# VISUALIZATION OF DENSE LINE DATA

MAARTEN H. EVERTS


Netherlands Organisation for Scientific Research

This research was supported by the Netherlands Organisation for Scientific Research (NWO) under project number 643.100.501.

The enclosed anaglyph 3D glasses were produced by http://www.3d-brillen.nl.

Cover: DTI brain fiber tracts visualized using the depth-dependent halo technique of Chapter 3.

Everts, Maarten H.

Visualization of Dense Line Data
Maarten H. Everts
Thesis Rijksuniversiteit Groningen

ISBN 978-90-367-5215-2 (printed version)
ISBN 978-90-367-5216-9 (electronic version)

RIJKSUNIVERSITEIT GRONINGEN

# VISUALIZATION OF DENSE LINE DATA

Proefschrift

ter verkrijging van het doctoraat in de
Wiskunde en Natuurwetenschappen
aan de Rijksuniversiteit Groningen
op gezag van de
Rector Magnificus, dr. E. Sterken,
in het openbaar te verdedigen op
vrijdag 9 december 2011
om 12.45 uur

door

MAARTEN HINDERIK EVERTS

geboren op 2 september 1981
te Hoogeveen

| | |
|---|---|
| Promotor: | Prof. dr. J.B.T.M. Roerdink |
| Copromotores: | Dr. H. Bekker |
| | Dr. T. Isenberg |
| Beoordelingscommissie: | Prof. dr. E. Gröller |
| | Prof. dr. G. Scheuermann |
| | Prof. dr. ir. J.J. van Wijk |

Real artists ship.

— Steve Jobs

# CONTENTS

# 1 INTRODUCTION

Visualization is the art and science of turning data into a depiction that helps its viewer to gain a new or better understanding of the data. While originally a process of manual labor, the advent of computers has made it possible to visualize data of ever increasing size and complexity. And not only that, computers allow us to go beyond static imagery by producing animations and supporting interactive exploration. Simultaneously, the computational revolution has resulted in a data explosion, both in terms of size and diversity. Consequently, Visualization as a scientific field has flourished, branching out into three main subfields: Scientific Visualization, Information Visualization, and Visual Analytics.

This thesis aims to contribute to the first, Scientific Visualization. In this field, the goal is to produce graphical representations of scientific concepts and phenomena to help scientists extract information and gain insight. The data can be the result of measurements or simulations and is typically three-dimensional in nature. Examples include visualizations of molecules, geospatial data, medical scans (e.g., MRI), and simulations of galaxies.

## 1.1 DENSE LINES

The type of data central to this thesis is information recorded in the form of *dense lines*, more specifically, diffusion tensor imaging (DTI) fiber tracts and flow streamlines. Although the domains are very different (medical and mathematical, respectively), the nature and the means of extraction of both types of lines are very similar.

### 1.1.1 *DTI fiber tracts*

Fiber tracts are representations of bundles of fibrous biological material, generated from a series of measurements (scans) in an MRI scanner. Typically, and also for this thesis, the fibrous material in question is the white matter of the brain, which are the bundles of thousands myelinated axons that interconnect different regions of the gray matter. The central assumption for the generation of fiber tracts is that water molecules diffuse more easily along the direction of the fibrous material than orthogonally to it. With a sequence of special MRI scans, the magnitude and direction of diffusion of water can be measured and, thus, can provide information on the directionality of the fibrous material.

This relatively recent technique is called diffusion-weighted magnetic resonance imaging (DW-MRI) and it gave rise to a number of analysis techniques.

One of those techniques is *deterministic tractography*, where the directional information from the diffusion weighted scans is used to produce three-dimensional lines. Commonly, the measurements are transformed to produce a volume of tensors (hence the often used term *diffusion tensor imaging*—DTI) to describe the diffusion of water molecules [Basser et al., 1994]. There are several methods for producing fiber tracts from tensor fields, but the basic idea is to start for each tract at a seed point and continuously follow the greatest local eigenvector while moving through the tensor field [Mori and van Zijl, 2002]. It is important to keep in mind that, because the resolution of DW-MRI is limited (up to 1 mm), the resulting fiber tracts are merely representations of bundles of neuron fibers, and not individual neuron fibers.

*Diffusion tensor*

$$D = \begin{bmatrix} D_{xx} & D_{xy} & D_{xz} \\ D_{yx} & D_{yy} & D_{yz} \\ D_{zx} & D_{zy} & D_{zz} \end{bmatrix}$$

### 1.1.2 *Flow streamlines*

While fiber tracts can be thought to represent actual physically existing components in biological tissue (e. g., bundles of neuron fibers), other lines such as streamlines do not have such a clear mapping to physical entities. Instead, streamlines are visual representations of the behavior and structure of (complex) three-dimensional flow. They are generated from velocity (vector) fields, which are in turn typically the result of large three-dimensional numerical simulations of the flow of fluids and gases. The domains in which such simulations play a role are diverse and include, for example, the aerodynamics of cars, the heat distribution in offices, and the airflow around falling ink droplets. Streamlines are one of the ways to study and analyze the results of these simulations.

The generation of streamlines is similar to the generation of the fiber tracts described above, that is, start at a seed point and iteratively take small steps in the direction of the local velocity vector until a stopping condition is reached. The resulting trajectories are locally tangent to the velocity field, which is what makes them suitable for conveying the behavior and structure of the flow. However, it has to be noted that streamlines represent the path of the massless particle if the underlying vector field describes a steady flow. If the underlying vector field is a snapshot of an unsteady turbulent flow, streamlines can still help in conveying information about the flow, but they do not represent the path a particle would take in such a flow. However, there are also types of flow lines that do take into account the time-varying nature of turbulent flow: streak lines and path lines. While they are not used in this thesis, the application of our techniques to these types of lines would be straightforward.

### 1.1.3 *Visualization of dense lines*

The similarity of fiber tracts and streamlines lies foremost in their three-dimensional nature and their typically dense arrangement. Consequently, for both types of lines one faces similar perceptual challenges when trying to depict them, including conveying spatial relationships and avoiding clutter and occlusion. A common and straightforward approach for the depiction of three-dimensional lines is to turn to three-dimensional objects such as tubes and ribbons (e. g., [Ueng et al., 1996; Petrovic et al., 2007; Zhukov and Barr, 2004; Zöckler et al., 1996]). Compared to simply drawing thin, single color lines, the size of such objects and consequently the possibility for shading allows for a better understanding of spatial relationships and an improved depth perception. However, with the increased size, the amount of detail that can be depicted decreases and some of the line-like feeling is lost. To some extent this can be overcome using illuminated lines [Zöckler et al., 1996], although the problem of conveying spatial relationships remains. One of the contributions of this thesis is a new approach to line visualization (Chapter 3) that does not depend on shading, but instead borrows from principles of illustrative visualization, which is discussed next.

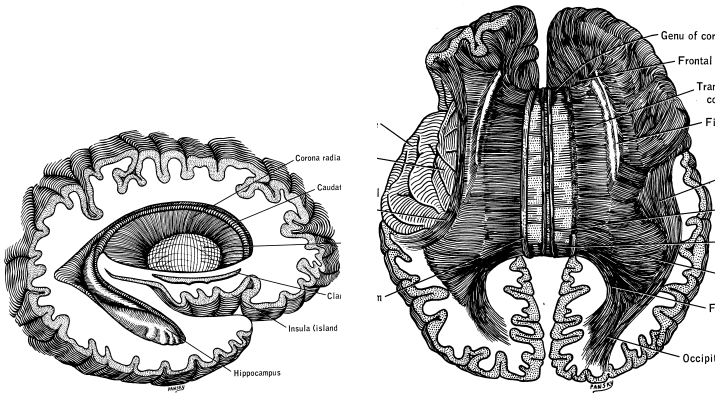
*In fact, in Chapter 2 we also use tubes for the depiction of 3D paths.*

## 1.2 ILLUSTRATIVE VISUALIZATION & ABSTRACTION

Illustrative visualization [Rautek et al., 2008] can be seen as as subfield of scientific visualization in which new approaches to visualization are developed that are inspired by the techniques of traditional illustrators. Long before the advent of computers, illustrators have been practicing and perfecting the art of visually communicating knowledge. Even with simple means such as pen-and-ink, illustrators are able to effectively convey shape, texture, and illumination. Figure 1.2 shows two illustrations from a medical handbook [House and Pansky, 1960] that demonstrate this effect. It is interesting to notice how, despite the lack of colors, these illustrations convey a certain clarity and crispness.

A large part of this clarity and crispness lies in the application of the principles of abstraction and emphasis. With the communication goal in mind, the illustrator can choose to leave out detail or even oversimplify things to show and emphasize higher level information. This process of depicting only the most relevant features is hard to capture algorithmically, but nonetheless, in the field of non-photorealistic rendering (NPR) a large variety of techniques have been developed that try to mimic the style and effectiveness of traditional illustration techniques.

Naturally, many approaches in illustrative visualization rely on these NPR techniques. However, one of the important remaining challenges is to determine from the underlying data what kind of abstraction and




(a) (b)

Figure 1.2: Examples of hand-drawn pen-and-ink illustrations in a medical handbook. From [House and Pansky, 1960].

what level of abstraction is most suitable for the visualization goal at hand. If the goal is, for example, to merely explain well-known phenomena to lay people, one can afford more (artistic) freedom to achieve an effective communication. On the other hand, if the visualization has a more exploratory, analyzing goal it is important to make sure no false impressions are given about the data being visualized.

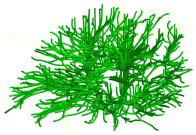
The concept and principle of abstraction plays an important role in this entire thesis. For example, one can view the shortest paths computed in Chapter 2 as abstract representations of the connections between the cortical regions in the brain. The abstraction introduced by means of the depth-dependent halo technique of Chapter 3 is a visual, more implicit kind of abstraction, whereas in Chapter 4 the aim is to achieve explicit abstraction through the bundling of DTI fiber tracts. Finally, the line styles introduced in Chapter 5 give users the means to interactively apply abstraction for the visualization of streamlines.

## 1.3 CONTRIBUTIONS & STRUCTURE OF THIS THESIS

This thesis presents four new approaches for the analysis and visualization of dense lines and underlying data. While these contributions are diverse, the key concept that binds them together is *abstraction*, as discussed in the previous section. In the rest of this section we briefly discuss the contents of the chapters that follow.

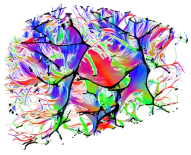
Chapter 2 introduces a new method for calculating trajectories of fiber bundles. Its main idea is to construct an undirected weighted graph from diffusion weighted magnetic resonance imaging data. In this graph, each node represents a voxel and the edge weights between

the nodes represent the white matter connectivity between the voxels. By applying Dijkstra's shortest path algorithm to this graph, we present a new method for calculating trajectories of fiber tracts as an alternative to determinstic and probabilistic tractography. In addition, we explore the visualization possibilities for the resulting tree structure of shortest paths.

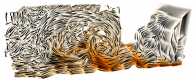


In Chapter 3, the underlying data remains the same —DWI/DTI data— but the focus changes to the visualization of fiber tracts, one of the possible derivatives of DTI data, as discussed in a previous section. More specifically, we present an illustrative visualization and rendering method for dense lines such as fiber tracts that is inspired by black-and-white illustrations. The *depth-dependent halos* technique we introduce here emphasizes collinear structures (e. g., bundles) and helps to improve depth perception without the use of shading or colors. We show that our fast GPU implementation of this technique is not only suitable for illustrative visualization of fiber tracts, but can also be applied to other types of line data.

In Chapter 4, we aim to achieve a more explicit kind of abstraction by locally bundling collinear fiber tracts. The visualization of this bundled state of fiber tracts gives insight to the structure of the white matter of the brain, in part because the resulting volumetric voids decrease the mutual occlusion of tracts. The bundling process itself consists of iteratively moving locally similar line (i. e., tract) segments towards each other. Our implementation supports an interactive and continuous transition between the original and the abstracted representations. In addition, we explore a number of options for interacting with the bundled fiber tracts.



Then, in Chapter 5 the focus returns to the rendering of lines, although the domain has changed: flow streamlines. Here we introduce a flexible illustrative line style model for the visualization of streamline data that builds upon the depth-dependent halo approach introduced in Chapter 3. In this model we partition view-oriented line strips into parallel bands whose basic visual properties can be controlled independently. The visual flexibility is further increased by supporting several mappings of local data attributes to these visual properties. We also introduce in this chapter the concept of a *line style transfer function* that maps local line attributes to line styles, creating additional possibilities for emphasis and abstraction.



Finally, Chapter 6 provides a summary and a discussion with general conclusions followed by an outlook on future work.

# 2 SHORTEST PATHS IN WHITE MATTER

ABSTRACT: *An undirected weighted graph may be constructed from diffusion weighted magnetic resonance imaging data. Every node represents a voxel and the edge weights between nodes represent the white matter connectivity between neighboring voxels. In this chapter we propose and test a new method for calculating trajectories of fiber bundles in the brain by applying Dijkstra's shortest path algorithm to the weighted graph. Subsequently, the resulting tree structure is pruned, showing the main white matter structures of the brain. The time consumption of this method is in the order of seconds.*

## 2.1 INTRODUCTION

The complexity of the human brain is enormous. In a volume of about 1.5 liter run some $10^5$ kilometers of myelinated nerve fibers, connecting many cortical brain regions. Most of these fibers are grouped into fiber bundles of various widths. Diffusion of water molecules in the longitudinal direction of these bundles is free while transverse diffusion is limited. With diffusion-weighted magnetic resonance imaging (DW-MRI) the per-voxel averaged directional diffusion of water in biological fibrous tissue can be measured, resulting in a symmetric diffusion tensor $\mathbf{D}$ [Basser et al., 1994]. For a voxel in an area with well-aligned nerve fibers the largest eigenvector of $\mathbf{D}$ points in the main fiber direction. In the last decades DW-MRI has matured to the extent that it is now possible to use this modality for detecting and visualizing fiber bundles in the millimeter range. It has been used, for example, to build atlases of the brain [Hagmann et al., 2003], to study chronic brain diseases [Rose et al., 2008], and to assess acute stroke [Lee et al., 2005]. For an overview see [Melhem et al., 2002] and [Mori and van Zijl, 2002].

In order to understand the workings of the healthy or affected brain it is important to determine to which cortical regions fiber bundles connect and which trajectories they follow. Many techniques have been proposed that, starting from the tensor field, visualize DTI data. A simple approach is to visualize the fractional anisotropy (FA), a scalar quantity representing a certain ratio of the eigenvalues of $\mathbf{D}$. It is useful for visualizing white matter density. More demanding is (deterministic) *fiber tracking*, a technique to reconstruct and visualize fiber bundles. In the most straightforward approach trajectories are generated by following the direction of the greatest local eigenvector, starting from a given voxel [Mori and van Zijl, 2002]. Instead of only using the greatest eigenvector of $\mathbf{D}$, some methods use the entire diffusion tensor [Lazar et al., 2003]. Also, level set methods have been applied [Tournier et al.,

2003], where the front propagates with a velocity depending on the eigenvectors of **D**.

What these methods have in common is that the tensor field is interpreted as a way of *locally* describing the direction of highest velocity, and in fiber tracking this direction is followed. Alternatively, we propose to approach fiber tracking as a way of finding a lowest-weight path in a graph, which is constructed by connecting each voxel to its neighboring voxels with a weighted edge. The weights are defined such that paths that follow the principal diffusion direction have a low weight. Having a weighted graph, a minimum-weight fiber tract between two given points may be calculated using Dijkstra's algorithm [Dijkstra, 1959]. In this chapter we report on our experiments with this approach, called *shortest path fiber tracking* (SPFT). Dijkstra's algorithm does not simply give a single shortest path, but, for a given source voxel, a tree of shortest paths. This allows us to not only show a single shortest path, but also to produce an overview of white matter density, structure, and direction, by visualizing a pruned version of this tree. Furthermore, we investigate a method to visualize SPFT results by clustering outer branches of the tree.

In the field of DTI visualization the SPFT method can be categorized as follows. In the first place it is a deterministic method; each run gives exactly the same result. However, it does not have the drawbacks of most other deterministic methods; paths are constructed using non-local information, yielding globally optimal paths. The paths generated by SPFT consist of edges connecting points of the voxel grid. This differs from most other methods, where paths are defined by non-grid points.

There are a few methods described in literature that also construct a graph from DW-MRI data and apply graph algorithms. In [Iturria-Medina et al., 2007] an iterative adaptation of Dijkstra's algorithm is used to determine most probable paths between voxels and to produce probabilistic brain anatomical connection maps. In a recent publication [Zalesky, 2008] a variation of Dijkstra's algorithm is used to compute optimal paths of maximum probability. The advantage of our method is its simplicity of assigning weights, and consequently its performance. On a similar dataset the method described in [Zalesky, 2008] is reported to take about 15 minutes, whereas our method runs in under 5 seconds. This makes our method suitable for interactive visualization.

The contribution of this chapter can be summarized as follows. We present and test a new fiber tracking method, SPFT, that

- gives a fast (in the order of seconds) first impression of global white matter structure
- uses global information
- can be used for clustering.

In the following section we explain how the weighted graph is constructed and how the paths are created. Section 2.3 reports on experiments performed. In Section 2.4 we discuss the advantages and disadvantages of our approach. Finally, in Section 2.5 we conclude this chapter and suggest possible future work.

## 2.2 CONSTRUCTING A WEIGHTED GRAPH AND SHORTEST PATHS FROM DTI DATA

Consider a diffusion tensor field $\mathbf{D}$ over the brain, where $\mathbf{D}_i$, $i = 1 \dots N$ is the diffusion tensor of voxel $i$ and $N$ the number of voxels. We assume that voxels are evenly spaced on a rectangular three-dimensional grid. Moreover, we assume that every diffusion tensor represents the average diffusion in the corresponding voxel, as measured by DW-MRI. In the following we use 26-connectedness, i. e., every voxel that is not at the boundary of the scanned volume has 26 neighboring voxels.

A mask is generated by using the brain extracting tool called `bet2` from the FSL package [Smith et al., 2004]. The mask is used to exclude areas without white matter, such as the skull and air surrounding the head. Also voxels in cerebrospinal fluid regions are excluded, characterized by a high value for the trace of the diffusion tensor. We used $9 \times 10^{-4}$ as the maximum trace value.

A diffusion tensor $\mathbf{D}$ represents local diffusion, which means that the local flux $\mathbf{J}$ of particles, due to diffusion, over an infinitesimal plane $A$ with unit normal $\mathbf{r}$ is given by the matrix-vector product

$$\mathbf{J} = \mathbf{D}\mathbf{r}. \tag{2.1}$$

In general $\mathbf{J}$ is not in the direction of $\mathbf{r}$. The flux $J_\mathbf{r}$ in the direction of $\mathbf{r}$ is given by

$$J_\mathbf{r} = \mathbf{r} \cdot \mathbf{D}\mathbf{r}. \tag{2.2}$$

Let $\mathbf{r}_{i,j} \equiv \mathbf{r}_j - \mathbf{r}_i$ and let $\hat{\mathbf{r}}_{i,j}$ be the corresponding unit vector, where $\mathbf{r}_i$ and $\mathbf{r}_j$ are the centers of voxels $i$ and $j$, respectively. Using (2.2), and following the notation in [Koch et al., 2002], the diffusion coefficient in voxel $i$ in the direction $\hat{\mathbf{r}}_{i,j}$ is denoted as

$$d(\mathbf{r}_{i,j}, i) \equiv \hat{\mathbf{r}}_{i,j} \cdot \mathbf{D}_i \hat{\mathbf{r}}_{i,j}. \tag{2.3}$$

According to [Koch et al., 2002] the connectedness $C$ between two neighboring voxels $i$ and $j$ can be defined as

$$C_{i,j} = \frac{d(\mathbf{r}_{i,j}, i) + d(\mathbf{r}_{j,i}, j)}{2}. \tag{2.4}$$

For SPFT it is required that a high $C_{i,j}$ value is transformed into a low edge weight $W_{i,j}$. In order to enhance the effect of high $C$-values with

respect to low values we use a nonlinear decreasing function $S$ to map connectedness to weights:

$$W_{i,j} = S(C_{i,j}). \tag{2.5}$$

In our experiments we used a decreasing sigmoidal function of the form

$$S(x) = \frac{1}{1 + e^{a(x-b)}}, \tag{2.6}$$

where $a$ is a positive constant that determines the steepness of the sigmoid and $b$ is a constant that determines the $x$-position of the steepest point of $S$.

In our experiments we used $a = 15$, but we found that any value in the range $14 \ldots 16$ provides the required steepness. The value of $b$ differs per data set and is determined as follows. First, all $C_{i,j}$ are scaled to the range $0 \ldots 1$ by dividing by the maximum of the $C$-values. Then, in the histogram of the $C$-values a value is chosen such that 98% is smaller, and $b$ is set to this. The reason for this is that outliers in the $C$-values will otherwise influence the scaling too much and almost all weights of the graph would be near 1.0.

Dijkstra's algorithm [Dijkstra, 1959] solves the single-source shortest path problem: given a weighted graph $G = (V, E)$, where $V$ is a set of vertices and $E$ a set of edges, find a shortest path from a given source vertex $s \in V$ to every vertex $v \in V$. A shortest path is defined as a path of which the sum of the weights of its edges, i. e., the path weight, is minimal. Dijkstra's algorithm returns for every vertex $v \in V$, the shortest path to $s$ and the weight of the path. The shortest paths are represented by a *shortest path tree* where each vertex has a reference to a predecessor in the shortest path from that vertex to the source vertex. Note that Dijkstra's algorithm only works for graphs with non-negative edge weights. By implementing the priority queue used in Dijkstra's algorithm with a Fibonacci heap, the complexity of the shortest path algorithm becomes $O(|V| \log |V| + |E|)$, where $|E|$ is the number of edges and $|V|$ the number of vertices [Cormen et al., 2001].

## 2.3 RESULTS

The DT-MRI data used in this chapter were acquired from a healthy volunteer on a 3T MRI system (Philips Intera). Diffusion Tensor Imaging was performed using a diffusion weighted spin-echo, echo-planar imaging technique. The DTI parameters were as follows: 240 mm × 240 mm field of view; 128 × 128 matrix size; 51 slices; $1.85 \times 1.85 \times 2$ mm$^3$ imaging resolution; 5485 ms repetition time; 74 ms echo time. Diffusion was measured along 60 non-collinear directions. For each slice and each gradient direction, two images with no diffusion weighting ($b = 0$ s/mm$^2$)

and diffusion weighting ($b = 800\,\mathrm{s/mm^2}$) were acquired, to measure APA and APP fat-shifts. The subject's consent was obtained prior to scanning. A tensor field was generated from the diffusion weighted data using the Diffusion Toolkit [Wang and Wedeen].

### 2.3.1 *Visualization*

The visualizations shown in this section were created using Python and the Visualization Toolkit (VTK). For better depth perception the paths and trees visualized are depicted using tubes with VTK's TubeFilter. In some of the images the paths are smoothed using approximating splines.

### 2.3.2 *Single Shortest Path*

A first step in visualizing SPFT results is to select one voxel in a region of interest and visualize its shortest path to the source voxel. This is illustrated in Figure 2.1, where a shortest path (red) is combined with a fiber tract produced by traditional deterministic fiber tracking (blue). The latter was created using a modified FACT [Mori et al., 1999] method provided by the `track` program of the Camino software package [Cook et al., 2006] and was seeded at the source voxel used for SPFT.

It can be seen that the SPFT path is very similar to the tract returned by the method for deterministic fiber tracking. The shortest path is not as smooth as the traditional fiber tract. This is partly because of the discretization necessary to be able to create a graph from the data. By interpolating the vertices of the shortest path by a spline the visual appearance can be improved. Note that in Figure 2.1 the shortest path is not smoothed to illustrate the jagged nature of the raw SPFT paths.

### 2.3.3 *Visualizing Brain Structure*

The next step after visualizing just one shortest path is visualizing the whole shortest path tree. This would of course create a very cluttered visualization, so instead we have experimented with visualizing a simplified, pruned tree and we propose two approaches for pruning. The first, pruning based on *tree size*, involves counting for each vertex the number of children in the shortest path tree. Then only those vertices are selected whose number of children in the shortest path tree are larger than a threshold $t_{\mathrm{size}}$. The second approach, pruning based on *tree depth*, involves calculating for each vertex the maximum depth of the tree starting from that vertex, that is, the maximum path length to a leaf vertex in the tree. That value is then used for thresholding ($t_{\mathrm{depth}}$). Both values, tree size and tree depth, are easily calculated recursively.

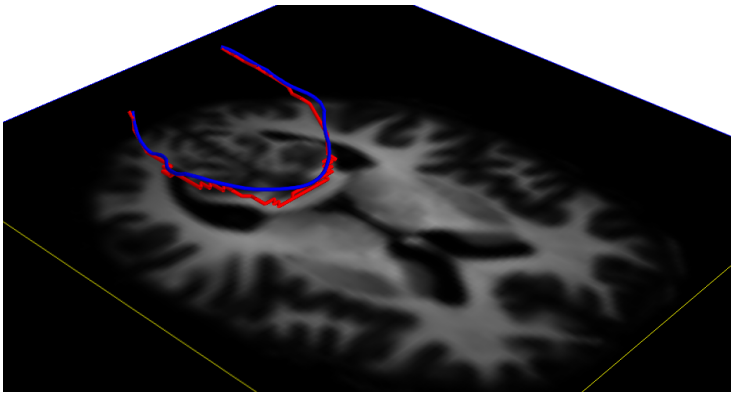


Figure 2.1: A shortest path (red) combined with a fiber tract from traditional deterministic fiber tracking (blue) seeded from the same voxel, shown together with a transverse plane showing fractional anisotropy for context. For the most part the tracts follow the same path.

Figures 2.2 to 2.6 show the results of applying pruning to the shortest path tree generated by SPFT. In these figures the pruned trees are smoothed using an approximating spline. A voxel near the brain stem is used as the source voxel. Figures 2.2 and 2.3 show the effect of the threshold parameter for both pruning based on size as well as depth. A large value (a), results in a very simple tree, and decreasing $t_{\text{size}}$ and $t_{\text{depth}}$ shows more and more detail ((b) and (c)).

In Figure 2.4 the pruned shortest path tree (size-based pruning) is shown together with image planes displaying fractional anisotropy (FA) information, where white means high FA. What we see here is that the edges of the pruned tree follow important white matter bundles to a reasonable approximation. This is also illustrated in Figure 2.5 where a transverse slice (four voxels thick) of a minimally pruned tree is combined with a plane showing FA information.

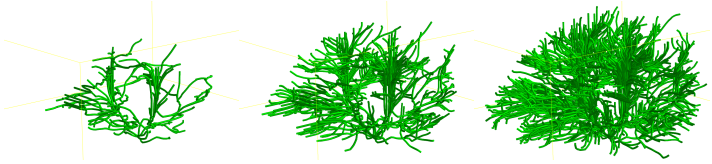
Of course the position of the source voxel influences what the resulting tree will look like. However, we found that when choosing two source voxels far apart, one near the brain stem and one in the corpus callosum, 72% of the edges of the shortest path trees match. Figure 2.6 shows what the pruned shortest path tree looks like when the source voxel is chosen in the corpus callosum.

### 2.3.4 *Voxel Clustering*

Instead of constructing paths we can look at which voxels connect to the outer voxels of the pruned tree and perform clustering based on that information. This means that we cluster voxels together whose shortest paths to the source voxel go through the same voxel. This

(a) $t_{\text{size}} = 1000$ (b) $t_{\text{size}} = 400$ (c) $t_{\text{size}} = 100$

Figure 2.2: Decreasing the threshold parameter for pruning based on tree size results in a visualization showing more detail.



(a) $t_{\text{depth}} = 25$ (b) $t_{\text{depth}} = 15$ (c) $t_{\text{depth}} = 10$

Figure 2.3: Decreasing the threshold parameter for pruning based on tree depth results in a visualization showing more detail.

