Eurographics/ IEEE-VGTC Symposium on Visualization 2010
G. Melançon, T. Munzner, and D. Weiskopf
(Guest Editors)

Volume 29 (2010), Number 3

Illustrative White Matter Fiber Bundles

Ron Otten[2], Anna Vilanova[1], and Huub van de Wetering[2]

[1]Department of Biomedical Engineering, Eindhoven University of Technology, The Netherlands
[2]Department of Mathematics and Computer Science, Eindhoven University of Technology, The Netherlands

Abstract

Diffusion Tensor Imaging (DTI) has made feasible the visualization of the fibrous structure of the brain white matter. In the last decades, several fiber-tracking methods have been developed to reconstruct the fiber tracts from DTI data. Usually these fiber tracts are shown individually based on some selection criteria like region of interest. However, if the white matter as a whole is being visualized clutter is generated by directly rendering the individual fiber tracts. Often users are actually interested in fiber bundles, anatomically meaningful entities that abstract from the fibers they contain. Several clustering techniques have been developed that try to group the fiber tracts in fiber bundles. However, even if clustering succeeds, the complex nature of white matter still makes it difficult to investigate. In this paper, we propose the use of illustration techniques to ease the exploration of white matter clusters. We create a technique to visualize an individual cluster as a whole. The amount of fibers visualized for the cluster is reduced to just a few hint lines, and silhouette and contours are used to improve the definition of the cluster borders. Multiple clusters can be easily visualized by a combination of the single cluster visualizations. Focus+context concepts are used to extend the multiple-cluster renderings. Exploded views ease the exploration of the focus cluster while keeping the context clusters in an abstract form. Real-time results are achieved by the GPU implementation of the presented techniques.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Bitmap and framebuffer operations, Display algorithms, Viewing algorithms

1. Introduction

In the last decades, visualization of the internal white matter structure in-vivo has become feasible thanks to the development of Diffusion Tensor Imaging (DTI) [BML94]. Several algorithms have been proposed to visualize this complex data [VZKL06]. One of the most popular techniques is to reconstruct the individual fibers from the tensor information, e.g., by tracing streamlines. Usually fibers are defined by manually setting seed points. In this case, the result is biased by the user who can miss important structures. However, white matter is a complex structure and the image gets easily cluttered if no regions are being defined.

Fibers form anatomically meaningful entities called bundles that define the connection of different grey-matter areas. Several authors have proposed to cluster the generated fibers to obtain bundles [MVW05, JDL09]. The enormous amount of unstructured individual fibers is grouped into a limited number of fiber clusters that are more manageable and understandable. The common visualization used for these bundles is still drawing individual fibers that are colored per cluster, see Figure 1. This does not produce a clear overview of the bundles and the image remains cluttered.

Once the clustering is obtained the white matter could be viewed at different levels of detail; a global view which shows the fiber clusters and a local view that shows more detailed information of the DTI data. For the global view, it is necessary to abstract from the individual fiber visualization to get insight into the relation between the clusters. The generation of surfaces that wrap the fibers of a cluster have been proposed in literature [ESM*05, CZCE08]. Although these methods avoid the individual fiber visualization per cluster, a polygonal surface needs to be generated, which is time consuming and error prone for complex cluster shapes. Technical illustrators and medical illustrators have mastered the art of simplification. They are able to leave out details that lead to cluttering and incomprehensibility, while retaining enough detail to propel the intention of the illustration. We propose to use illustrative techniques to present

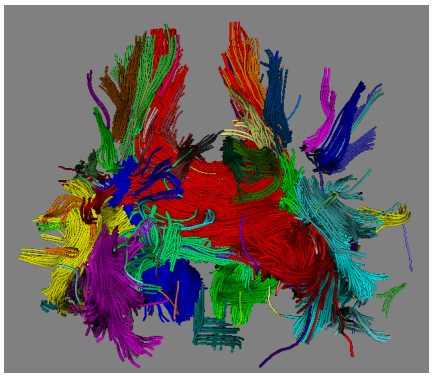
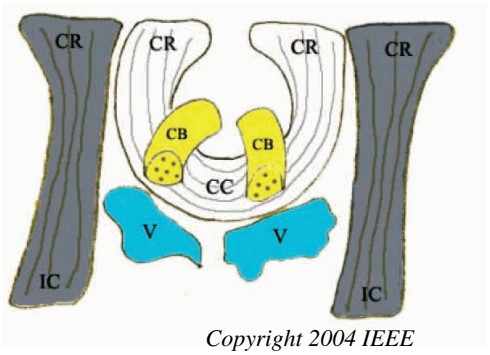


Figure 1: Results of hierarchical clustering where 49 clusters are rendered using color coding.



Copyright 2004 IEEE

Figure 2: Hand drawn illustration from Wenger et al. [WKZL03]

the clusters as a whole, and not as individual fibers. We aim at producing images similar to the hand drawn image presented by Wenger et al. [WKZL03](see Figure 2). In this image we observe that bundles are simplified to their silhouette together with a distinctive outline to enhance the borders of the cluster. They are rendered as flat shaded structures. We use the term *silhouette* for the solid shape formed by the projection of an object on a plane. The boundary of a silhouette is called *outline*. In Figure 2, the fibers are simplified to some sparse hint lines that indicate the global orientation of the fibers within the cluster. This illustrative style gives a good overview of the clusters where most of the detail, irrelevant to get a global view of the white matter, is hidden. We present techniques that allow to mimic this illustrative style and obtain the result shown in Figure 3. We also incorporate what we call *contours*, which are the silhouette points where the depth or projection distance is discontinuous. The *interior* contours are formed by the contour points that are not already on the outline. Figure 2 does not show contours, however, they can be seen in Figure 3. Especially the corpus callosum (i.e., the pink cluster) shows these contours giving a better shape perception of the cluster.

