FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Efficient Visualization and Processing of MRI Diffusion Tractography Data

using the Medical Imaging Interaction Toolkit

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Medizinische Informatik

eingereicht von

Ignaz Reicht
Matrikelnummer 0827782

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Dr. Yll Haxhimusa
Mitwirkung: Dr. Klaus H. Fritzsche

Wien, 08.05.2012

(Unterschrift Verfasser) (Unterschrift Betreuung)

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Efficient Visualization and Processing of MRI Diffusion Tractography Data

using the Medical Imaging Interaction Toolkit

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Medical Informatics

by

Ignaz Reicht
Registration Number 0827782

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Dr. Yll Haxhimusa
Assistance: Dr. Klaus H. Fritzsche

Vienna, 08.05.2012

(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Ignaz Reicht
Griesbachweg 46, 6370 Reith bei Kitzbühel

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Wien, 08.05.2012)

(Unterschrift Verfasser)

Danksagung

An dieser Stelle möchte ich mich recht herzlich bei meinen beiden Betreuern, Herrn Klaus Fritzsche und Herrn Yll Haxhimusa, für die reibungslose Umsetzung und Abwicklung meiner Diplomarbeit sowie für die tatkräftige organisatorische und fachliche Unterstützung bedanken. Weiteren Dank gilt Herrn Peter Neher und Markus Fangerau die mir beide programmiertechnisch immer zur Seite standen. Mein Dank sei auch an Herrn Bram Stieltjes gerichtet der dank seiner experimentierfreudigkeit aufschlussreiches Feedback zur praktischen Umsetzung dieser Arbeit lieferte. Generell möchte ich mich bei der Abteilung Prof. Meinzer am DKFZ bedanken welche dieses Projekt überhaupt erst möglich machte. In den letzten Stunden vor Fertigstellung und Abgabe meiner Arbeit wäre ich ohne die Hilfe meiner Verlobten, Dharanija Madhavan, nicht rechtzeitig fertig geworden - DANKE!

Abstract

Tractography is a technique providing information about anatomical connections of the brain using *Diffusion Weighted Magnetic Resonance Imaging* (DWI). Hence, fiber tracking shows high potential in the field of neuro science by facilitating the study of anatomical structures of the human brain as well as neuronal diseases such as *Alzheimer's Disease* (AD) and lately this technique is also used for monitoring cancer treatment. For the last couple of years, several tractography techniques show robust and clinically relevant results. However, neuro surgeons, radiologists and domain experts have to deal with complex information provided by tractography and DWI. In other words, large data sets need to be processed and combined efficiently as well as intuitively represented in 2D and 3D. Practical setups, especially clinical applications, require several conditions, e.g. usability, consistency in the medical context, accuracy, performance and compatibility. This work introduces a framework for efficient processing of fiber data including high performance visualization and interaction. The provided tools and techniques are designed to cover common tasks whereas their accuracy and performance show promising results for practicable applications in this field. Since fiber tracking algorithms are well established and available, this framework focuses on bringing tractography in combination with fiber processing into practice. So far, fiber tracking is used mainly in the radiological field which is why the framework concentrates on radiological applications. At the moment most fiber tracking solutions come with their own visualization and interaction mechanisms which are rarely adaptable to practical and clinical purposes. The outcome of this work presents an efficient solution for tractography post-processing, focusing on both scientifically and clinically relevant tasks. Compatibility to previously existing fiber tracking applications are also taken into consideration. To maximize availability, the framework is implemented into the diffusion imaging module of the open-source software platform, the *Medical Imaging Interaction Toolkit* (MITK), which is well known for medical image processing.