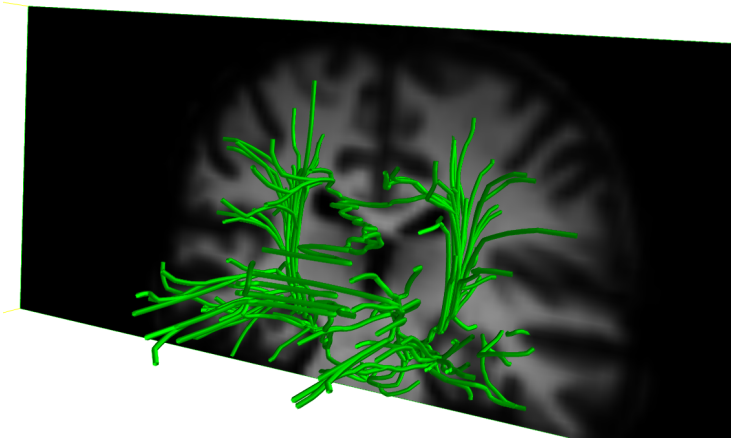


Figure 2.4: A pruned shortest path tree combined with a coronal plane showing fractional anisotropy. The tree shows the shape of part of the corpus callosum.

may give information on how certain regions are connected. Figure 2.7 shows a sagittal slice of colored clusters (a) and the same slice with FA information (b). For most regions in this image, the colored clusters match the white matter structure indicated by high FA values.

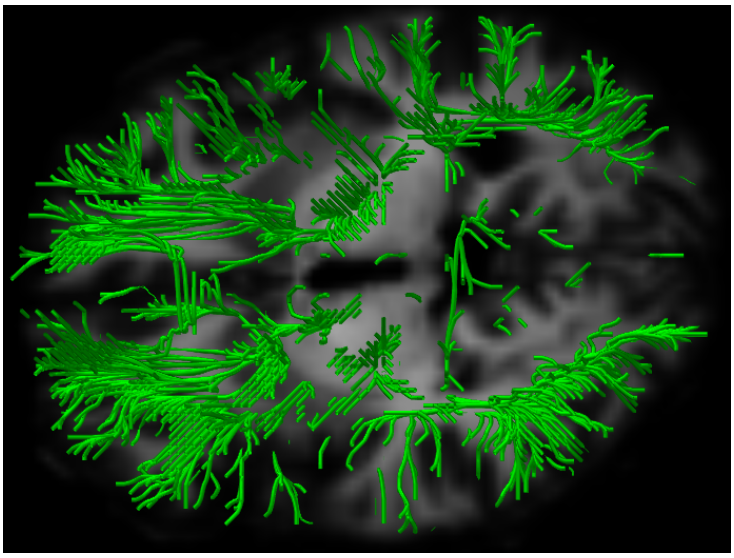
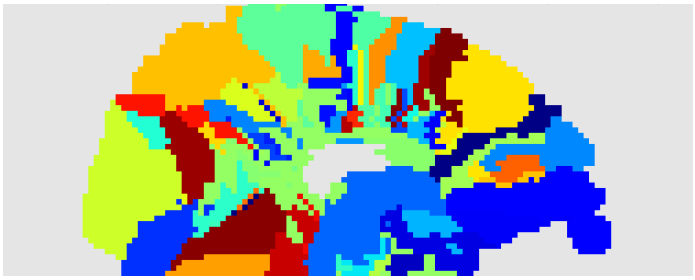


Figure 2.5: A 4-voxel thick transverse slice of a minimally pruned tree combined with a plane showing FA information.

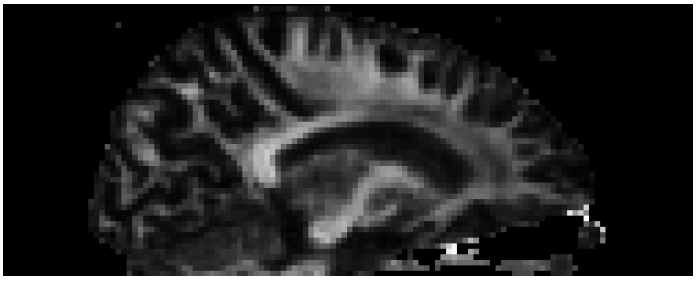
Figure 2.6: Choosing a source voxel on the corpus callosum results in a pruned shortest path tree that is similar to the pruned tree found by using a source voxel near the brain stem (see for example Figure 2.2).

### 2.3.5 *Path Weight Visualization*

Besides a shortest path tree, Dijkstra's algorithm returns for each voxel the weight of the shortest path to the source voxel, which is basically a scalar field we can visualize. Figure 2.8 shows a sagittal slice of this



(a) Clusters



(b) FA

Figure 2.7: A sagittal slice showing colored clusters and a slice showing FA information. The clusters appear to align with the white matter structure shown in the FA map.

weighted distance, colored using a red to yellow color scale. In addition we can calculate from the shortest path tree for each voxel the *length* of the path to the source voxel. Figure 2.9 shows this distance for the same sagittal slice. The neuroanatomical meaning and origin of the "parieto-occipital" red-yellow boundary in this figure is something that needs further analysis.

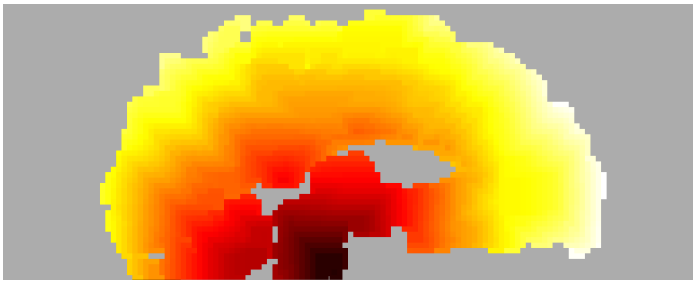


Figure 2.8: A sagittal slice showing for each voxel the weight of the shortest path to the source voxel.

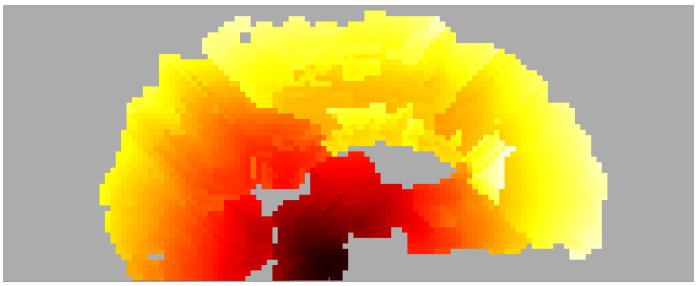


Figure 2.9: A sagittal slice showing for each voxel the length of the shortest path to the source voxel.

### 2.3.6 *Performance*

The graph construction and shortest path calculations are implemented in C, whereas the visualizations are created using a combination of Python and VTK.

Performance tests on a single core of an AMD Dual Opteron 280 (2.40 Ghz) were applied to a 128 ×128 ×51 tensor field. The graph building process, which only needs to be done once, takes on average 1.2 seconds, and the shortest path calculation takes *on average 0.37 seconds*. Pruning the shortest path tree is implemented in Python and needs about 1 to 2 seconds.

## 2.4 DISCUSSION

In our opinion SPFT should be considered as a fast fiber tracking method, both for previewing and interactive visualization. Computing time for the shortest path tree is under one second, so it is feasible to interactively change parameters to observe the effect. Even though the assignment of edge weights is simple, computing the shortest path tree already provides a good visual impression of brain structure.

The work in this chapter is of the proof-of-principle type, i.e., further verification and validation is required. Notably a comparison with other methods has to be performed and the method should be tested on both healthy and pathological cases. Also, because of the noisy nature of DTI data, sensitivity to noise is something to be investigated. Preliminary testing showed that the shortest path tree only significantly changes after adding Gaussian noise with $\sigma > 0.1$.

An important issue is the interpretation of the edge weights. Although our initial edge assignment is based on diffusion densities, subsequent rescaling makes that it is not obvious what the ensuing minimization precisely means in neurophysiological terms. A more principled edge weight assignment can be based on probabilistic methods, like Bayesian estimation or Markov Chain Monte Carlo sampling.

However, computation times of these all-paths tracking methods are much higher, ranging from 15 minutes [Zalesky, 2008], 30–40 minutes [Friman et al., 2006] to 18–24 hours [Behrens et al., 2003]. An interesting open problem is to include the probabilistic edge assignment of Zalesky [Zalesky, 2008] in our method without increasing computation time too much.

Dijkstra's algorithm gives for every voxel pair a shortest path, even for voxel pairs that are not actually connected by a nerve fiber. Probably, such a path will have one or more relatively high weights. We tried to determine which paths represent real nerve fiber bundles and which ones are fictitious by comparing the highest single-step weight of the path with the average weight of the path. So far, this did not solve the problem.

## 2.5 CONCLUSION AND FUTURE WORK

In this chapter we have presented a new fast method for analyzing and visualizing DW-MRI data based on Dijkstra's shortest path algorithm. With this method one can quickly obtain an approximate overview of important white matter bundles or view paths between regions of interest. There is however, room for improvement. First of all, the computation of the edge weights needs some more theoretical underpinning to give meaning to what is actually minimized (see section 2.4). This can then be combined with a more extensive validation of the results produced by SPFT. Also, the tensor model has some known drawbacks with regard to crossing fibers and we will try to combine other DW-MRI modalities such as Q-ball imaging [Tuch, 2004] with SPFT. Finally, our current visualizations are very simple and there might be interesting innovative ways to explore the shortest path tree returned by SPFT or variants thereof.

## ACKNOWLEDGEMENTS

We thank Cris Lanting and Pim van Dijk from the BCN NeuroImaging Center in Groningen, The Netherlands for the brain dataset used in this chapter. We also thank Alessandro Crippa for interesting discussions and test data.

This chapter is based on: Maarten H. Everts, Henk Bekker, and Jos B. T. M. Roerdink. Visualizing White Matter Structure of the Brain using Dijkstra's Algorithm. In Peter Zinterhof, Sven Lončarić, Andreas Uhl, and Alberto Carini, eds., *Proc. 6th International Symposium on Image and Signal Processing and Analysis* (ISPA 2009, September 16–18, Salzburg, Austria). Pages 575–580, 2009.

# 3

# DEPTH-DEPENDENT HALOS FOR DENSE LINE VISUALIZATION

ABSTRACT: *We present a technique for the illustrative rendering of 3D line data at interactive frame rates. We create depth-dependent halos around lines to emphasize tight line bundles while less structured lines are de-emphasized. Moreover, the depth-dependent halos combined with depth cueing via line width attenuation increase depth perception, extending techniques from sparse line rendering to the illustrative visualization of dense line data. We demonstrate how the technique can be used, in particular, for illustrating DTI fiber tracts but also show examples from gas and fluid flow simulations and mathematics as well as describe how the technique extends to point data. We report on an informal evaluation of the illustrative DTI fiber tract visualizations with domain experts in neurosurgery and tractography who commented positively about the results and suggested a number of directions for future work.*

## 3.1 INTRODUCTION

Illustrative depictions have been playing an essential role in the communication of knowledge for centuries. Traditionally, illustrators used graphic tools such as pen-and-ink to draw—with the goal to depict, e. g., the shapes of objects. The choice of tool was typically dictated by the means of reproduction, usually the printing in books. Therefore, illustrators often chose techniques that result in black-and-white imagery, for example pen-and-ink, woodcuts, or copper plates. Despite being limited to two "colors," these techniques still allowed illustrators to convey shape, material, and illumination through techniques such as stippling, hatching, or cross-hatching. More importantly, illustrators also made use of the fundamental illustration principles of *abstraction* and *emphasis* to effectively communicate their intentions.

With the advance of computer support, illustrators started to use general purpose graphics programs (e. g., Adobe's Illustrator™ or Photoshop™) for creating illustrations. One reason for this tool change is that general purpose programs, in many aspects, provide more freedom than traditional tools. In a separate development, the visualization community has created numerous successful techniques to solve specialized visualization problems, e. g., in the medical domain. In both cases, the availability of color processing and reproduction has invited the use of shading techniques, i. e., representing surfaces through shades of color, in contrast to the traditional black-and-white methods.

While illustrators in their use of general purpose tools can still apply the illustration principles of abstraction and emphasis, this is more difficult for automatic techniques as it is challenging to "teach" importance

to an algorithm. In the areas of non-photorealistic rendering (NPR) and illustrative visualization, however, abstraction and emphasis techniques have been investigated. Examples include the use of halos for simple line rendering [Appel et al., 1979; Elber, 1995] or shading [Luft et al., 2006; Tarini et al., 2006] and volume rendering [Bruckner and Gröller, 2007], the use of additional depth cueing by influencing line or shading parameters (e. g., [Elber, 1995]), interactive emphasis techniques (e. g., [Neumann et al., 2007; Strothotte et al., 1994; Winkenbach and Salesin, 1994]), or focus+context techniques (e. g., [Hauser et al., 2001; Viola et al., 2004]).

In this chapter we build on these previous approaches but focus on a specific subset of data—line datasets (see the example result in Figure 3.1). This type of data is generated in a number of application domains such as medical imaging (e. g., DTI fiber tract extraction), meteorology (e. g., particle traces in storm data or simulations), physics (e. g., particle tracts from 3D gas or fluid flow simulations), or astronomy (e. g., particle traces from mass distribution simulations for galaxy formation). In all these application areas, line data with a comparably high density of elements is generated and needs to be analyzed. This data lends itself more to the traditional black-and-white depiction techniques rather than shading-based methods because lines occupy much less space than shaded elements such as cylindrical shapes. In addition, detail in the visualizations is often important so that reducing the number of depicted lines may not be a suitable approach.

To address this problem of depicting dense line datasets in their full detail this chapter makes the following contributions: We show how to illustratively visualize dense line datasets at interactive frame-rates using modern graphics hardware. We introduce a conceptually simple technique that allows us to only render the front layer of the data, i. e., the lines or points that lie close to each other and closest to the viewer. These front elements are rendered such that they do not overlap each other but at the same time they occlude elements much further away from the viewer. This *depth-dependent halo* technique emphasizes bundles of co-linear line segments (which are likely to be important, see Figure 3.1) and abstracts from less structured segments. Moreover, we de-emphasize elements that are farther away to enhance depth perception, and filter the dataset for further emphasis of important structures. We further show how the discrete and black-and-white nature of the depicted elements lends itself to anaglyphic stereo rendering.

The remainder of the chapter is organized as follows. In Section 3.2 we place our work in the context of related approaches. Next, we present the technical details of the approach in Section 3.3. Then, in Section 3.4, we show examples of visualizations created with our technique for several application domains. In Section 3.5 we give details on an informal evaluation of our technique with domain experts in neurosurgery and

tractography and, finally, conclude this chapter in Section 3.6, where we also mention some avenues for future work.

## 3.2 RELATED WORK

To place our work on illustrative line rendering into context, we discuss techniques for both line visualization and illustrative rendering.

### 3.2.1 *Line Data Visualization Techniques*

Numerous techniques exist to depict paths of particles or other linear structures, in particular for flow visualization. For example, people have employed (shaded) lines, tubes, or strips (ribbons) whose color and shape can be changed depending on data properties such as velocity, flux, or direction [Post et al., 2002] (e. g., Figure 3.2a). Particularly related to our work are scalable, self-orienting surface techniques [Ma et al., 2002; Melek et al., 2006; Schussman and Ma, 2002; Stoll et al., 2005] which create shaded, view-aligned strips to visualize 3D vector fields. In contrast, our goal is to create high-resolution black-and-white visualizations for dense datasets using illustration principles. As an alternative to explicitly representing streamlines, texture-based methods [Laramee et al., 2004] such as line integral convolution [Cabral and Leedom, 1993] provide both a global and local impression of flow data. This technique was extended to 3D [Helgeland and Andreassen, 2004; Interrante and Grosch, 1998] where, related to our work, halos are used to increase depth perception. For a similar purpose, halos are used in streamline-based volume visualization [Wenger et al., 2004]. Both streamlines and texture-based techniques are employed in many other domains which rely on the visualization of line data resulting from real or simulated linear structures or particle traces. Examples include physics (e. g., electric or magnetic field lines), chemistry (e. g., protein structures), and meteorology (e. g., storm data).

In particular in the medical domain, line data such as tracts of muscle or brain fibers is important. Here, fiber tracts are estimated from diffusion weighted magnetic resonance imaging (DW-MRI) [Mori and van Zijl, 2002]. The fiber tracts represent, for example, bundles of neural axons connecting different parts of the brain. Such fiber tracts are typically rendered as lines or tubes with coloring or shading applied to them to enhance understanding of spatial relationships [Petrovic et al., 2007; Zhukov and Barr, 2004; Zöckler et al., 1996].

Rendering all possible fiber tracts or particle traces in a dataset is often not feasible with visualization techniques due to performance and occlusion issues. Thus, a selection is typically made, showing only fiber tracts that pass through a certain region of interest. Various methods are available for making this selection (e. g., [Blaas et al., 2005]).

Figure 3.1: Illustrative visualization of a subset of DTI fiber tracts with depth-dependent halos.

### 3.2.2 *Illustrative Visualization and Rendering Techniques*

In a number of line-based scientific visualization techniques, people use illustration principles. For instance, Joshi et al. [2009] employ techniques that enhance the boundary or silhouette to accentuate internal features in visualizations of hurricanes. Similar contour enhancing techniques have also been investigated for flow data [Svakhine et al., 2005] or medical volume data [Burns et al., 2005; Ebert and Rheingans, 2000]. Related to illustrative visualization is non-photorealistic rendering (NPR), where lines have been used as a means to illustrate surfaces (e. g., [Hertzmann and Zorin, 2000; Zander et al., 2004]) but are usually placed onto surfaces during rendering rather than being the original carrier of the depicted data or shape. Such techniques have been applied, for example, to medical volume visualization [Dong et al., 2003; Nagy et al., 2002; Treavett and Chen, 2000] as well as rendering of fiber and vessel structures [Klein et al., 2006; Ritter et al., 2006]. In the latter examples, line rendering techniques are used to enhance and supplement traditional techniques that are based on larger cylindric structures.

Techniques that enhance depth perception are important specifically in line rendering and have been introduced to NPR in its early days. Such techniques include the use of visibility information [Appel, 1967; Kaplan, 2007] as well as the use of halos [Appel et al., 1979; Elber, 1995], the illustration method we also use in our own work. Halos, however, can not only be used in line rendering but have been applied in more traditional visualization techniques based, e. g., on line integral convolution [Interrante and Grosch, 1998] or volume rendering [Bruckner

and Gröller, 2007] to enhance depth perception. Related to these approaches as well as to our own are techniques that use depth buffer manipulations to enhance the depth perception in the created images. Noteworthy in this respect are, in particular, depth buffer unsharp masking [Luft et al., 2006], depth cueing in molecular visualization [Tarini et al., 2006], and pen-and-ink tree rendering using depth discontinuities [Deussen and Strothotte, 2000]. In contrast to these techniques, our approach extends similar illustrative rendering principles to the domain of dense line or point data which are rendered without shading and which, thus, rely even more on cues to indicate depth relations.

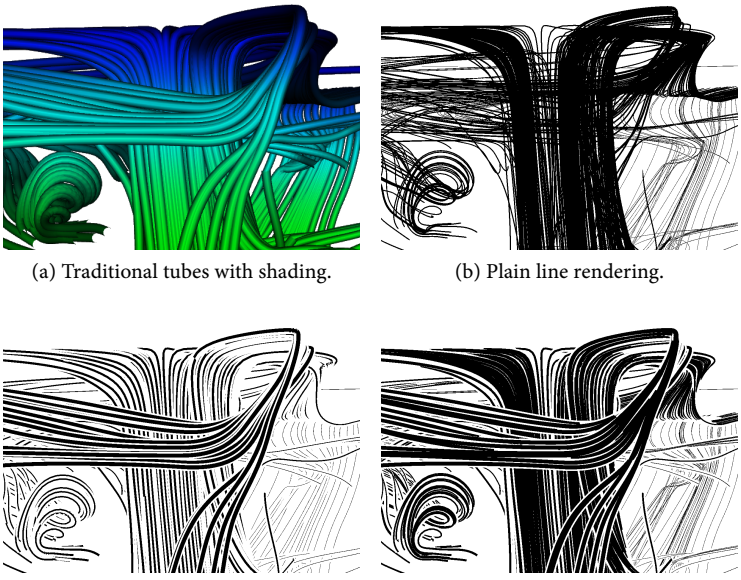
## 3.3 ILLUSTRATIVE 3D LINE RENDERING

Based on the previously discussed techniques to visualize line data, we combine these with principles of halo-based non-photorealistic rendering of lines. We focus on datasets where dense sets of lines are important, for example, DTI fiber tracts or particle traces in physical simulations. In this section we first give a general motivation and overview of the technique and then discuss its realization in detail. Next, we show how the technique is extended to point clouds, present a number of visual enhancements, and address data filtering.

### 3.3.1 *General Motivation and Technique Overview*

The rendering of line data requires, in particular, that the depth relation of the lines is clearly depicted. As discussed in the previous section, shaded cylindrical representations were traditionally employed for this purpose. This approach has a number of limitations. Due to the use of shading, each line needs to have a certain minimum width in order for viewers to be able to discriminate each individual line's orientation and location in space (Figure 3.2a). This limits the number of lines that can simultaneously be depicted and also hinders the visualization of natural line bundles if each line is to be visible individually. An alternative approach would be to use simple black lines on a white background (Figure 3.2b). On the one hand, a higher number of data elements can potentially be depicted because lines can easily be packed more tightly. Moreover, it also becomes more feasible to use such illustrations in print, because these do not require shading and thus do not rely on halftoning. On the other hand, this introduces a lot of visual clutter into the image and results in large regions of black being shown. Thus, it is no longer possible to distinguish foreground from background lines, which also means that the depth relation is lost.

To address these issues and to be able to use the advantages from both techniques, we employ line halos as previously used in line rendering [Appel et al., 1979; Elber, 1995]. Since we are dealing with dense datasets



(a) Traditional tubes with shading.

(b) Plain line rendering.

(c) All lines with halos.

(d) Lines with illustrative halos.

Figure 3.2: Comparison of rendering techniques for line datasets.

of lines, simply assigning a halo to each line is not sufficient (Figure 3.2c). Instead, we present a technique that assigns halos to all lines, but these halos are only rendered if the lines are sufficiently separated in depth (Figure 3.2d). This causes the halos of lines that lie in front of others to occlude lines further away. If lines have the same distance to the viewer, however, they do not occlude one another. This *depth-dependent halo* reduces visual clutter, emphasizes line bundles by visually clustering them, and depicts depth relations as in previous halo techniques [Appel et al., 1979; Bruckner and Gröller, 2007; Elber, 1995], resulting in an effective illustration of the data.

The general approach is to use view-aligned triangle strips. Each strip represents one of the lines and always faces the viewer (similar to billboards). Strips are textured so that the center is black, representing the line, and the perimeter is white to create the halo. In addition, each strip is bent away from the viewer as shown in Figure 3.3a. This way the part of the strip that is not black prohibits parts of other lines to be drawn that are close in image space but further away from the viewer in depth. This approach is related to the $\epsilon$-$z$-buffering used in point-based rendering [Botsch and Kobbelt, 2003; Gross and Pfister, 2007] that uses fragment-dependent depth corrections to determine the visibility of splats in a two-pass process. Our approach, however, makes use of

fragment depth manipulations in a single rendering pass to directly render lines that are close together without overlapping halos.

In practice, our approach for rendering 3D lines is a two-stage process. In the first stage we transform the lines into view-oriented triangle strips, while in the second stage we manipulate the shape of the strip and texture it. These two stages are mapped to the two stages in modern GPU processing: vertex shading and fragment shading.

### 3.3.2 *View-Oriented Triangle Strips*

Our goal is to display the line data as view-aligned strips that represent both the line itself as well as the halo around it. Therefore, before sending data to the GPU, we organize our input lines in the CPU as sequences of 3D vertices. To be able to later render the lines as triangle strips on the GPU, we create zero-width line strips on the CPU by duplicating each vertex but retaining the vertex locations. On the GPU, this strip needs to be widened by extending it perpendicular both to the viewing direction and the line direction so that it is always oriented to face the viewer. We derive the direction of the line locally at each of the vertices by taking the normalized average direction of both line segments adjacent to the vertex (or one segment for start and end of each line). This step occurs before the vertices are duplicated, and the direction is copied to the new vertex when the duplication is carried out. As a final preprocessing step, each of the vertices is assigned texture coordinates ($u$, $v$). For this purpose, the $u$-coordinate is interpolated along the length of the line, while the $v$-coordinate is set to 1 for the "left" side of the strip and to 0 for its "right" side. As a result, each vertex now has a position, texture coordinates, and a direction. This information in the form of zero-width triangle strips is transferred to the GPU as vertex buffer objects.

During the GPU rendering stage (once for each rendering pass), the strip is widened and view-aligned. For this purpose we extend the zero-width strip into a direction that is perpendicular to both the direction of the line $\mathbf{D}$ and the view direction $\mathbf{V}$. Thus, we compute the cross-product between $\mathbf{V}$ and $\mathbf{D}$, normalize the resulting vector, and move a vertex along this direction if $v = 1$ and in opposite direction if $v = 0$. Hence, the new vertex position $p_{\text{out}}$ is calculated as:

$$p_{\text{out}} = p_{\text{in}} + \mathbf{V} \times \mathbf{D}(v - 0.5)w_{\text{strip}}, \tag{3.1}$$

where $p_{\text{in}}$ is the input vertex position and $w_{\text{strip}}$ is the strip width. The result is that the strip always faces the viewer, and the centerline of the strip is located along the original line.

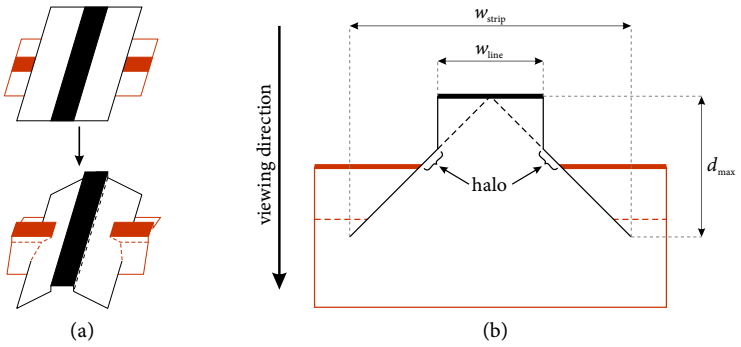



Figure 3.3: Schematic depiction of the rendering of the creation of depth-dependent halos for large depth discontinuities: (a) projected view of the depth displacement, (b) view of a line (black) being crossed by a perpendicular line (red) creating a halo. The fat parts are the actually rendered line pixels, the thin lines illustrate the $z$-buffer manipulation.

### 3.3.3 *Fragment Texturing and Depth Displacement*

The next step in the process is to assign either black or white to the individual pixels of the line strip so that both line and halo are created, but without the halo obstructing nearby lines. This occurs in the fragment shader after the previously created line strip has been rasterized. We first determine the distance ($s$) to the center of the strip for each fragment, again using the texture coordinates:

$$s = w_{\text{strip}}|v - 0.5|. \tag{3.2}$$

If this distance $s$ is smaller than half the line width ($w_{\text{line}}$), the fragment's output color is set to black and its depth value is left untouched. Otherwise, the fragment's output color is set to white and its depth is adjusted depending on the distance from the strip's center (Figure 3.3a):

$$d_{\text{new}} = d_{\text{old}} + d_{\text{max}} f_{\text{displacement}}(2|v - 0.5|). \tag{3.3}$$

Here, $d_{\text{new}}$ is the fragment's new depth, $d_{\text{old}}$ is the old depth, $d_{\text{max}}$ is the maximum displacement, and $f_{\text{displacement}}$ is a function that maps a scalar value $x \in [0,1]$ to $[0,1]$, representing the specific shape of depth displacement. A simple linear function has proven to be suitable, and we use $f_{\text{displacement}}(x) = x$ for all our examples (except Figure 3.17).

The effect of this depth displacement of fragments is illustrated in Figure 3.3b which shows one line (black) being rendered on top of another one (red). Because depth testing is enabled in the rendering, parts of the red crossing line are obscured by white fragments of the

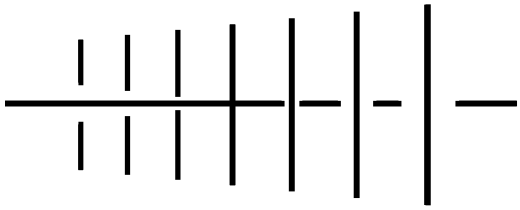


Figure 3.4: Illustration of how the line halos change depending on the distance of the lines with respect to each other.