This cluster visualization, however, is simplified to the point where it can no longer give useful detailed information for a specific cluster. In our approach a selected focus is given a more classical and detailed visualization, such as fibers or glyphs. The context is rendered with illustrative techniques and allows a fast navigation through the clusters. The interaction for the focus selection is done by simple clicking, and has not been studied in this paper. Given the complexity of white matter this is still not enough. Clusters might be occluded by other clusters, and it might be impossible to inspect them without getting clusters out of the way. Exploded views is a possible solution that we adapt and that is commonly used in illustration to show hidden objects. In summary, we present an exploration tool for white matter clusters based on illustrative rendering and the focus+context paradigm. For the context the fibers per cluster are simplified by using similar techniques to the one developed simultaneously by Everts et al. [EBRI09]. An image based extension is applied to be able to obtain the silhouettes and the contours of the clusters. This method has the advantage that it does not require generation of wrapping surfaces. Exploded views with interactive animations have been added to allow a non-occluded view to the focus cluster while still keeping context information. The techniques presented are implemented on the GPU to achieve interactive frame rates.

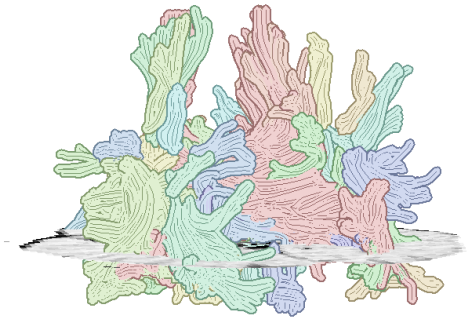
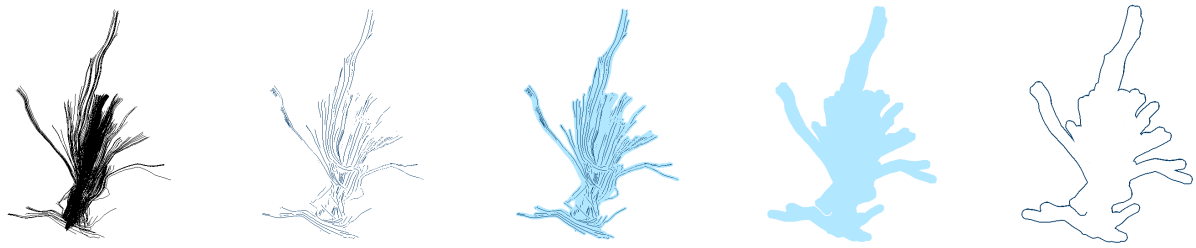


Figure 3: 49 illustrative clusters using the techniques presented in this paper

In the next section, we have an overview of related work. In section 3 we present the illustrative rendering of clusters and its GPU implementation. Section 4 describes our exploded-view techniques used for the focus+context paradigm. Finally, results and conclusions are presented.

2. Related Work

In this paper we present a visualization technique for the exploration of white matter clustering results. Interaction techniques have been proposed in literature to select and explore these white matter partitions, e.g., the work presented by Jianu et al. [JDL09]. Our work is complementary to these selection algorithms. We want to improve the 3D visualization of these complex structures. We try to improve the per-



(a) input polylines (b) hint lines (c) coloured halos (d) silhouette (e) outline

Figure 4: Examples of the components of our illustrative cluster visualization

ception of the fiber clusters by not visualizing the individual fibers but the clusters as such.

Enders et al. [ESM*05] and Chen et al. [CZCE08] presented techniques to generate wrapped surfaces around fiber bundles. These methods have problems with clusters that have complex shapes. We aim at a method where the clusters are visualized as a whole using the generated fibers directly without generating intermediate geometry.

We want to simplify our visualization to a comprehensive image where the clutter of the individual fibers is avoided. Illustrative rendering has been used for this purpose in visualization for several applications [VGH*05]. To enhance the visualization of line primitives several techniques have been proposed. Halo based methods for lines were first presented by Appel et al. [ARS79]. These techniques give better depth perception. Halos were applied to 3D flow data by Interrante and Grosch [IG97]. Later they were applied to white matter fiber structure by Wenger et al. [WKZL03] and Everts et al. [EBRI09]. One of the effects of halos is not just that they give a better perception but they also remove occlusion and generate a more sparse visualization of very densely packed lines. Inspired by the work of Busking et al. [BVW07] and using a technique similar to Everts et al. [EBRI09], we use halos for simplifying the amount of visible lines within a cluster. Stroke based methods [Her03] could also be used to obtain a sparse set of the fiber lines. However, these methods are based on optimization techniques that require iterative, computationally expensive processes. Furthermore, they are also concurrent with the generation of the line structures, which is not possible in our setting since the input are the results of clustering of already traced fibers.

Identifying silhouettes and finding their outlines is an important task used to improve the readability of images, e.g., for molecular visualization [TCM06] and parallel coordinates [MM08]. One group of methods work directly in a polygonal mesh while others work in image space by processing the rendered depth buffer per viewpoint [IFH*03]. McDonnell and Mueller [MM08] use illustrative techniques to present clustered polylines to enhance parallel coordinates. They generate, however, polygons based on the clustered polylines which we want to avoid for the complex configuration of the fibers. To the best of our knowledge, there are no methods in literature that use silhouette and its outline for visualization of clustered polylines without an intermediate surface. The approach we present is an image based method that builds on our technique for line simplification.

The principle of focus+context is to clearly present and suitably emphasize the object of interest in a visualization, called the focus, while providing a more subtle representation of the other objects as a context, serving to place the focus into perspective as part of the whole. The illustrative methods presented in this paper can be used for the context while the focus is rendered in more detail using traditional methods. However, occlusion problems remain.

