A Visualization Pipeline for Large-Scale Tractography Data

James Kress *Student Member, IEEE*

Fig. 1: Demonstrating our binning visualization technique on a volume of 25 clusters. The image on the left uses an isovalue of 0.5, while the image on the right uses an isovalue of 100.

Abstract— We present a novel methodology for clustering and visualizing large-scale tractography data sets. Tractography data sets are very large, containing up to hundreds of millions of tracts; making visualizing and understanding this data very difficult. Our method reduces and simplifies this data to create coherent groupings and visualizations. Our input is a collection of tracts, from which we derive metrics and perform a k-means++ clustering. Using the clustered data, we create a binning volume that contains the counts of the number of tracts that intersect each bin, from which we can perform standard visualization techniques. Our contribution is the visualization technique and methodology itself, as well as an extensive study and evaluation schema. Our study utilizes our evaluation schema to identify the best and most influential clustering metrics in a metric set, and an optimal number of clusters under varying user requirements.

Index Terms—Tractography, Visualization, Clustering

1 Introduction

Fields researching the function and structure of the brain have long been confronted with challenges regarding imaging and visualization of the complex data being studied [9]. Non-invasive methods, such as Magnetic Resonance Imaging (MRI), have been developed in order to safely generate three-dimensional representations of structural components of the living human brain. Typically, MRI data is used to provide differentiation between various tissue types (grey matter, white matter, and Cerebral Spinal Fluid (CSF)) [12]. Diffusion MRI (dMRI) builds on MRI technology to measure the diffusion of water throughout tissue [19]. Since white matter neurons are myelinated, their diffusion characteristics differ substantially from the similar grey matter neurons. Groups of these white matter neurons, or fiber tracts, form the basic connections between distant brain regions. It is believed that studying white matter fiber tracts will enable researchers to better understand the fine structure of the brain leading to a more complete understanding of how it works [27].

While tractography data is clearly useful, the size of this data often makes analysis difficult. A typical tractography data set consists of hundreds of thousands of tracts, and they can sometimes contain much more, even hundreds of millions. Further, each advance in technology allows more and more tracts to be identified. Each tract contains multiple line segments, typically around 300. In our study, we considered a data set with almost $500,000$ tracts and 150 million line segments.

• *James Kress, University of Oregon. E-mails: james@jameskress.com.*

Manuscript received 31 Mar. 2014; accepted 1 Aug. 2014; date of publication xx xxx 2014; date of current version xx xxx 2014.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

The problem with tractography data, then, is two-fold: (1) how to operate on large data sets? and (2) how to create meaningful results that do not visually overwhelm a medical researcher? Plotting each of the tracts and their line segments on the screen leads to a very complex scene; in the data set described above, there would be 15 line segments for every pixel of a 1000^2 image. Instead, techniques are needed that make the scale of the data more manageable. Specifically, techniques are needed that make tractography data smaller to operate on while also creating scenes that are more comprehensible for viewers.

With this work, we develop a novel approach for visualizing tractography data. The approach uses clustering to group similar tracts together, and then creates new data sets that can visualize the density of each cluster using traditional visualization techniques. The contribution of our study is the technique itself, as well as extensive analysis behind the best ways to carry out the technique.

The remainder of this paper is organized as follows: Section 2 surveys related work. Our new technique is described in Section 3. The technique is a framework with multiple "knobs"; Section 4 describes the metrics we use to evaluate which knob settings are best. Section 5 provides an overview of our experiment, and Section 6 describes our study over the knob values. This study is incorporated into Section 7, which describes our algorithm for picking the best knob values. Section 8 describes our experiences in practice, both in terms of carrying out the study and in terms of applying the algorithm to create new visualizations.

2 Related Work

2.1 Diffusion MRI

Understanding how the various parts of the brain are interconnected by neurons in white matter is an active topic of research [9]. dMRI

produces a series of volumetric images in which each image represents the directional strength of water diffusion *in vivo*. Due to the fibrous nature of myelinated white matter neurons, water diffuses more rapidly along these fibers than it does in other directions [11]. Once the dMRI is acquired, the various images of directional diffusion are coalesced into a single three dimensional image of high-order elements. Generally, these reconstructed images are referred to as diffusivity images. While there exist many methods to creating diffusivity images [14, 29, 32], each method requires different data acquisition protocols that may drastically change the time it takes to capture the overall dMRI scan. As this work focuses on the Diffusion Tensor Imaging (DTI) family of dMRI reconstructions, we restrict our discussion of fiber tractography methods to those that may use this representation. One of the simpler reconstruction techniques, DTI, uses symmetric, rank two diffusion tensors as basic elements in the diffusivity image. Scans to be used with DTI reconstructions are considered fast to acquire (approximately fifteen minutes), enabling their use with at risk populations, children, and patients that may not tolerate long scanning times [25].

2.2 White Matter Fiber Tractography

In order to study the connectivity of the brain, white matter fiber tracts must be estimated from the diffusivity image. Tractography reconstruction algorithms may be divided into two large classes: probabilistic strategies, and deterministic ones [10]. Probabilistic tractography alogrithms, such as Yendiki *et al.*'s TRACULA [32], use Bayseian frameworks to generate volumetric distributions of pathway likelihoods. A more complete discussion of probabilistic tractography is given by the work of Jbabdi *et al.* [18]. Deterministic approaches to tractography, such as the FACT method [22], integrate the diffusion tensor field to generate streamlines representing recovered fiber paths.

Both probabilistic and deterministic methods for tractography reconstruction may be *local* or *global* in nature. Local reconstructions seed positions at a small region of the brain in order to discover how a single cortical region is connected. Global methods, on the other hand, densely seed the white matter volume in order to capture a more holistic view of white matter fiber paths.

Unfortunately, global tractography methods may take hours to complete and produce large amounts of data. Only through the use of data reduction techniques can these data be analyzed to learn more about the brain's connectivity. In this work, we rely on the global tractography methods and acquisition parameters proposed by Scherrer *et al.* in order to generate the large data sets required to best understand the interconnectivity of the brain [28].

2.3 Visualization of Tractography

The visualization of whole brain tractography can generally be separated into two groups: (1) visualization of clustered tractography and (2) visualization of non-clustered tractography. We explore select closely related works in sections 2.3.1 and 2.3.2 respectively.

2.3.1 Clustered Tractography

Partitioning tracts into groupings is an important step in analyzing and understanding a tractography data set. Often, this partitioning is done by clustering. Tractography clustering generally breaks down to two components, the clustering method, and the tract similarity metrics used to perform the clustering.

Clustering methods can generally be broken down into two different general themes, Cartesian clustering and anatomical clustering. Each of these clustering approaches attempts to provide answers to the same general questions. Anatomical clusterings use existing knowledge of the brains structure to make assumptions about how a given data set should be partitioned. Cartesian clusterings on the other hand, solely use information that can be derived from individual tracts or the data set as a whole, to create metrics and a final clustering.