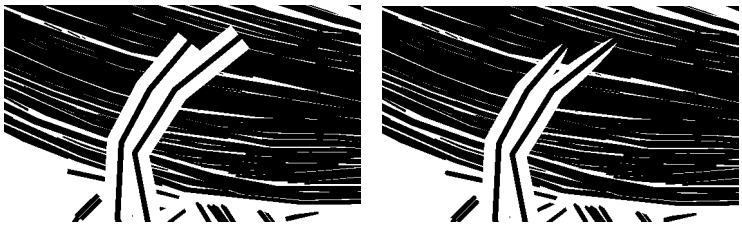
triangle strip belonging to the black foreground line. The visual effect is a halo around the black foreground line.

If the red line in Figure 3.3b were to move further back, the width of the halo would increase until the difference in depth between both lines is larger than $d_{\max}$, after which the halo width remains constant. Moving the background line toward the foreground line, in contrast, would decrease the halo width until the halo completely disappears. This happens if the background line is closer to the foreground line than $f_{\text{displacement}}(0.5w_{\text{line}})$. This effect is illustrated in Figure 3.4 in which a series of vertical lines of decreasing distance to the viewer are rendered with respect to a horizontal line. Notice how the changing halo widths enhance the depth perception in this case.

### 3.3.4 *Visual Enhancements*

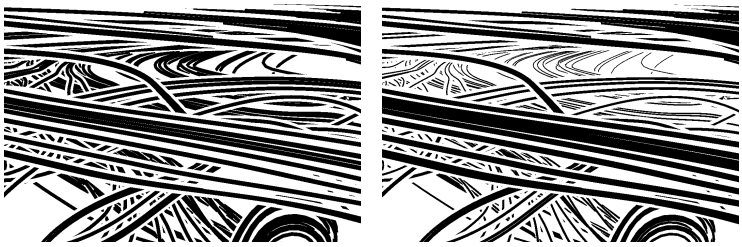
So far the technique correctly represents haloed lines that do not occlude each other if they lie close together. For an individual line that lies clearly in front of a bundle of other lines this has the effect of showing the rectangular strip—visible, in particular, at the line ends (Figure 3.5a)—which can be distracting. We improve the visual appearance of the lines by tapering (gradually narrowing line ends, Figure 3.5b). We place a stencil texture over the strip to be checked in the fragment shader which only lets fragments pass that are inside the mask. To make the amount of tapering independent from the line length, the $u$ texture coordinate is, in fact, interpolated non-linearly along each triangle strip. For the first $n_{\text{tapered}}$ vertices the $u$-coordinate is linearly interpolated between zero and $t_{\text{tapered}}$, assuming they are approximately equidistant. Similarly, the last $n_{\text{tapered}}$ vertices are mapped to $u$-values between $1 - t_{\text{tapered}}$ and one. We use $n_{\text{tapered}} = 2$ and $t_{\text{tapered}} = 0.2$.

One important aspect of visualization of 3D data is to correctly display spatial relationships. While the depth-dependent halo rendering already supports depth perception (Figure 3.6a), we further enhance this effect by employing depth cueing via adjusting the line width [Elber, 1995]. The perspective projection that we use already introduces some foreshortening of the line strips with growing distance from the



(a) (b)

Figure 3.5: Comparing rendering (a) without and (b) with tapering.



(a) (b)

Figure 3.6: Rendering (a) without and (b) with additional depth cueing.

viewer. We further emphasize this effect by manipulating the portion of the strip that is rendered as *line*, reducing it the further away a line section lies from the viewer (Figure 3.6b). This is realized in the fragment shader by modifying $w_{\text{line}}$ with respect to the current fragment's $z$-value. The degree of depth cueing can be adjusted w.r.t. the desired effect.

Finally, the visual quality of a line rendering depends to a large degree on the way the resulting image is represented. While using vector graphics could be advisable for the line renderings we are producing [Isenberg et al., 2005], we can also achieve similar results by rendering high-resolution 1-bit black-and-white images as used throughout this chapter. For on-screen rendering we either use a gradual transition from black to white in the fragment shader or employ full-scene anti-aliasing (FSAA). For example, Coverage Sampling Anti-Aliasing (CSAA) can be enabled to produce higher-quality images with less sampling artifacts.

### 3.3.5 *Extension to Point Cloud Data*

In addition to rendering line data, the technique can also be extended to visualize point clouds. This approach relates to point-based graphics

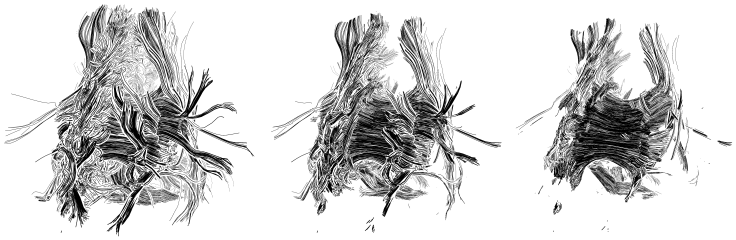
[Gross and Pfister, 2007] such as point splatting (e. g., [Botsch and Kobbelt, 2003]) or surfels [Pfister et al., 2000], the latter having previously been used in NPR [Schmidt et al., 2007]. To add explicit halos around points we render view-aligned geometry, i. e., quads, similar as before with lines. Thus, we quadruple each vertex during pre-processing but do not need to extract a direction. Instead, each zero-sized quad is oriented to the viewer in the vertex shader by extending it in the positive and negative $x$- and $y$-direction in screen-space. The actual size of the quad is determined in model-space, however, to guarantee a consistent behavior when zooming in and out.

For this purpose we back-project an up-vector (positive $y$-direction) and side-vector (positive $x$-direction) in screen-space to model-space, which are then used as basis-vectors for enlarging the quads. Again, with the texture coordinates of the zero-sized quad we determine the precise transformation for each vertex. In the fragment shader, the depth-displacement occurs similar to the case of lines, resulting in a cone shape being rendered to the $z$-buffer. The distance from the center is computed using the Euclidean distance, the same is also used for masking the halo to a circular shape. The overall processing is slightly more computationally and memory intensive than with lines, due to the quadrupling of points and the use of the Euclidean distance.

### 3.3.6 *Filtering*

As another means of illustrative abstraction (and in addition to selecting a subset of lines through a region of interest (ROI)) we implemented filtering of the data to remove selected parts, e. g., low velocity streamlines or fiber tracts in areas with low fractional anisotropy (FA). For this purpose we assign an extra scalar attribute to each vertex of the data based on which the filtering will occur. This filtering attribute is passed onto the GPU and replicated for each fragment in the rasterization stage. The fragment shader then compares each fragment's filtering attribute with a pre-determined threshold and discards fragments that do not pass this test. The threshold can now be changed at run-time and permits an interactive selection of the amount of filtering that is to occur.

Unfortunately, this process re-introduces the problem of the rectangular shape of line ends discussed in Section 3.3.4. To address it, we change the process of halo masking when filtering is enabled. Instead of using the actual interpolated $u$ texture coordinate we derive a new $u$-value from the interpolated filtering attribute and the currently active threshold such that the ends of the filtered lines are tapered. Limitations are that this assumes the filter attribute changes gradually and that it does not result in the same visual quality as before.



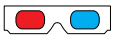
(a) (b) (c)

Figure 3.7: Three example stages of filtering, using the fractional anisotropy (FA) value of a DTI fiber tract dataset. With the growing filtering threshold for FA, more of the internal structure of the dataset is revealed.

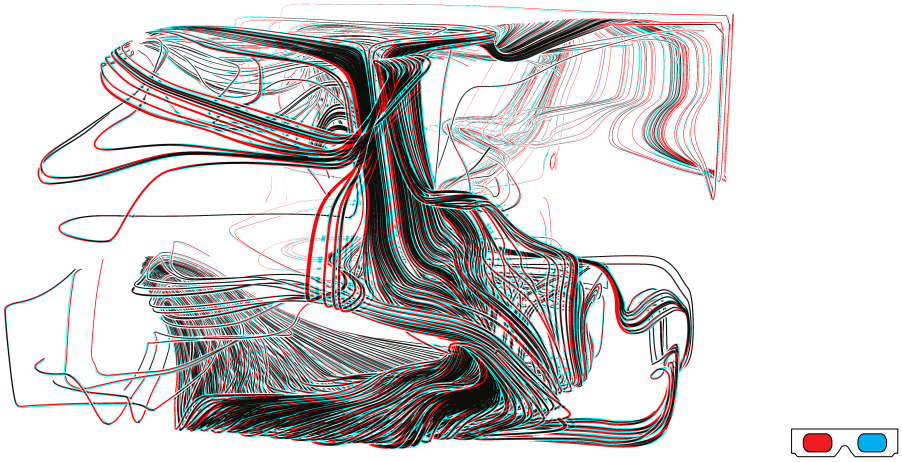
The scalar attribute used for filtering needs to be meaningful with respect to the specific dataset. For example, for DTI fiber tract data we use fractional anisotropy, which is a measure for the amount of directionality in the data (Figure 3.7). Since this value is defined based on a volume representation, it is interpolated for each vertex in the pre-processing stage based on the vertex' location in the volume.

### 3.3.7 *Anaglyphic 3D Rendering*

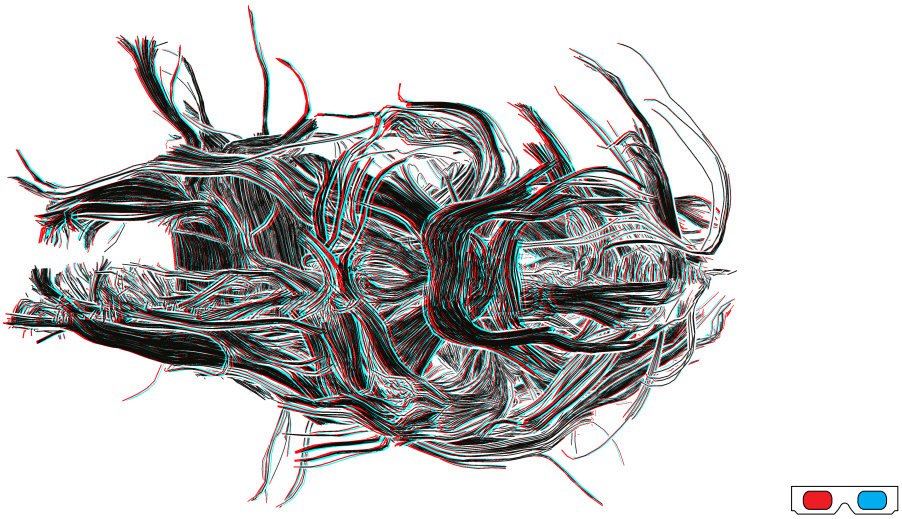
The three-dimensionality of the data is one important aspect that needs to be visualized, even beyond the support of halos. In regular rendering for visualization this is achieved through regular shading, potentially combined with special techniques such as depth buffer unsharp masking [Luft et al., 2006] or ambient occlusion [Tarini et al., 2006]. Alternatively, stereo rendering and projection can be used, i. e., computing separate images for each of the viewer's eyes. The black-and-white character of our visualization technique does not permit using shading-based techniques, but lends itself to stereo rendering without requiring complicated projection setups: anaglyphic rendering. For this purpose we render the scene from two different viewpoints, color these red and cyan, and overlay them on top of each other for use with red-cyan glasses (Figure 3.8).



The illustrative line rendering technique lends itself, in particular, to this stereo vision technique because it is monochrome and the discrete elements (lines and points) allow the human visual system to make an easy association between related elements. The halos around lines and line bundles enhance this effect because it makes the separation of individual elements that belong to each other easier. In addition to anaglyphic rendering, we also applied the technique to passive stereo



(a)



(b)

Figure 3.8: Anaglyphic stereo visualizations for two example datasets: (a) simulated air flow in an office and (b) DTI fiber tracts. For use with red-cyan or red-green glasses (red on the left eye).

rendering using polarized light, resulting in a comparable experience but without the small color artifacts caused by the red-cyan glasses.

## 3.4 DISCUSSION

After having described the illustrative line rendering technique and its implementation in detail, we now discuss some application aspects.

(a) Emphasis of dense bundles. (b) Abstraction for non-aligned lines.

Figure 3.9: Illustration principles at work. Detail regions from Figure 3.1.

In particular, we address how depth-attenuated halos incorporate the illustration principles of abstraction and emphasis, how specific visual results can be achieved, and how the technique can be applied to a number of application scenarios and input data types.

### 3.4.1 *Illustration Principles in Depth-Attenuated Halos*

Our illustrative rendering technique for lines has the effect that no or only small halos are created between lines that form concentrated bundles (Figure 3.9a). This effect emphasizes line bundles visually, through larger dark regions or tightly packed co-linear lines and through separation from the background. This emphasis may make the individual lines less distinguishable, but supports and highlights the importance and the coherency of such dense line bundles. Lines that do not form concentrated bundles (which are less aligned), in contrast, are visually de-emphasized and abstracted (Figure 3.9b): For crossing lines that are not at the same distance from the viewer many halos are generated, splitting these lines up into smaller segments. These emphasis and abstraction effects are enhanced further through the usage of additional depth cueing via line width attenuation, emphasizing front parts and de-emphasizing distant parts of the scene. These methods allow us to refrain from using any lighting while still illustrating spatial relations.

The illustrative line rendering technique using depth-attenuated halos has the additional effect that lines behind tight bundles are visually separated due to the halo that is rendered around the bundle. Thus, the technique renders the line data as implicit layers that are visually separated from each other through their halos. Occlusion situations are clearly visible through the halos surrounding the front line bundles, leading to a *visual depth clustering* similar to the halos in [Tarini et al., 2006]. Moreover, this effect also means that for bundles or other co-linear lines only the front-most layer of lines are displayed in the resulting images—visible, in particular, in videos, during interactive manipulation, or when viewing anaglyphic stereo images (Figure 3.8).

Therefore, despite the fact that no surfaces exist explicitly in the data, the rendering has the interesting effect that surfaces that implicitly exist in the data by means of densely packed bundles are visually noticeable.

### 3.4.2 *Case Studies of Application Scenarios*

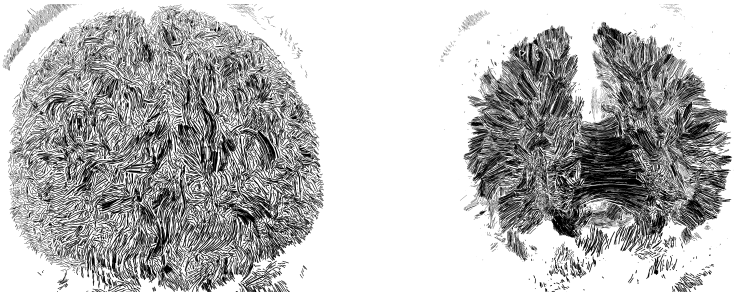
To illustrate the applicability of our technique we now discuss a number of case studies, using line datasets from a variety of domains.

#### 3.4.2.1 *DTI Fiber Tracts*

Nerve fiber tracts extracted from diffusion tensor imaging (DTI) give an indication of how actual bundles of axons connect different parts of the brain. Depending on the resolution of the underlying MRI scan, a large number of fibers can be extracted. In our example in Figure 3.10, 150 352 tracts with 1 625 472 vertices in total were extracted using the program Diffusion Toolkit, the small average vertex number per tract resulting from many short tracts. Because of such large dataset sizes, fibers are difficult to visualize with traditional techniques due to performance, rendering technique (shaded cylinders need a certain amount of space), and overview/occlusion issues. Hence, usually subsets of the fiber tracts are selected and visualized. Our illustrative line rendering technique can easily render whole datasets at interactive rates but also suffers from overview/occlusion issues (Figure 3.10a). Thus, we also select subsets (e. g., Figure 3.1 and 3.8b, where the ROI is a sagittal slice) and/or use filtering to cope with this type of data (e. g., Figure 3.7 and 3.10b). Nevertheless, our technique is able to display all fibers in a subset so that no automated data reduction technique is necessary that would cluster several lines into single ones. With our technique it is possible to *visually* distinguish single fibers from smaller or larger fiber bundles rather than requiring algorithmic support for this task.

#### 3.4.2.2 *Simulated Flows of Fluids or Gases*

Another domain where line data is generated is the simulation of fluids or gases. For example, we used VTK's "office" example dataset and extracted 786 streamlines from it using VTK (Figure 3.8a and 3.11). The visualization in Figure 3.11 shows where the simulated air flow is focused and where it branches off, highlighting concentrated bundles of air movement. Figure 3.12 shows two views of a similar simulation of water flow using 1 400 streamlines and 2 603 605 vertices in total. Here, the twisted flow of the water in a number of vortices is illustrated, while focused flow is still emphasized. The combination of these illustrative effects gives the illustrations a vividly three-dimensional appearance. Figure 3.13 shows two additional illustrative renderings of the same data (without and with filtering) but using a much higher number of stream-



(a) Full dataset. (b) Filtered dataset.

Figure 3.10: Fiber tracts from diffusion tensor imaging (DTI). Notice the visual bundling and depth clustering in (b).

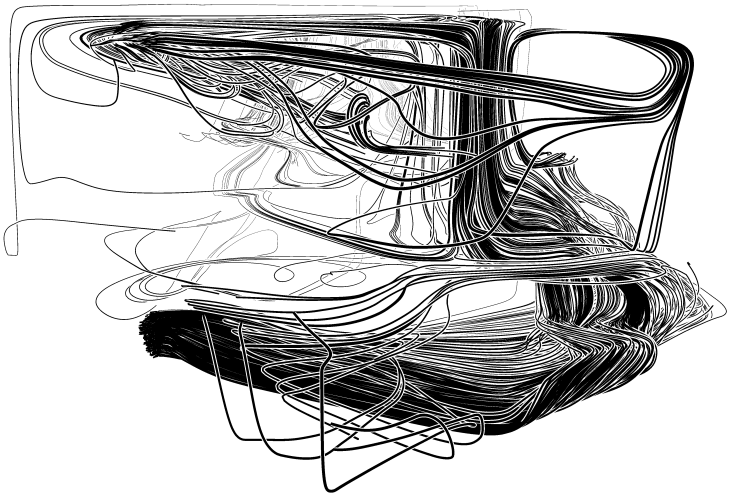


Figure 3.11: Simulated air flow in an office, same dataset as in Figure 3.8a.

lines (7 910 911 vertices and 4 475 streamlines), thus demonstrating how filtering on velocity can reveal inner structures.

#### 3.4.2.3 *Point Data: Elevation and 3D Scanning*

The extension to point cloud data lets us explore rendering datasets such as elevation scans from radar measurements or 3D object scans. Figure 3.14 shows an example of the former. The figure shows how points that are close together do not generate halos (planar areas), while for regions where points are further apart along the view direction more halos and, consequently, brighter shades are generated (e. g., note the trees in the foreground). Rows of trees are emphasized with a completely

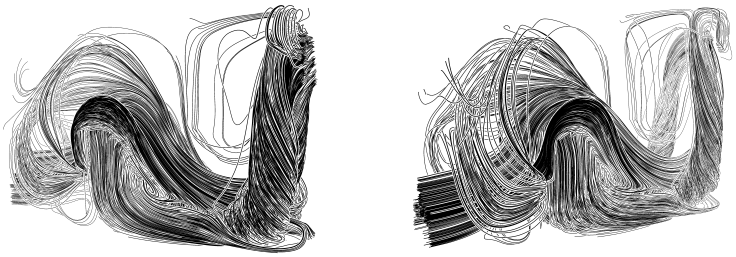


Figure 3.12: Two views of a selection of simulated water flow streamlines.

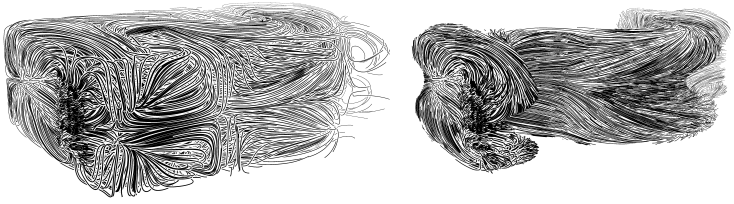


Figure 3.13: Illustrative line rendering of a denser set of streamlines of the data used in Figure 3.12, without and with filtering on velocity magnitude to reveal inner structures, showing the detail our technique can reveal.

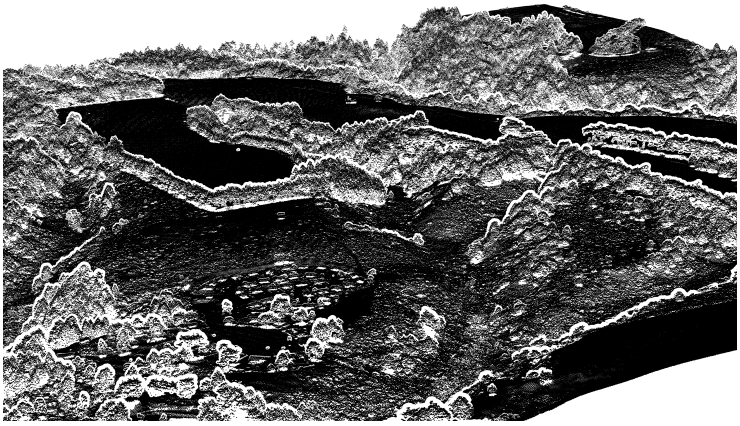
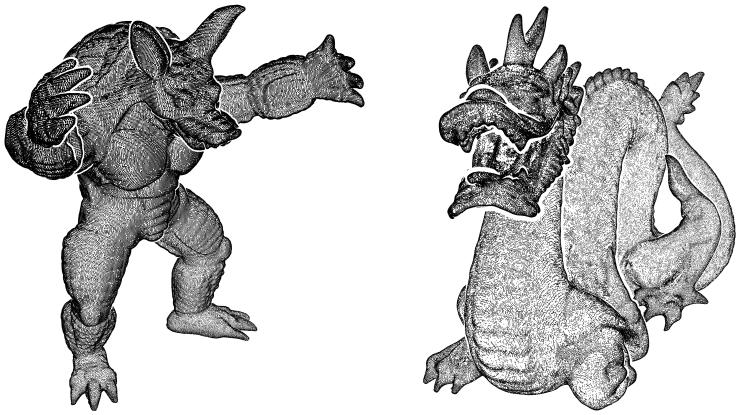


Figure 3.14: Visualization of point-based elevation data ($\approx 4.4 \cdot 10^6$ points; data obtained from http://www.OpenTopography.org/).

empty halo around them. Figure 3.15 shows how the technique can be applied to rendering "normal" 3D shapes that are represented by points on their surface. Notice here as well the effect of visually separating contiguous regions of points at depth discontinuities.



(a) (b)

Figure 3.15: Point datasets representing the surface of 3D shapes.

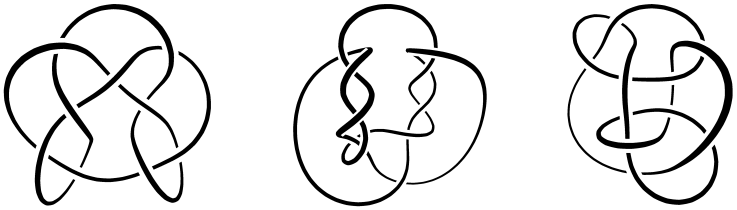


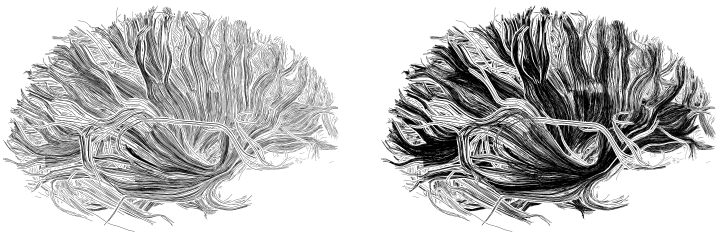
Figure 3.16: Visualizations of simple mathematical shapes from knot theory.

#### 3.4.2.4 *Mathematical Shapes*

As a last example, Figure 3.16 shows a selection of much simpler and less dense datasets from knot theory. This illustrates that our method can also handle simple shapes with similar results as in previous work [Elber, 1995].

### 3.4.3 *Parametrization*

A good set of parameters for a dataset depends to a certain degree on the specific dataset. For the illustrations shown in this chapter we chose $w_{\text{line}}$ and $w_{\text{strip}}$ such that $w_{\text{line}}$ is between 4 and 8 times as large as $w_{\text{strip}}$, for point data the factor was between 8 and 27. The size of $w_{\text{line}}$ also depends on the specific data and needs to be smaller for datasets with more elements. The maximum displacement, $d_{\text{max}}$, is set to 0.01 for all illustrations in this paper. Setting $d_{\text{max}}$ to zero has the effect of giving all lines a halo, potentially blocking other lines, see Figure 3.2c. In general, once a good setup for these parameters is found for a given



(a) $f_{\text{displacement}}(x) = x^2$ (b) $f_{\text{displacement}}(x) = \sqrt{x}$

Figure 3.17: The effect of using nonlinear displacement functions.

dataset, they do not need to be changed for interactive exploration and filtering or when visualizing lines resulting from a different ROI.

Unless otherwise indicated, we always use the same linear displacement function $f_{\text{displacement}}(x) = x$. Changing this function can easily be accomplished by adapting the function in the shader or by using a non-linear gray-ramp texture. The resulting visual effect differs to some degree from previously shown images as shown in Figure 3.17 and can be used to achieve the impression of more or less densely packed bundles. For example, using $f_{\text{displacement}}(x) = x^2$ results in less emphasis for the bundles but more individual lines being visible (Figure 3.17a), while $f_{\text{displacement}}(x) = \sqrt{x}$ has the opposite effect (Figure 3.17b).

### 3.4.4 *Performance and Limitations of the Technique*

In Table 1 we give average frame rates for a selection of datasets. The measurements were taken on a 3GHz Intel Core2 Extreme with 4GB RAM, running Windows Vista, and using an NVIDIA GeForce 8800 GTX graphics card. The rendering performance for both lines and points is determined mainly by the number of lines, the number of vertices, the size of the dataset on the screen, and the size of the strips or quads. For the measurements, we maximized each visualization on the $812 \times 600$ pixel rendering window and then measured the average speed for animating a rotation, with vertical sync disabled and without anti-aliasing or filtering. The offline pre-processing to extract streamlines from 3D vector data took in the order of several seconds to several minutes and was done with VTK.

Besides the obvious limit in the number of lines and vertices that the technique can handle as just indicated by the performance data, there are two aspects that are of further importance. One is that, while the technique works well when the data elements (lines or points) inherently form a surface or closely bundled structures, it performs not as well for datasets with less structured but still dense elements.

Table 1: Performance measurements, sorted by dataset size.

| Fig. | Lines | Vertices | Type | Anagl. | Frame rate (fps) |
|---|---|---|---|---|---|
| 3.15a | n/a | 172 973 | point | no | 190 |
| 3.1 | 11 306 | 260 836 | line | no | 123 |
| 3.8b | 11 306 | 260 836 | line | yes | 65 |
| 3.11 | 786 | 278 849 | line | no | 290 |
| 3.8a | 786 | 278 849 | line | yes | 163 |
| 3.15b | n/a | 437 645 | point | no | 65 |
| 3.10a | 150 352 | 1 625 472 | line | no | 24 |
| 3.12 | 1 400 | 2 603 605 | line | no | 43 |
| 3.14 | n/a | 4 440 900 | point | no | 12 |
| 3.13 | 4 475 | 7 910 911 | line | no | 11 |

For example, if a volume were to be filled with a dense set of random lines, the technique would not create meaningful visualizations. The second limitation is that the technique performs best for visualizing the outer layer of closely bundled elements. While filtering does allow to look at subsets of the data, it still does not allow to look beyond the surface of the visible structures. Transparency based on a given parameter cannot easily be added because this would require sorting based on individual line segments which would greatly diminish the technique's performance and, thus, its suitability to large and dense line datasets.

## 3.5 INFORMAL EVALUATION

To evaluate our illustrations we conducted a small informal study of DTI-based fiber tract visualizations with four medical domain experts, three of them neurosurgeons and one a researcher in tractography. We showed them still images and the interactive tool, both in the regular and the anaglyphic versions. We were interested in their initial response but also asked questions about the quality of the illustrations, the ability to show detail and depth relations, and the usefulness for specific applications. We also asked about shortcomings, possible improvements, and potential application scenarios for our technique.