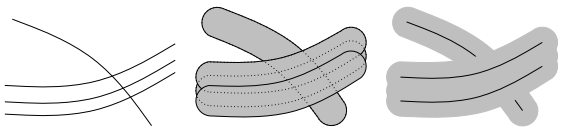
Exploded views [BG06] are popular means to provide focus+context avoiding occlusion. An exploded view pulls a composed object apart into its components. A mental map [JL83] is structured cognitive information a user builds up in his memory and helps with reasoning, comprehending, and navigating the higher level structure of a visualization. It is important that this mental map is preserved. We could use spatial distortion and retract the offending fiber clusters, or wedge them apart [CSC07]. This would closely mimic techniques that can be seen in surgical illustrations [CSC06]. The downside is that we end up warping (part of) the shape of the context. This is no problem if we have other visual cues left that can identify parts of an image, but a fiber cluster is a fairly abstract concept that is only recognizable by its general shape and relative position to other bundles of which an expert recognizes the general shape. Therefore, we want a technique that keeps this shape intact. Instead of using volume manipulation techniques such as retractions or wedges, which introduce alienating distortions to the data, we simply slide the offending bundles out of the way using a second exploded view.

3. Illustrative fiber clusters

We describe an illustrative rendering technique that is to be used for a context cluster in a focus+context rendering of a set of fiber clusters, where inspiration is taken from hand

drawn schematic illustrations. From those illustrations we learned that for schematically rendered fiber clusters to form comprehensible, clear, non-interfering visualizations, these visualizations should show large, clear shapes, have hard, well-defined boundaries, and hint subtly at the inner details of the clusters. As a consequence the proposed schematic illustrative rendering contains a silhouette to show the shape of a cluster, an outline to define the boundary of a cluster, and hint lines inside the silhouette to show some fiber details.

A fiber cluster is a list of polylines, where each polyline represents a fiber. Figure 4a shows the polylines of a fiber cluster. Note that our input data apparently consists of a dense set of lines without a clearly defined boundary. In section 3.1 we render hint lines as shown in Figure 4b, in section 3.2 we create a silhouette (see Figure 4d) and an outline (see Figure 4e) for a single cluster. We show how this can be implemented on the GPU in section 3.3, and we use it for rendering multiple fiber clusters in section 3.4.



(a) Unculled fibers (b) Overlapping halos (c) Culling fiber parts

Figure 5: Culling fibers using halos

3.1. Rendering hint lines

As a first step we convert a fiber cluster into a rendered image of a more sparse set of hint lines. If many fibers in a cluster cross or pass closely in parallel, an image of that cluster becomes cluttered. We cull enough fibers to remove clutter from the image, while leaving enough lines to hint at the cluster's overall fiber configuration. This is accomplished by adding an opaque halo around each polyline, performing a kind of hidden line removal.

The basic idea behind this approach is illustrated in Figure 5. We simply render all the polylines augmented with suitably wide halos. Each polyline culls parts of neighboring polylines that are behind its halo. This enforces open space between polylines by hiding parts of lines that are close.

The rendered polylines that survive the culling procedure, serve as a good spread of hint lines as can be seen in Figure 4b where the polylines are rendered with white halos. How far these lines are spread and how many remain is directly related to the radius r_{halo} of the halos. This makes the density of the hint lines a parameter that can be configured to the user's taste. Figure 4c shows the results with the halos added to the fibers when the halo color c_{fill} differs from the background color. To improve perception the halos are rendered as fins with depth in the viewing direction. This is similar to what Everts et al. [EBRI09] did for lines and Busking et al. [BVW07] for points. Unlike Everts et al. the ends

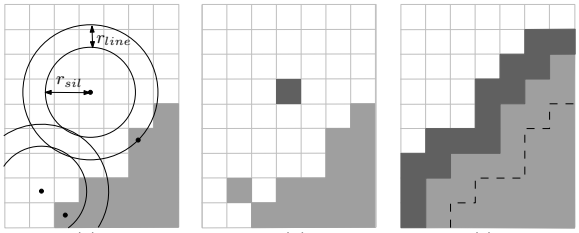


Figure 6: The compound structuring element is shown at two different pixels (a). The corresponding silhouette pixel (light) and outline pixel (dark) (b). The result after full compound dilation (c).

of the lines do have a round halo cap. The fins are oriented according to the view direction and therefore halos are view dependent.

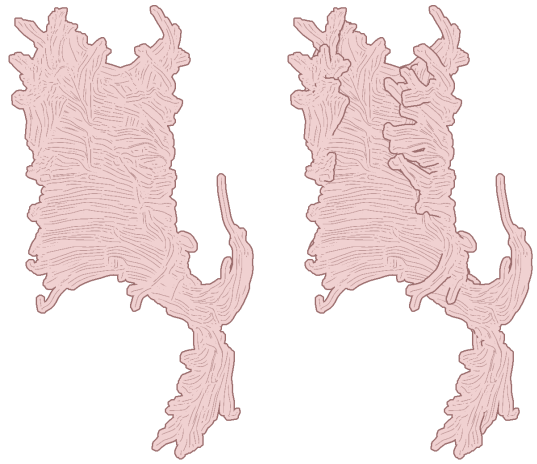
3.2. Silhouette, outline, and contours

The second step in our rendering technique is performed in image space and constructs a silhouette from the image of the first step. Figure 4c shows that the colored halos already form a thin, silhouette-like area which, however, is not coherent and contains holes. While we could increase the width of the halos, this would also reduce the number of hint lines. Instead, we use a dilation with a circular structuring element of radius r_{sil} to fatten the rendered halos to give the area more body and close the holes. The result is a silhouette as shown in Figure 4d.