Kurzfassung

Die vorliegende Arbeit wurde zusammen mit den Abteilungen Medizinische und Biologische Informatik und Radiologie am Deutschen Krebsforschungszentrum (DKFZ) erarbeitet. Tractographie, auch Fiber Tracking genannt, bezeichnet eine Bildverarbeitungstechnik die mittels diffusionsgewichteter Magnetresonanz Tomographie (DWI) Informationen über die anatomischen Strukturen des menschlichen Gehirns ableiten kann. Zusätzlich findet diese Technik auch Anwendung in der Therapieplanung und Verlaufskontrolle bei Tumorpatienten. Neue Ansätze und Optimierungen von Fiber Tracking Methoden konnten bereits im Vorfeld deren klinische Relevanz aufzeigen. Bis zum heutigen Zeitpunkt wurden noch keine praxistauglichen Endanwendungen entwickelt, welche sich der Komplexität von Tractographie annehmen und die Daten und Informationen dementsprechend verarbeiten um das Potential von Fiber Tracking in einem klinisches Umfeld voll ausgeschöpfen zu können. Der Hauptanwendungsbereich ist dem radiologischen Umfeld zuzuordnen in dem performante und interaktive Bildverarbeitungsmechanismen von hoher Genauigkeit sowohl in 2D als auch in 3D den Alltag bestimmen. Aus diesem Grund wurde bei der Umsetzung dieser Arbeit speziell auf diese Anforderungen eingegangen und ein effizientes Framework definiert welches die intuitive Visualisierung und Interaktion verschiedenster Tractographie Ergebnisdaten realisiert um so eine Basis zu schaffen, welche sowohl praxisnahen Anwendungen aus der Radiologie als auch Anforderungen aus dem Gebiet der Neurowissenschaften gerecht wird. Als Zielsetzung galt eine Anwendung bereitzustellen, welche sowohl effizient, intuitiv als auch funktionell umfangreiche Eigenschaften aufweist und durch die Verwendung bereits existierender Standards die Integration und Verwendung externer Fiber Tracking Verfahren motiviert. Zusammen mit Radiologen und Wissenschaftlern wurden die umgesetzten Aufgabenstellung auf Benutzbarkeit, Funktionalität, Effizienz und Leistung für einen praxisnahen Einsatz evaluiert. Als Entwicklungsplattform wurde das open-source Softwareprojekt Medical Imaging Interaction Toolkit (MITK) herangezogen, welches bereits eine beachtliche Community im wissenschaftlichen, kommerziellen als auch klinischen Umfeld betreut.

Contents

Nomenclature

AD Alzheimer's Disease

API Application Programming Interface

CT Computer Tomography

DICOM The Digital Imaging and Communications in Medicine

DKFZ Deutsches Krebsforschungszentrum, German Cancer Research Center

DSI Diffusion Spectrum Imaging

DTI Diffusion Tensor Imaging

DWI Diffusion Weighted Imaging

FA Fractional Anisotropy

FACT Fiber Assignment by Continuous Tracking

FB mitkFiberBundle

FBX mitkFiberBundleX

GDCM Grassroots DICOM

GFA Generalized Fractional Anisotropy

GLSL OpenGL Shading Language

GUI Graphical User Interface

ITK Insight Toolkit

MBI Institut of Medical and Biological Informatics

MITK The Medical Imaging Interaction Toolkit

MITK-DI MITK Diffusion Imaging module

MPR Multi Planar Reconstruction

MRI Magnetic Resonance Imaging

MRT Magnetic Resonance Tomography

PF mitkPlanarFigure

QSI q-Space Imaging

ROI Region of Interest

TBSS Tract Based Spatial Statistics

Tcl/TK Tool Command Language / Tool Kit

TDI Trackt Density Imaging

VTK Visualization Toolkit

CHAPTER 1

Introduction

Magnetic Resonance Tomography (MRT) is a well established noninvasive imaging technique allowing the acquisition of hydrogen containing tissues [1]. Based on the principles of MRT, *Diffusion Weighted Imaging* (DWI) [2] is able to measure diffusion of water molecules [3, 4]. Figure 1.1(a) shows the movement of a water molecule schematically. This method has assumed an important position in the field of both clinical routine and neurological science to study neurological diseases [5–11], and to monitor cancer treatment [12–15]. Studies show that the white matter of the human brain represents an ideal environment for DWI, enabling the investigation of the connectivity of anatomical structures within the brain [9, 16, 17].