The result from this evaluation was that all experts were impressed by the visualizations, commenting that they were "beautiful," "impressively good," that they "almost look like an anatomical specimen," and that they visualize "what the fibers would look like if a brain was dissected by an anatomist." All said that, compared to the tract visualizations

they are used to (i. e., lines or tubes colored based on direction), our illustrative visualizations "show better depth relation and structure" and that they emphasize bundles well. One commented on the visible detail saying "one can really see how structures are fanning out." Looking at anaglyphic images, all liked that the fibers "really come out and show structure" and that the images are "extremely clear." All confirmed our design decisions and said that our technique would be particularly useful for illustration purposes, for example for using the created images in educational contexts such as brain atlases.

All domain experts used the term "very suggestive" in their descriptions, meaning that the illustrative character of the technique is very powerful and shows the bundling and path as well as depth relations very well. However, they also noted that our technique suggests the path of the visualized fibers would actually be like the real neural fibers in the brain, something that the underlying DTI data may not be able to provide. Thus, to avoid the illustrations showing something that does not actually exist they suggested using a fiber tracking method that does a better job at detecting fiber crossings (e. g., Q-Ball). Additional suggestions were to provide context by combining the technique with other data visualization methods such as volume rendering or by adding relevant structures such as tumors to the visualization. Moreover, while they all liked the interactive application, they all requested even more interactivity such as being able to select subsets of smaller fiber bundles that, e. g., connect two regions of the brain. We plan to integrate these suggestions in a future version of the tool.

## 3.6 CONCLUSION

We have presented a technique for illustrative visualization of dense line datasets. We create depth-attenuated halos around lines that do not overlap each other if the lines are close in depth, but do occlude lines located further away in depth. This has the effect of emphasizing tightly bundled line structures and abstracting from less organized regions in the data. An additional important illustrative effect is the resulting visual depth clustering of visually connected regions. The visually connected regions portray surfaces implicitly existing in the dataset due to its inherent structure. These illustrative rendering techniques are augmented by traditional interaction and line rendering techniques such as filtering and depth cueing. While supporting the rendering at interactive to real-time frame rates depending on the size of the dataset, the technique is also capable of producing high-quality black-and-white renderings so that the results can easily be used in printed materials. In addition, it easily extends to point cloud datasets.

An informal evaluation with domain experts revealed that our visualizations are successful in illustrating brain fiber structures, in particular

showing detail, emphasizing fiber bundles, and depicting spatial relationships. While these results are encouraging, we need to continue our evaluation and explore both a richer collection of data sources (i. e., other than DTI) and evaluate the technique in the other domains for which we have created visualizations. Moreover, combining the visualization technique with other methods can provide both context and additional detail that is necessary in some domains.

Additional future work includes, for example, adapting the process of turning lines into view-oriented triangle strips which consists of a data duplication part that unnecessarily uses too much memory. This can be avoided by using modern graphics cards extensions such as geometry shaders or the instanced arrays extension. In the case of point data, the point sprite extension could be used. The visual appearance could also be improved by exploring alpha-blending, for instance guided by the filtering process. By employing line style rendering techniques such as used in [Zander et al., 2004] it may even be possible to maintain the black-and-white character necessary for high-quality print output.

*Creating view-oriented triangle strips with geometry shaders has been implemented for Chapter 4.*

## 3.7 ACKNOWLEDGMENTS

We thank Cris Lanting and Pim van Dijk from the BCN NeuroImaging Center in Groningen, The Netherlands, for the brain datasets used in this chapter. The turbulent flow simulation dataset is courtesy of Martin Rumpf, Univ. Bonn, Germany. The armadillo and dragon datasets are from the Stanford Computer Graphics Laboratory. We also thank Alessandro Crippa, Moritz Gerl, Wladimir van der Laan, and Alex Telea for interesting discussions.

This chapter is based on: Maarten H. Everts, Henk Bekker, Jos B. T. M. Roerdink, and Tobias Isenberg. Depth-Dependent Halos: Illustrative Rendering of Dense Line Data. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1299–1306, November/December 2009.

# 4 INTERACTIVE ABSTRACTION OF DTI FIBER TRACTS

ABSTRACT: *We present a visualization technique for brain fiber tracts from DTI data that provides insight into the structure of white matter through visual abstraction. We achieve this abstraction by analyzing the local similarity of tract segment directions at different scales using a stepwise increase of the search range. Next, locally similar tract segments are moved toward each other in an iterative process, resulting in a local bundling of tracts at a given scale. This not only leads to the abstraction of the global structure of the white matter as represented by the tracts, but also creates volumetric voids. This increase of empty space decreases the mutual occlusion of tracts and, consequently, results in a better understanding of the brain's three-dimensional structure. Our implementation supports an interactive and continuous transition between the original and the abstracted representations via various scale levels of similarity. We also support the selection of groups of tracts, which are highlighted and rendered with the abstracted visualization as context.*

## 4.1 INTRODUCTION

The task of uncovering the functionality of the human brain in all its detail has engaged researchers for centuries, resulting in neuroscience as an important domain in itself. Fundamental to gaining such an understanding is to comprehend the brain's complex anatomy which comprises ganglia, blood vessels, grey matter, as well as white matter. This white matter which contains connections between nerve cells in the grey matter and which consists of approximately $10^5$ km of myelinated axons is the focus of our work. Often, neuroscientists do not only want to understand its anatomy in general but rather based on patient-specific data. While modern imaging techniques provide non-invasive ways to obtain data about the white matter's anatomy, it remains difficult to understand the structure because of its complexity.

In this chapter we propose a new technique for showing this structure of the brain's white matter using fiber tracts extracted for the whole brain. The goal of our technique is to show all fiber tracts simultaneously and at the same time visually abstract them so that global patterns become clear for a given level of scale. Our approach is based on diffusion tensor imaging (DTI), a non-invasive medical imaging technology. DTI records information about the approximate anatomy of white matter in that it reflects the self-diffusion properties of water in fibrous material such as white matter. The resulting tensor field is used in a variety of analysis methods, of which fiber tracking [Mori and van Zijl, 2002] is a popular approach. The general idea of fiber tracking is, starting from a seed point, to generate a curve following the main diffusion direction

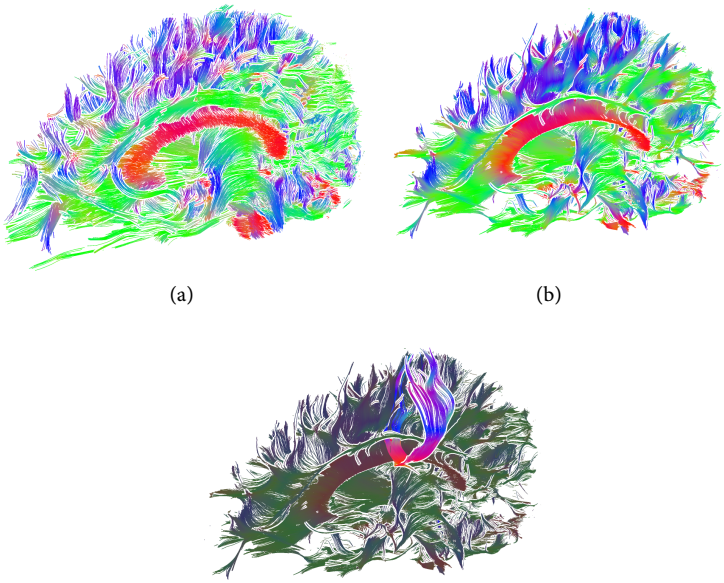
in the tensor field. Even though the resolution of DTI is not nearly high enough for tracing individual axons, fiber tracking can provide useful information on the structure of axon bundles and connectivity of brain regions.

Placing seeds to initialize fiber tracking is typically done in a particular region-of-interest or throughout the entire brain. We use the latter method which produces a huge collection of dense fiber tracts (typically $10^4$–$10^5$) that pass through the brain in complex ways (Figure 4.1a). When visualized in a naïve way, most of the global structure of the tracts including tube, sheet, or fan formations is occluded due to the high density of the tracts. If we look at the tracts in detail, however, we can observe that many neighboring tracts follow almost the same path for their entire length, others follow the same path only for subsections and then diverge to run parallel with other tracts. We use this observation that many tracts are locally virtually parallel as the basis of our approach and consider such sections to be similar to each other.

Specifically, we start by subdividing all tracts into small sub-segments of equal size. For every pair of vertices of nearby tracts we then determine the local tract similarity based on the direction of incident segments. This processing is done for vertices in a local neighborhood whose size is determined by the scale in which we are interested. For every vertex pair with a similarity above a certain threshold we record this information by adding an edge to a similarity graph that connects these vertices and, thus, the fiber tracts. Based on this graph we iteratively bundle tracts by relocating tract vertices to be closer to similar ones. The result of this process is a bundling of fiber tract structures which leads to a visually abstracted representation (Figure 4.1b).

While the reduction of visual information is useful in itself, it is also accompanied by the creation of volumetric voids; space previously occupied by many parallel tracts becomes mostly empty through the abstraction process (Figure 4.1b). These added voids throughout the brain decrease occlusion of the fiber tract patterns by the fibers themselves so that it becomes possible to see deeper into the brain and better interpret its structure. To further enhance the understanding we allow users to interactively and continuously move between the original and the bundled state of the fiber tracts which makes it easier to relate the two states. Furthermore, we allow users to select subsets of fibers to explore local patterns in the bundled state (Figure 4.1c).

The remainder of this chapter is organized as follows. First, we relate our work to existing approaches in Section 4.2. Then, we explain how we create our bundled fiber tract configurations in Section 4.3 and discuss their visualizations in Section 4.4. Next, we present our results in Section 4.5 and we discuss the merits and limitations of our technique in Section 4.6. Finally, we conclude this chapter in Section 4.7 and mention some possibilities for future work.



(a) (b)

(c)

Figure 4.1: DTI fiber tract abstraction based on similarity and proximity: (a) not processed; (b) bundled; (c) subset highlighted.

## 4.2 RELATED WORK

There exists a wide variety of techniques for generating abstractions and visualizations that provide an understanding of the brain's white matter structure. The vast majority of these approaches rely on DTI data because it captures the local directionality of the myelinated axons in the brain.

One way to understanding the brain's white matter structure is to start from the fractional anisotropy (FA) derived from DTI. This scalar field is a measure for the directionality in the tensor field. The tract-based spatial statistics approach by Smith et al. [2006] produces a 3D volumetric skeleton from a mean FA volume as an abstract representation of white matter. A related representation—anisotropy crease surfaces—was proposed by Kindlmann et al. [2007], also based on the FA values directly. In a complementary approach, Schultz et al. [2007] analyze the topology of the tensor field based on probabilistic fiber tracking, thus illustrating the structurally important information.

In contrast to these techniques and similar to our own method, many approaches that visualize white matter first employ deterministic tractography [Mori and van Zijl, 2002] and then visualize the extracted

fiber tracts [Everts et al., 2009; Petrovic et al., 2007; Schultz et al., 2008], possibly processing them further. Visualizing all tracts resulting from full-brain tractography typically not only leads to occlusion and clutter, but also obscures the structually important aspects of the data. The visual clutter in full-brain tractography can be reduced by using techniques for more careful seeding [Merhof et al., 2005; Vilanova et al., 2004; Zhang et al., 2003]. An alternative option is to simply visualize a subset of all generated tracts, based on seeding at regions-of-interest and interactive tract selection [Akers et al., 2004; Blaas et al., 2005]. With our technique we can both include all generated fiber tracts in the visualization such that the overall structure is revealed and at the same time make use of tract selection for emphasis.

An important way to obtain an abstract representation of white matter structure is to use clustering [Brun et al., 2004; Klein et al., 2006; Moberts et al., 2005; O'Donnell and Westin, 2007; Tsai et al., 2007; Zhang et al., 2008], potentially incorporating interaction with the clustered data to support exploration [Chen et al., 2009; Jianu et al., 2009]. To represent the structure captured in the abstraction visually, for example, Chen et al. [2008], Enders et al. [2005], and Merhof et al. [2009] wrap surfaces around the fiber tract clusters. The use of clustering is related to our own approach because in both cases the similarity of tracts is important. However, while clustering determines the similarity based on the whole length of the tract, we establish similarity at a local level.

This aspect of local similarity is related to the visualization of fiber tract coherence by Hlawitschka et al. [2010] who calculate a coherence measure based on the deviation of fiber tracts in small neighborhoods. The resulting scalar field is not only visualized directly, but also used as a transparency mask to enhance fiber tract visualization. Our goal is different in that we aim to show the overall structure while keeping the connectivity intact. This structure can also be captured in terms of an abstract network [Hagmann et al., 2008] which nicely illustrates the connectivity of regions but loses the relation to the actual anatomy to some degree.

Finally, our approach of bundling fiber tracts is related to techniques used in 2D graph drawing, including force-directed layouts [Di Battista et al., 1999] and, in particular, the graph edge bundling by Holten et al. for hierarchical [Holten, 2006] and general graphs [Holten and van Wijk, 2009]. The latter is especially relevant due to its use of an iterative force-directed approach. An important difference with our approach is that for the edges in the 2D graph the path is a feature denoting the connection of two nodes, whereas for our fiber tracts the path location is relevant and important as an anatomical feature.

## 4.3 ANALYSIS AND BUNDLING OF FIBER TRACTS

Our goal is to visualize DTI fiber tracts in their entirety such that the depictions assist viewers in understanding the structure of white matter, at different levels of scale. Our technique is based on fiber bundling that takes the local conditions into account but also considers the location of the original paths to provide anatomic relevance. We achieve these goals through a two-stage process. First, an analysis stage inspects the fiber tracts to collect information on the local similarity of tracts with respect to proximity and direction. Next, an iterative process uses this similarity information to move fiber tract sections towards similarly oriented neighboring tracts, resulting in a bundled state. In interactive exploration, users can move between the unbundled and the bundled state via a number of scale levels.

### 4.3.1 *Analysis of Local Direction Similarity*

We consider two tracts to be locally similar if the directions of the considered sub-tracts are approximately parallel and they are not located far from each other. We capture this similarity information in a graph whose edges connect fiber tracts where they are locally similar. Hence, the nodes of the similarity graph are vertices of the fiber tracts, while the graph's edges represent similarity relations between these vertices and, thus, between tracts. The goal of the analysis stage is to create the similarity graph which will later be used to inform the bundling stage.

To be able to faithfully describe the local similarity for a set of fiber tracts with the similarity graph we rely on the polygonal fiber tracts being evenly tessellated. This means we expect vertices to be equidistant and, because the fiber tracts resulting from tractography not necessarily have these properties, we first resample each tract such that each pair of neighboring vertices has the same distance $d_{\text{sample}}$.

Based on the set of resampled tracts we now capture their local similarity. While such similarity can be established for a more general case of points on two planes [Jones et al., 2000], we use a more direct heuristic because we only deal with vertices on linear structures. We process each possible pair of fiber tracts $A$ and $B$ to find, for each vertex on $A$, its nearest neighbor vertex on $B$, and vice versa. This step yields a list $L$ of vertex pairs of the form $(v, nn(v))$, where $v$ is a vertex on tract $A$ or $B$ and $nn(v)$ is $v$'s nearest neighbor on the other tract. At this stage, the number of pairs in $L$ is equal to the sum of the number of vertices in $A$ and $B$. However, $L$ may also contain superfluous entries, i. e., a pair $(p, q)$ can be removed if $(q, p)$ also exists in $L$.

*We experimented with variations of this heuristic, but what we describe here proves to strike a good balance between simplicity and effectiveness.*

*Such a brute force approach leaves room for improvement in terms of efficiency, see the discussion in Section 4.6.1.*

Next, we filter the entries of $L$ based on three conditions, a distance condition, an angle condition, and a combinatorial condition, before

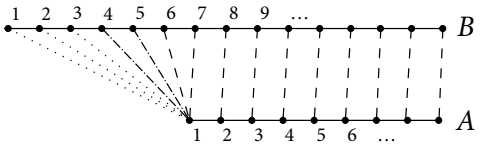



Figure 4.2: Illustration of how vertices of tracts are connected by edges. Only lines with pure dashes connect vertices that form an edge in the similarity graph, the other pairs of vertices are discarded because of Condition 1 (dots) and Condition 3 (dash-dot).

creating edges in the similarity graph. We create an edge if and only if a vertex pair $(p, q)$ in $L$ meets all the following conditions:

1. the distance between $p$ and $q$ is smaller than a given length $d_{\max}$;
2. at least one of the segments incident to $p$ is approximately parallel to one of the segments incident to $q$, where "approximately parallel" means that the angle between the segments is smaller than a predefined angle $\theta_{\text{par}}$; and
3. the nearest neighbor relation of $p$ and $q$ is approximately mutual, that is, it holds that $|index(nn(q) - index(p)| \leq 1$ and $|index(nn(p)) - index(q)| \leq 1$, where the function $index(v)$ returns the index of $v$ on the tract to which it belongs.

This last condition requires some additional motivation. Figure 4.2 shows a common configuration of tracts. Assume that tract $A$ and $B$ are approximately 3 mm apart, $d_{\text{sample}} = 1$ mm and $d_{\max} = 6$ mm. If only Conditions 1 and 2 are used then the pairs $(A_1, B_1)$, $(A_1, B_2)$, $(A_1, B_3)$ would not define an edge because their distances are all greater than $d_{\max}$, but the pairs $(A_1, B_4)$, $(A_1, B_5)$, $(A_1, B_6)$, $(A_1, B_7)$ would each lead to an edge in the similarity graph. The latter is undesirable because this would mean that during the actual bundling process vertices $A_1$ and $B_4$ would move toward each other, $A_1$ towards $B_5$, $A_1$ towards $B_4$, etc. However, we only want $A_1$ and $B_6$ to move towards each other and $A_1$ and $B_7$ to move towards each other. Condition 3, therefore, prevents $(A_1, B_4)$ and $(A_1, B_5)$ from being turned into edges and to cause unwanted edge bundling later on. The value of one in the condition is somewhat arbitrary. It has to be greater than zero but not too large, and a value of one proved to work well in our experiments.

Each newly added edge is also given a real-valued attribute that captures the distance between the vertices in the pair. This value is later used in the iterative bundling process.

### 4.3.2 *Iterative Bundling of Fiber Tracts*

In the second stage, the information in the similarity graph is used to iteratively move vertices towards each other. In general, this has the effect that tracts, or parts of tracts, move towards nearby tracts of similar direction. The process is realized iteratively because a vertex in the similarity graph is, in general, connected to multiple other vertices. During each iteration, we accumulate the influence of all incident graph edges for each vertex in a displacement vector.

Specifically, one iteration consists of the following steps. First, the displacement vector of each vertex is initialized to zero. Similar to force-directed layouts [Di Battista et al., 1999, Ch. 10] we then calculate the displacement for each graph edge $(p, q)$ that will move $p$ and $q$ towards each other such that they will meet halfway, i. e., at $(\boldsymbol{r}_p + \boldsymbol{r}_q)/2$, with $\boldsymbol{r}_p$ and $\boldsymbol{r}_q$ being the current position vectors of vertices $p$ and $q$, respectively.

If $p$ and $q$ would not be connected to any other vertices they would end up at the same position, i. e., both halfway between $p$ and $q$. However, that will not be the case in general because $p$ and $q$ are also connected to other vertices.

Because such a simple force-directed approach can potentially lead to run-away situations, we divide the displacement vector by the number of edges connected to the processed vertex to compute the average, similar to what is done in the barycenter force-directed method [Di Battista et al., 1999, Ch. 10.2]. In addition, we remove noise along a fiber tract (which occurs in cases such as the ones illustrated in Figure 4.3) by applying a small Gaussian kernel to the displacements along each tract. This smoothing also addresses the issue of otherwise large transitions where merging tracts come into the influence range of other tracts (Figure 4.4, top), and the issue of the occasional double-linking (Figure 4.4, bottom). In a final adjustment, we ensure that vertices can only be moved perpendicularly to the tract's local direction. We obtain this direction from the average orientation of the two tract edges incident to the processed vertex and project the accumulated and filtered displacement vectors onto the orthogonal plane. After all displacement vectors have been computed they are applied to their respective vertices and the next iteration is started. Several iterations are necessary to achieve a bundled state; we typically use in the order of 40 iterations.

The bundling as described here deforms the tracts themselves: the fiber tracts are not simply subjected to rigid transformations but instead are locally modified. This is an inherent property of the method and the confined deformations make it possible to force tracts to locally follow the same paths. This creates voids between bundles which is one of the main goals of the method. However, the amount of deformation is kept to a minimum by moving vertices mainly perpendicularly to the tract direction and by applying Gaussian smoothing.

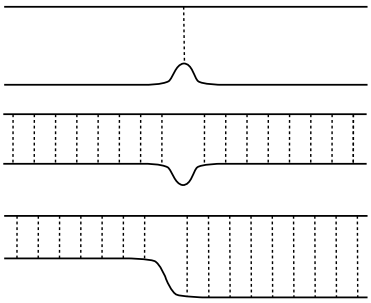


Figure 4.3: Illustration of noise (dents, bulges, steps) in neighboring tracts. We remove this by applying a Gaussian filter to the displacements.

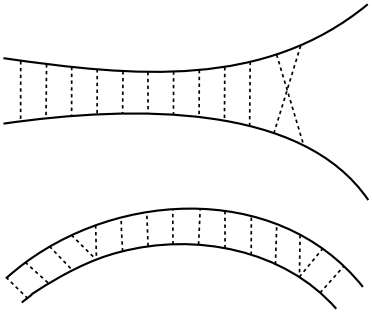


Figure 4.4: Illustration of diverging tracts (top) and double-linking in the similarity graph (bottom).

### 4.3.3 *Scale-Dependent Abstraction as a Step-Wise Process*

At this stage, the fiber tract analysis and bundling incorporates a notion of scale due to the use of $d_{\max}$ to limit the neighborhood of what is being considered similar. Of course, selecting different values for $d_{\max}$ will result in different notions of scale, bundling only close structures for small values of $d_{\max}$, or merging larger structures for larger values of $d_{\max}$. To better support the exploration of scale in our process we select a fairly large value for $d_{\max}$ (we typically use the equivalent of 5 mm, and have explored values of up to 8 mm) and compute the similarity graph based on this value. During bundling, however, we only process edges from the graph whose length is smaller than a given $d_{\max,i} \in [0, d_{\max}]$. By performing the bundling for increasing values of $d_{\max,i}$ we can create bundled fiber tract representations for growing scales. Because we use the same fiber tract tessellations for each of these computations we can relate the position of a fiber tract vertex in the un-bundled location to the position of the same vertex at a bundled location at any of the computed scales.

## 4.4 VISUALIZATION AND INTERACTION

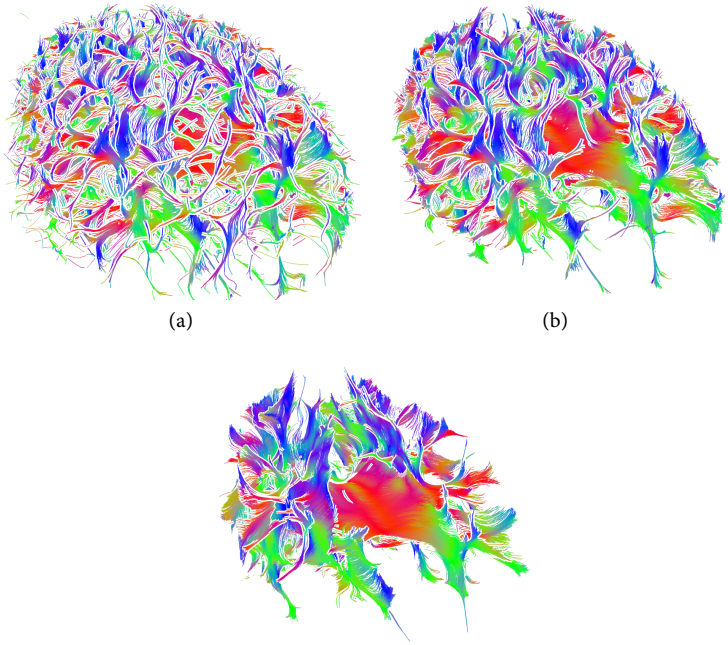
The result of the bundling process is that for each tract not only its original vertex positions have been stored but also the positions of its vertices at the different bundling scales. Based on this data we provide a visualization that enables the user to interactively and seamlessly move from one bundling scale to the next to explore white matter structure. In addition, we provide means to employ filtering, a 2D lens tool, and tract selection.

### 4.4.1 *Rendering and Visualization at Multiple Scales*

In the user interface of our visualization users can interactively control the scale of bundling. Even though we only compute a discrete number of scales (typically in 1 mm increments, so when $d_{\max} = 5$ mm we compute 5 discrete bundled representations), we provide a continuous transition by linearly interpolating the tract vertex positions between consecutive scales. This continuous transition prevents sudden jumps of tracts while users are interacting with the scale slider and helps them to follow the tracts across scales.

For rendering we store the tract positions for the different scales on the graphics card and perform linear interpolation between consecutive scales in a vertex shader. This allows us to render the tract representation and to move between scales at interactive speeds. Furthermore, our specific rendering is based on depth-dependent halo visualization for dense line data [Everts et al., 2009] with one small adaptation. Instead of duplicating vertices to create a view-oriented triangle strip from the lines, we use a geometry shader for this purpose. The main reason for this change is that the creation of the view-oriented triangle strips depends on the local direction of the line at each vertex, which was precomputed in the original implementation. However, pre-computation is unfeasible in the current case because the interpolation between scales results in the local direction of the lines depending on the chosen scale. This problem is solved by calculating the direction and creating the triangle-strip on the fly in a geometry shader. This approach has the additional advantage that it requires much less data to be transferred to the graphics card.

The complete process employs a vertex shader that interpolates vertex positions depending on the chosen scale, then transfers the positions to the geometry shader, which turns the line strip into a view-oriented triangle strip. The fragments that result from rasterization are processed in a fragment shader that assigns white for halos, black for lines, and performs the depth manipulation as explained in [Everts et al., 2009]. In addition to the original black-and-white rendering, the lines can also be colored based on the direction as is common in many DTI tract



(a) (b)

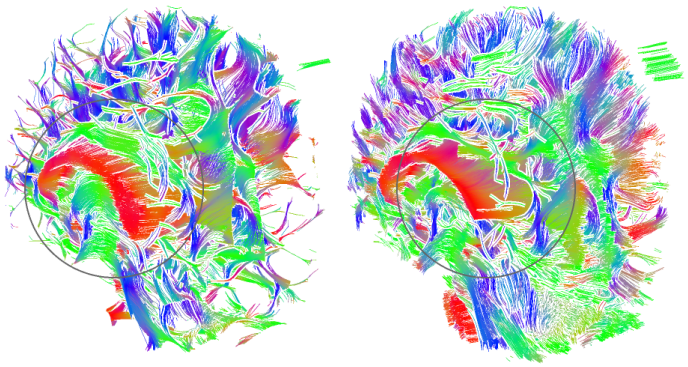
(c)

Figure 4.5: Filtering on vertex degree in the similarity graph reduces occlusion more. Here we show tracts (a) unfiltered, (b) with a filter threshold of 160, and (c) a filter threshold of 560.

visualization techniques (e. g., Figure 4.1). The additional colors assist viewers in distinguishing directions in dense regions after bundling: in strongly bundled regions the difference in depth is minimal so that no halo is generated which would otherwise support depth perception.

### 4.4.2 *Filtering for additional abstraction*

Bundling tracts based on local similarity results in volumetric voids which reduce occlusion artifacts (e. g., Figure 4.1b). We explored decreasing the occlusion even more by interactively filtering out smaller, thinner structures that do not have as many neighboring tracts with similar direction (Figure 4.5). The filtering attribute we use is the number of similarity graph edges that are connected to a vertex of a fiber tract, i. e., the degree of the vertex in the graph. By passing the degree as an attribute to each vertex, the filtering can be done in the fragment shader. This has the added effect that even though the attribute passed with its vertex is a natural number, the interpolation done in the rasterization stage enables filtering over a continuous range.



(a) (b)

Figure 4.6: Lens interaction with the bundled fiber tracts. The lens can have two modes, one (a) where the lens reveals the original state and one (b) that does the reverse, revealing the bundled state.

### 4.4.3 *Focus+Context with a 2D Lens*

While the scale control described in the previous subsection affects the whole scene, we also allow users to explore the effects of scale locally with a two-dimensional virtual lens [Bier et al., 1993]. If this lens is placed over the scene in screen-space (Figure 4.6) it affects the scale value that is used to derive the positions of the vertices beneath it, while other vertices remain at their normal position as determined by the global scale setting. For this purpose we first check for each rendered vertex' screen position whether it is inside the lens and, based on their location inside the lens, compute an attenuation factor (which is one in a center region and zero at the perimeter). Next, we reposition affected vertices to the location determined by the lens scale setting and the vertex attenuation factor. This attenuation factor ensures a gradual change from global to local scale. We implemented the lens in the same vertex shader that is also responsible for the scale-dependent interpolation of vertices to ensure fast processing. As a result, users can move the lens over the scene in screen-space to locally explore the effect of the bundling.