An additional dilation that draws the added pixels in a distinct color c_{out} and uses a circular structuring element of radius r_{line}, produces an outline of the silhouette as shown in Figure 4e. Later in this section we combine these two dilations into a single compound operation. Note that in order for this to work, setting a pixel to the silhouette color has to take preference to setting a pixel to the line color, since the silhouette dilation was performed before the outline dilation. This compound dilation uses for each pixel p a circular structuring element B_p of radius $r_{sil}+r_{line}$, that is split into two parts S_p and O_p (see Figure 6) defined as follows.

$$\begin{aligned} S_p &= \{q \mid 0 \le |p-q| \le r_{sil}\} \\ O_p &= \{q \mid r_{sil} < |p-q| \le r_{sil}+r_{line}\} \\ B_p &= S_p \cup O_p \end{aligned}$$

Outlines alone may not be enough when dealing with more complex cluster shapes. They only show the boundary of the silhouette. This significantly reduces the clearness of the image if the cluster exhibits self-occlusion. In such a case it is favorable to add the *interior* contours. Interior contours convey the three dimensional nature of the fiber cluster much better [IFH*03]. To appreciate this, compare Figure 7a with an outline to Figure 7b with added interior contours. The remainder of this section describes how to generate silhouette, outline, and interior contours in a single pass. When a fiber



(a) Outline only (b) Outline and interior contours

Figure 7: A self-occluding fiber cluster

cluster occludes itself, one or more fibers *pass in front of* one or more other fibers. With only the 2D image I of halo-augmented fibers produced in the first step, we have no way of detecting this. Therefore we assume the first processing step creates two outputs: an image I with colors and a *depth buffer* D with depth values for each pixel.

A pixel p has a single depth in D, so detection of multiple layers passing over one another is not feasible. However, the detection of differences in depth between pixels under the structuring element B_p is possible. This is in fact analogous to how image-space edge detection can be used to draw outlines and contours [IFH*03].

If a difference between D_p and D_b of a pixel $b \in B_p$ exceeds a given threshold t, we assume b belongs to a different fiber halo passing in front or behind (depending on the sign of the difference). We define the operator $<_t$, where $D_a <_t D_b$ means $D_b - D_a > t$ or 'a is *in front of* b'. We assume that for background pixel c holds that $D_c = \infty$. Furthermore, we define that $\infty <_t \infty$ is false.

For all pixels p of image I all pixels b in its structuring element are visited for the computation of the new color and depth at pixel p, taking care of both the silhouette and outline dilation and the generation of interior contours. To decide on the depth order the minimum D_b for $b \in B_p$ is needed. Furthermore, in case of the combined dilations we give precedence to coloring pixel p with the silhouette color c_{fill}, when a pixel b is found within radius r_{sil}. Depth now plays a role in deciding precedence, so this condition can no longer be upheld in that form. We explicitly record the minimum depth in both S_p and O_p:

$$\begin{aligned} DS_p &= \min\{D_q \mid q \in S_p\}, \text{ and} \\ DO_p &= \min\{D_q \mid q \in O_p\} \end{aligned}$$

The precedence between silhouette and contour can be handled correctly by comparing the minimum depths and the depth D_p in pixel p. Table 1 shows the new color and depth for pixel p for all cases. As a first case, if neither $DS_p <_t D_p$ or $DO_p <_t D_p$ holds, the color and depth of pixel p remain as they were. If only $DS_p <_t D_p$ holds, the silhouette is in front and the color and depth of pixel p change to c_{fill} and DS_p, respectively. Similarly, if only $DO_p <_t D_p$ holds, the contour is in front and the color and depth of pixel p change to c_{out} and DO_p, respectively. If both conditions hold, the silhouette takes precedence as before, unless the contour is in front of the silhouette, i.e., $DO_p <_t DS_p$.

	$\neg s$	s
$\neg o$	(I_p, D_p)	(c_{fill}, DS_p)
o	(c_{out}, DO_p)	$\begin{cases} (c_{out}, DO_p) & \text{if } DO_p <_t DS_p \\ (c_{fill}, DS_p) & \text{otherwise} \end{cases}$

Table 1: Compound dilation cases at pixel p showing the new color and depth for p where $s = DS_p <_t D_p$ (silhouette in front of p) and $o = DO_p <_t D_p$ (contour in front of p)

With a suitable threshold t, images as in Figure 7b can now be produced in a single pass, given the result of hint lines: image I and the corresponding depth buffer D. Note that with lower thresholds the contours would become more pronounced with more showing up, whereas with higher thresholds less would be shown. With $t = \infty$ the result image shows only the outline.

3.3. GPU implementation

To prove that illustrative clusters can be generated at interactive speed, an efficient GPU-based implementation has been constructed using OpenGL. In Figure 8 the illustrative rendering technique is mapped on the GPU's graphics processing pipeline. Halo rendering is handled by a geometry shader, which creates the halos on the fly. The rendering output of this phase is redirected to a frame buffer object, which records both the color channel and depth channel into GPU memory as textures. Both textures are subsequently offered to a fragment shader, which performs the compound dilation and outputs the result on a screen-aligned quadrilateral.

Halo rendering The geometry of our halos is view-dependent, so they need to be rebuilt when the viewpoint changes. A one-time generation of the halos as static geometry would therefore be useless. Instead our implementation generates the halos on the fly, using geometry shaders, which can generate new geometry on the fly as data is processed by the graphics pipeline. The geometry shader is given the polylines as an OpenGL's `GL_LINES_ADJACENCY` primitive. This gives the shader a current segment, the previous segment, and the next segment of a polyline, and allows it to construct a smooth halo geometry.