Modeling diffusion

In an anisotropic environment diffusion can be described as a tensor [18] where the direction and magnitude of diffusion are acquired by measuring different gradients. This can be illustrated as a 3D ellipsoid [19] where size, shape and orientation are derived from the tensor. Figure 1.1(b) shows ellipsoids and 3D tensors. In *Diffusion Tensor Imaging* (DTI) the diffusion of water molecules in one voxel is modeled using a 3D gaussian function [18]. The tensor describes the anisotropy of diffusion in a voxel [20]. An approach that illustrates diffusion data in 3D is the visualization of the *Orientation Distribution Function* (ODF) which is shown in figure 1.1(c). However, DTI shows limitations in resolving fiber crossings, bendings and twistings since a 3D tensor represents only six independent elements where three describe the orientation and the remaining three the magnitude of diffusion [21]. In contrast, *q-Space Imaging* (QSI) estimates diffusion without the Gaussian approach overcoming the limitations of DTI since this approach samples the diffusion signal directly onto a 3D Cartesian coordinate system or spherical shell. The two techniques based on these approaches are *Diffusion Spectrum Imaging* (DSI) and *q-Ball Imaging* [22, 23] respectively. Both DTI and QSI provide the basis for tractography.

Tractography

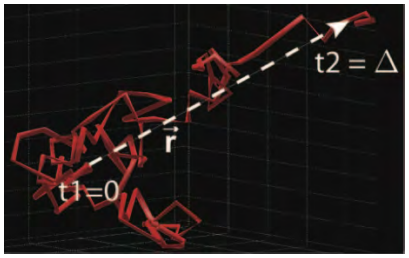
Information about the flow of molecules is processed by computational post-processing algorithms, also known as tractography or fiber tracking, allowing the reconstruction of corresponding pathways [24, 25]. Especially the acquisition of moving water molecules inside and in between axons provide the input for fiber tracking [26, 27]. It is to be noted that the outcome of fiber tracking algorithms represent not actual anatomical fibers, but reconstructed fibers representing areas of diffusion, assuming that anatomical fibers also occur accordingly. The impact of tractography has been noticed over recent years since the results became more reliable and of clinical relevance. [28,29]. In principal, tractography algorithms are divided into two groups, local [30,31] and global methods [32–35]. Local methods require a starting region from where the fiber is assembled by voxelwise information step by step, whereas global methods reconstruct the whole fiber network simultaneously. Although the global methods are computationally more challenging compared to local methods, they promise more robust results. An example for a voxelwise, step by step tracking is illustrated in figure 1.1(d).

Current situation

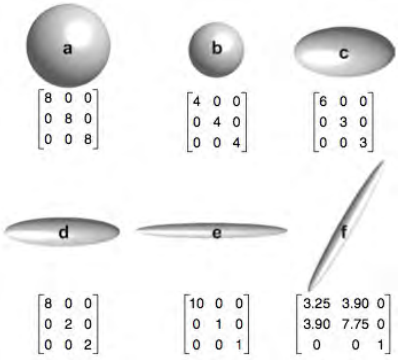
However, tractography has not yet arrived into clinical practice so far because efficient methods for visualization and tools for accurate interaction with fiber structures are not well established according to the expectations of end users, such as radiologists, neuro surgeons and scientists. In radiology, viewing images from different perspectives including sagittal, coronal and transversal in 2D are more prevalent than 3D representation. *So far, fiber visualization is performed mostly in 3D but intuitive and efficient 2D visualization still is challenging especially when it comes to large data sets* [36]. Performance issues as well as inconsistent representation of fiber structures in 2D and 3D lead to limitations of accurate fiber post-processing and interaction. For investigating medical images, image processing in radiology contains several segmentation and region of interest (ROI) related tasks. Therefore, to establish acceptance from the end users, similar methods and techniques are required for fiber processing tasks. However, the above mentioned limitations and issues of tractograpy post-processing give a reasonable explanation why a seamless combination of fiber tracking and practically relevant post-processing still is challenging and needs to be solved.

