template by sampling a series of sizes, continuing the minimization of h. The position of the template is then updated by taking a step along the estimated filament trajectory based on the estimated orientation. In order to segment capillary networks, we use a cylindrical template. The orientation and size of the template is adjusted as it is moved down each filament. Intersections are detected based on the proximity of two segments.

Unlike network data, cells vary much less in size and cell bodies are mostly rotationally invariant. Therefore, we can locate cell positions using standard template matching with a small Gaussian correlation kernel [7]. Although this technique can be error-prone, we note that cells exist in vast numbers in biological tissue and highly accurate information is not necessary to demonstrate our visualization methods. We therefore manually select cells in our sample data set in order to provide input to our visualization framework. The segmentation of cellular structures is a well understood problem and several other automated methods are available [19, 11].

3 Volumetric Encoding

Using the vector tracking techniques described previously (Section 2), we construct a graph G describing the position (through estimates of the central axis of capillaries) and connectivity of the microvascular network. In addition, the tracking algorithm provides an estimate of the filament radius at each node. Using standard graph analysis algorithms [5], we now query G for structural and statistical information.

Although some basic visualization of the network can be provided using the information in the graph [3], G does not contain any volumetric information and the surface structure can only be estimated from the radius of each node. This surface information can be particularly important in physically-based simulations of fluid dynamics

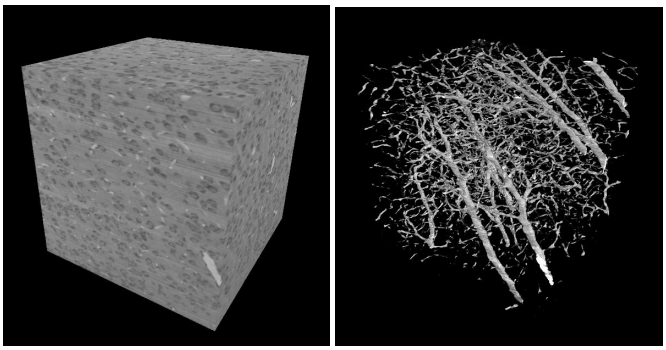


Fig. 4. Orthogonal sections from Figure 3 shown in context (left) and an isosurface representing the microvasculature.

through the network. The ability to store volumetric information also extends to other types of networks, such as those created by neurons, which have important surface features. Finally, the volumetric information associated with the vascular network provides more accurate and interesting visualization.

3.1 Prior Work

Encoding volumetric network data is a unique and challenging problem. Although there are many techniques for creating a bounding volume around the network using structures such as cylinders [13], spline models [27], and various other primitives [28], these methods are primarily used for simplified visualization and do not provide any means of encoding the volumetric data. We are aware of only one method [18] for storing data associated with filament structures as a series of attached axis-aligned bounding boxes, although this is primarily used as a means of compression and there are no efficient algorithms for accessing the volumetric data.

3.2 Dynamic Tubular Grids

The method that we use is based on Dynamic Tubular (DT) Grids [22], which were designed to encode a narrow band of volumetric data around an implicit surface computing advancing fronts [25]. In our case, we use DT-Grids to encode the region surrounding the network skeleton defined by G.

DT-Grids store an array of points representing the volume data in lexicographic order in each dimension $x_1, x_2, \ldots x_n$. When constructing a DT-Grid, the points must be inserted in this order. Insertions and deletions are not allowed elsewhere in the structure. The data is represented recursively as a series of projections onto a lower dimension. The data itself is stored in a single array in lexicographical order while an underlying data structure keeps track of the coordinates of each value.

DT-Grids have several algorithmic features. Some of the most important for our visualization techniques are:

- Constant time ($O(1)$) access when values are accessed in lexicographical order.
- Constant time access to neighboring voxels when using a template.
- Logarithmic time ($O(\log n)$) random access.

Additionally, DT-Grids provide significant compression when used to store large sparse data sets. Despite the complexity of our volumetric networks, the scalar data describing the network occupies a small percentage of the entire volume. Provided that we limit ourselves to operations on DT-Grids that can be done in constant time, this provides significant speedup over processing the entire volumetric data set. An extensive implementation of DT-Grids and a discussion of their algorithmic properties is given in the paper by Nielson, et al. [22].

3.3 Mapping Volume Data to a Graph

Our segmentation and tracing has given us a graph, G, describing the topology of the capillary network. This graph G consists of a set of nodes and connecting edges. Each node in the graph represents a sample point on the network skeleton while each edge describes the connectivity between neighboring points. We first define some terms that are used to refer to anatomical structures in G:

- a *branch* point is a node with three or more incident edges. These points represent branches in the network.
- a *termination* point is a node with only one incident edge. These represent a capillary termination, which can be due to an error in the tracking algorithm, a developing capillary, or a capillary leaving the bounds of the data set.
- a *segment* is a series of edges between branch or termination points. These edge sequences represent a single capillary.

For any point in G, we can look up a corresponding point in the volumetric data, Φ. However, what we actually need is a way of associating larger volumetric regions with the individual components (vertices and edges) of G. This can be a complicated computation, and so we precompute the associations, and use DT-Grids to store this correspondence. We will describe this preprocessing (i.e. storing the correspondence) here.