There has been substantial work on clustering methods and metrics in the past, especially in Cartesian clustering. Visser *et al* [30] and Moberts *et al* [21] both employ the use of hierarchical clustering using variations of the pairwise distance between tracts as their tract similarity metrics. This work is important in that it does not use anatomical knowledge to perform the clustering, but relies solely on the characteristics of the data set at hand. The drawback of this method however, is that it does not consider multiple aspects of the tract or the data set. There are other metrics that can be calculated on a per tract basis that could lead to a more comprehensive clustering.

Brun *et al* [5] and Batchelor *et al* [3] address this issue of low order clustering metrics, by each using multiple metrics. Brun *et al* creates a feature vector representing each tract using the mean of coordinates of all points on the tract, as well as the covariance of the coordinates in a 3D space. Using this feature vector, pairwise tract comparisons are performed to create a weighted undirected graph, and partitions this space using normalized cuts. Whereas, Batchelor *et al* takes it a step further, and define more metrics, by using curvatures, torsions, and Fourier descriptors.

O'Donnel *et al* [23] and Voineskos *et al* [31] take a slightly divergent path, and use derived clustering metrics, but do so only for selected regions of interest where they perform their clusterings. They employ a spectral clustering technique that uses a similarity metric that is a modification of the Hausdorff distance (the upper bound of the minimal point-to-point distance between tracts), using high distances as low similarity and low distances as high similarity. This approach is taken primarily to increase clustering speed, while potentially sacrificing the insights that can be gained from whole brain clustering and visualization.

Often, Cartesian clustering is extended through the use of anatomical maps. One such example comes from Guevara *et al* [16]. They defined a robust clustering system for tractography data composed of a five step process, two steps of which are partitioning and Cartesian clustering. The partitioning is used to break the brain down into anatomical regions, and hierarchical clustering is performed separately in each region. The preprocess step of subsets does allow for faster clustering, but may hide more natural clusterings of the data, i.e., where tracts from neighboring subsets intersect.

Another example comes from Ros *et al* [26]. They proposed a clustering method using a hybrid of hierarchical clustering and an atlas-based classification. Their clustering classification is unique in that they develop a method called CASTOR (Cluster Analysis Through Smartly Extracted Representatives), which reduces the clustering space overhead. This allows for faster clusterings, but relies on the soundness of the representatives in creating coherent and meaningful clusters.

Summarizing the previous related work, all previous efforts use metrics derived from tracts as input to their clustering algorithms. However, many of the works do not describe their metrics, or, alternatively, describe metrics that are not suitable for large data (for example, the work of [21, 30] considers pairwise metrics between all tracts). For our study, our focus in on the methodology that transform tractography data into a smaller form for interactive visualization. Our methodology is conceptually capable of dealing with any per-tract metric, and we consider six such metrics in our own experiments.

2.3.2 Non-Clustered Tractography

Very little work has been done in the area of whole brain tractography visualization. Most often, simple line or tube representations are used to portray the data. These techniques have drawbacks however, the most prominent of which are a lack of depth and locality information.

One recent work by Petrovic *et al* [24] extends the tube representation to not only include enhanced depth information, but also an intricate in-image tract labeling system. This work does provide a well defined sense of locality of the tracts within the data set, and is implemented as a GPU-based renderer. One important contribution of this work is a Level of Detail management system that occludes low level tracts when the user is too far away to meaningfully view them. This enhances the speed and performance of the system, but lacks a fine tunning ability for a user to directly dial in the exact level of detail they need.

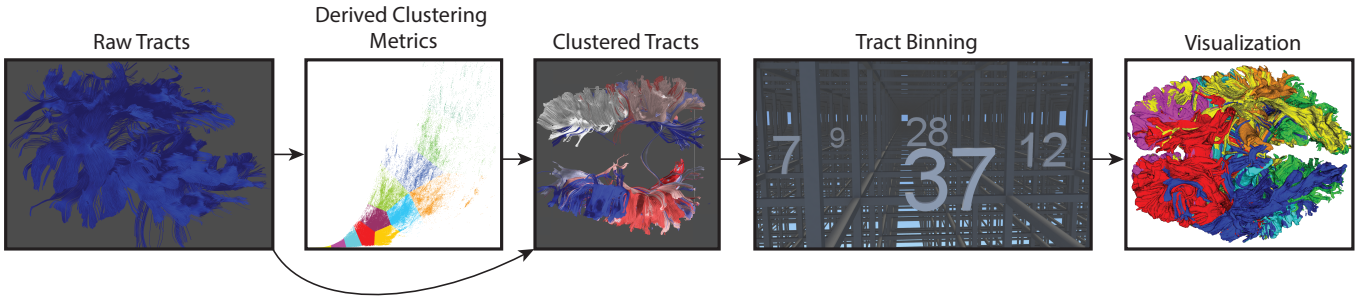


Fig. 2: A representation of the flow of data through our clustering and visualization pipeline. Demonstrating the transformation of raw tracts to derived metrics, to clustered tracts, to a binned volume, to the final visualization solution.

Two other works [13, 15] follow the same general pattern of Petrovic *et al*, and provide new and different ways of emphasizing tracts within the view plane. However, one item missing from each of these methods, is that they do not focus on data reduction or gaining insight into the structural qualities of the brain. Instead, they focus on the beautification of the very dense data they display.

With our study, we present a new visualization technique that incorporates important aspects from each of these works. We extend those works by enhancing the interactive and level of detail abilities of the rendering, by allowing the user to interactively dial in their desired level of detail. This ability combined with the space saving size of the visualization files, combine to form an intuitive visualization method for large-scale tractography data sets.

3 Methodology

Our method transforms tractography data into a set of three-dimensional histograms. These histograms can then be visualized with traditional techniques. The transform has free parameters, i.e., knobs that affect how the transform is carried out. As the output of the transform varies greatly based on parameter choices, a key part of our methodology is to locate the parameter values that optimize the output.

In the following sections, we describe the details of the transform and the parameters that affect it (Section 3.1), how we evaluate whether one set of parameters is better than another (Section 3.2), and lastly, we discuss the visualization options for our histograms (Section 3.3).

3.1 Transform

The transform occurs over three distinct phases: (1) calculating metrics on individual tracts, (2) clustering the tracts using these metrics, and (3) binning the tractography data for each cluster into a three-dimensional histogram. The resulting data can then be visualized using traditional scientific visualization techniques. Fig. 2 illustrates the transform.

The following subsections describe each of the phases in the transform. The free parameter associated with each phase is identified at the end of its section.