To allow rendering of the rounded end caps by the same shader, the first and last segment need to be distinguishable

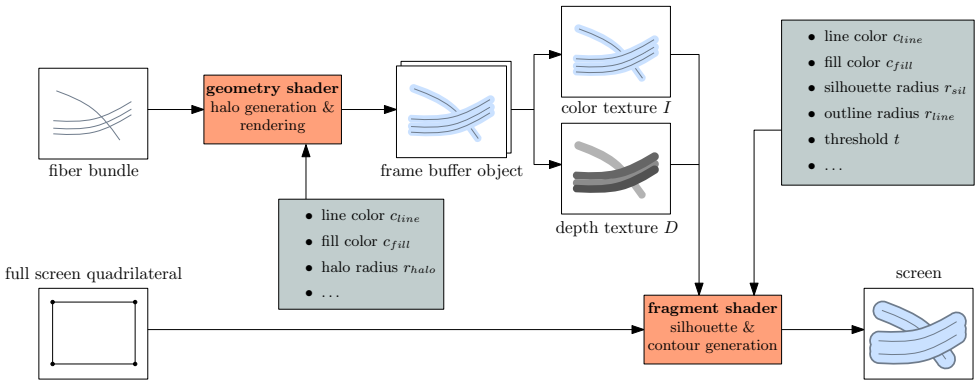


Figure 8: GPU implementation pipeline

from the others. The first and last vertex of a polyline are duplicated, enabling the shader to render an end cap only if the previous or next segment has length 0. The polyline itself needs to be rendered in a separate pass since the geometry shader can not output multiple primitive types. This second pass disables the geometry shader and renders the polyline directly. We use a 'less than or equal' test on the scene depth buffer, dismissing those parts of the polylines that are positioned behind any halo geometry rendered in the first pass.

Silhouette and contour rendering The compound dilation operator may be implemented as a fragment shader, where the outer loop of the algorithm (for all pixel $p \in$ image I) is executed in parallel. The fragment shader needs the textures that are retrieved from the frame buffer object resulting from the halo rendering. The compound dilation can be implemented fairly straightforward on the fragment shader. There are two issues relating to the depth buffer D.

The first issue concerns the effective range of the depth buffer. We assumed that the depth buffer values range from 0 at the viewpoint to infinity for a background pixel. In practice, a depth buffer has limited accuracy. This accuracy can be controlled by setting near and far planes. Our implementation puts the background pixels on the far plane, which we set to be slightly further away then any content which *should* be visibly rendered.

The second issue concerns the depth values stored in the depth buffer D if a perspective projection is used. Instead of storing z the OpenGL depth buffer actually stores $\frac{z'}{w'}$, where z' and w' result from the perspective projection of homogeneous point (x,y,z,w). Since in the compound dilation we use the differences of depth values, the depth values stored in the depth buffer need to be converted. To map the depth value D_p of a pixel back to a proper z-coordinate z'' in camera space, we have to apply the inverse of OpenGL's perspective projection matrix on the z coordinate:

$$z'' = \frac{D_{near} D_{far}}{D_{far} - D_p(D_{far} - D_{near})}, \qquad (1)$$

where D_{near} and D_{far} are the depths of the near and far plane, respectively.

Setting up a canvas for the fragment shader The fragment shader receives the color image I and the depth image D of the haloed fibers. To have the fragment shader map its results to the screen correctly, we use a simple screen aligned quadrilateral spanning the entire viewport as a canvas. For this to work efficiently a screen aligned bounding box for the cluster is computed and used as a clipping mask inside the fragment shader, having the shader discard any fragments that fall outside the mask.

3.4. Rendering multiple fiber clusters

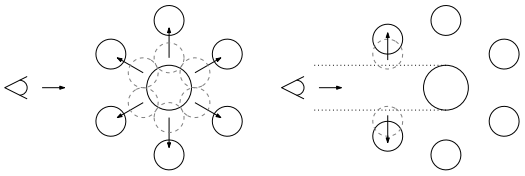
The methods presented so far operate on a single cluster, but can simply be used for multiple clusters. Actually, illustrative clusters work together nicely with any object being composed into the same scene. For each pixel the fragment shader implementing the silhouette generation process, writes new values to the pixel's depth value before it is written to the scene. These values override the regular depth values of the screen aligned quad the shader uses as a canvas. In this way the proper functioning of the OpenGL depth buffer is kept intact and multiple clusters (or other objects) can be rendered one by one to obtain results as in Figure 9.

4. Focus+context

If users load multiple fiber clusters in a scene, the cluster they are most interested in, called the focus, might be invisibly situated in the midst of the other clusters. To obtain an unobstructed view on the focus some of the other clusters have to be manipulated such that they still can function as context for the focus, yet do not occlude it. To achieve this the occluding clusters are pushed away from the focus to obtain an unobstructed view on the focus. Figure 12 illustrates both the problem and a possible solution.

We present a technique that keeps the shape of the clusters intact and tries to preserve their relative positions. Figure

Figure 9: Multiple illustrative fiber bundles



(a) Explode ... (b) ... then slide

Figure 10: Basic idea behind our focus+context technique

10 shows the concepts of our solution: first we explode the clusters uniformly relative to the focused cluster, then we slide clusters that continue to occlude the focused cluster out of the way in a second explosion, the force of which differs per cluster.