We apply a method proposed by Mayerich and Keyser [15] to construct a bounding volume around each segment using a series of truncated generalized cones (TGC) (Figure 5). Each TGC is defined by two adjacent nodes along a segment and has a medial axis defined by the edge connecting the points. The TGCs are connected end-to-end with the end caps oriented using normals defined as

$$\mathbf{n_b} = \frac{1}{2}\left(\frac{\mathbf{b}-\mathbf{a}}{||\mathbf{b}-\mathbf{a}||} + \frac{\mathbf{c}-\mathbf{b}}{||\mathbf{c}-\mathbf{b}||}\right) \tag{3}$$

where $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ are three consecutive points on the estimated medial axis of a filament. The radius of each end cap is equal to the radius stored at each node in G.

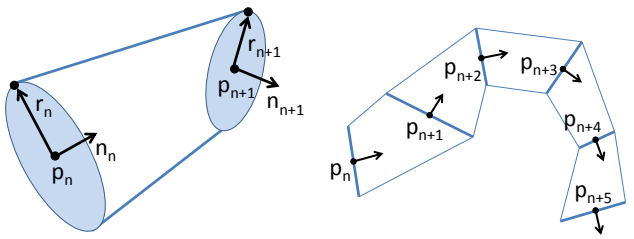


Fig. 5. Truncated generalized cones (left) fit end-to-end to construct the bounding volume for a filament (right).

In order to determine if a given point exists within the bounding volume, we first define the region within a TGC. Assuming that there is a point $\mathbf{p_x}$ under consideration, we want to determine if it lies within the TGC specified by the end caps $(\mathbf{p_n}, \mathbf{n_n}, r_n)$ and $(\mathbf{p_{n+1}}, \mathbf{n_{n+1}}, r_{n+1})$ where $\mathbf{p_n}$ is the point at the center of the end cap, $\mathbf{n_n}$ is the end cap normal calculated from Equation 3 and r_n is the end cap radius (Figure 6). We first find the plane that passes through points $\mathbf{p_x}$, $\mathbf{p_n}$, and $\mathbf{p_{n+1}}$ with normal

$$\mathbf{n_{plane}} = \frac{\mathbf{p_{n+1}}-\mathbf{p_n}}{||\mathbf{p_{n+1}}-\mathbf{p_n}||} \times \frac{\mathbf{p_x}-\mathbf{p_n}}{||\mathbf{p_x}-\mathbf{p_n}||} \tag{4}$$

For each end cap, we then find the end point of the line segment that represents the intersection of the end cap with the plane:

$$\mathbf{p_{r,n}} = \mathbf{p_n} + r_n(\mathbf{n_{plane}} \times \mathbf{n_n}) \tag{5}$$

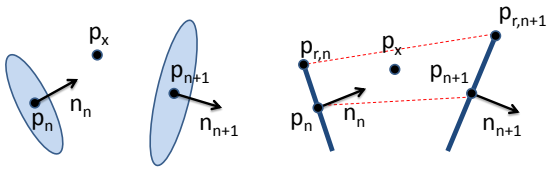


Fig. 6. (a) End caps of a TGC and (b) the intersection of the TGC with the plane defined by $\mathbf{p_x}$, $\mathbf{p_n}$, and $\mathbf{p_{n+1}}$.

we know that the point $\mathbf{p_x}$ is within the TGC if it lies within the polygon formed by $(\mathbf{p_n}, \mathbf{p_{r,n}}, \mathbf{p_{r,n+1}}, \mathbf{p_{n+1}})$.

Having constructed the TGCs, we now associate all values in the volume image Φ as either inside or outside the bounding volume of G. We wish to avoid comparing all N voxel positions with the TGCs of all M edges, since both N and M are large numbers. Instead, we "rasterize" each TGC onto a new grid, Λ of the same resolution as the original volume Φ. Each point of the grid Λ contains a unique number identifying the segment that the corresponding point in Φ is associated with. For each point inside the bounding volume of a filament, we write an identifier indicating the associated segment in G. Due to possible overlaps between neighboring filament bounding volumes, particularly at intersections, the values of Λ may be overwritten multiple times. Since the vascular volume is relatively small, this generally only happens at intersections and poses no problem for visualization and storage. In fact, this eliminates redundancy in the encoding scheme since each voxel in Φ is associated with at most a single network segment. We therefore ensure that the minimum amount of data is stored in the resulting DT-Grids based on the bounding volume of the network.

After rasterizing all TGCs, we iterate through all elements of Λ in lexicographical order. If a value was written to $\Lambda(x,y,z)$, we push the associated volumetric value $\Phi(x,y,z)$ into the DT-Grid representing the segment identified by $\Lambda(x,y,z)$. Each value in Φ is either ignored or added into at most one DT-Grid representing a single segment in G.

At this point, we now have a graph G containing the original medial axis, connectivity, and radius information. Additionally, each segment in G has an associated DT-Grid containing the original scalar volume values in Φ that make up that segment. Even with the large overhead of the DT-Grid data structure, our experiments still show significant compression over storing the original scalar data Φ as a three-dimensional grid (Figure 7).