3.1.1 Tractography Metrics: Phase I

The purpose of this phase is to augment each tract in the data set with descriptive values. We do this by calculating metrics on a per tract basis, and we considered six metrics in this study. In order to prevent some metrics from overwhelming others, all metrics were normalized to values between 0 and 1. The six metrics were:

- **Tract Area (A)**, computed by taking the area of the bounding box around a tract.
- **Tract Length (L)**, computed by summing the individual line segment lengths between each pair of points that compose a tract.
- **Tract Curvature (C)**, computed by evaluating the maximum curvature along a tract. Specifically, we considered each pair of connected line segments within a tract, calculated its curvature, and then assigned the tract the maximum value.
- **Tract Linear Distance (LD)**, computed by calculating the linear distance between the starting and ending points of a tract.
- **Tract Start Position (SP)**, computed by calculating the linear distance from the starting point in a tract to a reference point. This was actually a family of metrics, measuring distance from three different reference points. Each of the reference points coincided with the bounding box of the overall data set.
- **Tract End Position (EP)**, computed similarly to **Tract Start Position**, but using the last point in a tract.

While it is possible to use all six of these metrics, it is not clear that they are all useful, i.e., that they lead to better clusterings. So we treat the metrics as one of our free parameters, i.e., which metrics should be used to cluster? We allow for all combinations except for the choice where none of the six metrics are used, meaning $2^6 - 1$.

Free Parameter for Phase I: metrics used (63 total options)

3.1.2 Tractography Clustering: Phase II

The purpose of this phase is to cluster tracts, and this is done using the metrics from Phase I. The output of Phase II is k clusters, with the clusters forming a partition over the original tractography data.

To perform the clustering, we opted to use the k-means++ algorithm. The goal of k-means++ is to partition n observations (tracts) into k clusters which minimize intra-cluster variance. K-means++ operates similarly to the k-means algorithm, only differing in the selection of initial seed locations. Tracts are represented by their metric values; these values combine to form a position in the Cartesian coordinate system. The intra-cluster variance is calculated based on the clustering domain in this Cartesian space. The algorithm starts by distributing k centroid points in the clustering domain. The points are placed according to the updated initialization algorithm developed by Arthur and Vassilvitskii [2]. The algorithm then iterates through a series of steps that update the positions of the k centroid points, attempting to minimize the intra-cluster sum of squares.

While k can be as low as one (meaning one cluster total) and as high as N_{tracts}, the number of tracts (meaning one cluster per tract), both of these extremes are likely sub-optimal in terms of maximizing user insight. We leave k as a free parameter for our subsequent optimization phase.

Free Parameter for Phase II: k (N_{tracts} total options)

3.1.3 Tractography Binning: Phase III

The purpose of this phase is to create bins of the tractography data, i.e., a three-dimensional histogram. The binning of the tracts is accomplished in two steps:

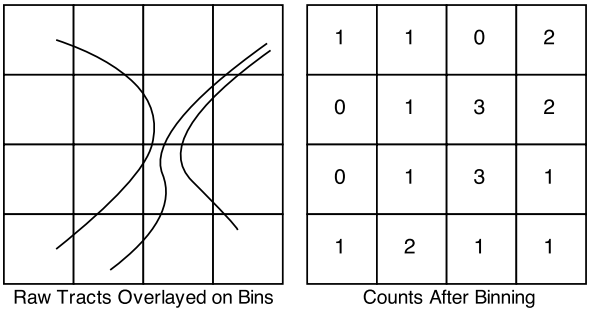


Fig. 3: Example showing three tracts being binned in two dimensions.

1. A binning volume is created to store a count of the number of tracts that cross any given bin in the volume. This volume is sized to be large enough to encompass the minimum and maximum extents of the entire input tractography data set.
2. Counts for each bin are calculated. This is done by considering each segment of each tract, determining the bins that each segment overlaps, and increasing the counts in those bins accordingly.

Fig. 3 demonstrates the binning in two-dimensions.

One control in this process is the granularity of the histogram, i.e., the total number of bins in the volume. If the total number of bins is high, then the storage costs are higher, but the subsequent visualizations are at a finer resolution. On the other hand, if the total number of bins is low, then the storage costs are lower, but the subsequent visualizations are coarser.

For our tests, we fixed the grid resolution to be $420 \times 420 \times 420$. This resolution gave a good representation of the underlying tractography data set, without being overly coarse. With a lower resolution, we would have lost many of the finer structures within the data set. We feel that this resolution is representative of the level of detail that would be needed for this data, and thus, did not treat grid size as a free parameter during the optimization phase.

3.2 Choices for Free Parameters

Let (m,k) be a choice in the parameter space, such that:

- m is a Boolean tuple. In our study, the tuple had six elements, since we considered six metrics. The value of $m[i]$ was true if the i^{th} metric was used as an input to the clustering and false otherwise. This was the free parameter associated with Phase 1 (Section 3.1.1).
- k is an integer denoting the total number of clusters. This was the free parameter associated with Phase 2 (Section 3.1.2).

We then choose an (m,k) configuration and run the clustering and evaluation steps based on these inputs.

Discussion of our approach to evaluating the optimum set of free parameters can be found in Section 4.

3.3 Interactive Cluster Exploration

Using the cluster histograms from our three-phase process, we are now able to interactively explore the entire tractography data set as a whole, or each cluster individually. There are multiple end user tools for large-scale visualization that can be used to accomplish this task. Examples include VisIt [7], ParaView [1], EnSight [8], and FieldView [20]. These tools provide interactivity through parallelization: parallel I/O requests, parallel processing, and parallel rendering [6]. As an additional benefit, these tools provide rich sets of algorithms. One particularly useful algorithm for our study was the ability to identify connected components on large data sets [17], and to then discard small components.

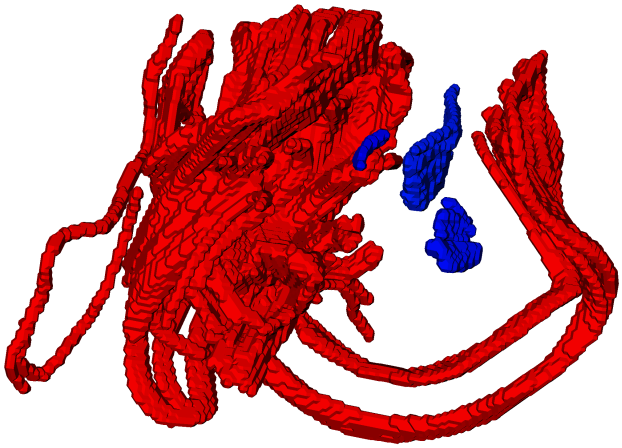


Fig. 4: Single cluster demonstrating multiple distinct components contained within the cluster.

For our study we utilized VisIt. Using the VisIt isosurfacing filter on cluster histograms readily shows the areas that have high concentrations of tracts, and hides areas with low concentrations. This ability effectively provides a "knob" for setting the amount of detail for accomplishing explorative tasks.

4 Selecting Free Parameters

Given the free parameters, our goal is to select an (m,k) such that the resulting histograms from the transform process is optimized for the user.