### 4.4.4 *Tract Selection*

As an additional means of exploring white matter structure and its connectivity we support a simple interactive selection of groups of fiber tracts. Selected tracts are highlighted with respect to the bundled state by rendering the selection with regular colors while the remaining tracts are rendered with less vivid colors (Figure 4.1c and 4.7). Specifically,

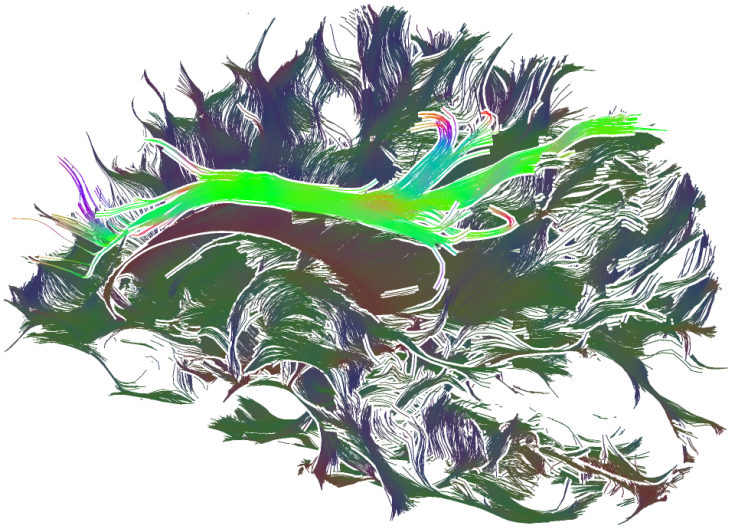


Figure 4.7: The selected subset of tracts is shown in its unbundled state, while the remainder is shown bundled. In addition, a large amount of filtering was applied to the context only, showing only vertices with a degree in the similarity graph of more than 210. Notice the sheet-shape of the *corpus callosum* while the *cingulum* is in its original state (compare it with its bundled shape, e. g., in Figure 4.1c).

we let the tract direction affect the color in a range between 20% and 40% of each color channel. This has the effect that the fiber tracts in the context are considerably darker than the ones in focus. The context's subdued color scheme enforces the emphasis of the selection which still uses vivid colors. Alternatively, we also permit users to only show selected tracts or to render selected tracts at their original locations as shown in Figure 4.7.

The interactive selection of fiber tract sub-sets makes it easier for viewers to follow the path and position of the selected tracts and relate these to their bundled state. We use a sphere as the interaction object that can be placed in 3D space; tracts that pass through it *in the unbundled state* are selected [Akers et al., 2004; Blaas et al., 2005]. For the positioning of the sphere we use an approach similar to that found in many 3D modeling programs (such as Blender). Users can click on the sphere and, while the mouse button is down, three lines are drawn through the center of the sphere (Figure 4.8), each representing one coordinate axis. Upon moving the mouse, the sphere is moved along the line closest to the current mouse position. With repeated clicks the sphere can be positioned anywhere in the dataset. For an improved depth perception of the sphere placement we extend the three coordi-

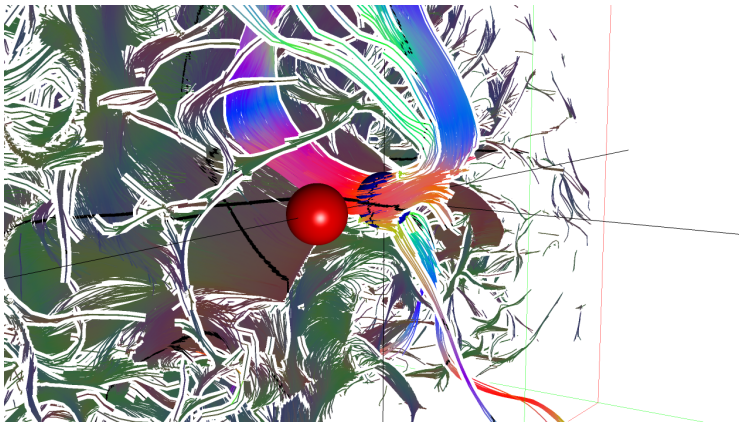


Figure 4.8: Fiber bundle selection using a sphere and the visualization of the coordinate axes. The blue sphere (inside the selected tracts) represents the current selection while the red one is being moved and will specify a new region of interest.

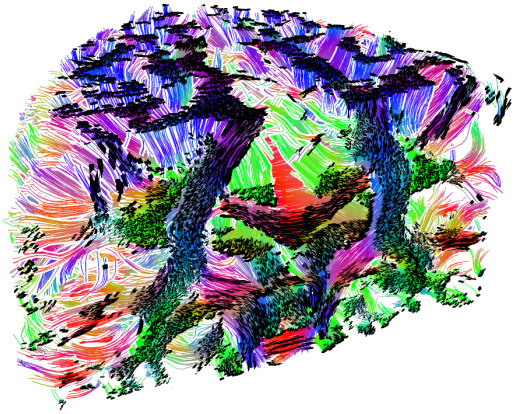
nate axis lines to where they intersect the bounding box of the whole dataset (see Figure 4.8).

## 4.5 RESULTS

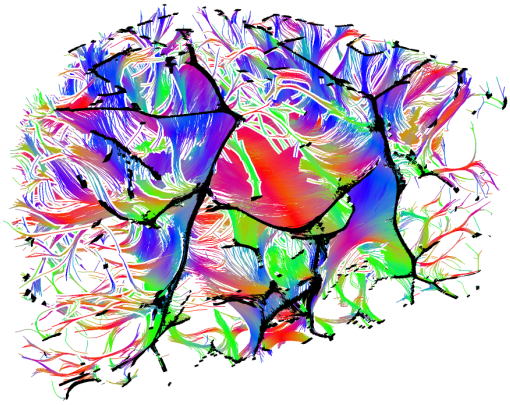
Having described how we create a bundled representation of fiber tracts and how to visualize them, we now discuss a number of examples of the technique to illustrate how the method can be employed.

These results and the remaining images shown in this paper are based on DT-MRI data that was acquired on a 3T MRI system (Philips Intera) from a healthy volunteer whose consent was obtained prior to scanning. Diffusion Tensor Imaging was performed using a diffusion weighted spin-echo, echo-planar imaging technique. The DTI parameters were as follows: 240 mm × 240 mm field of view; 128 × 128 matrix size; 51 slices; $1.85 \times 1.85 \times 2\ \text{mm}^3$ imaging resolution; 5,485 ms repetition time; 74 ms echo time. Diffusion was measured along 60 non-collinear directions. For each slice and each gradient direction, two images were acquired, one with no diffusion weighting ($b = 0\ \text{s/mm}^2$) and one with diffusion weighting ($b = 800\ \text{s/mm}^2$) to measure APA and APP fat-shifts. The fiber tracts were generated by the Diffusion Toolkit program, using the FACT algorithm [Mori and van Zijl, 2002].

Figure 4.1a shows the fibers of one half of the brain in its unbundled state, with the color values indicating the local direction of the tracts. While it is possible to identify certain structures such as the *corpus callosum*, much of the structure of the fiber tracts below and above this region is hidden by other tracts. Figure 4.1b shows the same view,



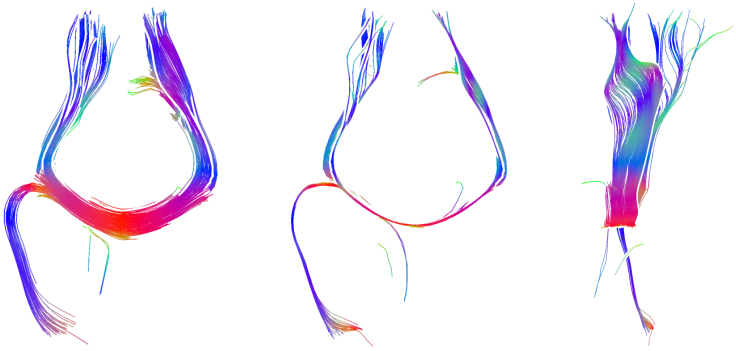
(a)



(b)

Figure 4.9: The sheetness of the underlying tract data in both (a) the original and (b) the bundled state is emphasized by using a cutting plane. To further highlight this characteristic, parts of the fiber tracts close to the cutting planes are rendered in black.

bundled to a scale of 5 mm. In this visualization, the sheet-like structure of the *corpus callosum* becomes evident and more detail is visible for the upper regions. Similarly, the *cingulum* is now visualized as a pronounced structure and tracts emerging from it to the outer regions of the brain are visible. Also, the structure in the central region of the brain is shown much more clearly, the merging and splitting of fiber bundles is well depicted. Finally, a selection on the *corpus callosum* was made in Figure 4.1c which shows the fiber tracts in bundled form emerging from this location. Here we show the whole length of the tracts in focus, while the context of unselected tracts is partly removed using a cutting plane.



(a) (b) (c)

Figure 4.10: A selection of fiber tracts in the *corpus callosum* (a) before and (b) after bundling. In (c) the result from (b) was turned. Notice that the bundling is done with respect to all tracts, not only the selection.

Figure 4.9 further emphasizes the aspect that most of the brain's fiber tracts are organized in sheets rather than bundles. Here we render the parts of the fiber tracts that are close to an active cutting plane in black to be able to clearly see where the tracts intersect the planes. While the unbundled state in Figure 4.9a still shows large regions with intersections, the visualization of the bundled state in Figure 4.9b clearly illustrates the brain's sheet-like structure. In addition, the added space that results from the iterative bundling allows us to look further into the brain in Figure 4.9b, and thus to see smaller structures such as the ones that emerge from the *cingulum*.

Figure 4.10 shows a selection of tracts in the *corpus callosum* both (a) before and (b) after bundling. It is important to note that the selection of tracts only affects the rendering of tracts, not the bundling which is still done for all generated fiber tracts. Therefore, the selection provides a nice example for the sheet-like structure in the neighborhood of the selected tracts, apparent in particular in the turned view in Figure 4.10c.

As stated above, the volumetric voids created by our bundling approach assist in reducing occlusion. The application of filtering based on the degree of the vertices in the similarity graph decreases the occlusion even more, as illustrated in Figure 4.5. The three images in Figure 4.5a–c show the fiber tracts in a bundled state for no filtering and two different filter thresholds, respectively. Notice how only the major structures of the white matter remain in Figure 4.5c.

The 2D lens interaction of Section 4.4.3 is shown in Figure 4.6. We support two modes for the lens. One mode shown in Figure 4.6a shows the tracts under the lens in the original positions while the rest of the tracts is in its bundled state. The other mode shown in Figure 4.6b is

the reverse of the first, now the context is in its unbundled state and the lens allows the user to see the bundled state of the tracts inside the lens.

Based on these results, we presented our fiber tract bundling approach to a neuroscientist and we now report on his thoughts. His initial reaction was enthusiastic, he could clearly distinguish, for example, the *cingulum* bundle which he found clearly separated from the *corpus callosum*. However, he found the lateral parts more difficult and could not really distinguish specific bundles. He did, on the other hand, appreciate the bundled visualization of a selection of tracts, as shown in Figure 4.10b, stating it nicely conveyed the sheet-like nature of the white matter structure. Furthermore, he was interested in seeing our visualization together with FA slices and he would like to compare it to the FA skeleton by Smith et al. [2006].

## 4.6 DISCUSSION

In this section we further discuss some aspects of the technique. In particular, we report on performance, review the specific parameters that we used, and mention a few limitations.

### 4.6.1 *Performance and Parameters*

Both the graph generation and the edge bundling were written in C, while the visualization program was written in Python, making extensive use of shaders, vertex arrays, and vertex buffer objects. The latter part, therefore, is interactive, but the graph generation and bundling stages are done in a pre-processing step.

The first part of this preprocessing, the graph generation process, is relatively easy to parallelize because each pair of tracts can be compared independently. We choose a simple thread-based approach for parallelization and achieve a near-linear speedup with an increasing number of CPU cores. Nevertheless, due to the amount of data and the computational complexity of our approach, both the graph generation and iterative bundling take a considerable amount of time.

The generation of the similarity graph requires that each tract will be compared to each other tract, so the number of tract comparisons is quadratic with respect to the number of tracts. For each pair, finding the nearest neighbor for each vertex requires a quadratic amount of operations with respect to the number of vertices in both tracts. On average, this amounts to a complexity of $O(N^2M^2)$, where $N$ is the number of tracts and $M$ the average number of vertices per tract. We ran the process on a machine on which up to 4 Intel® Xeon® X7350 processors could be used, each with 4 cores, running at 2.93 GHz and using 128 GiB of memory. On this setup and using 4 threads, the graph

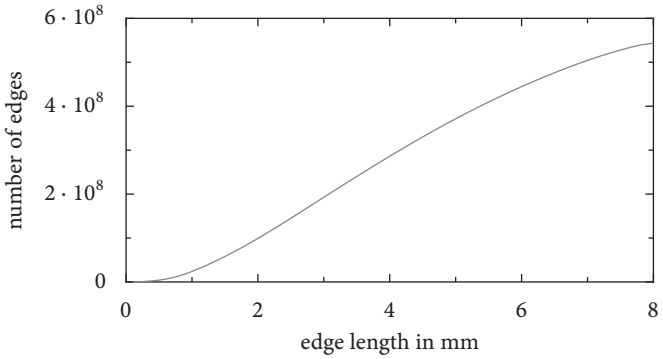



Figure 4.11: Number of edges in the similarity graph as a function of the edge length.

generation process took 76 minutes to complete for 77 389 tracts with a total of 1 944 570 vertices ($d_{\text{sample}}$ was set to 1 mm).

An iteration of the iterative bundling process is linear in the number of edges in the similarity graph, but the number of edges in the graph is typically large. In the given example, a similarity graph with ca. $345 \cdot 10^6$ edges was generated, using a direction similarity threshold $\theta_{\text{par}}$ of 11.48 ° and a maximum search range $d_{\text{max}}$ of 5 mm. With this graph we created 5 scale steps, each with $d_{\text{max}}$ increased by 1 mm. This computation took 4 hours and 45 minutes on a single core of the aforementioned machine.

Figure 4.11 shows a graph of the number of edges in the graph as a function of the edge length. It can be seen that for a length of ≈ 8 mm the graph levels off, which justifies our choice to use in the preprocessing phase a value of $d_{max}$ =8 mm. In the same figure it can be observed that over a range of approximately 1–6 mm the graph is more or less linear. This behavior is probably caused by conditions 2 and 3 from Section 4.3.1, and by the sheet-like nature of the brain's structure.

### 4.6.2 *Limitations*

One important limitation of our technique is the computational complexity of the graph generation and iterative bundling, in terms of running time and memory usage. As reported in Section 4.6.1, the running time of both steps has to be measured in hours. An option for improving this situation, for example, is to employ a space-partitioning data structures for finding which tract segments are similar.

Even though we aim to keep the amount of deformation of the tracts minimal, our bundling approach does change the positions of the vertices of the tracts, making them not entirely anatomically correct. Also, topological properties may change as a result of bundling. Let us, for

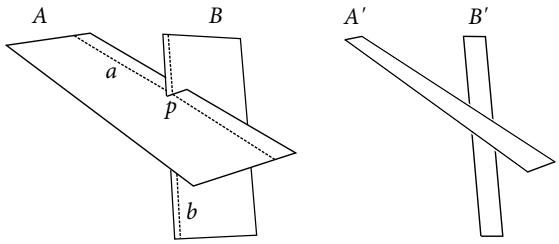



Figure 4.12: Illustration of how the topology may change during bundling.

*Note that, for the tractography method used here (FACT), intersections of fiber tracts are not possible. However, intersections are possible with other fiber tracking methods.*

example, consider two ribbon shaped fiber groups $A$ and $B$ (see Figure 4.12) which intersect each other perpendicularly. Now assume that the leftmost tract $b$ of $B$ intersects some tract $a$ in $A$ in point $p$, i. e., $a$ and $b$ have the point $p$ in common. During bundling, the tracts do not affect each other because of the perpendicular arrangement but the width of each of the ribbons decreases while their overall spatial positions remain more or less the same. So it is reasonable to assume that the bundled ribbons $A'$ and $B'$ do not intersect anymore, that is, the tracts $a'$ and $b'$ do not have a point in common. This shows that bundling may change the property '$a$ and $b$ have a point in common.' In the same way, properties like 'between' and 'in front of' may change under bundling. When answering topological questions of this kind it is advisable to make a number of transitions between bundled and original state and determine what the situation is in the original state.

## 4.7 CONCLUSION

We have presented a visualization method for DTI fiber tracts that helps users in understanding white matter structure through visual abstraction. It differs from previous approaches in that we are able to process and visualize all fiber tracts extracted for a brain, deriving an abstracted representation through neighborhood similarity analysis based on local tract orientation and iterative bundling. The resulting representation of the fiber tracts allows people to get a better insight into the structure of the white matter of the brain. Increased voids between the bundled tracts not only reduce the occlusion issues normally arising from visualizing all fibers in the brain but also reveal the sheet-like structure of the brain's white matter. While the pre-processing requires a considerable amount of processing time, it is possible to explore the resulting data interactively. Because we compute the bundling for a number of stages for increasing neighborhood sizes, users can seamlessly transition between the different scales of abstraction. In addition, we support filtering, tract selection, and a virtual lens as interactive tools to explore the effects of the bundling.

Future work includes, of course, improving the running time of the pre-processing steps, as mentioned in Section 4.6.2. This would also make it easier to further explore the effect of parameter settings on the bundling. Furthermore, our method needs more validation with the help of neuroscientists. Finally, we are planning to combine our bundling approach with fiber tract clustering which could yield interesting results.

## 4.8 ACKNOWLEDGMENTS

We thank Cris Lanting and Pim van Dijk from the BCN NeuroImaging Center in Groningen, The Netherlands, for the brain dataset used in this chapter.

# 5

# INTERACTIVE ILLUSTRATIVE LINE STYLES FOR FLOW VISUALIZATION

ABSTRACT: *We present a flexible illustrative line style model for the visualization of streamline data. Our model partitions view-oriented line strips into parallel bands whose basic visual properties can be controlled independently. We thus extend previous line stylization techniques specifically for visualization purposes by allowing the parametrization of these bands based on the local line data attributes. Moreover, our approach supports emphasis and abstraction by introducing line style transfer functions that map local line attribute values to complete line styles. With a flexible GPU implementation of this line style model we enable the interactive exploration of visual representations of streamlines. We demonstrate the effectiveness of our model by applying it to 3D flow field datasets.*

## 5.1 INTRODUCTION

The flow of fluids and gases plays an important role in a wide variety of real-world phenomena. Examples include the aerodynamics of cars, the heat distribution in offices, and the airflow around falling ink droplets. Consequently, flow has been extensively studied, typically through three-dimensional simulations. These simulations yield large amounts of data containing information at multiple scales; for some applications the general structure of the flow is most relevant, for others the small local deviations are the subject of study. Visual representations of flow data help in understanding its behavior and over the years a large number of methods have been developed for this purpose. Initially, most flow visualization methods employed photorealistic rendering techniques, but later on also methods that borrow principles from scientific illustration were developed.

Inspired by such illustrative visualization techniques [Rautek et al., 2008], we present a flexible method for illustratively depicting streamlines generated from 3D vector fields. We achieve this flexibility by introducing a line style model whose parameters can be interactively manipulated, thus facilitating the interactive exploration of the parameter space of visual streamline representations. This allows the user to select and generate the representations that are most suitable for the data and communication goals at hand. In addition, we introduce *line style transfer functions* that map scalar values derived from the 3D simulation to line styles. These line style transfer functions also allow us to actively control the introduced amount of abstraction, which is an important principle of illustrative visualization [Rautek et al., 2008].

In order to achieve flexible parametrization of line styles we generalize a previous illustrative approach for line visualization [Everts et al.,

2009], by subdividing the view-oriented line strips that represent the streamlines. These strips are split into bands that are arranged orthogonally to a line's direction, and whose shape, color, relative distance to the viewer, and width can be independently controlled. In addition, we allow these line parameters to individually depend on local data attributes such as temperature or velocity. The same local data attributes can also be used as input for the line transfer function, controlling which line style is being used for which part of a line. Together, the individual line parameter control and the line style transfer functions facilitate a flexible line-based illustrative visualization.

In summary, the contributions of this chapter are a flexible line style model for use in scientific streamline visualizations, line style transfer functions to control the application of a specific style depending on an external parameter, and a fast yet flexible implementation of this model on the GPU. We demonstrate the power of our approach for a number of 3D flow datasets that exhibit complex flow patterns.

## 5.2 RELATED WORK

In this section we discuss related work in the fields of flow visualization and illustrative visualization.

### 5.2.1 *Flow Visualization*

Being one of the most fundamental subjects for visualization, a broad range of methods have been developed for the visualization of flow datasets. McLoughlin et al. [2010] survey both flow visualization in general and integration-based, geometric flow visualization in particular. A key component of integration-based flow visualization methods is the use of geometric objects to depict the properties and structure of the flow. These objects are generated by integrating over the underlying velocity field—starting from a set of seed points.

Lines are the most widely used primitives for this purpose, and in the context of steady flow such trajectories are called streamlines, whereas for unsteady flow streak and pathlines are used. The challenge for the visualization of three-dimensional streamlines is overcoming perceptual problems. Simply rendering large numbers of lines can quickly lead to clutter and occlusion, not to mention the fact that the general thinness of a line makes it hard to convey depth and spatial relationships.

Solutions to deal with these perceptual challenges include careful placement of streamlines through seeding strategies (see [McLoughlin et al., 2010] for an overview), illuminated streamlines [Zöckler et al., 1996], and shaded tubes or ribbons [Ueng et al., 1996]. Particularly relevant for this chapter are the approaches that employ textured view-oriented triangle strips to mimic shaded tubes [Schussman and Ma,

2002; Stoll et al., 2005]. Such shaded primitives need to have a certain width for the depth perception to work and have only a limited number of visual variables to convey additional information: typically only width and color, although textures can also be used to convey information about the flow [Stoll et al., 2005]. In terms of flexibility in controlling the appearance of flow streamlines, the approach by Shen et al. [2004] that uses 3D flow textures is relevant.

### 5.2.2 *Illustrative Visualization and Line Stylization*

Illustrative visualization methods [Rautek et al., 2008] use and apply the principles of (scientific) illustrators to achieve the clarity and effectiveness found in traditional illustrations. Naturally, many of these methods employ techniques from the field of non-photorealistic rendering (NPR). NPR methods particularly relevant to our work aim to replicate line drawings and, for this purpose, support different line styles.

Dooley and Cohen [Dooley and Cohen, 1990], for example, use dashing for illustrating geometric models, whereas the difference vectors of Schlechtweg et al. [1998] permit a larger palette of styles. Other line style parametrization methods include stroke texturing [Northrup and Markosian, 2000; Kalnins et al., 2002, 2003], multi-resolution curves [Finkelstein and Salesin, 1994], skeletal strokes [Hsu et al., 1993; Hsu and Lee, 1994], and programmable line styles [Isenberg and Brennecke, 2006; Grabli et al., 2010]. These NPR styles are typically applied to contour and feature lines of 3D objects, aim to replicate marks made by traditional tools, and—if used in an illustration—may carry a meaning (such as parts being hidden). In our work, in contrast, we use line styles to specifically visualize data properties of streamlines in a flow.

Two concepts from the field of illustrative visualization are important for our work. The first is halos [Appel et al., 1979; Stoll et al., 2005; Tarini et al., 2006; Everts et al., 2009] that make objects (including lines) easier to discern from the background, thus improving depth perception. The second is the use of transfer functions, well known from volume rendering [Kniss et al., 2002], to display multiple styles in one visualization, e. g., as used for volumetric style transfer functions [Bruckner and Gröller, 2007]. In the context of flow visualization, other illustrative methods related to our work include stroke- and painting-inspired visualizations of 2D flow fields [Kirby et al., 1999; Li et al., 2008], illustrative 3D volume rendering [Svakhine et al., 2005], stylized streamlines [Li and Shen, 2007], as well as animated, dashed streamlines [Laramee and Hauser, 2005] and dashtubes [Fuhrmann and Gröller, 1998].

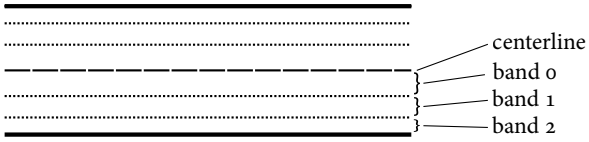



Figure 5.1: A view-oriented strip subdivided into a number of bands mirrored around the centerline.

## 5.3 LINE STYLES FOR VISUALIZATION

As we have seen, line-based flow visualizations are problematic due to their limited number of visual variables as well as the occlusion that is introduced if more than a few line primitives are being used. We address these two major issues by introducing an extended line style model for visualization purposes (this section) and the use of line style transfer functions (Section 5.4) based on our extended line model.

### 5.3.1 *Line Partitioning Into Line Bands*

Such an extended illustrative line style model needs to increase the number of visual variables to allow the specification and parameterization of a variety of effects that can be flexibly used in visualization. For this purpose and inspired by halo-based line visualizations [Everts et al., 2009], we partition the line strips that are used to render the data lines into several bands (Figure 5.1). By that separation we provide the granularity that is necessary to allow us to define visually distinguishable styles, each of the bands increasing the number of visual variables that can be controlled. These bands run parallel to the centerline and together define the visualization line style.

Specifically, we represent each line from the 3D dataset by a view-oriented line strip as done in many previous line-based rendering systems. This strip is subdivided into two mirrored sets of bands, one on each side of the line (Figure 5.1). The three visual properties that we control per band are color, width, and distance offset with respect to the viewer, each of which can be controlled independently. While the distance offset is not actually a *visual* property, it has an effect when used as a depth-dependent halo [Tarini et al., 2006; Everts et al., 2009]. In that case the halo line band is folded back, away from the viewer. The effect is that the *visible* width of the halo depends on the difference in distance between two lines with respect to the viewer, improving the depth perception. Therefore, our extended line style model can be seen as a generalization of the depth-dependent line halo technique [Everts et al., 2009].

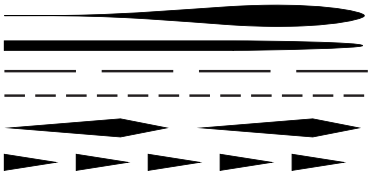


Figure 5.2: Example line shapes.

### 5.3.2 *Local Attribute Mapping*

This basic line model allows us to specify a wide range of visual effects, notably by its capability to convey information about the data in the visualization by mapping data attributes such as temperature, pressure, etc. to a line's visual properties. Specifically, we control each line style band's color and width based on the value of local scalar line attributes by means of mapping functions.

For the color property, this mapping is encoded in conventional color maps that assign input values $\in [0,1]$ to RGB colors. We provide a selection of pre-defined color maps from which the user can choose a color map most suitable for that particular attribute type and the desired visual style. Similar to controlling the color of a band a user may adjust the width of a band to convey more information in the visualization. For example, mapping local velocity to band width yields wide bands where the velocity is high and thin bands where the velocity is low. To control this mapping, both a minimum and a maximum value can be set for the band width.

### 5.3.3 *Flexible Band Shapes*

The control of the line width property of a band can also be used to create bands with repeating line shape patterns such as dashes, droplets, etc. (e. g., Figure 5.2). Moreover, the local density (or frequency) of these repeating shape patterns can be used to provide additional means for conveying local flow properties; this is particularly useful for the velocity property. For this purpose we employ (1) a *shape mapping function* and (2) a *dedicated line shape attribute*.