First explosion Let F be a focus cluster and let B be a possibly conflicting cluster that hinders the user to view F. The amount of overlap between F and B determines how much B has to move to undo this overlap. To avoid costly operations we replace F and B by their oriented bounding boxes (OBB) OBB_F and OBB_B, respectively. For a spherical explosion B will be moved along the direction $\mathbf{d}$ from the center of OBB_F to the center of OBB_B. The amount of displacement along $\mathbf{d}$ depends on the penetration depth of OBB_B in OBB_F along direction $\mathbf{d}$, which can be computed using a separating axis test [Got96, GLM96]. This test relies on running a series of interval tests on projections of polyhedra onto a set of potential separating axes $\mathcal{A}$. A separating axis is a direction on which the projected intervals are disjoint, proving that the polyhedra do not intersect and the axis is said to *separate* the polyhedra. If such an axis does not exist, the polyhedra intersect and for every axis $\mathbf{a} \in \mathcal{A}$ the penetration depth p_a along $\mathbf{a}$ is given by the length of the overlap of the projections of the polyhedra on the axis. The displacement δ_a along direction $\mathbf{d}$ (see Figure 11) needed to undo the penetration p_a along axis $\mathbf{a}$ is then given by

$$\delta_a = \frac{p_a}{\mathbf{d} \cdot \mathbf{a}}.$$

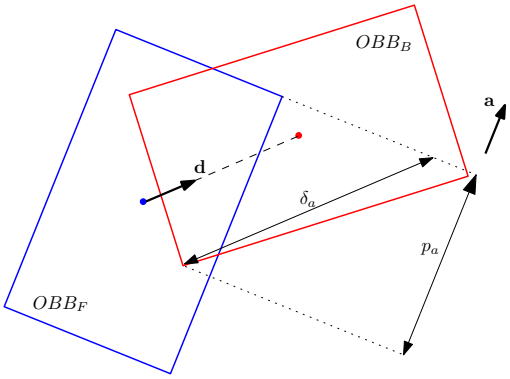


Figure 11: Construction of the displacement δ_a along direction $\mathbf{d}$ based on the penetration depth p_a along axis $\mathbf{a}$. $\mathbf{d}$ is the direction from the center of OBB_F to the center of OBB_B.

To undo the overlap between the focus F and the cluster B and simultaneously avoid unnecessary displacement, it is sufficient to move B

$$\delta_B = min\{\delta_a | \mathbf{a} \in \mathcal{A}\}$$

along $\mathbf{d}$. For a uniform explosion of a set $\mathcal{B}$ of clusters the displacement δ along the directions between the center of F and each cluster in $\mathcal{B}$ is given by

$$\delta = max\{\delta_B | B \in \mathcal{B}\}.$$

This assures that the context clusters are not pushed into one another, but will move uniformly outward without disrupting their relative position too much.

Second explosion During this explosion clusters that still occlude the focus F, slide along a vector parallel to the screen. To apply the same techniques as for the first explosion, we consider the projected OBB's of the clusters and find displacements that undo the overlap of the projections. The projections are convex shapes for which a 2-dimensional separating axis test can be applied. Similarly to the first explosion we compute δ_P for each projected cluster P with respect to the projected focus cluster. Since the second explosion interferes with the relative position of the context clusters on a greater scale, we keep its effect to a minimum. So each context cluster is only displaced as far as strictly necessary for that cluster to clear any occlusion with the focus cluster over a distance δ_P along a direction between the projected centers of the focus cluster and the displaced cluster.

4.1. Animation

The focus+context technique as presented so far, does not preserve a user's mental map very well. Suddenly moving around the clusters adversely affects comprehension of the visualization. To prevent unexpected changes in the visualization's layout and thus allowing the mental map of a user to be updated accordingly [BB99], the important transitions

ought to be animated. Animated transitions in our implementation are used in the following cases, the user enabling and disabling the focus+context technique, the user picking a new focus cluster, and context displacements taking place during the second explosion.

Our focus and context technique is stateless; when the viewpoint changes, computation of the focus+context layout starts from the neutral situation where the algorithm has not yet been applied. This makes it easy to introduce animation tracks. Instead of immediately translating the clusters in the rendered scene, we use our displacement algorithms to compute a set of target end positions for each cluster. An animation track for a transition then simply consists of moving a fiber cluster's scene representation from its current position to the current target position.

5. Results

In the previous sections, we have presented an illustrative rendering technique for fiber clusters and a related focus+context system. In this section we give an overview of the performance of their implementation.

Our illustrative rendering technique consists of two steps. First unnecessary fibers are culled with the halo rendering technique. This step has a complexity of $O(v)$, where v is the total number of vertices in the fiber polylines of the input data. Secondly, an image space algorithm is applied to the rendered result. It analyzes and compares information present in the rendered image's depth buffer and appropriately applies dilation to grow a silhouette with contours. This step has a complexity of $O(bm)$, where m is the total number of pixels in the viewport and b is the number of bundles.

image resolution (px)	$r_{sil}+$ r_{lin} (px)	1000 f. 1 bundle (fps)	4000 f. 1 bundle (fps)	3000 f. 10 bundles (fps)
1100	5	28.84	16.06	13.90
1100	3	58.64	21.40	20.77
730	5	53.76	20.05	17.34
730	3	56.30	22.49	25.29

Table 2: Performance for 1 bundle with 1000 fibers, 1 bundle with 3000 fibers, and 10 bundles with 4000 fibers, with different viewport resolutions and structuring element size.

For a balanced picture of our technique's performance we have recorded performance data where we vary the screen resolution, the combined size of the clusters, and the number of clusters. This reflects the dependency on m, v, and b, respectively. Furthermore, we have varied the thickness of the silhouette and contour lines, which affects the radius of our structuring element. This change significantly affects the results, as it has a large impact on the number of required texture samplings. We measure execution times using the CPU clock. To combat the loss of accuracy and anomalies due to interference of the OS scheduler we have run the same test case multiple times in sequence. Each test case has been set up to run 1000 iterations of the rendering technique and to report back the average time taken and the derived frame rate. All measurements were taken using a NVidia Geforce 9800 GX2 graphics card.

We note here that our implementation has an optimization built in that aims to reduce the number of processed pixels m to a particular subset of pixels on which the image space algorithm could have an actual effect. This optimization makes the algorithm dependent on the shape, orientation and scale of the displayed fiber bundles. As the results in Table 2 show, the illustrative rendering technique continues to deliver results at rates suitable for real-time interaction. Even in the cases of a large 1100×1100 pixel viewport and a large silhouette radius, we retain frame rates over 10 fps.