4.1 Evaluation Metrics

We considered two metrics to evaluate this optimization:

1. *The depth complexity:* This metric captures, on average, the number of cluster components stacked up in depth along a pixel. If the depth complexity is low, then the scene is likely comprehensible for the viewer. However, a low choice also may force unrelated things to be grouped together into the same cluster.
2. *The average number of connected components per cluster:* See Fig. 4, which shows a single cluster that contains multiple components, i.e., distinct regions that do not touch. Ideally, each cluster would have exactly one component, meaning the clustering algorithm grouped only very similar things together. However, in practice, each cluster contains multiple components; achieving one component per cluster requires increasing the total number of clusters (k) to a point that increases the depth complexity.

Our two metrics, then, are in tension. Minimal depth complexity is achieved by setting k to one (and thus having many connected components per cluster) and minimal connected components per cluster is achieved by setting k to be the same as the number of tracts (and thus having very high depth complexity). Our approach was to allow the user to set a cutoff for acceptable depth complexity. Our thinking was that the user would want the most information that they could comprehend, and that the depth complexity should be fixed to be at that point.

4.2 Search Space

Optimizing the selection of the free parameters required many different data runs to be conducted in a search space that contains up to $N_{tracts} \times 63$ possible configurations. We evaluated which values for the free parameters, (m,k), produced optimized clusterings, and present our findings in Section 6.

5 Experimental Overview

5.1 Clustering Software

Three distinct pieces of software were used in this work. We developed the first two pieces of software, and then utilized an existing clustering software for the third

The first piece of software, creates the derived clustering metrics. This code reads through the entire tractography data set, and creates metrics for each tract. These metrics are then given to the clustering software to perform clustering.

The piece of software is the binning code. This code creates a 3D grid space with a given resolution, and then bins an input tractography data set. This binning is performed by calculating which bins the tract intersects by tracking which bin faces the tract intersects.

The clustering software that we used is contained in the ALGLIB Free Edition package [4]. ALGLIB is a cross-platform numerical analysis and data processing library. Specifically, we used the ALGLIB k-means++ clustering implementation for all clustering tests.

5.2 Data Set

The data set that we worked with during this project was generated was provided by Electrical Geodesics Inc. The data set contained a total of 49,6646 tracts, for a total of 3.6 Gigabytes.

5.3 Experimental Machines

Two different machines were used during the development and evaluation stages of our work:

- A desktop computer containing two 2.60 GHz Intel Xeon(R) E5-2650 v2 8 core CPUs and a total of 64 GB of memory.
- The parallel Oak Ridge National Laboratory Sith machine, containing 39 nodes. Each node contains four 2.3 GHz 8 core AMD Optetron processors and 64 GB of memory, configured with an 86 TB Lustre file system for scratch space.

We ran more than $15,000$ different test configurations during the course of our study, and used more than $50,000$ node hours.

6 Exploring Interaction over (m,k)

Our analysis of (m,k) consisted of two distinct experiments. In the first, we explored the relationships and patterns in our tractography metrics from Section 3.1.1. In the second, we explore the effects of varying the value for (k). The analysis of these two experiments are in Sections 6.1 and 6.2, respectively.

6.1 Selecting Optimal Values for (m)

As the target value for (k) varies, the best set of metrics (m) may also vary. That is, for very few clusters, one set of metrics may be best, and, for very many clusters, another set of metrics may be best. With this first part of our analysis, we wanted to understand how the various configurations of (m) affected clustering quality for varying values of (k).

To determine the relationship, we set up a series of tests using every possible combination of (m), with eight different values for (k). From these runs we then plotted and evaluated the clustering performance at each value of (k), using the metrics we defined in Section 3.1.1. The plots show a very persistent pattern in the quality of the clusterings produced, see Fig. 5. There are six distinct groups that form for every value of (k) that we used. These groups demonstrate that the quality of the clusterings produced by different values of (m) are persistent across different values for (k). We demonstrate this persistence with the colored values in each of the plots that show three of the best values for (m), and how they track across various (k).

From this data we can say which metrics (discussed in Section 3.1) are more useful than others, and which combinations of metrics produce the best clusterings under our evaluation schema. The absolute worst clusterings are featured in the upper four groupings in each of the graphs (see Fig. 5). Consistently, the worst metric was curvature used by itself. In every test, this produced the number one worst result. In fact, when Area, Length, Curvature, and Linear Distance are used alone, their performance is worse than when any combination of metrics is used, and significantly worse, than when Start Position and End Position are used on their own. Table 1 demonstrates the performance of each of the clustering metrics used singularly. This table is representative of the results seen at other values of (k).

Table 1: Table showing the performance of each of the metrics used singularly for $(k = 50)$ in terms of the average number of connected components and total surface area.

Metric	Total Surface Area	Average # of Con. Comp.
Curvature	9,148,415	74.2
Linear Distance	6,382,081	45.8
Length	6,253,787	48.1
Area	5,698,423	8.1
Start Position	***1,591,580***	***4.8***
End Position	***1,412,429***	***4.7***

We were able to conclude, that to have a clustering that performed well under our evaluation, it had to include Start Position, End Position, or both. Used singularly with other metrics, they performed well, but not as well when used together. Further, when both are used in conjunction with either Length, Area, or both, we saw the best performing clusterings.

A further clear trend in this analysis is that we do not have the case where we have to perform a trade-off, of either picking metrics that reduce the number of connected components, or metrics that reduce the total surface area of our clusterings. We are always able to optimize for both.

Persistent performance is the best result we could have gotten with these tests. It means that we are able to eliminate the majority of (m) from consideration when optimizing (k). Three of the top performing metrics were:

- **L/SP/EP:** Length, Start Position, End Position
- **SP/EP/A:** Start Position, End Position, Area
- **L/SP/EP/A:** Length, Start Position, End Position, Area

We classified these three metrics as optimal configurations for the remaining tests in our study. We could have also chosen SP/EP, L/C/SP/EP, or L/C/A/SP/EP, but they were less persistent across tests, and showed more variation.

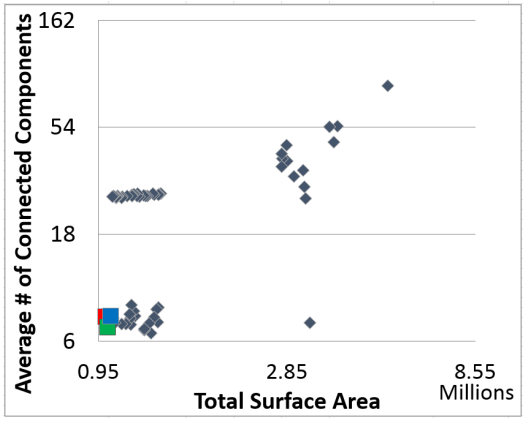
6.2 Studying the Effects of (k)

As the value for (k) varies, the performance of our clusterings, based on the metrics from Section 3.1.1, is going to vary. In order to understand the trends in performance, we ran tests on 200 values for (k), using the top three optimal metrics as determined in Section 6.1. From these runs we then plotted and evaluated the clustering performance at each value of (k).