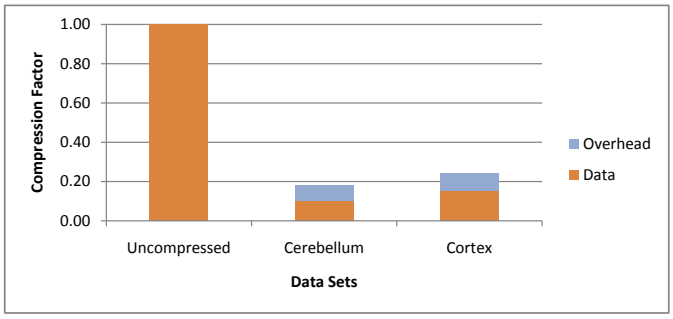


Fig. 7. Compression achieved after processing the volume function Φ into a series of DT-Grids associated with each segment. All volumetric data describing the network is stored while $\approx 90\%$ of the volume data representing surrounding tissue is culled. We compare the size of a full volume to sample volumes in two regions of the rat brain.

4 Visualization

The key to being able to visually convey complex data is to selectively render related structures, thereby limiting the amount of visual information a user sees at one time. We now describe methods for selectively visualizing the volumetric data associated with our network structure. We also note that the overhead size is constant and independent of voxel size. As the voxel size increases (e.g. by storing colors or gradients), the compression provided by a DT-Grid relative to a uniform grid increases.

4.1 Previous Work

Several techniques exist for the visualization of filament and fiber data that can also be applied to vascular networks. Several surface construction methods discussed previously (Section 3.1) are designed for visualization. A large body of work is also available for estimating vascular surfaces [8, 17, 4] for visualization. However, these techniques do not capture surface features.

Oriented imposter methods such as stream lines are very good at representing highly organized fibers, such as those found in Diffusion Tensor Imaging [30, 32, 26]. These techniques have little applicability, however, in complex networks that have highly uncorrelated orientations as are found in microvasculature. Although these techniques can be adapted to selectively highlight filaments [20], unselected data still interferes with visualization. Additionally, these visualization techniques do not capture surface features, which can be important when exploring relationships between microvasculature and surrounding tissue.

4.2 Volume Visualization and Isosurfacing

Both ray-casting and isosurface rendering are important for volume visualization but do not offer many opportunities for selective visualization. Isosurfaces provide a mesh that can be rendered efficiently using hardware acceleration while direct volume rendering provides a way to visualize fine surface details that would be lost when selecting a single isovalue. The easiest way to selectively visualize structures in volume visualization is to cull away regions, focusing on local areas of interest. This is of little help in microscopy data sets since vascular structures can span the entire data set. Isosurfaces, in contrast, are often represented as triangles and therefore independent surfaces can be selectively visualized. This still causes problems with microvascular networks, however, since the structures are highly connected, requiring some means of separating the surface into useful components. In addition, selecting an isosurface value that fully captures the structure while limiting misclassifications can be impossible. We will now discuss how to adapt our algorithms to selectively visualize complex networks using standard volume visualization and isosurfacing techniques while eliminating most problems with noise.

The total volume of brain microvasculature is known to make up only a small percentage ($< 3\% - 6\%$) of the total volume in biological tissue. Since Φ is on a discrete grid, we must store all voxels that intersect the vascular surface. The estimated radius of our vector tracking algorithm (Section 2) and subsequent encoding based on a bounding volume (Section 3) therefore retains a slightly larger portion of the data set ($\approx 10\%$). This prevents important capillary features from being culled from the final image. Although noise in microscopy is highly correlated from section to section [16], it tends to occur randomly throughout a volume. Therefore, by culling over 90% of the volume, we have eliminated large quantities of volume noise that would interfere with isosurfacing and volume rendering methods. In the case of isosurfacing, we can now use more liberal isovalues without generating surface artifacts elsewhere in the data set.

Isosurfacing algorithms are straightforward to implement directly on a DT-Grid. For example, implementation of the Marching Cubes algorithm [12] requires that each element in the grid be visited, along with its neighbors. Although random access to any given neighbor requires logarithmic time (Section 3.2), the use of a 3x3 template across all elements allows the isosurface to be reconstructed in time linear to the number of values in the DT-Grid. This provides a net gain in efficiency since the DT-Grid only contains values near the network. Since geometry can be sent arbitrarily to the graphics card, the isosurface can be updated interactively based on the user's selection criteria.

The major problem encountered when rendering DT-Grids using volume rendering methods such as ray-casting is that random access into a DT-Grid requires logarithmic time. This is further complicated by our structure since G contains a separate DT-Grid for each segment. One way to combat this problem is to create a union of all selected segments and render them as a single DT-Grid. Creating the union of all segments requires time linear to the number of voxels in the union, which can be very efficient. However, the resulting structure is still a DT-Grid and requires logarithmic time for random access. We note that Nielson et al. [22]. have shown that, for large data sets, the logarithmic time random access is often faster than the constant time random access provided by a uniform grid. This is due to cache coherence. Unfortunately, there are no predefined algorithms to use DT-Grids with graphics hardware, so such methods would be limited to CPU-based rendering.