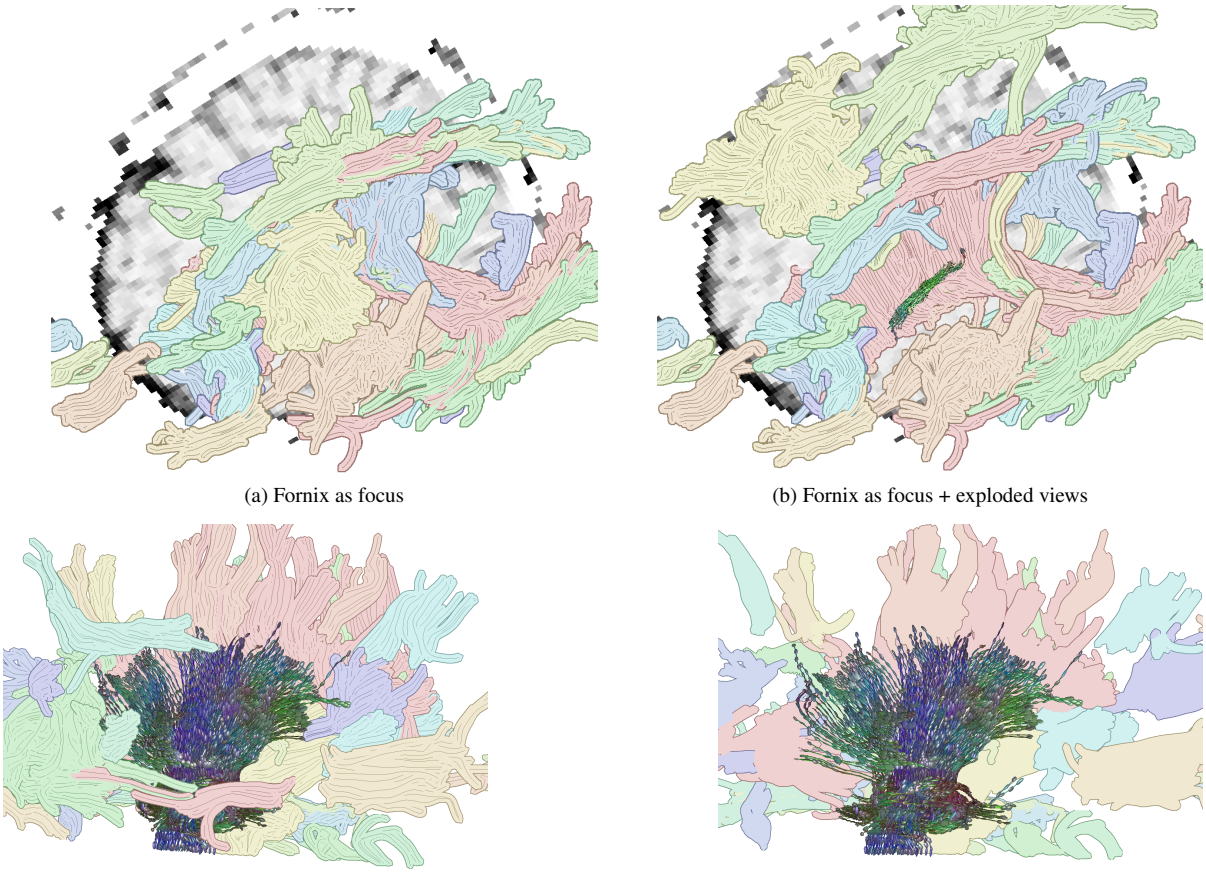
Figure 3 shows the rendering of 49 illustrative clusters. These could be rendered interactively. With the methods presented, the style of the hand drawn example given in section 1 can be mimicked quite well, if somewhat more detailed. If we compare our illustration with a regular rendering procedure for clustered fiber bundles, as in Figure 1, we note that thanks to the silhouette and contours it has become easier to identify the boundaries of clusters. The silhouette also gives the bundles a more solid look which somewhat helps with perceiving the depth order. Reduced crowdedness of lines makes it possible to follow structure of the cluster.

The capability to define separate rendering styles for the focus and context improves perception. The illustrative methods have been integrated in a general DTI tool that allows the use of different rendering styles for the focus. Figure 12 shows results for the fornix and the corona radiata.

6. Conclusions and Future Work

Illustrative techniques for the visualization of white matter fiber bundles have been presented. This visualization improves the perception of the white matter clusters by simplifying the presentation of its structure. The focus+context paradigm is applied. Exploded views and animations are used to avoid occlusion of the focus while trying to keep the mental map.

A user study would be necessary to analyze the real value of the proposed techniques. There are different user tasks that are interesting for the exploration of white matter fiber bundles, e.g., validation of the correctness and completeness of a specific bundle, definition of clustering parameters to obtain a correct bundle segmentation, bundle identification, and selection of a focus bundle. The parameters of the visualization (e.g., illustrative styles) could directly be studied in tasks as simple as: counting the number of clusters, estimating size or shape of bundles, in exploded views, naming or pointing at the bundles that would overlap a given bundle. The accuracy and user speed performing these tasks could



(a) Fornix as focus

(b) Fornix as focus + exploded views

(c) Corona Radiata

(d) Corona Radiata as focus + exploded views

Figure 12: Focus+context examples using different styles for context and focus, and showing the effect of the exploded views.

be compared to the ones using a classic fiber visualization. Switching on and off some of the features of the presented techniques could also help identify their individual value.