A *shape mapping function* maps a line data attribute ($\in [0,1]$) to the width ($\in [0,1]$) of the band at that point of a line. As such, it defines the shape of a band. We combine this mapping function with a *dedicated line shape attribute*:

$$s_x = \frac{x}{l} \bmod 1, \tag{5.1}$$

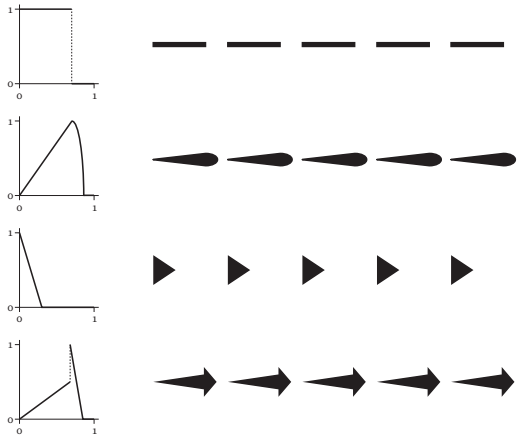


Figure 5.3: Shape mapping functions and corresponding line styles.

where $l$ defines the size of the shape pattern on the line and $x$ is a line attribute that is monotonically increasing along the line. The modulo operation ensures that the shape is repeated along the line.

The choice of $x$ in Eq. 5.1 determines the local 'density' of the patterns. For example, choosing the distance along the data line to the seed point results in constant size patterns. However, choosing the integration time $t$ makes the frequency of the patterns depend on the local velocity of the field: high velocity will result in a lower frequency of patterns (i. e., elongated patterns), providing additional means for visualizing the local velocity.

Together, the line shape attribute and the shape mapping function provide a flexible way to achieve a wide range of line shapes. Figure 5.3 shows a number of examples of mapping functions and illustrates how the mapping function influences the shape of a band and, thus, the line style.

### 5.3.4 *Directional Color Patterns*

The flexible band shapes (line band width mapping) affect the overall shape of the line as a whole. For example, an adjustment of the mapping of an inner band will always have an effect on the outer band layers. To provide a related visual effect without the influence of the overall line shape we provide—as the final aspect to our visualization line style model—*directional color patterns* realized as procedural textures that control the line color of the bands. These textures can be employed to produce similar shapes and patterns (e. g., Figure 5.4) as the flexible band shapes but without affecting the line width. Similar to the band shapes, the directional color patterns can also convey additional in-

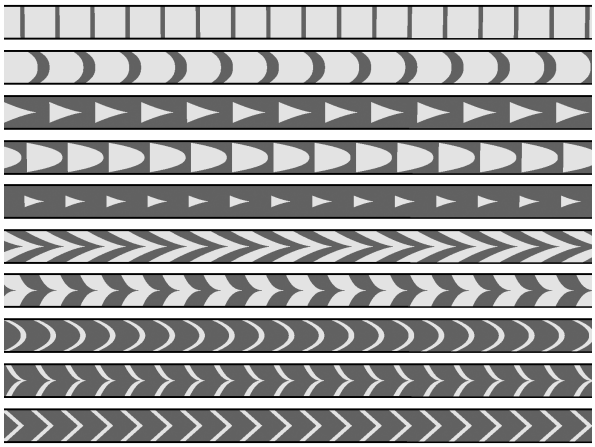


Figure 5.4: Examples of color patterns to apply to bands.

formation about the streamlines such as direction, velocity, and other parameters.

The approach we use to achieve this type of patterns is to apply a function that, based on the position on the band, determines which band color to use. For example, a decision function that captures the different aspects of the patterns illustrated in Figure 5.4 is:

$$f_d(x, y) = \left(\left(\frac{x}{l} + a\, y^c\right) \bmod 1\right) - w, \tag{5.2}$$

where $x$ is again a line attribute that is monotonically increasing along the line, defining the longitudinal position on the strip, $y \in [0,1]$ is the lateral position on the band, $l$ is the length of the repeating pattern, $a$ is the steepness of the slope, $c > 0$ defines the shape of the slope, and $w \in [0,1]$ determines the relative width of the colors. When $f_d$ is smaller than zero, one color is used, otherwise a second color. Similar to the band shape and centerline patterns, choosing $x$ to be the integration time in Eq. 5.2 allows the color pattern to convey velocity through the local frequency of the pattern.

### 5.3.5 *The Extended Line Style Model for Visualization*

Together, the line bands with their color and width control, the means to parametrize the band width with mapping functions, and the directional color patterns extend the number of visual variables available for visualization of line data. Most of these visual variables can be combined in one visualization and convey multiple aspects of the streamline data in one image.

For example, consider a line style with the following settings for its two bands. On the center-most bands, the temperature is mapped to

color via a color map. The resulting color is used in a directional color pattern to convey both velocity and directionality. The second band (on both sides) is used as a contour line and is simply black. The width of both bands depends on the pressure. In such a visualization, the viewer can discern four different aspects of the flow field (besides the path of the streamlines): temperature, pressure, velocity, and direction—all aspects discernible in the mapping.

This combination of information comes at a price: the resulting image may be overwhelming and cluttered. However, we do not necessarily need to use all visual variables for all streamlines. One could imagine that, depending on the illustrative visualization goal, we would like to see extra information about the flow only for streamlines that have certain properties (e. g., local attribute values). To cater to this need, we introduce *line style transfer functions* as described next.

## 5.4 LINE STYLE TRANSFER FUNCTIONS

Similar to regular transfer functions in volume rendering [Kniss et al., 2002], *line style transfer functions* assign pre-defined line styles to line regions with certain local line attributes. Because we use our extended model for these line styles, the styles assigned to an attribute range are typically not static such as traditional (NPR) line styles but themselves show changes in the attribute values. Therefore, line style transfer functions allow us to visualize attribute values on two levels: ranges of attribute values and changes within these ranges. Moreover, these line style transfer functions also facilitate the explicit control of abstraction in streamline visualization. By choosing a line style that maps fewer attributes using a less complex line style (which can be as simple as a single thin line, maybe with a dashing pattern) for regions of the dataset that are less important (as indicated by a given attribute), we can highlight important regions with a more elaborate style that encodes more attributes for a detailed visualization.

As an example of explicitly applying abstraction, let us consider a visualization user who is interested in the behavior of streamlines in high vorticity areas, but still would like to see the streamlines in the remainder as context. So he or she creates two line styles: one style with thick colormapped lines (to show temperature) combined with halos and one with thin, dashed, light gray lines. The line style transfer function is then configured such that high vorticity values use the first line style and low vorticity values use the latter. Thus, low vorticity lines are thin and dashed so one can see through them, yet they provide context for the more pronounced high vorticity regions of the dataset.

Conceptually, a line style transfer function is a simple mapping from ranges of line attributes to line styles. As such, because the transfer

function depends on local line attributes, one line in the visualization can have multiple line styles.

## 5.5 IMPLEMENTATION

Several design decisions of the conceptual line model and transfer functions were driven by implementation considerations. More specifically, because we aim for the interactive exploration of line styles even when applied to large datasets, the line style model should be suitable for implementation in shaders on modern GPUs. Our implementation consists of two parts, each implemented in a different type of shader. The first is the generation of view-oriented triangle strips (geometry shader) and the second is the application of the line style to the strip (fragment shader).

The transformation of the input lines (stored in GPU memory) into view-oriented triangle strips is done each rendering pass in a geometry shader. The width of these triangle strips is (pre-)calculated by multiplying a global scaling factor with the maximum of the line style widths. The width of a line style is calculated through a summation of the maximum widths of its bands. With the line strips in place as the 'canvas' for the line style, the next step is to apply the style model.

The actual application of the line style is done in a fragment shader. The main goal of this fragment shader is to decide which band of which line style should be applied to the fragment. To determine this it uses the position on the strip, the shape mapping functions, and the values of the relevant line attributes. Then, based on the settings for that band, the color (either from a color map or from a color pattern) and the depth offset of the fragment can be determined.

One additional aspect of our implementation is the use of templated shaders. The main reason for this is that the flexibility of our extended line style model yields a large number of options, which without templated shaders would result in a large number of expensive conditional statements in the shader. The templated shaders (implemented using the existing templating library Jinja2, see http://jinja.pocoo.org/docs/) allow us to flexibly include only the necessary shader code, based on the chosen style configurations. This approach has the additional benefit of making the shader used for rendering as small as possible.

## 5.6 RESULTS

To illustrate the broad range of possible visual representations of lines that can be achieved with our line style model, we apply a number of different line styles to two sets of streamlines. The first set is generated from a snapshot of a numerical simulation of a heat driven cavity (Dataset 1), the other set is generated from a snapshot of a simulation

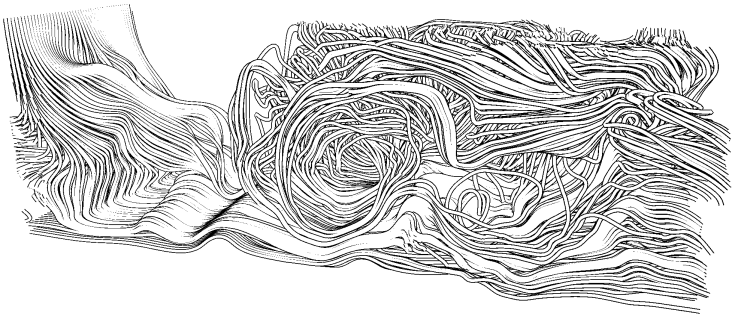


Figure 5.5: A simple black-and-white line style applied to streamlines from Dataset 1.

of turbulent flow around a cube (Dataset 2). It is important to note that although the streamlines that we visualize here may give the impression of a steady flow, they are merely a visualization of the flow field at one particular time step.

We start with the application of a single simple black-and-white line style to Dataset 1 (see Figure 5.5). This line style has two bands. The center-most band is white and fairly wide, whereas the outer band is thin and black. In addition, this outer band acts as a depth-dependent halo, although in this case it can also be considered a depth-dependent contour. The first thing to notice in Figure 5.5 is how, despite the fact that no color has been used, the spatial relationships of the lines are still clear. Also, the depth manipulation ensures that collinear streamlines (e. g., the laminar flow at the bottom) blend together, emphasizing such collinear structures and yielding a crisper visualization. The close-up of the same dataset in Figure 5.6 illustrates this aspect further.

The next step is to employ the visual parameters that our line style model introduces to convey additional information about the flow. Figures 5.7 and 5.8 show the application of a color map to streamlines. In Figure 5.7 a blue-purple color map is used to display velocity in Dataset 2; and in Figure 5.8 a grayscale color map is used to convey local temperature in Dataset 1. Again, the halo allows us to omit shading and still have good depth perception, making direct application of color maps possible without a potential shading that affects the perception of the colors.

Besides color maps, the other way our line style model can convey additional information is through the size and frequency of shape patterns. Figure 5.9 illustrates how an arrow shape can be used to convey both direction and velocity in a black-and-white visualization. Shape and color maps can also be combined, as illustrated in Figure 5.10, where light gray arrows are combined with a fairly wide halo to which a color map is applied. Again the size (length) of a pattern indicates the local velocity of the flow. Besides an indication of direction, the arrow

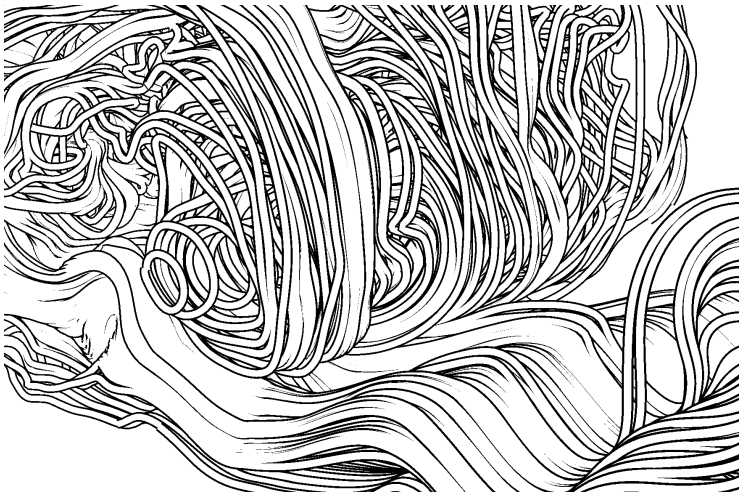


Figure 5.6: Close-up of streamlines with a simple black-and-white line style applied to them. Notice how the depth-dependent halo (contour) emphasizes collinear streamlines.

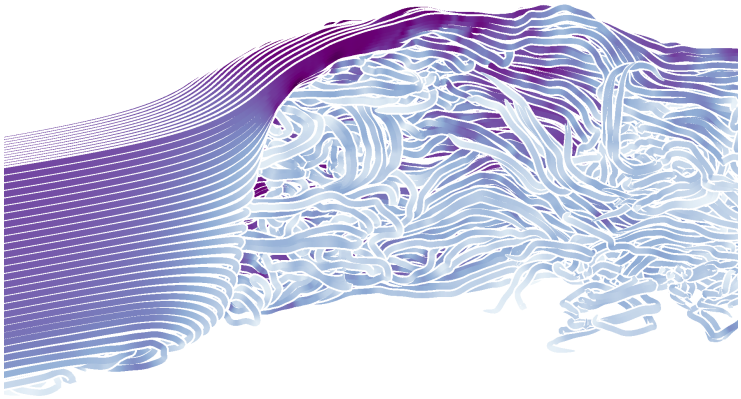


Figure 5.7: Streamlines depicting flow around a cube, colored with a blue-purple color map to show velocity, combined with white halos for depth perception.

shape also gives the visualization a certain feel of motion. A similar effect is achieved with the tadpole shape shown in Figure 5.11 where also a color-mapped halo is used, but with a constant shape length.

The color pattern is the final addition to our line style model, of which an application is shown in Figure 5.12. The base color comes from a yellow-green color map conveying temperature, the alternate color is black. Together, these two colors form a directional pattern, whose length, as in the images shown previously, correlates with the

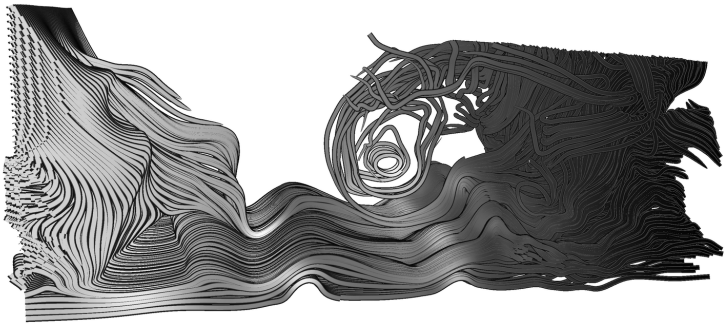


Figure 5.8: Gray scale color map applied to streamlines depicting their local temperature.

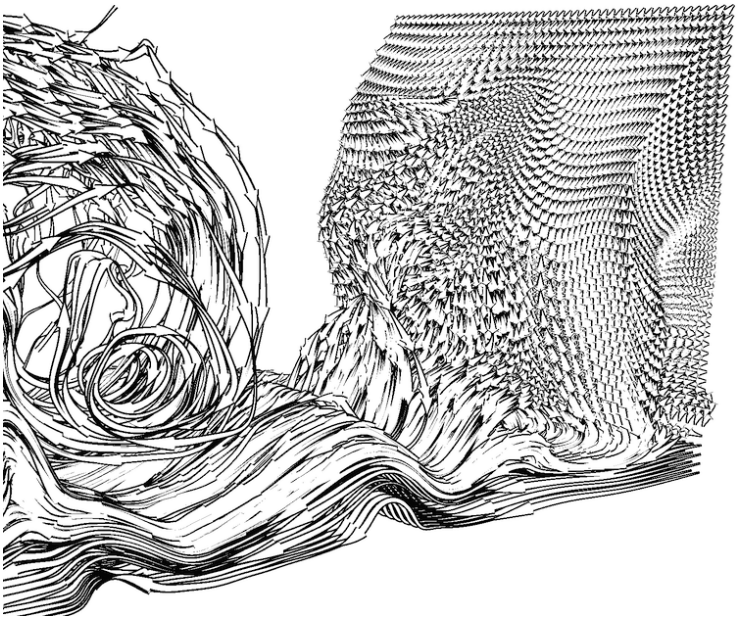


Figure 5.9: An arrow shape mapping function applied with a simple black-and-white style. The size of the arrow indicates velocity.

local velocity. This time, however, the width of the centermost band also depends on the velocity, emphasizing the effect.

Finally, in Figure 5.13 two line styles are shown in one visualization, illustrating the use of a line style transfer function. Here, the visualization or illustration goal is to study the breakup of laminar flow into more turbulent flow. Simply rendering all the lines (as in Figure 5.5) will obscure the transition areas from laminar to turbulent flow; however, by removing the vortex part the context would be lost. For this

Figure 5.10: Streamlines depicted through light gray arrow shapes combined with a halo colored with a color map to depict velocity.

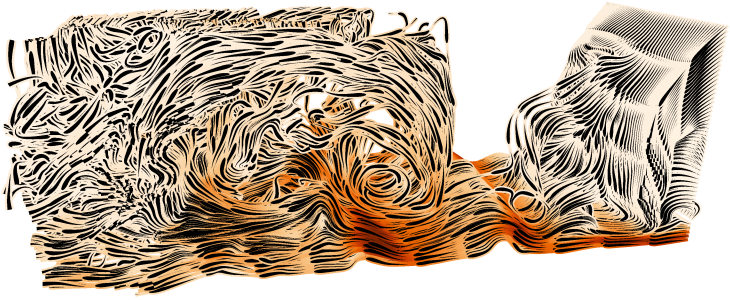


Figure 5.11: Streamlines depicted with tadpole-shaped, fixed size patterns, combined with a halo colored using a color map (velocity).

purpose we defined two line styles for this visualization. One, being the focus of the visualization, is fairly thick and has a color map applied to it (velocity); the other, for providing context, is light gray, thin, and dashed. For this example we use the distance-to-seed-point attribute as the guiding attribute for the line style transfer function, where low values get mapped to the more pronounced line style and high values are mapped to the thin dashed lines providing context. Because of the dashing, one can see through the context-providing lines, revealing the flow behind it.

## 5.7 DISCUSSION

As illustrated by the results in the previous section, our parametrization of line styles allows for a wide variety of visual representations of lines, accompanied by visual variables to show additional information about

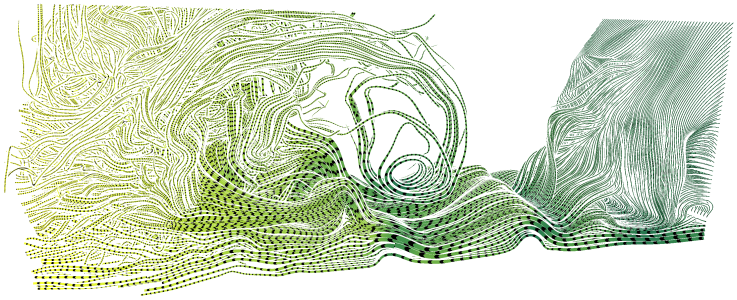


Figure 5.12: A color map (depicting temperature), combined with a color pattern whose control attribute is the integration time, yielding larger patterns where the velocity is high. The width of the band depends on the local velocity.

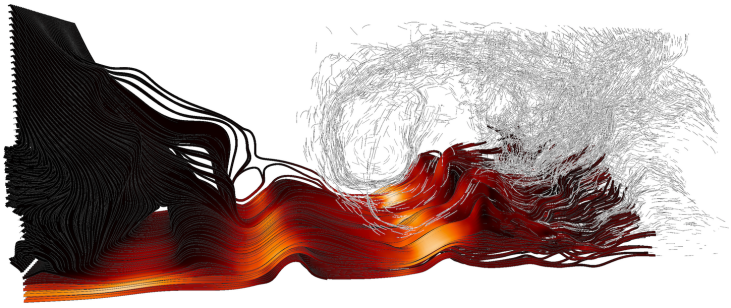


Figure 5.13: Application of a line style transfer function to emphasize the transition from laminar to turbulent flow while the gray dashed lines provide context.

the flow. In this section we discuss further aspects, observations, and limitations of our line style model.

In terms of performance, we find that on a fairly modern graphics card (NVIDIA GeForce GTX 285), we can interactively manipulate the line style parameters whilst displaying fairly large datasets, facilitating the interactive exploration of different visual representations of lines. For reference, the two datasets in Section 5.6 consist of 2500 streamlines (2.5M vertices) and 390 streamlines (250k vertices), respectively.

The repeating directional patterns in Figures 5.9, 5.10, 5.11, and 5.12 give a sense of motion to the visualizations. We experimented with this by generating animations where these patterns move along the line, further illustrating the motion (an example is included in the supplementary material). There is, however, a certain danger in this because it would give the false impression of a steady flow, whereas the datasets we use are a single snapshot of a simulation of unsteady flow. As an experiment to capture the behavior of an *unsteady flow* using our

streamline visualization, we generated line visualizations for a number of consecutive snapshots and combined those in a short animation (included in the supplementary material). Although the ends of the streamlines behave somewhat erratically, together, the streamlines seem to capture the behavior of this unsteady flow.

An additional observation is that in our visualizations where the length of a (shape) pattern depends on the local velocity, the patterns are longer in high velocity areas. Whether this effect is intuitive seems to depend on the people who are asked and the kind of shape being used, as some people correlate high (pattern) frequency with velocity. A related observation is that when the difference in velocity is large, the shape might become too small in low velocity areas, see for example the right side of Figure 5.9. Other rendering artefacts are possible, for example when (shaped) line strips overlap in a certain way, resulting in oddly shaped patterns. Also, occasionally there are small artefacts when the view-vector is parallel to the line direction, though the effect is minimal and methods exist to remedy this artifact [Stoll et al., 2005].

Finally, we presented our visualization results to a fluid mechanics expert in an informal discussion. In his initial reaction he commented on the "prettyness" of the images and found the visualizations very suitable for illustration purposes (e. g., classroom usage) because they illustrate well-known phenomena very well. Interestingly though, he liked the simple black-and-white visualizations (such as Figure 5.5) best, mainly because of their simplicity and expressive power.

## 5.8 CONCLUSION

We have presented a flexible illustrative line style model for the visualization of streamline datasets. By partitioning line strips into parallel bands whose basic visual properties can be independently controlled, we create a parametrization that allows us to represent a broad range of visual styles for line data visualization. This approach is combined with line attribute mapping functions for color and width to facilitate flexible line shapes and means to convey additional information about the flow. Moreover, the line style transfer function we introduced enables emphasis and abstraction by mapping local line attributes to pre-defined line styles.

Future work includes combining our exploration of line styles with interactive streamline seeding strategies to further improve the exploration of flow datasets for visualization and illustration. Furthermore, we would like to experiment with alternative automatic line style mapping strategies to create more flexible line style transfer functions.

## 5.9 ACKNOWLEDGEMENTS

We thank Roel Verstappen and F. Xavier Trias Miquel for the datasets as well as their discussion and helpful feedback.

This chapter is based on: Maarten H. Everts, Henk Bekker, Jos B. T. M. Roerdink, and Tobias Isenberg. Illustrative Line Styles for Flow Visualization. In *Short Paper Proceedings of the 19th Pacific Conference on Computer Graphics and Applications* (Pacific Graphics 2011, September 21–23, 2011, Kaohsiung, Taiwan). Goslar, Germany. Eurographics Association, pages 105–110, 2011.

# 6 CONCLUSION

This thesis has presented a quartet of approaches and techniques for the visualization of lines and line-like data. Two of those approaches (Chapter 2 and Chapter 4) focused on the analysis of the data, whereas the other two techniques (Chapter 3 and Chapter 5) addressed visualization and rendering aspects. Nonetheless, all four methods share a common goal of creating abstracted visual representations of line data that help viewers to gain a better understanding of the data. This chapter summarizes the contributions of this thesis and discusses perspectives on future work.

## 6.1 GRAPH-BASED TRACTOGRAPHY

In Chapter 2 we introduced a new method for calculating fiber tract paths from DTI data. Central to this method is the application of Dijkstra's shortest path algorithm to a large graph derived from DTI data in which each node represents a voxel and the edge weights represent the white matter connectivity between neighboring voxels. We investigated our hypothesis, that by selecting a proper mapping for the edge weights, the resulting shortest paths would represent likely fiber tract trajectories. Besides the most likely path between two points, Dijkstra's algorithm produces a tree structure of shortest paths, annotated with path weights, which can be the source of further visualization. For example, we showed that a (pruned) shortest path tree has a strong resemblance with the expected white matter structure and gives a good first quick overview, although further validation is needed. This shortest path approach is a first step that provides possibilities for further study.

One of the main issues of our shortest path approach for creating fiber tracts is that in our conversion from DTI data to edge weights, the resulting values become unitless, making it hard to reason about the paths found. Zalesky [2008] solved this problem by making a slight modification of Dijkstra's algorithm, allowing him to use probabilities for edge weights resulting in most probable paths, with accompanying probabilities. Such a probabilistic approach combined with the continuation of our exploration of visualization possibilities for the shortest path tree could yield interesting results.

Another issue is the coarseness of the paths that is produced by such a graph-based approach. One possible solution is to upsample the underlying data to a higher resolution and produce graphs for that. Although from an information-theoretic standpoint the amount of

available information would not change, it should yield less jagged trajectories. Moreover, as Dijkstra's algorithm on such regular graphs is fast, we expect that applying it to higher resolution graphs will not result in prohibitively long running times.

## 6.2 DEPTH-DEPENDENT HALOS FOR LINES

In Chapter 3 the focus changed to visualization and rendering. We presented a new illustrative visualization technique for dense line datasets such as DTI fiber tracts or flow streamlines. We showed that the depth-dependent halos we added to simple black lines improve depth perception and visually emphasize collinear structures of lines (e. g., bundles and surfaces that implicitly exist in the line data). Combined with additional depth-cueing, the visualizations we produced with this method mimic the clear pen-and-ink illustrations that are created by human illustrators as hand-drawn images. The effectiveness of our technique was supported by positive comments in an informal evaluation by domain experts who particularly noted the possibilities in showing detail, emphasizing bundles, and depicting spatial relationships. In addition, our GPU implementation of this technique allowed us to achieve interactive to real-time frame rates, depending on the size of the dataset.

Our informal evaluation with DTI fiber tract domain experts about the depth-dependent halo technique for lines yielded positive feedback, but one important comment was the request for showing context together with the fiber tracts. This addition of context has since been realized [Svetachov et al., 2010]. Figure 6.1 shows an illustration where depth-dependent halo lines are combined with hatched and stippled surfaces to create context in a similarly clear black-and-white style.

## 6.3 FIBER TRACT BUNDLING

After noticing that bundles and surfaces of fiber tracts are emphasized by the depth-dependent halos in Chapter 3, we aimed to produce more explicit abstractions for these implicitly existing bundles and surfaces in Chapter 4. We created these abstractions in a two-step process. The first step consists of analyzing the local similarity of tract segment directions at different search ranges. Then, locally similar segments are iteratively moved towards each other. This bundling process creates volumetric voids between the tracts which decreases the mutual occlusion of tracts and gives insight into the general white matter structure. We explored a number of ways to interact with these abstracted representations, including one where we allowed a seamless transition between the original and abstracted representations.

As discussed in Chapter 4, one of the main opportunities for future work for our fiber tract bundling technique lies in reducing the running

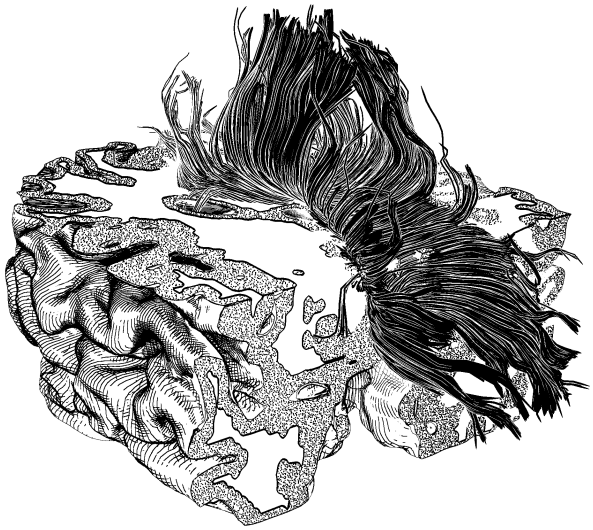


Figure 6.1: Example of combining depth-dependent halo fiber tracts with hatched and stippled surfaces for context. From [Svetachov et al., 2010].

time and computational complexity of the pre-processing steps. There are certainly possibilities for achieving that, including the use of space partitioning data-structures and optimizations in the iterative bundling process. In addition, it is important to further analyze the deformation of shape that the bundling process introduces and to validate the abstract representations we produce with domain experts.

Another avenue for future work with respect to bundling is *interaction*. The interaction possibilities we introduced so far focused mainly on exploring the abstracted representations and comparing them with the original tract configuration. We believe that, besides being an abstraction that helps communicating the important structures in the brain, the bundled representation can be a tool for exploring large datasets where the bundles provide handles for selection and interaction.