Aim of this work

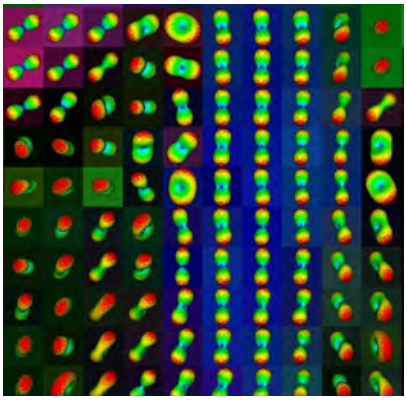
In general the pipeline of tractography and fiber processing starts at the MRI scanner, where physicists experiment with several sequences to provide high quality data for tractography algorithms. After that, methods to process the generated output are needed, including visualization and interaction with fiber structures. On the one hand, designing fiber tracking algorithms still contributes to the major challenges in this field. However, techniques to explore their output cannot be trivialized because current visualization and interaction techniques are mainly designed for specific scientific purposes. Limitations of usability, performance, visualization and interaction are analyzed and a practicable solution is proposed. This work introduces a framework for fiber processing and visualization overcoming current limitations related to this discipline. Tools



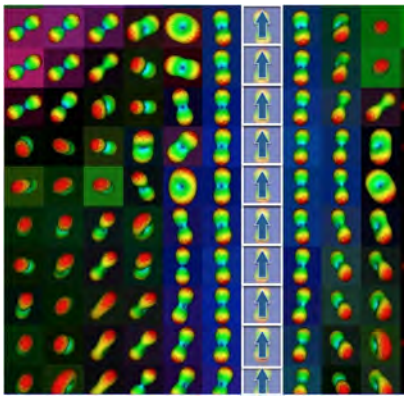
(a) Random movement of a single molecule Image taken from [27].



(b) Shapes and 3D tensor matrices. Image taken from [26].



(c) ODFs representing the displacement distribution for each voxel.



(d) Illustration of tracking a fiber based on DTI.

Figure 1.1: This figure illustrates the movement of a water molecule during diffusion where r describes the direction vector between timestamp $t1$ and $t2$ (a). ODFs are shown in (b) (c). A schematic illustration of DTI based fiber reconstruction is shown in (d).

for intuitive and efficient interaction with fiber structures are introduced, empowering end-users to accomplish common tasks of fiber processing techniques and features which are adopted from medical image processing in radiology. Additionally, the potential increase of productivity and efficiency opens ways to advanced applications in this area, raising scientific as well as clinical interest in porting tractography into practice. This work is implemented into the open-source software platform *Medical Imaging Interaction Toolkit* (MITK) [37] which is developed and maintained by the division of *Medical and Biological Informatics* (MBI) at the *German Cancer Research Center* (DKFZ). It specializes in medical imaging and provides a framework supporting the development of clinical applications. Features such as multiple render windows for 2D and 3D visualization and interaction as well as a data management to store and manage medical data are provided. Two tractography algorithms are incorporated into MITK. A stochastic fiber tacking algorithm proposed by Ngo [38], as well as a cutting edge global approach, namely

Gibbs tracking proposed by Reisert et.al. [35], are provided by the diffusion imaging module in MITK. Together with clinicians and radiologists from clinical settings, the diffusion imaging application in MITK undergoes continuous development and improvements towards bringing fiber tracking closer to clinical practice.

Overview of this document

In chapter 2 current frameworks and toolkits for fiber visualization and processing are analyzed. Based on this, requirements for practical usage in radiology are defined. Chapter 3 describes the methodology how the fiber framework was implemented. The results of this work are in chapter 4. Chapter 5 discusses the results based on the requirements defined in chapter 2. A conclusion is given in chapter 6.

CHAPTER 2

State of the Art

This part describes the software environment in relation to this thesis and gives an overview about related solutions for fiber visualization and fiber interaction. An investigation and evaluation of the major established applications are mentioned here.