In order to implement hardware-accelerated volume rendering of our networks, we use a more direct approach. We store a uniform grid as a three-dimensional texture in graphics memory. When the user makes any updates to the selected filaments, the DT-Grids for each selected segment are copied to a 3D "canvas" in main memory. This canvas is then copied to the graphics card, updating the rendered texture. Copying the DT-Grids to the canvas is a highly efficient operation and can be performed interactively. In this case, updating the texture map in graphics memory is the bottleneck and causes some frame stuttering, depending on bus speed, when the user changes the selection criteria.

4.3 Results

These techniques allow us to perform structured visualization of complex network data. We are able to visualize the actual volumetric or isosurface data interactively based on user-selection. Examples of user selection operations include:

- Picking filaments from the network,
- Expanding a list of components using a breadth first search,
- Selecting filaments based on statistical queries of G (such as radius, branch angle, etc.),
- Visualization of the shortest path between two points in the network,
- and visualization of the diffusion of a substance based on simulated flow through the network.

Operations such as breadth first search allow us to understand the high level of connectivity that has been theorized to exist throughout the microvascular system. By selecting an initial sample of filaments based on these techniques, we use a breadth first search to explore network connectivity (Figure 8). By dynamically expanding and contracting the network, a user can view capillaries in the context of the entire network and then refine the visualization to include only regions of interest.

5 Cellular-Vascular Relationships

One of the most interesting relationships between microvasculature and the surrounding tissue is between capillaries and nearby cells. The ability to explore these relationships would provide a valuable tool for scientists interested in the effects of microvasculature on cell nutrition and modulation.

5.1 Distance Metrics

Nutrients and other chemicals travel from capillaries to nearby cells using both diffusion and active transport (where mediating proteins carry molecules). Both of these transport methods are highly dependent on the distance of the cell to the capillary surface. The distance of the cell surface from the surface of nearby capillaries is therefore an effective metric for measuring the relationship between a cell and capillary.

It is straightforward to compute the distance from any point in the tissue to the central axis of any capillary, since this axis is extracted during segmentation. However, since capillaries vary in size, we should also take the radius of the capillary into account. We do this by using the distance to the computed bounding volume of the network (Section 3). In order to compute the distance from any specified point p_x interactively, we compute the distance between p_x and the TGC surrounding each edge in G. In order to determine the distance between a point p_x and a TGC, we use Equation 4 and Equation 5. We then compute the distance between p_n and the line formed by $p_{r,n}$ and $p_{r,n+1}$ (Figure 6).

5.2 Visualizing Distance Queries

We use this metric to visualize a network of capillaries closely associated with a selected point in the surrounding tissue as follows. By specifying a point in the volume Φ, we can select nearby components of G based on their distance from the specified point. Since often the effects of individual capillaries (not just portions thereof) are of interest to biologists, we can further adjust the selection to identify segments based on the distance of their nearest components in G (Figure 9). For measuring the distance from points to capillaries, we define the segment distance as the distance from the closest component edge (the minimum edge distance).

We show how this can be effectively used to cull excess microvascular information from a volume visualization (Figure 10). In this figure, the same region of tissue is visualized. The first image shows the entire region while the following images show only segments near the selected point. As the "radius of interest" is reduced, the structure of the surrounding capillaries becomes more clear.

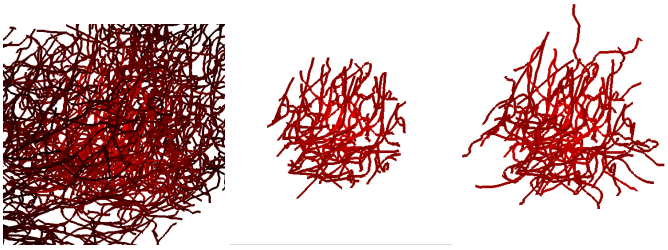
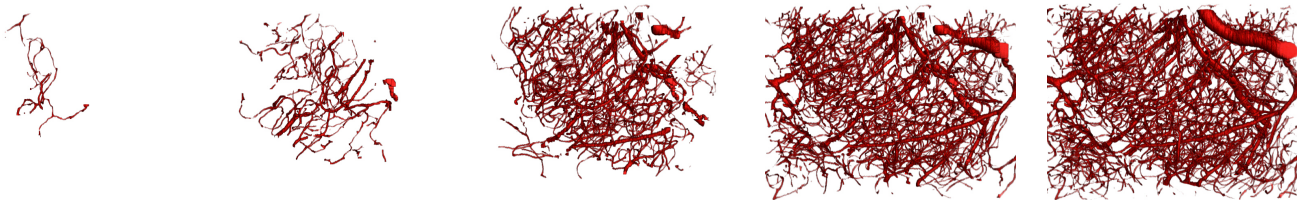


Fig. 9. Vertex and filament distance from a point. A region of cerebellar cortex is rendered using oriented billboards. Segments are colored based on the distance of the closest point to a cell (left). We then clip edges to a region $\approx 100\mu m$ from a cell. We can either select only edges within that distance (center) or any segment that contains an edge/vertex within that distance (right).

When selecting specific cells in order to determine their distance to neighboring capillaries, we must also take the radius of the cell into account. Since our cell segmentation is limited to a center point and radius, we simply subtract the cell radius r from the computed distance between the vascular surface and the center point p.