It is probably difficult to use real data for such a study, since a considerable amount of data sets would be necessary. However, for these tasks also artificial data sets could be generated where the level of challenge for a given task is similar to real data. The amount of specialists (e.g., medical researchers, surgeons) for a user study is also relatively low. Instructions could be given to non-specialists users, e.g., instead of using a bundle name show a bundle and ask the user to find the bundle with a given clustering. By these means the user population could be increased. Evaluate how appealing the specialists find the framework and its individual elements is also of interest. Structured interviews based on rating scales could be used to obtain this information.

Apart of the user studies, several enhancements could still be made, e.g., improved focus selection interaction, addition of contours between bundles, and addition of depth cues [EBRI09]. The clusters are flat shaded and adding shading would enhance the 3D shape of the clusters. Note that this is not trivial, since our method does not directly use surfaces.The implementation of the GPU based rendering of multiple fiber clusters uses a render pass per cluster. It would be more efficient to implement the techniques in a single pass.

The bundles have complex forms and using bounding boxes to calculate the transformation for the exploded views is limited. Using tighter bounding elements, or more accurate intersection calculations, would allow to minimize the displacement needed to avoid occlusion.

Finally, the illustrative techniques presented are not limited to white matter bundles. They could be used for any application where dense clustered line primitives need to be visualized, e.g., flow visualization or parallel coordinates.

Acknowledgements

We thank C. van Pul from Máxima Medical Center in Veldhoven for providing the data sets used in this paper. This study was financially supported by NWO-VENI program.

References

[ARS79] APPEL A., ROHLF F. J., STEIN A. J.: The haloed line effect for hidden line elimination. In *SIGGRAPH '79* (1979), ACM, pp. 151–157.

[BB99] BEDERSON B. B., BOLTMAN A.: Does animation help users build mental maps of spatial information? In *INFOVIS '99* (1999), p. 28.

[BG06] BRUCKNER S., GRÖLLER M. E.: Exploded views for volume data. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 1077–1084.

[BML94] BASSER P., MATTIELLO J., LEBIHAN D.: Estimation of the effective self-diffusion tensor from the NMR spin echo. *MR Journal 103*, 3 (1994), 247–54.

[BVW07] BUSKING S., VILANOVA A., WIJK J. J. V.: Particle-based non-photorealistic volume visualization. *The Visual Computer 12* (2007), 335–346.

[CSC06] CORREA C., SILVER D., CHEN M.: Feature aligned volume manipulation for illustration and visualization. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 1069–1076.

[CSC07] CORREA C., SILVER D., CHEN M.: Illustrative deformation for data exploration. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1320–1327.

[CZCE08] CHEN W., ZHANG S., CORREIA S., EBERT D. S.: Abstractive representation and exploration of hierarchically clustered diffusion tensor fiber tracts. *Computer Graphics Forum 27*, 3 (2008), 1071–1078.

[EBRI09] EVERTS M. H., BEKKER H., ROERDINK J. B., ISENBERG T.: Depth-dependent halos: Illustrative rendering of dense line data. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (2009), 1299–1306.

[ESM*05] ENDERS F., SAUBER N., MERHOF D., HASTREITER P., NIMSK C., STAMMINGER M.: Visualization of white matter tracts with wrapped streamlines. In *IEEE Visualization Conference Proceedings* (2005), pp. 51–58.

[GLM96] GOTTSCHALK S., LIN M. C., MANOCHA D.: OBBTree: a hierarchical structure for rapid interference detection. In *SIGGRAPH '96* (1996), pp. 171–180.

[Got96] GOTTSCHALK S.: *Separating axis theorem*. Tech. rep., Departement of Computer Science, UNC Chapel Hill, 1996.

[Her03] HERTZMANN A.: A survey of stroke-based rendering. *IEEE Computer Graphics and Applications 23* (2003), 70–81.

[IFH*03] ISENBERG T., FREUDENBERG B., HALPER N., SCHLECHTWEG S., STROTHOTTE T.: A developer's guide to silhouette algorithms for polygonal models. *Computer Graphics and Applications, IEEE 23*, 4 (2003), 28–37.

[IG97] INTERRANTE V., GROSCH C.: Strategies for effectively visualizing 3d flow with volume lic. In *IEEE Visualization Conference Proceedings* (1997), pp. 421–424.

[JDL09] JIANU R., DEMIRALP C., LAIDLAW D.: Exploring 3D DTI fiber tracts with linked 2D representations. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (2009), 1449–1456.

[JL83] JOHNSON-LAIRD P. N.: *Mental models: towards a cognitive science of language, inference and consciousness*. Cambridge University Press, 1983.

[MM08] MCDONNELL K. T., MUELLER K.: Illustrative parallel coordinates. *Computer Graphics Forum 27*, 3 (2008), 1031–1038.

[MVW05] MOBERTS B., VILANOVA A., WIJK J. V.: Evaluation of fiber clustering methods for diffusion tensor imaging. In *IEEE Visualization Conference Proceedings* (2005), pp. 65–72.

[TCM06] TARINI M., CIGNONI P., MONTANI C.: Ambient occlusion and edge cueing for enhancing real time molecular visualization. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 1237–1244.

[VGH*05] VIOLA I., GRÖLLER M. E., HADWIGER M., BÜHLER K., PREIM B., SOUSA M. C., EBERT D. S., STREDNEY D.: Illustrative visualization. In *IEEE Visualization Conference Proceedings* (2005), p. 124.

[VZKL06] VILANOVA A., ZHANG S., KINDLMANN G., LAIDLAW D.: *Visualization and Image Processing of Tensor Fields*. Springer Verlag series Mathematics and Visualization, 2006, ch. An Introduction to Visualization of Diffusion Tensor Imaging and its Applications, pp. 121–153.

[WKZL03] WENGER A., KEEFE D. F., ZHANG S., LAIDLAW D. H.: Interactive volume rendering of thin thread structures within multivalued scientific datasets. *IEEE Transactions on Visualization and Computer Graphics 10* (2003), 664–672.