2.1 Related work

In the last couple of years several fiber tracking algorithms have been developed and techniques for fiber visualization and interaction are available in several solutions for tractography. Major applications for tractography and fiber visualization are described in this sections.

3D Slicer: 3D Slicer [39] is an open source software package specializing in medical image analysis and visualization. The application is written in C++ and includes components of VTK, ITK and Tcl/TK (Tool Command Language / Tool Kit) [40]. Its tractography module provides a streamline fiber tracking algorithm based on the Runge-Kutta method [41].

MedINRIA: The MedINRIA [42] platform offers a collection of applications for medical image processing. The included DTI Track application provides tools for DTI analysis and fiber tracking. The implemented tractography algorithm relies on a local and probabilistic approach, also knows as *Fiber Assignment by Continuous Tracking* (FACT) [43, 44]. For fiber visualization, MedINRIA uses its own VTK based open source visualization library, vtkINRIA3D.

MeVisLab: MeVisLab [45] represents a development environment for medical image processing and visualization. The software includes well known third party libraries and technologies such as Qt, Open Inventor, ITK and VTK. The provided tractography toolkit, known as *Tensor-Tractography*, comes with two different fiber tracking algorithms. Both of them perform on DTI data and use a local tractography approach [31, 46].

Apart from the above mentioned, there exist many other applications such as MRI/DTIStudio [47, 48], Camino [49], ExploreDTI [50], TrackVis [51]. The *Oxford Centre for Functional MRI of the Brain* (FMRIB) also develops the software application *FMRIB's Diffusion Toolbox* (FDT) but is not further considered in this work, since there is no visualization of fiber structures supported. Visualization of fibers can be accomplished using *Track Density Imaging* (TDI) [52]. In contrast to the methods introduced in this work, TDI uses a different approach by using post-processing methods to achieve higher resolution of the acquired images to generate so called track-density maps which represent probability of tracts. These maps are volumetric images like MRI and *Computer Tomography* (CT) images where 2D visualization is already provided.

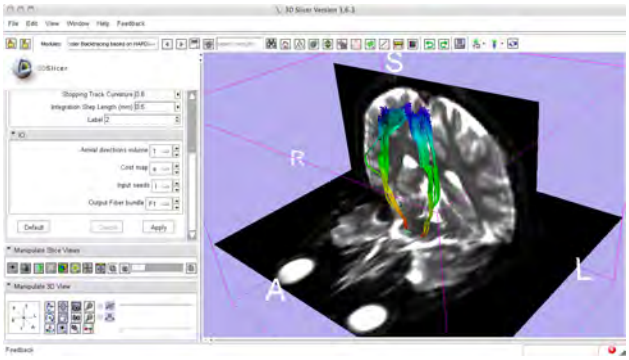
Features of current fiber processing applications

This work focuses on intuitive and practical handling of fiber structures including visualization and fiber interaction which can be used in a clinical environment. Many efforts have been invested in the visualization of fiber tracts using CPU and GPU accelerated methods for realistic 3D representation [53–55]. 3D visualization can be performed by all the above mentioned applications except Camino which only supports exporting fiber information to external applications such as Geomview [56] and Paraview [57]. However, tractography and fiber processing are mostly applied in radiology where interaction and representation in 2D are performed. So far only MeVisLab and MedINRIA can perform 2D visualization of fiber structures. MedINRIA maps fibers into 2D when the user clicks on the designated button. Fiber structures in 2D only support coloring of a single color only leading to inconsistency between 2D and 3D representation as shown in figure 2.2(a). Certain fibers can be selected by loading additional segmentation and ROI images which cannot directly be modified after importing. 3D Slicer and TrackVis provide a 3D sphere which shows all fibers crossing it. In TrackVis this sphere can also be adjusted directly across the 2D slides of an underlying DTI image. However, both solutions are not consistently representing fiber data in 2D and 3D. Unfortunately, MeVisLab could not be tested for similar parameters since no free version was available. In practice, radiologists are used to manually segmenting ROIs, e.g. circles and polygons, on 2D images with direct feedback in 3D. Such features are not provided by any of the introduced solutions. The limitations of visualization and interaction as well as missing features for practical and performant fiber processing in radiology give the backdrop for this work.