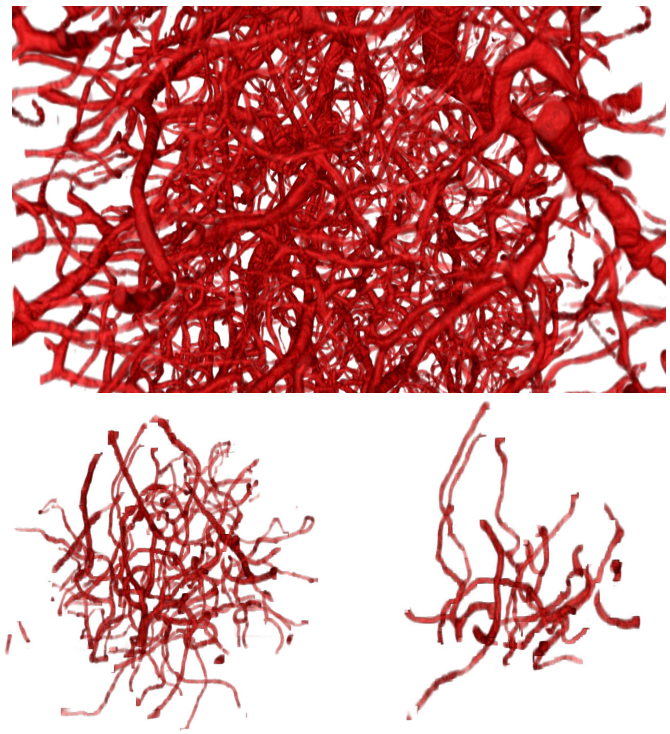
5.3 Voronoi Diagrams

Although the selection described above allows us to interactively explore the structure of vessels that support and modulate a single cell, many theories concerning the function of microvasculature concern the number and positions of cells surrounding microvessels. Of primary concern are cells that are most directly associated with a capillary. Using the techniques described in the previous section (Section 5.1) would involve computing the distance between every cell and blood vessel and sorting all values to find the closest relationships. Since both the number of cells and the number of edges in G are large, this process can be time consuming. Additionally, other aspects of the surrounding tissue may be of interest besides cell positions. For example, understanding the size and shape of a region closely related to a capillary may be interesting in studying tumor growth. This would require comparing each voxel in Φ to the bounding volume of G. The



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

Fig. 8. Network searching and selection. A small bundle of filaments was picked by the user (a). Successive expansion of the network using a breadth-first search (b-d). The complete network (e) contains filaments at the edge of the data set that are unconnected to the main network.



(a) (b)

Fig. 10. Volume rendering of the region in Figure 9. The entire volumetric network is shown (top) as well as a cradle of vessels $\approx 100\mu m$ (a) and $\approx 50\mu m$ (b) from a specified point.

following method allows us to efficiently associate regions of tissue in Φ with the closest segments in G.

We label these regions by constructing an implicit Voronoi diagram based on G. Several methods are available for constructing Voronoi diagrams [24, 2] but most are designed to work with point sets rather than implicit or geometric surfaces. Fast methods have been proposed for computing implicit Voronoi diagrams from geometric primitives [9], however these are algorithmically complex, requiring $O(PN)$ where P is the number of primitives and N is the number of voxels in Φ.

We compute the Voronoi diagram by propogating an identifier for the Voronoi region while constructing a signed distance function (SDF). We use the standard definition of a SDF

$$\Psi = \begin{cases} d_\Psi(\bar{x},\Gamma) & \text{if } \bar{x} \text{ is outside } \Gamma \text{ and} \\ -d_\Psi(\bar{x},\Gamma) & \text{if } \bar{x} \text{ is inside } \Gamma. \end{cases} \quad (6)$$

where Γ is the vascular surface and $d_\Psi(\bar{x},\Gamma)$ is the distance between $\bar{x}$ and Γ.

There are several well understood methods for constructing signed distance functions on a discrete grid. The most commonly used algorithm is the Fast Marching Method [31] which requires $O(n \log n)$ time to evaluate. The Fast Sweeping algorithm [34] converges in $O(n)$ time, although it requires that every point in the implicit function Ψ be visited multiple times. Since we wish to evaluate every point in Ψ, we use Fast Sweeping for efficiency.

Fast sweeping requires that we specify initial boundary conditions from which the final SDF is iteratively computed. We first initialize Ψ to some large value $+\infty$. We then rasterize each edge of G into Ψ using a three-dimensional line drawing algorithm [6]. At each sample point in Ψ intersected by an edge, we write

$$\Psi(\bar{x}) = -r_G(\bar{x}) \quad (7)$$

where r_G is the radius measured at the edge and $\bar{x}$ is the position in G along the edge. Both values are computed by interpolating between the graph nodes defining the edge. We then use Fast Sweeping to propagate the distance values through the rest of the function (Figure 11). Implementation details as well as the Ψ update function for uniform grids with cubic voxels are provided in the paper by Zhao [34]. Additional details describing the construction of SDFs can be found in Osher and Fedkiw [25].