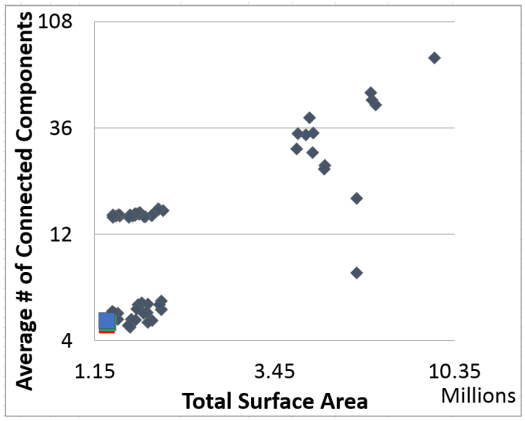
The plot in Fig. 6 shows the two trends in performance for our evaluation criteria. The lower the value of (k), the more connected components per cluster. While the higher the value of (k), the fewer connected components. The initial decreases in average number of connected components as the values for (k) vary from $(k = 1)$ to $(k = 25)$ are substantial, dropping from 66 to approximately 7.7

Another trend in Fig. 6, is that surface area increases as (k) increases. As each new cluster is introduced, the surface area rises, giving us a constantly increasing surface area for larger and larger values of (k). This is opposite the trend seen with average number of connected components, which constantly decreases with higher values of (k).

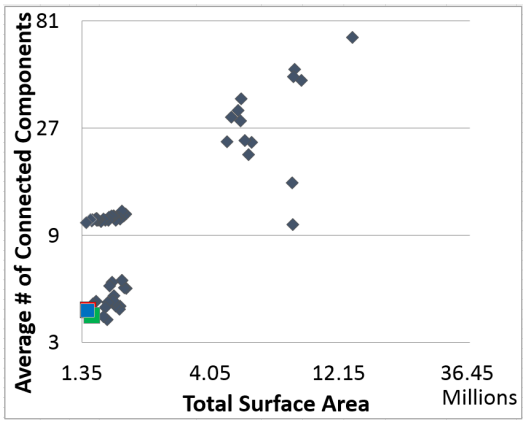
An important observation that can be made from Fig. 6, is that the benefits of increasing numbers of clusters diminishes extremely quickly. The average number of connected components for $(k = 30)$ using the *L/SP/EP/A* metric is 6.3. Whereas, using the same metric at $(k = 200)$, the average number of connected components only drops to 2.75, while surface area rises from (1.08) million to (1.98) million. The drop in average connected components is extremely small



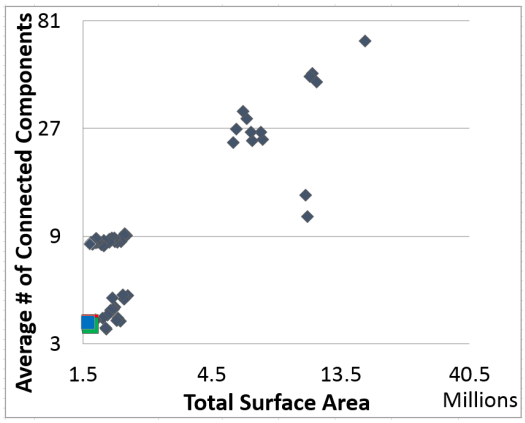
(a) 25 Clusters



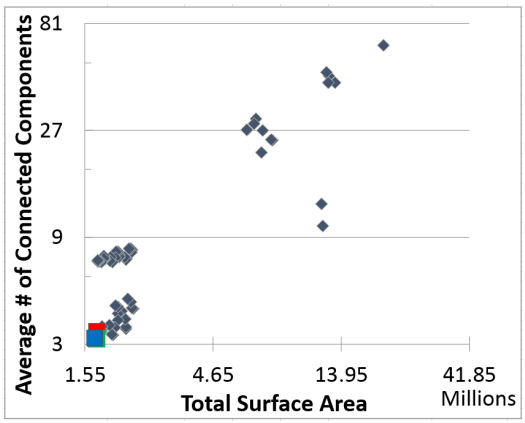
(b) 50 Clusters



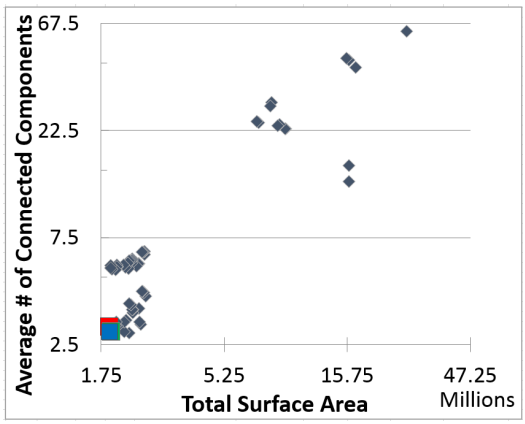
(c) 75 Clusters



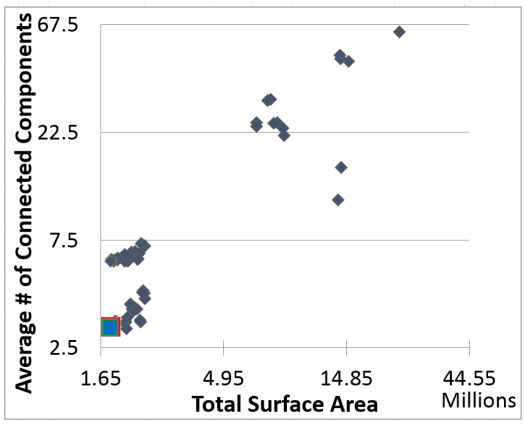
(d) 100 Clusters



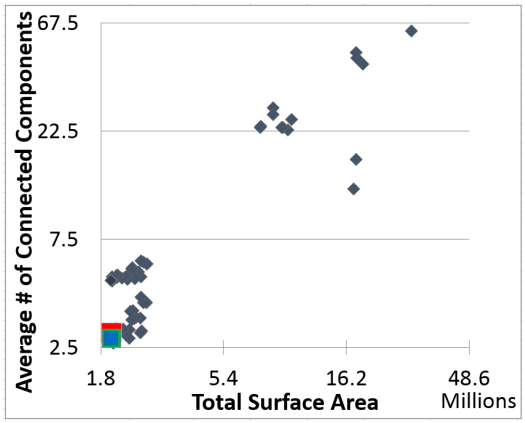
(e) 125 Clusters



(f) 150 Clusters



(g) 175 Clusters



(h) 200 Clusters

Fig. 5: Average number of connected components compared against surface area for all 63 clustering metrics. The best metrics that we chose are shown as square: red is *L/SP/EP*, green is *SP/EP/A*, blue is *L/SP/EP/A*