## 6.4 LINE STYLES FOR FLOW VISUALIZATION

In Chapter 5 our focus returned to rendering and visualization to introduce a flexible illustrative line style model for the visualization of streamlines. In this model, a generalization of the depth-dependent halo technique of Chapter 3, we subdivide line strips into parallel bands whose visual properties can be controlled individually. We further extended the range of possible depictions by allowing local line attributes (e.g., velocity or temperature) to be mapped to visual properties (no-

tably width and color). In addition, we introduced the concept of a line style transfer function, which can map local line attributes to predefined line styles, giving users the ability to emphasize certain lines. Moreover, because this model was conceived with GPUs in mind, our GPU implementation supports fairly large datasets.

For this work on line styles there is also room for extension with respect to interacting with the visualization. Of course, we already provide interactive manipulation of the line styles, but the interaction with the lines itself is still limited. Right now, the line data is static and being able to influence the number and density of the lines would give the user more control over the visualization, for example by supporting interactive seeding strategies and/or interactive line-filtering strategies.

Furthermore, we have only touched the surface of what is possible with respect to the line style transfer function we introduced in Chapter 5. There are two important possibilities for further research. The first is—in collaboration with domain experts—to determine or define local line attributes best suited for line style mapping. The second is to create and explore alternative automatic line style mapping strategies that give users more flexibility in expressing how local line attributes should be mapped to line styles.

## 6.5 GENERAL CONCLUSIONS AND PERSPECTIVES

Lines are the geometric shapes central to this thesis and the vehicles for conveying information about the underlying data (e. g., DTI tensor fields and flow fields). Compared to alternative methods such as other geometric shapes [McLoughlin et al., 2010] or volumetric LIC approaches [Helgeland and Andreassen, 2004], lines have the advantage of being conceptually simple and being able to convey both general structure and detail. There is, of course, plenty of opportunity to combine (our) line visualization methods with other approaches, in particular in the light of providing context.

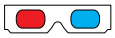
There lies, however, also a danger in being able to capture details; the clarity of simple (black) lines can give an impression of accuracy and certainty, without this certainty being backed-up by the underlying data. For example, the lines generated by deterministic DTI tractography are the result of a chain of measurements and transformations, each introducing inaccuracies. As such, it would be interesting to further explore whether it is possible to maintain the simplicity and abstractive power while, at the same time, conveying the degree of uncertainty. Using the line rendering technique of Chapter 5, it may be possible to convey this uncertainty through line styles. However, such an approach raises (new) questions about the perception of uncertainty. In general, the perceptual effects of line styles is something to be examined further.

Going from (raw) data to knowledge is typically a process of analysis, visualization, and interaction. As became clear in the previous section of this chapter, all three activities interconnect and support each other; advances in one area spur new challenges in the other, solutions to which create new possibilities for the former. When analysis, visualization, and interaction intuitively work together, a feedback loop is created that allows domain experts to really use the techniques to their full potential. Each of the chapters in this thesis contributes towards this goal, but there is much work to be done before we can put it in the hands of end-users.

# A

## APPENDIX: SUPPLEMENTAL IMAGES

This appendix contains a number of high-resolution anaglyphic result images of the Depth-Dependent Halo technique described in Chapter 3. These images are to be viewed with red-cyan or red-green glass (with the red filter used for the left eye).



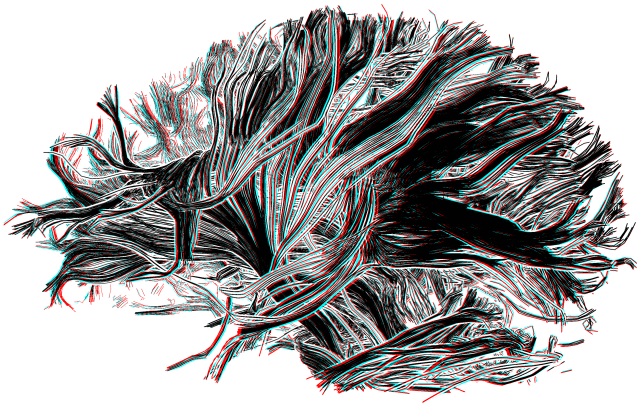


Figure A.1: Illustrative visualization of a subset of DTI fiber tracts in a human brain. Note the emphasis of compact fiber bundles and how the "fanning out" of these is clearly visualized with depth-dependent halos. The dataset comprises 11 306 tracts and 260 836 vertices.

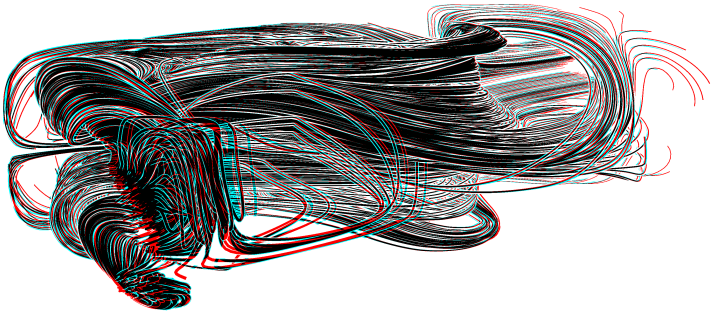


Figure A.2: Illustrative visualization of streamlines showing water flow. The dataset uses 1 400 streamlines and 2 603 605 vertices and was generated using VTK.

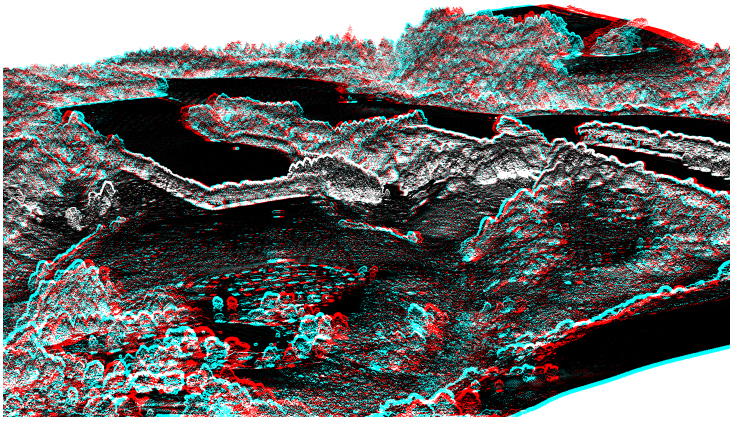


Figure A.3: Illustrative visualization of an elevation dataset with 4 440 900 points, rendered using points with halos. Note the trees being visible due to the halo effect, either as rows of trees (center and right) or even as individual trees (bottom left).

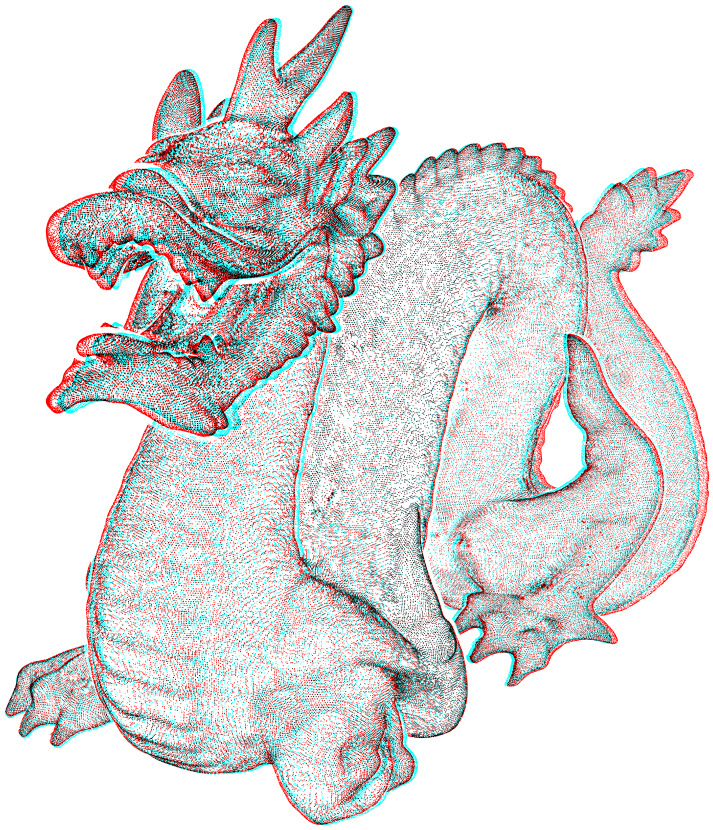


Figure A.4: Point dataset with 437 645 points representing the surface of a 3D shape. Notice the effect of visually separating contiguous regions of points at depth discontinuities.

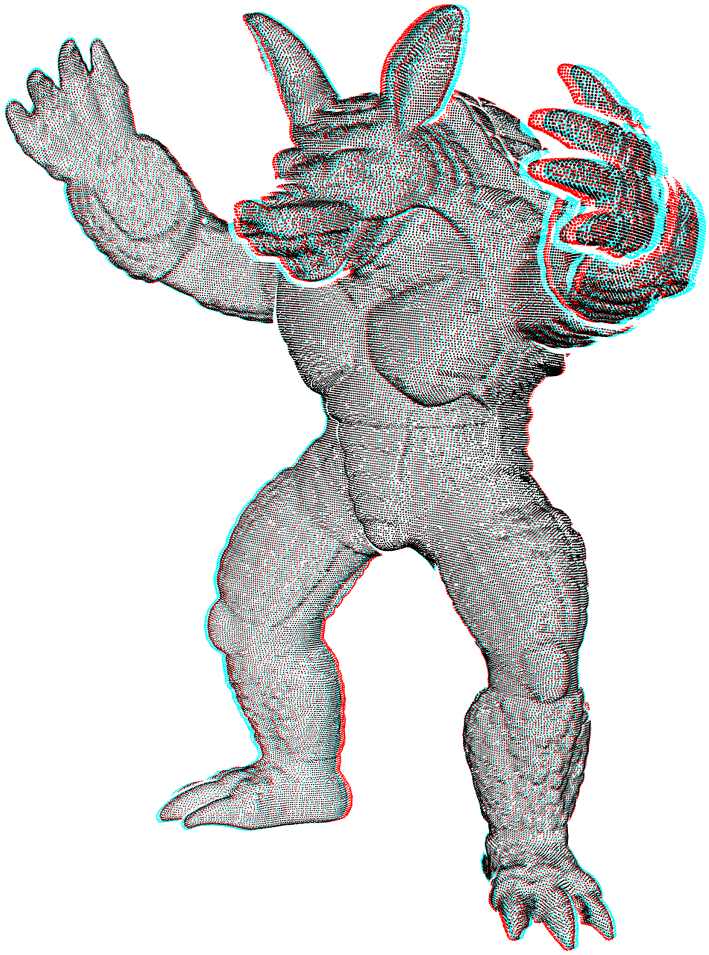


Figure A.5: Point dataset with 172 973 points representing the surface of a 3D shape. Notice the effect of visually separating contiguous regions of points at depth discontinuities.

# B

## APPENDIX: SUPPLEMENTAL MOVIES

THIS appendix contains links to number of online movies/videos that support the chapters in this thesis. For the convienence of those with a smartphone, the URLs are also placed in QR codes. Simply scan these 2D barcodes and you will be taken to a website showing the video.

Figure B.1: QR-code linking to an online movie showing the depth-dependent halo technique as shown in Chapter 3.
URL: http://youtu.be/UF4ZYDj2C5k?hd=1

Figure B.2: QR-code linking to the anaglyphic version of an online movie showing the depth-dependent halo technique as shown in Chapter 3.
URL: http://youtu.be/pbO_AAwNp5g?hd=1

# BIBLIOGRAPHY


D. Akers, A. Sherbondy, R. Mackenzie, R. Dougherty, and B. Wandell. Exploration of the Brain's White Matter Pathways with Dynamic Queries. In *Proc. Visualization*, pages 377–384, Los Alamitos, 2004. IEEE Computer Society. doi> 10.1109/VISUAL.2004.30

A. Appel. The Notion of Quantitative Invisibility and the Machine Rendering of Solids. In *Proc. 22nd ACM National Conference*, pages 387–393, New York, 1967. ACM. doi> 10.1145/800196.806007

A. Appel, F. J. Rohlf, and A. J. Stein. The Haloed Line Effect for Hidden Line Elimination. *ACM SIGGRAPH Computer Graphics*, 13(3):151–157, August 1979. doi> 10.1145/800249.807437

P. J. Basser, J. Mattiello, and D. LeBihan. MR Diffusion Tensor Spectroscopy and Imaging. *Biophysical Journal*, 66(1):259–267, January 1994. doi> 10.1016/S0006-3495(94)80775-1

T. Behrens, M. Woolrich, M. Jenkinson, H. Johansen-Berg, R. Nunes, S. Clare, P. Matthews, J. Brady, and S. Smith. Characterization and Propagation of Uncertainty in Diffusion-Weighted MR Imaging. *Magnetic Resonance in Medicine*, 50(5):1077–1088, 2003. doi> 10.1002/MRM.10609

E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose. Toolglass and Magic Lenses: The See-Through Interface. In *Proc. SIGGRAPH*, pages 73–80, 1993. doi> 10.1145/166117.166126

J. Blaas, C. P. Botha, B. Peters, F. M. Vos, and F. H. Post. Fast and Reproducible Fiber Bundle Selection in DTI Visualization. In *Proc. Visualization*, pages 59–64, Los Alamitos, 2005. IEEE Computer Society. doi> 10.1109/VIS.2005.40

M. Botsch and L. Kobbelt. High-Quality Point-Based Rendering on Modern GPUs. In *Proc. Pacific Graphics*, pages 335–343, Los Alamitos, 2003. IEEE Computer Society. doi> 10.1109/PCCGA.2003.1238275

S. Bruckner and E. Gröller. Enhancing Depth-Perception with Flexible Volumetric Halos. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1344–1351, 2007. doi> 10.1109/TVCG.2007.70555

S. Bruckner and E. Gröller. Style Transfer Functions for Illustrative Volume Rendering. *Computer Graphics Forum*, 26(3):715–724, September 2007. doi> 10.1111/J.1467-8659.2007.01095.X

A. Brun, H. Knutsson, H.-J. Park, M. E. Shenton, and C.-F. Westin. Clustering Fiber Tracts Using Normalized Cuts. In *Proc. MICCAI*, Lecture Notes in Computer Science, pages 368–375, Berlin, Heidelberg, 2004. Springer. doi> 10.1007/B100265

M. Burns, J. Klawe, S. Rusinkiewicz, A. Finkelstein, and D. DeCarlo. Line Drawings from Volume Data. *ACT Transactions on Graphics*, 24(3):512–518, July 2005. doi> 10.1145/1073204.1073222

B. Cabral and L. C. Leedom. Imaging Vector Fields using Line Integral Convolution. In *Proc. SIGGRAPH*, pages 263–270, New York, 1993. ACM. doi> 10.1145/166117.166151

W. Chen, S. Zhang, S. Correia, and D. S. Ebert. Abstractive Representation and Exploration of Hierarchically Clustered Diffusion Tensor Fiber Tracts. *Computer Graphics Forum*, 27(3):1071–1078, May 2008. doi> 10.1111/J.1467-8659.2008.01244.X

W. Chen, Z. Ding, S. Zhang, A. MacKay-Brandt, S. Correia, H. Qu, J. A. Crow, D. F. Tate, Z. Yan, and Q. Peng. A Novel Interface for Interactive Exploration of DTI Fibers. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1433–1440, November/December 2009. doi> 10.1109/TVCG.2009.112

P. Cook, Y. Bai, S. Nedjati-Gilani, K. K. Seunarine, M. G. Hall, G. J. Parker, and D. C. Alexander. Camino: Open-Source Diffusion-MRI Reconstruction and Processing. In *Proc. ISMRM*, page 2759, May 2006.

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 2nd edition, September 2001. ISBN 0262032937.

O. Deussen and T. Strothotte. Computer-Generated Pen-and-Ink Illustration of Trees. In *Proc. SIGGRAPH*, pages 13–18, New York, 2000. ACM. doi> 10.1145/344779.344792

G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, Upper Saddle River, NJ, USA, 1999.

E. W. Dijkstra. A Note on Two Problems in Connexion With Graphs. *Numerische Mathematik*, 1(1):269–271, December 1959. doi> 10.1007/BF01386390

F. Dong, G. J. Clapworthy, H. Lin, and M. A. Krokos. Nonphotorealistic Rendering of Medical Volume Data. *IEEE Computer Graphics & Applications*, 23(4):44–52, July/August 2003. doi> 10.1109/MCG.2003.1210864

D. L. Dooley and M. F. Cohen. Automatic Illustration of 3D Geometric Models: Lines. In *Proc. I3D*, pages 77–82, New York, 1990. ACM. ISBN 0-89791-351-5. doi> 10.1145/91385.91422

D. Ebert and P. Rheingans. Volume Illustration: Non-Photorealistic Rendering of Volume Models. In *Proc. Visualization*, pages 195–202, Los Alamitos, 2000. IEEE Computer Society. doi> 10.1109/VISUAL.2000.885694

G. Elber. Line Illustrations ∈ Computer Graphics. *The Visual Computer*, 11(6):290–296, June 1995. doi> 10.1007/S003710050022

F. Enders, N. Sauber, D. Merhof, P. Hastreiter, C. Nimsky, and M. Stamminger. Visualization of White Matter Tracts with Wrapped Streamlines. In *Proc. Visualization*, pages 51–58, Los Alamitos, 2005. IEEE Computer Society. doi> 10.1109/VISUAL.2005.1532777

M. H. Everts, H. Bekker, J. B. T. M. Roerdink, and T. Isenberg. Depth-Dependent Halos: Illustrative Rendering of Dense Line Data. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1299–1306, November/December 2009. doi> 10.1109/TVCG.2009.138

A. Finkelstein and D. H. Salesin. Multiresolution Curves. In A. Glassner, editor, *Proc. SIGGRAPH*, Computer Graphics Proceedings, Annual Conference Series, pages 261–268, New York, 1994. ACM Press. ISBN 0-89791-667-0. doi> 10.1145/192161.192223

O. Friman, G. Farneback, and C.-F. Westin. A Bayesian Approach for Stochastic White Matter Tractography. *IEEE Transactions on Medical Imaging*, 25(8):965–978, August 2006. doi> 10.1109/TMI.2006.877093

A. Fuhrmann and E. Gröller. Real-time Techniques for 3D Flow Visualization. In *Proc. IEEE Visualization*, VIS '98, pages 305–312, Los Alamitos, 1998. IEEE Computer Society Press. ISBN 1-58113-106-2. doi> 10.1109/VISUAL.1998.745317

S. Grabli, E. Turquin, F. Durand, and F. X. Sillion. Programmable Rendering of Line Drawing from 3D Scenes. *ACM Transactions on Graphics*, 29:18:1–18:20, April 2010. doi> 10.1145/1731047.1731056

M. Gross and H. Pfister, editors. *Point-Based Graphics*. Elsevier, 2007. ISBN 978-0-12-370604.

P. Hagmann, J.-P. Thiran, L. Jonasson, P. Vandergheynst, S. Clarke, P. Maeder, and R. Meuli. DTI Mapping of Human Brain Connectivity: Statistical Fibre Tracking and Virtual Dissection. *NeuroImage*, 19(3): 545–554, July 2003. doi> 10.1016/S1053-8119(03)00142-3

P. Hagmann, L. Cammoun, X. Gigandet, R. Meuli, C. J. Honey, V. J. Wedeen, and O. Sporns. Mapping the Structural Core of Human Cerebral Cortex. *PLoS Biology*, 6(7):1479–1493, July 2008. doi> 10.1371/JOURNAL.PBIO.0060159

H. Hauser, L. Mroz, G. I. Bischi, and M. E. Gröller. Two-Level Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):242–252, July–September 2001. doi> 10.1109/2945.942692

A. Helgeland and O. Andreassen. Visualization of Vector Fields Using Seed LIC and Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 10(6):673–682, November/December 2004. doi> 10.1109/TVCG.2004.49

A. Hertzmann and D. Zorin. Illustrating Smooth Surfaces. In *Proc. SIGGRAPH*, pages 517–526, New York, 2000. ACM. doi> 10.1145/344779.345074

M. Hlawitschka, C. Garth, X. Tricoche, G. Kindlmann, G. Scheuermann, K. I. Joy, and B. Hamann. Direct Visualization of Fiber Information by Coherence. *International Journal of Computer Assisted Radiology and Surgery*, 5(2):125–131, April 2010. doi> 10.1007/S11548-009-0302-5

D. Holten. Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, September/October 2006. doi> 10.1109/TVCG.2006.147

D. Holten and J. J. van Wijk. Force-Directed Edge Bundling for Graph Visualization. *Computer Graphics Forum*, 28(3):983–990, June 2009. doi> 10.1111/J.1467-8659.2009.01450.X

E. L. House and B. Pansky. *A Functional Approach to Neuroanatomy*. McGraw-Hill Book Company, New York, 1960.

S. C. Hsu and I. H. H. Lee. Drawing and Animation Using Skeletal Strokes. In *Proc. SIGGRAPH*, pages 109–118, New York, 1994. ACM. doi> 10.1145/192161.192186

S. C. Hsu, I. H. H. Lee, and N. E. Wiseman. Skeletal Strokes. In *Proc. UIST*, pages 197–206, New York, 1993. ACM. doi> 10.1145/16894.168662

V. Interrante and C. Grosch. Visualizing 3D Flow. *IEEE Computer Graphics & Applications*, 18(4):49–53, July 1998. doi> 10.1109/38.689664

T. Isenberg and A. Brennecke. G-Strokes: A Concept for Simplifying Line Stylization. *Computers & Graphics*, 30(5):754–766, October 2006. doi> 10.1016/J.CAG.2006.07.006

T. Isenberg, M. S. T. Carpendale, and M. C. Sousa. Breaking the Pixel Barrier. In *Proc. CAe*, pages 41–48. Eurographics Association, 2005. doi> 10.2312/COMPAESTH/COMPAESTH05/041-048

Y. Iturria-Medina, E. J. Canales-Rodriguez, L. Melie-Garcia, P. A. Valdes-Hernandez, E. Martinez-Montes, Y. Aleman-Gomez, and J. M. Sanchez-Bornot. Characterizing Brain Anatomical Connections Using Diffusion Weighted MRI and Graph Theory. *NeuroImage*, 36(3): 645–660, July 2007. doi> 10.1016/J.NEUROIMAGE.2007.02.012

R. Jianu, C. Demiralp, and D. Laidlaw. Exploring 3D DTI Fiber Tracts with Linked 2D Representations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1449–1456, November/December 2009. doi> 10.1109/TVCG.2009.141

S. E. Jones, B. R. Buchbinder, and I. Aharon. Three-Dimensional Mapping of Cortical Thickness Using Laplace's Equation. *Human Brain Mapping*, 11(8):12–32, August 2000. doi> 10.1002/1097-0193(200009) 11:1<12::AID-HBM20>3.0.CO;2-K

A. Joshi, J. Caban, P. Rheingans, and L. Sparling. Case Study on Visualizing Hurricanes Using Illustration-Inspired Techniques. *IEEE Transactions on Visualization and Computer Graphics*, 15(5):709–718, 2009. doi> 10.1109/TVCG.2008.105

R. D. Kalnins, L. Markosian, B. J. Meier, M. A. Kowalski, J. C. Lee, P. L. Davidson, M. Webb, J. F. Hughes, and A. Finkelstein. WYSIWYG NPR: Drawing Strokes Directly on 3D Models. *ACM Transactions on Graphics*, 21(3):755–762, July 2002. doi> 10.1145/566654.566648

R. D. Kalnins, P. L. Davidson, L. Markosian, and A. Finkelstein. Coherent Stylized Silhouettes. *ACM Transactions on Graphics*, 22(3): 856–861, July 2003. doi> 10.1145/882262.882355

M. Kaplan. Hybrid Quantitative Invisibility. In *Proc. NPAR*, pages 51–52, New York, 2007. ACM. ISBN 978-1-59593-624-0. doi> 10.1145/1274871.1274879

G. Kindlmann, X. Tricoche, and C.-F. Westin. Delineating White Matter Structure in Diffusion Tensor MRI with Anisotropy Creases. *Medical Image Analysis*, 11(5):492–502, October 2007. doi> 10.1016/J.MEDIA.2007.07.005

R. M. Kirby, H. Marmanis, and D. H. Laidlaw. Visualizing Multivalued Data from 2D Incompressible Flows Using Concepts from Painting. In *Proc. IEEE Visualization*, pages 333–340, Los Alamitos, 1999. IEEE Computer Society. doi> 10.1109/VISUAL.1999.809905

J. Klein, F. Ritter, H. K. Hahn, J. Rexilius, and H.-O. Peitgen. Brain Structure Visualization using Spectral Fiber Clustering. In *Research Posters of SIGGRAPH*, article no. 168, New York, 2006. ACM. doi> 10.1145/1179622.1179816

J. Kniss, G. Kindlmann, and C. Hansen. Multidimensional Transfer Functions for Interactive Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):270–285, 2002. doi> 10.1109/TVCG.2002.1021579

M. A. Koch, D. G. Norris, and M. Hund-Georgiadis. An Investigation of Functional and Anatomical Connectivity Using Magnetic Resonance Imaging. *NeuroImage*, 16(1):241–250, May 2002. doi> 10.1006/NIMG.2001.1052

R. S. Laramee and H. Hauser. Geometric Flow Visualization Techniques for CFD Simulation Data. In *Proc. SCCG*, pages 221–224, New York, 2005. ACM. ISBN 1-59593-204-6. doi> 10.1145/1090122.1090158

R. S. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F. H. Post, and D. Weiskopf. The State of the Art in Flow Visualization: Dense and Texture-Based Techniques. *Computer Graphics Forum*, 23(2): 203–221, June 2004. doi> 10.1111/J.1467-8659.2004.00753.X

M. Lazar, D. M. Weinstein, J. S. Tsuruda, K. M. Hasan, K. Arfanakis, M. E. Meyerand, B. Badie, H. A. Rowley, V. Haughton, A. Field, and A. L. Alexander. White Matter Tractography Using Diffusion Tensor Deflection. *Human Brain Mapping*, 18(4):306–321, April 2003. doi> 10.1002/HBM.10102

J. S. Lee, M.-K. Han, S. H. Kim, O.-K. Kwon, and J. H. Kim. Fiber Tracking by Diffusion Tensor Imaging in Corticospinal Tract Stroke: Topographical Correlation With Clinical Symptoms. *NeuroImage*, 26(3):771–776, July 2005. doi> 10.1016/J.NEUROIMAGE.2005.02.036

L. Li and H.-W. Shen. Image-based streamline generation and rendering. *IEEE Transactions on Visualization and Computer Graphics*, 13:630–640, May/June 2007. doi> 10.1109/TVCG.2007.1009

L. Li, H.-H. Hsieh, and H.-W. Shen. Illustrative Streamline Placement and Visualization. In *Proc. PacificVIS*, pages 79–86, 2008. doi> 10.1109/PACIFICVIS.2008.4475462

T. Luft, C. Colditz, and O. Deussen. Image Enhancement by Unsharp Masking the Depth Buffer. *ACM Transactions on Graphics*, 25(3): 1206–1213, July 2006. doi> 10.1145/1141911.1142016

K.-L. Ma, G. Schussman, B. Wilson, K. Ko, J. Qiang, and R. Ryne. Advanced Visualization Technology for Terascale Particle Accelerator

Simulations. In *Proc. Supercomputing*, pages 19–30, Los Alamitos, 2002. IEEE Computer Society. doi> 10.1109/SC.2002.10007

T. McLoughlin, R. S. Laramee, R. Peikert, F. H. Post, and M. Chen. Over Two Decades of Integration-Based, Geometric Flow Visualization. *Computer Graphics Forum*, 29(6):1807–1829, September 2010. doi> 10.1111/J.1467-8659.2010.01650.X

Z. Melek, D. Mayerich, C. Yuksel, and J. Keyser. Visualization of Fibrous and Thread-like Data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1165–1172, September/October 2006. doi> 10.1109/TVCG.2006.197

E. R. Melhem, S. Mori, G. Mukundan, M. A. Kraut, M. G. Pomper, and P. C. van Zijl. Diffusion Tensor MR Imaging of the Brain and White Matter Tractography. *American Journal of Roentgenology*, 178 (1):3–16, January 2002.