It should be noted that the voxels of many actual biological scans (including ours) are not cubic. This means that a more complicated update function is required to correctly perform fast sweeping, or the data must be resampled to a uniformly spaced grid ($dx = dy = dz$). If a more complex update function is used, this involves solving the quadratic equation:

$$\left(\frac{\bar{x}_{x,y,z} - \bar{x}_{x-1,y,z}}{dx}\right)^2 + \left(\frac{\bar{x}_{x,y,z} - \bar{x}_{x,y-1,z}}{dy}\right)^2 + \left(\frac{\bar{x}_{x,y,z} - \bar{x}_{x,y,z-1}}{dz}\right)^2 = F \quad (8)$$

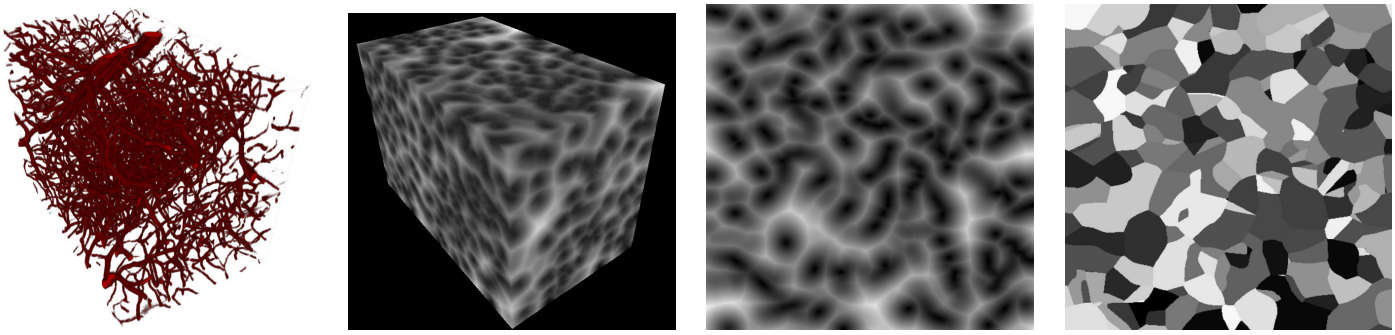
for $\bar{x}_{x,y,z}$, which is the value in Ψ at point (x,y,z).

We construct the implicit Voronoi diagram by propagating a unique identifier for each filament along with the distance function. This is done during the Fast Sweeping procedure. In addition to the function Ψ, we maintain a separate implicit function Ψ_{VOR} that stores an identifier at each voxel indicating from which segment the distance value at that point originated. After completing all iterations of Fast Sweeping (eight for $\Psi(\bar{x})$ if $\bar{x} \in \mathbb{R}^3$), Ψ_{VOR} contains the complete Voronoi diagram where each point contains an identifier to the segment closest to that point.

Finally, we use the computed Voronoi diagram to assign each cell to the appropriate capillaries (segments in G). Frequently, the cell will overlap multiple Voronoi regions. Depending on the desired visualization, the cell can be associated with one or many capillaries. In our visualizations, we elected to simply associate the cell with the region in which the cell's center was positioned. We can then render cell positions along with capillaries as either simplified spheres (representing the cell position and radius) or using volumetric data from the original data set (Figure 12). In the latter case, we stored the volumetric data in a DT-Grid associated with each cell as described in Section 3, using a sphere as a bounding volume.

6 Discussion

Although there are several methods available for visualizing vasculature on a macroscopic scale, microvasculature is a significantly more



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

Fig. 11. Vascular network (a) and associated three-dimensional implicit signed distance function (b and c). The associated Voronoi diagram is constructed by propogating an identifier for each segment along with the distance value during computation of the Eikonal equation.

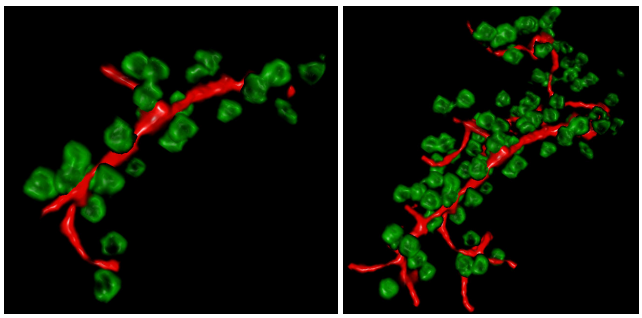


Fig. 12. A series of connected capillaries in the mouse somatosensory cortex along with their associated cell bodies.

complex and interconnected structure. In order to interpret complex networks, biologists have been limited to statistical models describing features such as branch angles and vessel direction. Although these features are important for comparing biological models, they do little to elucidate local characteristics, such as the positions of cells around individual capillaries. We provide a tool that allows biologists to explore the connectivity and complexity of tissue microvasculature without sacrificing important high-resolution details, such as surface structure. This provides a tool for biologists to explore several potential areas, including:

- visualization of *angiogenesis*, or the formation of capillaries in the microvascular system
- visualization of the surface structure of capillaries, particularly around bifurcations
- analysis and visualization of microscopic aneurisms and other abnormalities visible in the microvascular surface
- detailed visualization of differences in microvascular structures between two types of tissue (control and chronic)

These features would be impossible to visualize using simplified primitives, such as cylinders or billboards. In addition, our selective visualization techniques allow a user to exclude excessive amounts of data and focus on local regions of interest.