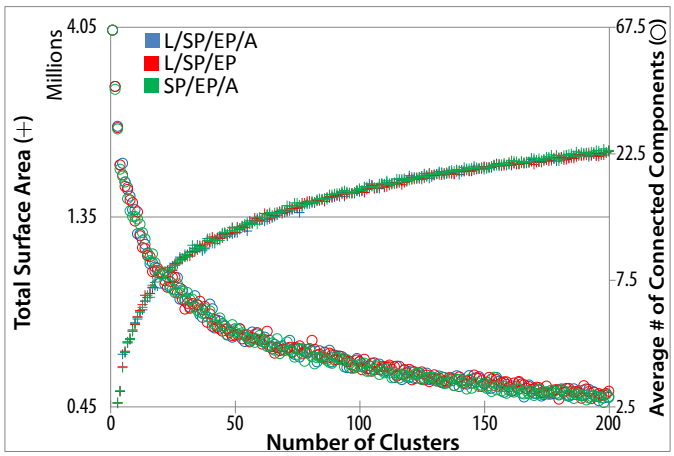


Fig. 6: Average number of connected components per cluster (o) versus the total surface area of all clusters (+) using the three optimum clustering metrics.

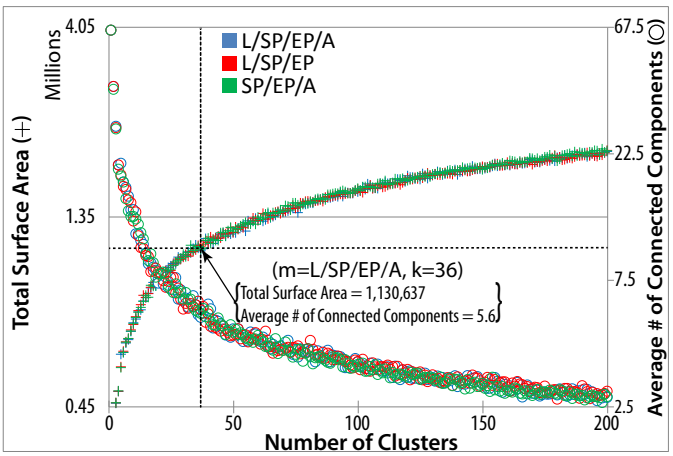


Fig. 8: Drawing the horizontal and vertical lines at the user specified depth complexity to determine the optimal value for (m,k)

when compared to the large rise in total surface area, and the added visual complexity of 170 more clusters. We describe our algorithm for selecting (m,k) pairs in Section 7.

7 Algorithm for Selecting (m,k)

Given the optimal set of clustering metrics that we learned in Section 6.1, and the clustering performance curves that we learned in Section 6.2, we can now describe an algorithm for selecting (m,k) pairs.

Given a depth complexity requirement, we can select a value for (m,k) that produces the optimal configuration. In our study, we use total surface area of all clusters to analogously represent depth complexity. We do this by making the following assumptions:

- The tracts were binned on a 420^3 grid;
- An orthographic projection;
- The final image fills the screen;
- The user sets a depth complexity requirement per pixel (e.g., 6.5 triangles per pixel in depth on average).

Fig. 7 demonstrates the depth complexity of the scene for eight different values, from $(k = 25)$ to $(k = 200)$, in increments of 25.

Using our assumptions, we can then start by selecting a value for (k). Assuming the user asks for a depth complexity of 6.5, we start by multiplying 6.5 by the number of grid cells in a single plane of the data, 420^2, which gives us our analogous surface area value $1,146,600$. Using this value, we search for a (k) that creates clusterings as close to this value as possible. Visually, we draw a horizontal line across the total surface area curve from our chart at the total surface area value of $1,146,600$, as in Fig. 8. We then draw a vertical line down to the x-axis, to get our values for (m) and (k). This line gives us a value of $(k = 36)$, and we then select the (m) at that point that produces the fewest number of connected components, $(m = L/SP/EP/A)$.

8 Method in Practice

Our software pipeline consists of three distinct phases. (1) Creating Derived Metrics; (2) Clustering; and 3) Binning. Each stage of this pipeline is constrained by different time and size bounds. Some stages are time constrained by the input data size, while others are time constrained by the number of clusters, and further still, some by a function of both. In Table 2, we present the time complexity and output file size of each stage in Big O notation.

The most time intensive operation in our pipeline is clustering as it is dependent on the number of clusters, the number of input tracts, and the dimensionality of the derived clustering metrics. As more clusters, tracts, or dimensions are introduced, the clustering will take more time. The current clustering implementation presents a bottleneck for tractography data sets that contain hundreds of millions of tracts. However, this can be addressed through the use of parallel clustering, which we did not address in the scope of this work.

Table 2: Table showing the Big O time and storage size complexity of each phase of our pipeline. n denotes the number of tracts to be clustered, s denotes the average number of segments per tract, k denotes the number of clusters, d denotes the dimensionality of the derived clustering metrics, and g denotes the binning grid size.

	Derived Metrics	Clustering	Binning
Time:	$\mathcal{O}(n \times s \times d)$	$\mathcal{O}(n^{d \times k+1} \times logn)$	$\mathcal{O}(n \times s + k \times g^3)$
Size:	$\mathcal{O}(n \times d)$	$\mathcal{O}(n)$	$\mathcal{O}(k \times g^3)^*$

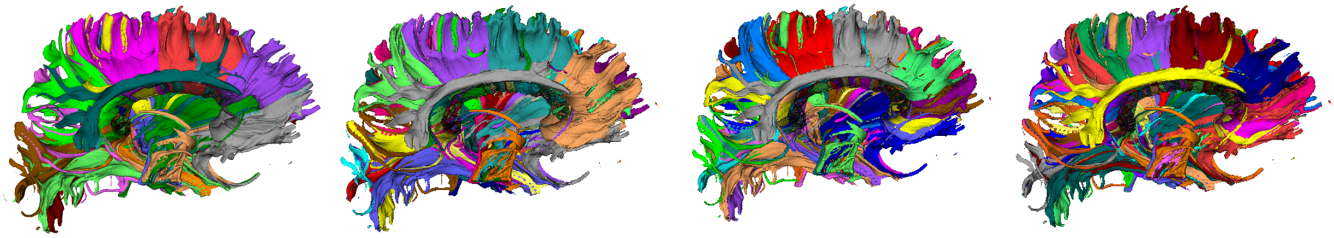
(Represents uncompressed sparse grid size)*

The operation in our pipeline that generates the most data is binning. A new file is created for each cluster generated, at a fixed size per file, dependent on the grid size. These files represent regular grids, and due to the data locality introduced by clustering, these files end up with large very sparse regions. Using compression, we are able to compress these files at a minimum ratio of 370 to 1, using our current input data set. This compression leaves us with clustered binary files that are much smaller than the input data set, reducing the storage overhead associated with the raw tract files. Table 3 shows the total visualization file sizes and run times for the clustering section of the pipeline for eight varied values of (k).