D. Merhof, M. Sonntag, F. Enders, V. P. Hastreiter, R. Fahlbusch, C. Nimsky, and G. Greiner. Visualization of Diffusion Tensor Data Using Evenly Spaced Streamlines. In *Vision, Modelling and Visualization*, pages 79–86, 2005.

D. Merhof, M. Meister, E. Bingöl, C. Nimsky, and G. Greiner. Isosurface-Based Generation of Hulls Encompassing Neuronal Pathways. *Stereotactic and Functional Neurosurgery*, 87(1):50–60, February 2009. doi> 10.1159/000195720

B. Moberts, A. Vilanova, and J. J. van Wijk. Evaluation of Fiber Clustering Methods for Diffusion Tensor Imaging. In *Proc. Visualization*, pages 65–72, Los Alamitos, 2005. IEEE Computer Society. doi> 10.1109/VIS.2005.29

S. Mori, B. J. Crain, V. P. Chacko, and P. C. van Zijl. Three-Dimensional Tracking of Axonal Projections in the Brain by Magnetic Resonance Imaging. *Annals of Neurology*, 45(2):265–269, February 1999. doi> 10.1002/1531-8249(199902)45:2<265::AID-ANA21>3.0.CO;2-3

S. Mori and P. C. M. van Zijl. Fiber Tracking: Principles and Strategies – A Technical Review. *NMR in Biomedicine*, 15(7-8):468–480, November–December 2002. doi> 10.1002/NBM.781

Z. Nagy, J. Schneider, and R. Westermann. Interactive Volume Illustration. In B. Girod, H. Niemann, H.-P. Seidel, G. Greiner, and T. Ertl, editors, *Proc. Vision, Modeling and Visualization*, pages 497–504, Berlin, 2002. Akademische Verlagsgesellschaft Aka GmbH.

P. Neumann, T. Isenberg, and S. Carpendale. NPR Lenses: Interactive Tools for Non-Photorealistic Line Drawings. In *Proc. Smart Graphics*,

pages 10–22, Berlin, Heidelberg, 2007. Springer-Verlag. doi> 10.1007/978-3-540-73214-3_2

J. D. Northrup and L. Markosian. Artistic Silhouettes: A Hybrid Approach. In *Proc. NPAR*, pages 31–37, New York, 2000. ACM. doi> 10.1145/340916.340920

L. O'Donnell and C.-F. Westin. Automatic Tractography Segmentation Using a High-Dimensional White Matter Atlas. *IEEE Transactions on Medical Imaging*, 26(11):1562–1575, 2007. doi> 10.1109/TMI.2007.906785

V. Petrovic, J. Fallon, and F. Kuester. Visualizing Whole-Brain DTI Tractography with GPU-based Tuboids and LoD Management. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1488–1495, November/December 2007.

H. Pfister, M. Zwicker, J. van Baar, and M. Gross. Surfels: Surface Elements as Rendering Primitives. In *Proc. SIGGRAPH*, pages 335–342, New York, 2000. ACM. doi> 10.1145/344779.344936

F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch. Feature Extraction and Visualization of Flow Fields. In *Eurographics 2002 State of the Art Reports*, pages 69–100. Eurographics Assoc., Aire-la-Ville, Switzerland, 2002.

P. Rautek, S. Bruckner, E. Gröller, and I. Viola. Illustrative Visualization: New Technology or Useless Tautology? *ACM SIGGRAPH Computer Graphics*, 42(3):4:1–4:8, August 2008. doi> 10.1145/1408626.1408633

F. Ritter, C. Hansen, V. Dicken, O. Konrad, B. Preim, and H.-O. Peitgen. Real-Time Illustration of Vascular Structures. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):877–884, September/October 2006. doi> 10.1109/TVCG.2006.172

S. E. Rose, A. L. Janke, and J. B. Chalk. Gray and White Matter Changes in Alzheimer's Disease: A Diffusion Tensor Imaging Study. *Journal of Magnetic Resonance Imaging*, 27(1):20–26, January 2008. doi> 10.1002/JMRI.21231

S. Schlechtweg, B. Schönwälder, L. Schumann, and T. Strothotte. Surfaces to Lines: Rendering Rich Line Drawings. In *Proc. WSCG*, volume 2, pages 354–361, 1998.

R. Schmidt, T. Isenberg, P. Jepp, K. Singh, and B. Wyvill. Sketching, Scaffolding, and Inking: A Visual History for Interactive 3D Modeling. In *Proc. NPAR*, pages 23–32, New York, 2007. ACM. doi> 10.1145/1274871.1274875

T. Schultz, N. Sauber, A. Anwander, H. Theisel, and H.-P. Seidel. Virtual Klingler Dissection: Putting Fibers into Context. *Computer Graphics Forum*, 27(3):1063–1070, May 2008. doi> 10.1111/j.1467-8659.2008.01243.x

T. Schultz, H. Theisel, and H.-P. Seidel. Topological Visualization of Brain Diffusion MRI Data. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1496–1503, November/December 2007. doi> 10.1109/tvcg.2007.70602

G. Schussman and K.-L. Ma. Scalable Self-Orienting Surfaces: A Compact, Texture-Enhanced Representation for Interactive Visualization of 3D Vector Fields. In *Proc. Pacific Graphics*, pages 356–365, Los Alamitos, 2002. IEEE Computer Society. doi> 10.1109/pccga.2002.1167879

H.-W. Shen, U. Bordoloi, and G.-S. Li. Interactive Visualization of Three-Dimensional Vector Fields With Flexible Appearance Control. *IEEE Transactions on Visualization and Computer Graphics*, 10(4):434–445, July/August 2004. doi> 10.1109/tvcg.2004.13

S. M. Smith, M. Jenkinson, H. Johansen-Berg, D. Rueckert, T. E. Nichols, C. E. Mackay, K. E. Watkins, O. Ciccarelli, M. Z. Cader, P. M. Matthews, and T. E. Behrens. Tract-Based Spatial Statistics: Voxelwise Analysis of Multi-Subject Diffusion Data. *Neuroimage*, 31(4):1487–1505, July 2006. doi> 10.1016/j.neuroimage.2006.02.024

S. M. Smith, M. Jenkinson, M. W. Woolrich, C. F. Beckmann, T. E. J. Behrens, H. Johansen-Berg, P. R. Bannister, M. De Luca, I. Drobnjak, D. E. Flitney, R. K. Niazy, J. Saunders, J. Vickers, Y. Zhang, N. De Stefano, J. M. Brady, and P. M. Matthews. Advances in Functional and Structural MR Image Analysis and Implementation as FSL. *NeuroImage*, 23 Suppl 1:S208–19, 2004. doi> 10.1016/j.neuroimage.2004.07.051

C. Stoll, S. Gumhold, and H.-P. Seidel. Visualization with Stylized Line Primitives. In *Proc. IEEE Visualization*, pages 695–702, Los Alamitos, 2005. IEEE Computer Society. doi> 10.1109/vis.2005.124

T. Strothotte, B. Preim, A. Raab, J. Schumann, and D. R. Forsey. How to Render Frames and Influence People. *Computer Graphics Forum*, 13(3):455–466, August 1994. doi> 10.1111/1467-8659.1330455

N. A. Svakhine, Y. Jang, D. S. Ebert, and K. Gaither. Illustration and Photography Inspired Visualization of Flows and Volumes. In *Proc. Visualization*, pages 687–694, Los Alamitos, 2005. IEEE Computer Society. doi> 10.1109/vis.2005.53

P. Svetachov, M. H. Everts, and T. Isenberg. DTI in Context: Illustrating Brain Fiber Tracts In Situ. *Computer Graphics Forum*, 29(3):1024–1032, June 2010. doi> 10.1111/j.1467-8659.2009.01692.x

M. Tarini, P. Cignoni, and C. Montani. Ambient Occlusion and Edge Cueing for Enhancing Real Time Molecular Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1237–1244, September/October 2006. doi> 10.1109/TVCG.2006.115

J.-D. Tournier, F. Calamante, D. G. Gadian, and A. Connelly. Diffusion-Weighted Magnetic Resonance Imaging Fibre Tracking Using a Front Evolution Algorithm. *NeuroImage*, 20(1):276–288, September 2003. doi> 10.1016/S1053-8119(03)00236-2

S. M. F. Treavett and M. Chen. Pen-and-Ink Rendering in Volume Visualisation. In *Proc. Visualization*, pages 203–210, Los Alamitos, 2000. IEEE Computer Society. doi> 10.1109/VISUAL.2000.885696

A. Tsai, C.-F. Westin, A. O. Hero, and A. S. Willsky. Fiber Tract Clustering on Manifolds With Dual Rooted-Graphs. In *Proc. CVPR*, Los Alamitos, 2007. IEEE Computer Society. doi> 10.1109/CVPR.2007.383096

D. S. Tuch. Q-ball Imaging. *Magnetic Resonance in Medicine*, 52(6):1358–1372, December 2004. doi> 10.1002/MRM.20279

S.-K. Ueng, C. Sikorski, and K.-L. Ma. Efficient Streamline, Streamribbon, and Streamtube Constructions on Unstructured Grids. *IEEE Transactions on Visualization and Computer Graphics*, 2(2):100–110, June 1996. doi> 10.1109/2945.506222

A. Vilanova, G. Berenschot, and C. van Pul. DTI Visualization with Stream Surfaces and Evenly-Spaced Volume Seeding. In O. Deussen, C. Hansen, D. A. Keim, and D. Saupe, editors, *EG/IEEE TCVG Symposium on Visualization*, pages 173–182, 2004.

I. Viola, A. Kanitsar, and M. E. Gröller. Importance-Driven Volume Rendering. In *Proc. Visualization*, pages 139–145, Los Alamitos, 2004. IEEE Computer Society. doi> 10.1109/VISUAL.2004.48

R. Wang and V. J. Wedeen. trackvis.org.

A. Wenger, D. F. Keefe, S. Zhang, and D. H. Laidlaw. Interactive Volume Rendering of Thin Thread Structures within Multivalued Scientific Data Sets. *IEEE Transactions on Visualization and Computer Graphics*, 10(6):664–672, November/December 2004. doi> 10.1109/TVCG.2004.46

G. A. Winkenbach and D. H. Salesin. Computer-Generated Pen-and-Ink Illustration. In *Proc. SIGGRAPH*, pages 91–100, New York, 1994. ACM. doi> 10.1145/192161.192184

A. Zalesky. DT-MRI Fiber Tracking: A Shortest Paths Approach. *Medical Imaging, IEEE Transactions on*, 27(10):1458–1471, 2008. doi> 10.1109/TMI.2008.923644

J. Zander, T. Isenberg, S. Schlechtweg, and T. Strothotte. High Quality Hatching. *Computer Graphics Forum*, 23(3):421–430, September 2004. doi> 10.1111/J.1467-8659.2004.00773.X

S. Zhang, C. Demiralp, and D. H. Laidlaw. Visualizing Diffusion Tensor MR Images Using Streamtubes and Streamsurfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(4):454–462, 2003. doi> 10.1109/TVCG.2003.1260740

S. Zhang, S. Correia, and D. H. Laidlaw. Identifying White-Matter Fiber Bundles in DTI Data Using an Automated Proximity-Based Fiber-Clustering Method. *IEEE Transactions on Visualization and Computer Graphics*, 14(5):1044–1053, September/October 2008. doi> 10.1109/TVCG.2008.52

L. Zhukov and A. H. Barr. Oriented Tensor Reconstruction. In C. D. Hansen and C. R. Johnson, editors, *The Visualization Handbook*, chapter 15, pages 313–326. Elsevier, Oxford, UK, 2004.

M. Zöckler, D. Stalling, and H.-C. Hege. Interactive Visualization of 3D-Vector Fields Using Illuminated Stream Lines. In *Proc. VIS*, pages 107–113, Los Alamitos, 1996. IEEE Computer Society. ISBN 0-89791-864-9. doi> 10.1109/VISUAL.1996.567777

# PUBLICATIONS

## JOURNAL PAPERS

Tobias Isenberg, Maarten H. Everts, and Jens Grubert and Sheelagh Carpendale. Interactive Exploratory Visualization of 2D Vector Fields. *Computer Graphics Forum*, 27(3):983–990, May 2008.

Maarten H. Everts, Henk Bekker, Jos B. T. M. Roerdink, and Tobias Isenberg. Depth-Dependent Halos: Illustrative Rendering of Dense Line Data. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1299–1306, November/December 2009. Best Paper Award at IEEE Visualization 2009.

Pjotr Svetachov, Maarten H. Everts, and Tobias Isenberg. DTI in Context: Illustrating Brain Fiber Tracts In Situ. *Computer Graphics Forum*, 29(3):1024–1032, June 2010.

Lingyun Yu, Pjotr Svetachov, Petra Isenberg, Maarten H. Everts, and Tobias Isenberg. FI3D: Direct-Touch Interaction for the Exploration of 3D Scientific Visualization Spaces. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1613–1622, November/December 2010.

## PEER-REVIEWED CONFERENCE PAPERS

Maarten H. Everts, Henk Bekker, and Jos B. T. M. Roerdink. Visualizing White Matter Structure of the Brain using Dijkstra's Algorithm. In Peter Zinterhof, Sven Lončarić, Andreas Uhl, and Alberto Carini, eds., *Proc. 6th International Symposium on Image and Signal Processing and Analysis* (ISPA 2009, September 16–18, Salzburg, Austria). Pages 575–580, 2009.

Maarten H. Everts, Henk Bekker, Jos B. T. M. Roerdink, and Tobias Isenberg. Illustrative Line Styles for Flow Visualization. In *Short Paper Proceedings of the 19th Pacific Conference on Computer Graphics and Applications* (Pacific Graphics 2011, September 21–23, 2011, Kaohsiung, Taiwan). Goslar, Germany. Eurographics Association, pages 105–110, 2011.

## WORKSHOP PAPERS AND POSTERS

Maarten H. Everts, Henk Bekker, Adrei Jalba, and Jos B. T. M. Roerdink. Particle Based Image Segmentation with Simulated Annealing. In *Proc. of the Thirteenth Annual Conference of the Advanced School for Computing and Imaging* (ASCI CONFERENCE 2007, June 13–15, 2007, Het Heijderbos, Heijen, The Netherlands). 2007.

Maarten H. Everts, Henk Bekker, Adrei Jalba, and Jos B. T. M. Roerdink. Particle Based Image Segmentation with Simulated Annealing. In *SIREN: Scientific ICT Research Event Netherlands*, 30 October 2007, TU Delft. 2007. Poster.

Tobias Isenberg, Jens Grubert, and Maarten H. Everts and Sheelagh Carpendale. Hands-On Analysis and Illustration: Interactive Exploratory Visualization of Vector Fields. In *Proc. of the Fourteenth Annual Conference of the Advanced School for Computing and Imaging* (ASCI Conference 2008, June 11–13, 2008, Heijen, The Netherlands). Pages 222–229, 2008.

Maarten H. Everts, Henk Bekker, and Jos B. T. M. Roerdink. Graph Algorithms for Fast Preview of Global Brain White Matter Structure. In *SIREN: Scientific ICT Research Event Netherlands*, 29 September 2008, VU Amsterdam. 2008. Poster.

Maarten H. Everts, Henk Bekker, Jos B. T. M. Roerdink, and Tobias Isenberg. Illustrative Rendering of Dense Line Data using Depth-Dependent Halos. In *SIREN: Scientific ICT Research Event Netherlands*, (November 5, 2009, University of Twente, The Netherlands). 2009. Poster. Best Poster Award at SIREN 2009.

Maarten H. Everts, Henk Bekker, Jos B. T. M. Roerdink, and Tobias Isenberg. Illustrative Rendering of Dense Line Data. In *Proc. of the Sixteenth Annual Conference of the Advanced School for Computing and Imaging* (ASCI Conference 2010, November 1–3, 2010, Veldhoven, The Netherlands). 2010. Paper number 13.

# SAMENVATTING

Visualisatie is de kunst en de wetenschap van het maken van plaatjes van grote verzamelingen gegevens op zo'n manier dat kijkers een nieuw en/of beter begrip krijgen van deze gegevens. Met de steeds maar groeiende rekenkracht van computers wordt de hoeveelheid gegevens ook steeds groter en speelt visualisatie een steeds prominentere rol in het verkrijgen van inzicht in deze gegevens. In dit proefschrift richten wij ons specifiek op de visualisatie van een bepaald type gegevens: verzamelingen van opeengepakte (gekromde) lijnen. Dergelijke gegevens komen in velerlei domeinen voor, waarvan er twee in dit proefschrift expliciet benoemd en behandeld worden.

Als eerste noemen we het medische domein: hierbij bekijken we zogenaamde fiber tracts die worden gegenereerd vanuit Diffusion Tensor Imaging (DTI) data. DTI is een variant van Magnetic Resonance Imaging (MRI) die het mogelijk maakt om de mate en de richting van diffusie van watermoleculen te meten in vezelachtig biologisch materiaal zoals de witte massa in de hersenen. De bijbehorende aanname is dat op basis van de diffusieinformatie iets gezegd kan worden over de ligging en de structuur van het vezelachtige materiaal. Een van de manieren om dat inzichtelijk te maken is door middel van de zogenaamde deterministische fiber tracking methoden. Het basisidee van deze methoden is om vanuit een startpunt een klein stapje te doen in de richting van de grootste diffusie op dat punt en vervolgens dit proces te herhalen om zo een driedimensionale kromme (een zogenaamde fiber tract) te verkrijgen. Door dit vanuit meerdere startpunten te doen is het resultaat een grote verzameling lijnen die kunnen helpen om inzicht te krijgen in de structuur en ligging van de (bundels van) vezels.

Het tweede domein met soortgelijke verzamelingen van lijnen is het deelgebied van de wiskunde dat zich bezig houdt met de simulatie van stromingen van vloeistoffen en gassen; denk bijvoorbeeld aan simulaties om de aerodynamica van een auto te onderzoeken. Het resultaat van dergelijke simulaties zijn snelheidsvelden waaruit stroomlijnen berekend kunnen worden op een vergelijkbare manier als bij de eerder beschreven fiber tracts. Deze stroomlijnen kunnen we zien als een (visuele) representatie van het gedrag en de structuur van de (vaak complexe) drie-dimensionale stroming. Zo'n voorstelling kan helpen bij de interpretatie van de simulatieresultaten.

Voor zowel fiber tracts als stroomlijnen geldt dat wanneer ze op een naïeve manier gevisualiseerd worden, er al gauw sprake zal zijn van een warboel van overlappende lijnen waaruit weinig inzicht te halen is. Daarom zijn in de loop van de jaren visualisatiemethoden ontwikkeld die helpen bij het inzichtelijk maken van de structuur en de onderlinge

ruimtelijke relaties van de lijnen. De verschillende hoofdstukken in dit proefschrift leveren bijdragen op dit gebied.

In hoofdstuk 2 ligt de nadruk op een nieuwe methode voor het extraheren van lijnen (fiber tracts) uit DTI data. Het centrale idee is om Dijkstra's kortste pad algoritme toe te passen op grote gewogen grafen die zijn afgeleid van DTI data. In deze grafen stellen de knopen de voxels voor (uit de DTI data) en stellen de gewogen verbindingen tussen de knopen de mate van connectiviteit tussen naastgelegen voxels voor. Onze hypothese was dat de resulterende kortste paden een goede indicatie kunnen geven van de meest waarschijnlijke banen van vezelbundels in de hersensen. Op basis van eerste experimenten met deze methode hebben we voorzichtig kunnen concluderen dat deze hypothese aannemelijk blijkt te zijn, maar dat verder onderzoek noodzakelijk is.

In hoofdstuk 3 richten we ons meer op de *visualisatie* van lijnen. We introduceren een nieuwe visualisatiemethode voor grote verzamelingen lijnen die gebaseerd is op traditionele illustratieve technieken. De belangrijkste vernieuwing is de diepte-afhankelijke halo: een witte rand om de zwarte lijnen waarvan de dikte afhangt van de relatieve afstand tot achterliggende lijnen. Hierdoor wordt de diepte-waarneming verbeterd en worden structuren van lijnen (bijvoorbeeld bundels) visueel benadrukt. De lijnvisualisaties geproduceerd met deze methode lijken qua stijl op met de hand gemaakte zwart-wit illustraties. Bovendien maakt het zwart-wit karakter de resulterende visualisaties erg geschikt voor drukwerk en anaglyph (rood-cyaan) 3D plaatjes. Ook in een informele evaluatie door domeinexperts (neuro-wetenschappers) werden de resultaten van onze methode positief ontvangen.

De visualisaties uit hoofdstuk 3 benadrukken impliciet aanwezige structuren van lijnen zoals bundels en vlakken. In hoofdstuk 4 creëren we voor dergelijke structuren expliciete abstracties. De methode die wij daarvoor hebben geïntroduceerd bestaat uit twee stappen. In de eerste stap wordt geanalyseerd welke lijnsegmenten qua richting en positie veel op elkaar lijken. Vervolgens worden in de tweede stap lokaal op elkaar lijkende lijnsegmenten iteratief naar elkaar toegetrokken. Dit proces van bundelen creëert leegtes tussen de fiber tracts waardoor er minder sprake is van storende overlapping van lijnen en de globale structuur van de witte massa duidelijker wordt. Wij hebben geëxperimenteerd met verschillende manieren voor het interactief werken met de geabstraheerde representaties van de fiber tracts. In de belangrijkste van deze methoden kan naadloos geschakeld worden tussen de originele en geabstraheerde representatie.

Tenslotte ligt in hoofdstuk 5 de nadruk meer op de afbeelding van lijnen. Hiervoor hebben we een flexibel illustratief lijnstijlmodel geïntroduceerd. In dit model, een generalisatie van de diepte-afhankelijke halos uit hoofdstuk 3, verdelen we lijn-stroken in paralelle banden waarvan de visuele eigenschappen individueel ingesteld kunnen worden.

We vergroten de visuele flexibiliteit verder door het mogelijk te maken om lokale stroomlijn-eigenschappen (zoals snelheid en temperatuur) te koppelen aan visuele eigenschappen (in het bijzonder breedte en kleur). Daarnaast hebben we het concept "line style transfer function" geïntroduceerd dat lokale stroomlijn-eigenschappen kan koppelen aan eerder gedefinieerde lijnstijlen. Dit maakt het mogelijk voor gebruikers om bepaalde soorten lijnen te benadrukken. En omdat deze methode gebruikmaakt van de grafische kaart ondersteunt onze implementatie grote datasets.

Lijnen staan centraal in dit proefschrift en ze vormen het vehikel om informatie uit de onderliggende gegevens over te brengen. In vergelijking met andere methoden hebben lijnen het voordeel dat ze conceptueel simpel zijn en tegelijkertijd zowel globale structuur als detail kunnen uitdrukken. Maar er zijn natuurlijk ook vele mogelijkheden voor het combineren van de in dit proefschrift beschreven methoden met andere aanpakken, met name met het oog op het voorzien van context.

# DANKWOORD

Het komen tot een proefschrift is een hele reis, met onvoorziene afslagen, zijweggetjes en soms doodlopende wegen. Velen hebben me hierbij bijgestaan en via deze weg wil ik mijn dank betuigen.

Allereerst richt ik mij tot mijn promotor Jos Roerdink. Jos, dank voor de kans om bij jou promotieonderzoek te doen en het vertrouwen dat je in mij hebt gehad en gehouden. Ook waardeer ik het zeer dat je me de vrijheid hebt gegeven om zelf richting te geven aan mijn onderzoek en dat je tegelijkertijd met rust en wijsheid het grote plaatje in de gaten hield. Daarnaast was je taalgevoel en kritische blik van onschatbare waarde bij het schrijven van de artikelen en dit manuscript.

Tijdens mijn onderzoek had ik naast mijn promotor ook nog de ondersteuning van twee dagelijks begeleiders: Henk Bekker en Tobias Isenberg. Henk, de donderdagen waren altijd een speciaal ankerpunt van de week: met een stapeltje lege A4'tjes en beide een vulpotlood in de aanslag exploreerden wij dan al schetsend de wetenschap. Ik heb veel geleerd van je vrije, no-nonsense manier van wetenschap beoefenen, wars van regels en conventies, en je hebt me getoond dat ik daarbij mijn intuitie mag (moet?) gebruiken. Dank daarvoor!

Tobias, hoewel je pas later in een later stadium bij mijn promotie-onderzoek betrokken raakte, is jouw invloed op de richting van mijn onderzoek groot gebleken. Je drive en passie is erg inspirerend en ik heb veel van je geleerd, zowel over onderzoek in het algemeen als over de kleine details met betrekking tot het schrijven van artikelen. Daarnaast was je ook goed gezelschap, ik kijk bijvoorbeeld met veel plezier terug op de door jou geïnitieerde bios-bezoekjes met de groep, het jou helpen met de Nederlandse taal (vandaar dat ik het aandurf dit in 't Nederlands te schrijven) en het maken van de fast-forward filmpjes.[1]

I would like to thank the members of the reading committee, Prof. dr. E. Gröller, Prof. dr. G. Scheuermann, and Prof. dr. ir. J.J. van Wijk for reading my thesis and the very helpful feedback. I would also like to thank professor Thomas Ertl and the other members of the Institut für Visualisierung und Interaktive Systeme (VIS) for their hospitality and and scientific input during my one month visit.

Als je als AIO iets voor elkaar wilt krijgen binnen de universiteit dan vormen twee groepen mensen een onmisbare ondersteuning. De eerste bestaat uit de secretaresses: Ineke, Esmee, Marike, Helga en in het bijzonder Desiree, bedankt voor al het regelen, hulp met formulieren en de gezelligheid. De tweede bestaat uit de systeembeheerders: Peter,

1 Zie http://youtu.be/MicCqBR70Fs and http://youtu.be/C5idOGZkq6Y.

Jurjen en Harm, dank voor de ondersteuning en flexibiliteit in een veranderende (IT-)organisatie.

Verder vormden natuurlijk de leden van de onderzoeksgroep een belangrijke bron van inspiratie, advies en prettig gezelschap, te beginnen met de seniors Andrei (dank voor de begeleiding in het 1e jaar), Michel en Alex. Bedankt!

The same holds for my fellow PhD students: I really enjoyed your company, advice, support, inspiration and the general sharing of your different cultures and languages. So thank you, Ronald (mijn eerste kamergenoot, die mij wegwijs heeft gemaakt bij het AIO zijn), Michael ten Caat (leuk dat we nog steeds contact hebben), Wladimir (OpenGL & graphics kennis en gezelschap bij onze avonturen in Salzburg en Stuttgart), Bilkis (sharing of your culture and the funny stories about your son), Ale (food and photography tips), Yun (thank you for teaching me a few words of your language), Moritz, George, and Deborah. A special thank you goes out to Ozan, my officemate for the last few years, for your company, teaching me bits and pieces of Turkish (sağ olun!), and agreeing to be one of my paranymphs.

Mijn dank gaat ook uit naar de studenten die bij mijn onderzoek betrokken zijn geweest: Pjotr en Eric.

Mijn nieuwe collega's bij TNO wil ik ook bedanken voor hun aanmoediging en begrip bij de laatste loodjes van het afmaken van dit manuscript.

Een belangrijk component van onderzoek is het onderbewuste af en toe rustig z'n werk laten doen. Voor de daarvoor benodigde rust en ontspanning kon ik terecht bij vrienden en familie. Michiel, dank voor de nuchterheid & afleiding, etentjes, avondjes gamen (Rockband!), bordspellen, de geleende TV-serie afleveringen en natuurlijk voor het aanvaarden van de taak van paranimf. Ook Bob, Marijn, Martijn, en de schoonfamilie (Bert, Jennie en Paulien) wil ik bedanken voor het gezelschap, de aanmoedigingen en de interesse in m'n onderzoek.

Lieve pa & ma, jullie vormen het thuis-thuis, een plek van rust & ontspanning waar ik waar ik altijd terecht kan voor hulp en wijsheid, groot of klein, van klussen tot sociale protocollen. Dank daarvoor! Met hen wil ook ook m'n lieve en slimme zusjes Hanneke en Anneroos bedanken voor hun interesse, aanmoediging en steun.

Ten slotte, m'n lieve meis, m'n lieve Kaatje, onmisbaar deel van m'n leven: dank voor al je liefde, steun, begrip, aanmoediging, hulp en inspiratie tijdens deze reis. <3

*Maarten* *Groningen, 2011-10-20*

## COLOPHON

This thesis was typeset with LaTeX 2ε using Robert Slimbach's *Minion Pro* type face. The style of this thesis is based on André Miede's excellent Classic Thesis Style.