We also associate the tissue surrounding the microvascular network with individual capillaries. This is of particular interest to biologists studying the relationships between capillaries and individual cells. This allows researchers to visualize several features that are not yet well understood, including:

- the number and types of cells directly associated with a single microvessel
- the regions of tissue that may be affected by aneurisms in either the macrovascular or microvascular system
- differences in the volume associated with a microvessel based on tissue type and region

We are unaware of any methods in the literature that allow biologists to explore the relationships between a network and its surrounding tissue. We have therefore provided a new valuable tool that allows researchers to visualize information that is not well understood.

7 Conclusion and Future Work

In this paper, we discuss a framework for visualizing the complex structure of a microvascular network. We then extend this framework to visualize the network's anatomical relationship to the surrounding tissue.

Due to the large size and high data rate of KESM, we were careful to limit our visualization algorithms to those with time complexity no greater than $O(N)$ where N identifies some large number of features. In our framework, this includes the number of cells, the number of nodes and edges in G, and the number of voxels in the data set Φ.

We provide the time required for several pre-processing and interactive stages in our framework (Table 1). We did not implement any automated methods for cell segmentation, since this is a well-understood problem and many methods are available in the literature [19, **?**, 7]. In order to test our visualization methods we manually labeled cells in the data set. Segmentation of the microvascular network is performed automatically and approximately 97% of the network is labeled correctly while 3% of the capillaries were missed by the tracking algorithm.

All data sets were 512x512x512 voxels and rendered using a Geforce 7900 graphics board. We were able to maintain interactive frame rates while looking at any specified network. When the user changed the network, the appropriate DT Grids are copied onto a pre-allocated 3D canvas and sent to the GPU. This time required to perform this copy is highly dependent on the size of the network but was less than 0.2 seconds in most cases. This results in a small amount of stuttering when the network is changed, but the interactive frame rate returns to after the copy.

Additionally, filament networks occur frequently in other high-throughput microscopy data sets. In particular, networks of neuronal fibers found in brain tissue can be visualized using techniques similar to those discussed in this paper. Storing the volumetric data associated with neuronal fibers is particularly important since surface features play an important role in identifying their function (Figure 13).

Finally, as data sets created using high-throughput microscopy become more frequently available, we expect to see many anatomical structures that are significantly more complex than those found at the macroscopic scale using traditional imaging methods. The density and high-frequency nature of these structures will require the use of underlying data structures similar to ours that can be used to query and selectively visualize anatomy.

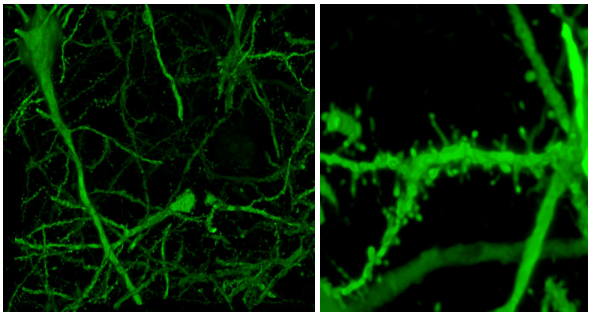


Fig. 13. Neurons imaged using Array Tomography [21] (left). Storing the network using DT-Grids preserves important surface details (right).

Table 1. Time required for various stages of our framework using a 512x512x512 cerebellar data set (Figure 8) on an NVIDIA Geforce 7900. An updated filament volume is sent to the GPU only when the user changes the selection criteria.

Stage	Time
Preprocessing	
Vascular Segmentation	229.248 s
Cell Segmentation	Manual
Create DT-Grids	42.192 s
Interactive	
Update Filaments on the GPU	0.188 s
Render Time	0.047 s

8 Acknowledgements

We would like to thank Stephen Smith, Kristina Micheva, and Brad Busse for their Array Tomography data. This work was supported in part by NIH/NINDS grant 1R01-NS54252.

References

[1] K. Al-Kofahi, S. Lasek, D. Szarowski, C. Pace, G. Nagy, J. Turner, and B. Roysam. Rapid automated three-dimensional tracing of neurons from confocal image stacks. *IEEE Transactions on Information Technology in Biomedicine*, 6:171–186, 2002.

[2] F. Aurenhammer. Voronoi diagrams: A survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23:345–405, 1991.

[3] G. D. Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall PTR Upper Saddle River, NJ, USA, 1998.

[4] J. Bloomenthal and K. Shoemake. Convolution surfaces. *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques*, pages 251–256, 1991.

[5] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications*. Elsevier Science Ltd., 1976.

[6] J. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 4:25–30, 1965.

[7] A. D'Souza. *Automated Counting of Cell Bodies Using Nissl Stained Cross-Sectional Images*. Master's thesis, Texas A&M University, 2007.

[8] G. Gerig, T. Roller, G. Szekely, C. Brechbuhler, and O. Kubler. *Symbolic description of 3-D structures applied to cerebral vessel tree obtained from MR angiography volume data*, volume 687 of *Lecture Notes in Computer Science*, pages 94–111. Springer Berlin, 1993.