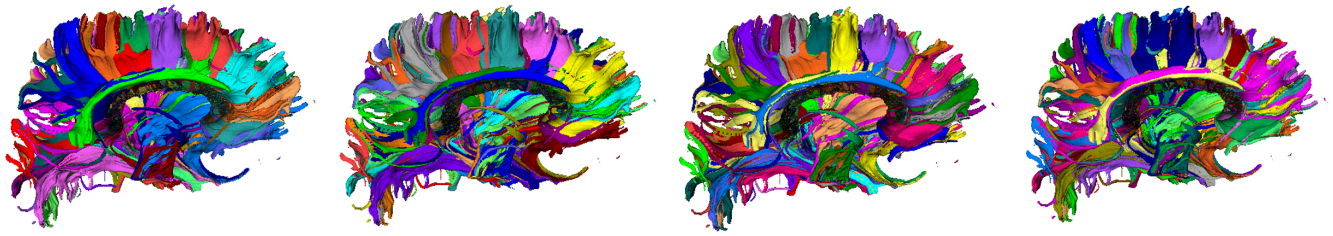
As demonstrated in Table 3, file sizes for the binary clustered files become quite large as the number of clusters grows beyond 50. Large

Table 3: Table showing the run time for clustering (the most time intensive section of the pipeline), the total size of all of the binary files used in post-hoc visualization in raw and compressed form, and the achieved compression factor.

Num Clusters	Clustering Time (min)	Binary File Size	gzip Compressed	Compression Ratio
25	4.9	7.2 GB	19.4 MB	371.1
50	9.4	14.5 GB	29.2 MB	496.6
75	20.5	21.7 GB	38.5 MB	563.6
100	25.9	28.9 GB	47.5 MB	608.4
125	32.5	37.0 GB	56.3 MB	657.2
150	45.0	43.4 GB	65.1 MB	666.7
175	58.1	50.7 GB	73.8 MB	687.0
200	72.8	57.9 GB	82.5 MB	701.2



(a) 25 Clusters (b) 50 Clusters (c) 75 Clusters (d) 100 Clusters



(e) 125 Clusters (f) 150 Clusters (g) 175 Clusters (h) 200 Clusters

Fig. 7: Plots showing an isosurface (value = 10), clipped in half. This demonstrates the varying depth complexity for different values of (k) using metric *L/SP/EP/A*

enough in fact, that visualizing this data on a single commodity node becomes prohibitive due to RAM limitations. This obstacle can be overcome by utilizing a visualization tool that operates on sparse data in compressed form, however, we did not investigate this path. Instead, we accomplished the visualization for large numbers of clusters using a distributed memory version of VisIt, running across multiple nodes on Sith. One effective visualization technique that emphasizes areas of high tract concentration is contouring. In Fig. 9 you can see the areas of very high tract density emphasized as higher isovalues are applied.

8.1 Example Workflow

Each stage in our pipeline is presented below, along with a representative amount of time spent in that section using a value of $(k = 50)$.

- Creating Derived Metrics (82 seconds)
  - This stage calculates the derived metrics for each tract so that clustering can be performed. Using this data set, this file is 38 MB.
- Clustering (564 seconds)
  - This stage is where the clusters are determined based on the derived clustering metrics. Each cluster is saved for reuse as well as the next stage, at approximately 5 MB per file.
- Binning (100 seconds)
  - Using the clustered tracts from the previous stage, separate binary files containing the binned tracts for each cluster are created and saved to allow for post-hoc visualization. At current bin resolution of 420^3, this is 294.6 MB per cluster uncompressed, or 584 kB compressed.
- Visualization (70 seconds)
  - Performing an isosurface on the 50 binned binary files running a parallel memory version if VisIt on 15 Sith nodes. This time may be reduced by utilizing a visualization tool that operates on sparse data.

Overall, the total time to generate 50 clusters and perform an initial visualization with the current input data size is approximately 13.6 minutes. This time can still be reduced by perusing parallel clustering and a visualization tool that operates on compressed sparse data.

9 Conclusions and Future Work

We have described a methodology for the interactive visualization of tractography data. We have shown the creation of whole brain tractography clusters based on the creation of derived metrics and K-means++ clustering. We have shown a process under which the set of optimal clustering metrics chosen from the initial set of clustering metrics can be chosen, as well as an evaluation of the optimal number of clusters. Our clusterings combined with our binning visualization technique, provide a unique visualization and space saving solution.

Using compression, we are able to drastically reduce the size of our binned datasets, below that of the size of the original input data set. This space savings will likely be even larger on data sets with 10's of millions of tracts, and when the number of clusters is increased.

In terms of future work, we would like to implement a method for parallel clustering and perform an analysis on the effect of varying the binning grid resolution. Parallel clustering combined with either adaptive grid resolution in the binning mesh, or a visualization system that works on sparse data sets, would allow for a very large speedup in our pipeline. Having accomplished this speedup, we would then like to confirm our analysis using much much larger tractography datasets, of 10's to 100's of millions of tracts.

References

[1] J. Ahrens, B. Geveci, and C. Law. Visualization in the paraview framework. In C. Hansen and C. Johnson, editors, *The Visualization Handbook*, pages 162–170, 2005.

[2] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

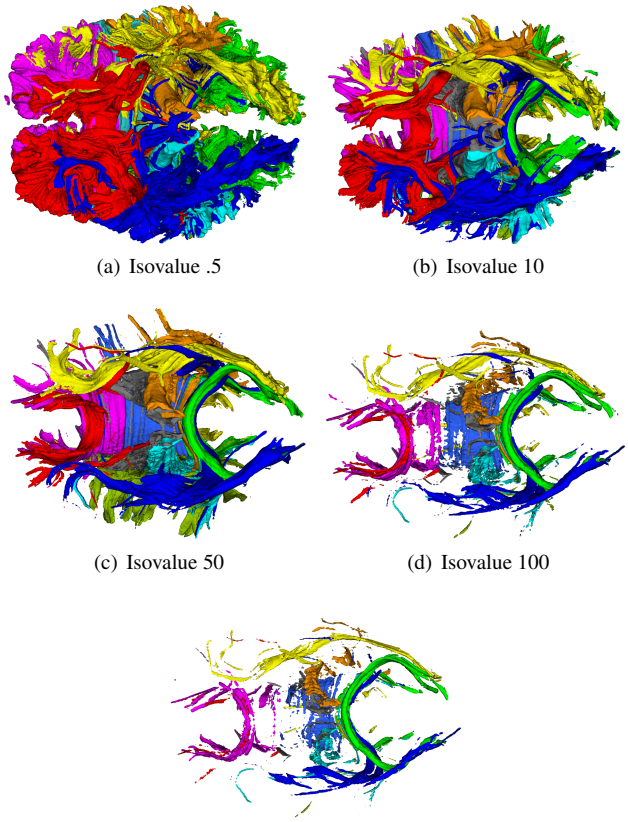
[3] P. Batchelor, F. Calamante, J.-D. Tournier, D. Atkinson, D. Hill, and A. Connelly. Quantification of the shape of fiber tracts. *Magnetic Resonance in Medicine*, 55(4):894–903, 2006.