[9] K. E. Hoff III, T. Culver, J. Keyser, M. Lin, and D. Manocha. Fast computation of generalized voronoi diagrams using graphics hardware. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, pages 277–286, 1999.

[10] C. Kirbas and F. Quek. A review of vessel extraction techniques and algorithms. *ACM Computing Surveys*, 36:81–121, 2004.

[11] G. Li, T. Liu, J. Nie, L. Guo, J. Chen, J. Zhu, W. Xia, A. Mara, S. Holley, and S. Wong. Segmentation of touching cell nuclei using gradient flow tracking. *Journal of Microscopy*, 231:47–58, 2008.

[12] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, pages 163–169, 1987.

[13] Y. Masutani, T. Schiemann, and K. H. Hohne. Vascular shape segmentation and structure extraction using a shape-based region-growing model. *Proceedings of the 1st International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 1242–1249, 1998.

[14] D. Mayerich, L. C. Abbott, and B. H. McCormick. Knife-edge scanning microscopy for imaging and reconstruction of three-dimensional anatomical structures of the mouse brain. *Journal of Microscopy*, 231:134–143, June 2008.

[15] D. Mayerich and J. Keyser. Filament tracking and encoding for complex biological networks. *Proceedings of the ACM Solid and Physical Modeling Symposium*, pages 353–358, 2008.

[16] D. Mayerich, B. H. McCormick, and J. Keyser. Noise and artifact removal in knife-edge scanning microscopy. *Proceedings of the 4th IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pages 556–559, 2007.

[17] A. Mazziotti, A. Cavallari, and J. Belghiti. *Techniques in Liver Surgery*. Greenwich Medical Media, 1997.

[18] B. McCormick, B. Busse, P. Doddapaneni, Z. Melek, and J. Keyser. Compression, segmentation, and modeling of filamentary volumetric data. In *Proceedings of the 9th ACM Symposium on Solid Modeling and Applications*, pages 333–338, 2004.

[19] D. P. Mccullough, P. R. Gudla, K. Meaburn, A. Kumar, M. Kuehn, and S. J. Lockett. 3d segmentation of whole cells and cell nuclei in tissue using dynamic programming. *Proceedings of the 4th IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pages 276–279, 2007.

[20] Z. Melek, D. M. Mayerich, C. Yuksel, and J. Keyser. Visualization of fibrous and thread-like data. *IEEE Transactions on Visualization and Computer Graphics*, 12:1165–1172, 2006.

[21] K. D. Micheva and S. J. Smith. Array tomography: A new tool for imaging the molecular architecture and ultrastructure of neural circuits. *Neuron*, 55:25–36, 2007.

[22] Nielsen and Museth. Dynamic tubular grid: An efficient data structure and algorithms for high resolution level sets. *Journal of Scientific Computing*, 26:261–299, 2006.

[23] N. Niki, Y. Kawata, H. Satoh, and T. Kumazaki. 3d imaging of blood vessels using x-ray rotational angiographic system. In *IEEE Nuclear Science Symposium and Medical Imaging Conference*, volume 3, pages 1873–1877.

[24] A. Okabe, B. Boots, and K. Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, Inc. New York, NY, USA, 1992.

[25] S. J. Osher and R. P. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2002.

[26] V. Petrovic, J. Fallon, and F. Kuester. Visualizing whole-brain dti tractography with gpu-based tuboids and lod management. *Transactions on Visualization and Computer Graphics*, 13:1488–1495, 2007.

[27] A. Pommert, K. H. Hohne, B. Pflesser, E. Richter, M. Riemer, T. Schiemann, R. Schubert, U. Schumacher, and U. Tiede. Creating a high-resolution spatial/symbolic model of the inner organs based on the visible human. *Medical Image Analysis*, 5:221–228, 2001.

[28] A. Puig, D. Tost, and I. Navazo. An interactive cerebral blood vessel exploration system. *Proceedings of the 8th Conference on Visualization '97*, pages 433–436, 1997.

[29] Y. Sato, S. Nakajima, N. Shiraga, H. Atsumi, S. Yoshida, T. Koller, G. Gerig, and R. Kikinis. Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images. *Medical Image Analysis*, 2:143–168, 1998.

[30] P. Schroder and G. Stoll. Data parallel volume rendering as line drawing. In *Proceedings of the 1992 Workshop on Volume Visualization*, pages 25–32, 1992.

[31] J. A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge University Press, 1999.

[32] C. Stoll, S. Gumhold, and H.-P. Seidel. Visualization with stylized line primitives. In *Proceedings of the 16th Conference on Visualization 2005*, pages 695–702, 2005.

[33] T. Tozaki, Y. Kawata, N. Niki, H. Ohmatsu, K. Eguchi, and N. Moriyama. Three-dimensional analysis of lung areas using thin slice ct images. In *Proceedings of SPIE*, volume 2709, pages 1–11, 1996.

[34] H. Zhao. A fast sweeping method for eikonal equations. *Mathematics of Computation*, 74:603–627, 2004.

[35] B. Zlokovic. The blood-brain barrier in health and chronic neurodegenerative disorders. *Neuron*, 57:178–201, 2008.