[4] S. Bochkanov and V. Bystritsky. Alglib, June 2015.

[5] A. Brun, H. Knutsson, H.-J. Park, M. E. Shenton, and C.-F. Westin. Clustering fiber traces using normalized cuts. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2004*, pages 368–375. Springer, 2004.

[6] H. Childs. Parallel Visualization Frameworks. In *High Performance Visualization—Enabling Extreme-Scale Scientific Insight*, pages 9–24. Oct. 2012.

[7] H. Childs, E. Brugger, B. Whitlock, J. Meredith, S. Ahern, D. Pugmire, K. Biagas, M. Miller, C. Harrison, G. H. Weber, H. Krishnan, T. Fogal,



(a) Isovalue .5 (b) Isovalue 10

(c) Isovalue 50 (d) Isovalue 100

(e) Isovalue 125

Fig. 9: Isovalue series showing a contour of the binned tractography at five different values.

A. Sanderson, C. Garth, E. W. Bethel, D. Camp, O. Rübel, M. Durant, J. M. Favre, and P. Navrátil. VisIt: An End-User Tool For Visualizing and Analyzing Very Large Data. In *High Performance Visualization—Enabling Extreme-Scale Scientific Insight*, pages 357–372. Oct. 2012.

[8] Computational Engineering International, Inc. *EnSight website*, June 2015.

[9] F. DellAcqua and M. Catani. Structural human brain networks: hot topics in diffusion tractography. *Current opinion in neurology*, 25(4):375–383, 2012.

[10] M. Descoteaux, R. Deriche, T. Knosche, and A. Anwander. Deterministic and probabilistic tractography based on complex fibre orientation distributions. *Medical Imaging, IEEE Transactions on*, 28(2):269–286, 2009.

[11] P. Douek, R. Turner, J. Pekar, N. Patronas, and D. Le Bihan. Mr color mapping of myelin fiber orientation. *Journal of computer assisted tomography*, 15(6):923–929, 1991.

[12] R. R. Edelman and S. Warach. Magnetic resonance imaging. *New England Journal of Medicine*, 328(10):708–716, 1993.

[13] M. H. Everts, H. Bekker, J. B. Roerdink, and T. Isenberg. Depth-dependent halos: Illustrative rendering of dense line data. *Visualization and Computer Graphics, IEEE Transactions on*, 15(6):1299–1306, 2009.

[14] A. Goh, C. Lenglet, P. M. Thompson, and R. Vidal. Estimating orientation distribution functions with probability density constraints and spatial regularity. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2009*, pages 877–885. Springer, 2009.

[15] A. J. Golby, G. Kindlmann, I. Norton, A. Yarmarkovich, S. Pieper, and R. Kikinis. Interactive diffusion tensor tractography visualization for neurosurgical planning. *Neurosurgery*, 68(2):496, 2011.

[16] P. Guevara, C. Poupon, D. Rivière, Y. Cointepas, M. Descoteaux, B. Thirion, and J.-F. Mangin. Robust clustering of massive tractography datasets. *Neuroimage*, 54(3):1975–1993, 2011.

[17] C. Harrison, J. Weiler, R. Bleile, K. Gaither, and H. Childs. A Distributed-Memory Algorithm for Connected Components Labeling of Simulation Data. In *Topological and Statistical Methods for Complex Data – Tackling Large-Scale, High-Dimensional, and Multivariate Data Sets*, pages 3–21. Springer, Dec. 2014.

[18] S. Jbabdi, M. Woolrich, J. Andersson, and T. Behrens. A bayesian framework for global tractography. *Neuroimage*, 37(1):116–129, 2007.

[19] H. Johansen-Berg and T. E. Behrens. *Diffusion MRI: from quantitative measurement to in vivo neuroanatomy*. Academic Press, 2013.

[20] S. M. Legensky. Interactive Investigation of Fluid Mechanics Data Sets. In *VIS '90: Proceedings of the 1st conference on Visualization '90*, pages 435–439. IEEE Computer Society Press, 1990.

[21] B. Moberts, A. Vilanova, and J. J. van Wijk. Evaluation of fiber clustering methods for diffusion tensor imaging. In *Visualization, 2005. VIS 05. IEEE*, pages 65–72. IEEE, 2005.

[22] S. Mori, B. J. Crain, V. Chacko, and P. Van Zijl. Three-dimensional tracking of axonal projections in the brain by magnetic resonance imaging. *Annals of neurology*, 45(2):265–269, 1999.

[23] L. ODonnell, M. Kubicki, M. E. Shenton, M. H. Dreusicke, W. E. L. Grimson, and C.-F. Westin. A method for clustering white matter fiber tracts. *American Journal of Neuroradiology*, 27(5):1032–1036, 2006.

[24] V. Petrovic, J. Fallon, and F. Kuester. Visualizing whole-brain dti tractography with gpu-based tuboids and lod management. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1488–1495, 2007.

[25] Y. Rathi, O. Michailovich, K. Setsompop, S. Bouix, M. E. Shenton, and C.-F. Westin. Sparse multi-shell diffusion imaging. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2011*, pages 58–65. Springer, 2011.

[26] C. Ros, D. Güllmar, M. Stenzel, H.-J. Mentzel, and J. R. Reichenbach. Atlas-guided cluster analysis of large tractography datasets. *PloS one*, 8(12):e83847, 2013.

[27] M. Rubinov and O. Sporns. Complex network measures of brain connectivity: uses and interpretations. *Neuroimage*, 52(3):1059–1069, 2010.

[28] B. Scherrer and S. K. Warfield. Toward an accurate multi-fiber assessment strategy for clinical practice. In *Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on*, pages 2140–2143. IEEE, 2011.

[29] D. S. Tuch, T. G. Reese, M. R. Wiegell, and V. J. Wedeen. Diffusion mri of complex neural architecture. *Neuron*, 40(5):885–895, 2003.

[30] E. Visser, E. H. Nijhuis, J. K. Buitelaar, and M. P. Zwiers. Partition-based mass clustering of tractography streamlines. *Neuroimage*, 54(1):303–312, 2011.

[31] A. N. Voineskos, L. J. O'Donnell, N. J. Lobaugh, D. Markant, S. H. Ameis, M. Niethammer, B. H. Mulsant, B. G. Pollock, J. L. Kennedy, C. F. Westin, et al. Quantitative examination of a novel clustering method using magnetic resonance diffusion tensor tractography. *Neuroimage*, 45(2):370–376, 2009.

[32] A. Yenkiki, P. Panneck, P. Srinivasan, A. Stevens, L. Zöllei, J. Augustinack, and B. Fischl. Automated probabilistic reconstruction of white-matter pathways in health and disease using an atlas of underlying anatomy. *Frontiers in Neuroinformatics*, 5(23), 2011.