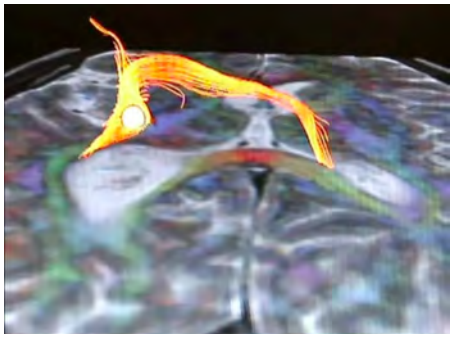
2.2 Toolkits

Insight Toolkit (ITK)

MITK uses ITK [58] primarily for digital image processing and has adopted several features and concepts such as the application framework, filter framework and the class concept into the main infrastructure of MITK. This features implementation of commonly used design patterns and fully supports multi threading and filesystem access. ITK provides the main superclass `Object` which is the superclass of all MITK classes. So called `itkSmartPointers` allow automatic memory management. ITK comes with readers and writers for many image and



(a) 3D Slicer fiber tracking application. GUI to adjust tracking parameters (left) 3D representation of fibers (right) with ROIs (green segmentation at bottom of fibers).



(b) Selecting fibers using 3D sphere. The sphere is moved by mouse interaction and shows all fibers crossing the ball.

Figure 2.1: This figure shows fiber visualization and interaction in 3D Slicer.

surface file formats and wraps external libraries such as *Grassroots DICOM* (GDCM) [59] for *Digital Imaging and Communications in Medicine* (DICOM) support, libpng [60], etc.

Visualization Toolkit (VTK)

VTK [61] specializes in several disciplines of scientific visualization and also in the medical field. Advanced rendering algorithms are available for different backends, supporting software-only rendering and graphics-card-accelerated rendering via the *Application Programming Interface* (API) in OpenGL [62]. MITK uses mainly VTK for 3D and 2D render windows and provides following main features:

- Visualization of images and surfaces which contain sets of various geometric primitives such as points, lines and triangles.
- Image filter algorithms which are used to preprocess datasets before rendering, e.g. transfer functions to apply a level window.
- Camera interaction in 2D and 3D render windows including panning, rotating and zooming within the scene.

Qt

Qt [63] was developed to provide a platform independent *Graphical User Interface* (GUI) for PCs. By now, it also covers support for mobile platforms like Nokia and Android devices. It has grown to a full application environment for C++ providing its own GUI editor. A central feature of Qt is the signal and slot concept which manages events and actions, e.g. calling a specific method when a button is clicked. Additionally, Qt provides a rich library of GUI elements as well as methods for multithreading and tools for visualization from which MITK benefits.

MITK and other libraries

MITK is an open source, non restrictive licensed software toolkit developed and hosted by MBI at DKFZ, Heidelberg. The MITK platform facilitates the integration and interaction of software from different disciplines of the medical image processing field. A repertoire of selected toolkits and mechanisms provide a powerful environment to support rapid prototyping, as well as developing efficient applications. The major purpose of MITK is to reduce efforts in constructing specifically tailored and interactive applications for medical image analysis. It combines ITK, VTK and the *Common Toolkit* (CTK) [64] with an application framework. A strong hierarchy defines the architecture of MITK with one important level being the layer of modules. A module in MITK is defined as an accumulation of related tasks within a discipline. This contains data structures, algorithms, classes for loading, rendering and interacting with medical images. For example, the diffusion imaging module includes algorithms to process DWI data, perform tractography, *Tract Based Spatial Statistics* (TBSS), etc. The interface to users is realized by so called bundles, also known as plugins. They provide all necessary methods to enable user interaction with the framework, e.g. executing fiber tracking on the selected data and adjusting parameters for DTI as well as a GUI using the Qt framework. One of the major bundles is the datamanager which provides access to loaded data and file system. Figure 2.2 shows the MITK application with the datamanager and bundles for diffusion imaging. In terms of modularity, MITK uses the CTK plugin framework which provides a C++ OSGi™ [65] implementation following the principle of lazy loading. MITK is in use by several groups in universities, research institutes, and also in commercially available products. MITK offers the following main functionalities:

- Interaction with common medical image formats including DICOM.
- Visualization of volumetric images in 2D and 3D.
- Advanced medical image processing algorithms on datasets, e.g. fully automatic segmentation and registration of anatomical structures.
- Interfaces for sensor input devices e.g. optical and electromagnetic tracking devices which are used in intra-operative navigational applications.

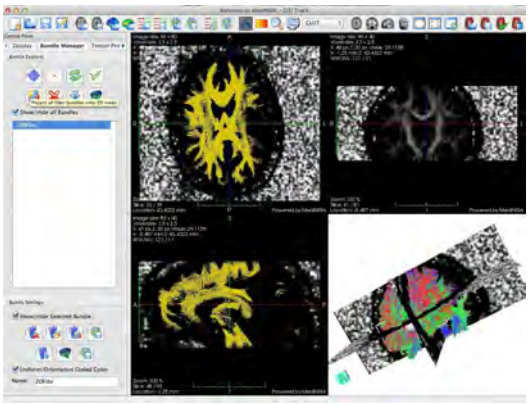
MITK is based on two major toolkits which are developed by Kitware. ITK provides image processing algorithms specializing in medical segmentation and registration, whereas VTK focuses on visualizing medical datasets of different kind, e.g. volumetric images, slices, meshes and surfaces. MITK extends and combines these algorithms for image processing and visualization for analysis in a medical context. The concept of MITK allows a transparent interchange between ITK and VTK. A MITK image for instance can be used for both processing via ITK and displaying via VTK. Additionally, a large repertoire of mechanisms to interact with medical datasets is provided.

2.3 Requirements for fiber processing in practise

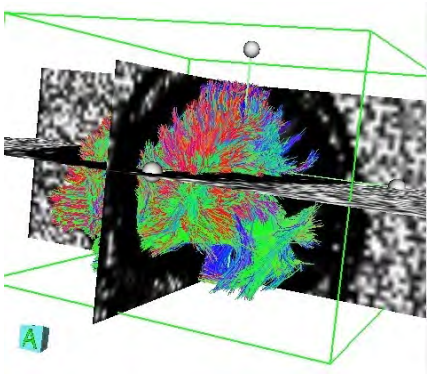
Since medical image processing is a domain of radiology, it seems to be reasonable that their visualization and interaction techniques are adapted to fiber processing. This supports the integration of fiber processing into workflows of radiology. Main features and requirements are summarized in table 2.1.

Requirements	Description
Consistent 2D and 3D visualization	Fiber structures in 3D shall be mapped accordingly to 2D. Appearance of color and shape need to be equivalent.
Accurate 2D and 3D interaction	When interacting with 2D slices, an immediate feedback shall be provided in 3D. For example, if a ROI is drawn onto a 2D slice (transversal view), this ROI shall also be visible in 3D. This is an intuitive method to monitor adjustments of segmentations in 2D and 3D.
Practical performance	Smooth and judder-free interaction with fiber structures as well as accomplishing tasks in minimal time.
Handling of large data sets	No noticeable performance issues when dealing with large fiber sets.
Features	*Multi Planar Reconstruction (MPR)*, adjusting slice thickness in 2D (thick slab), visualizing orientation based and functional parameters, e.g. orientation based coloring of fibers and coloring fibers according to scalar values. Drawing ROIs and combining them as well as extracting, joining and subtracting certain fibers and fiber structures.
Intuitive user interface	An appropriate GUI and self explaining GUI elements shall be provided.

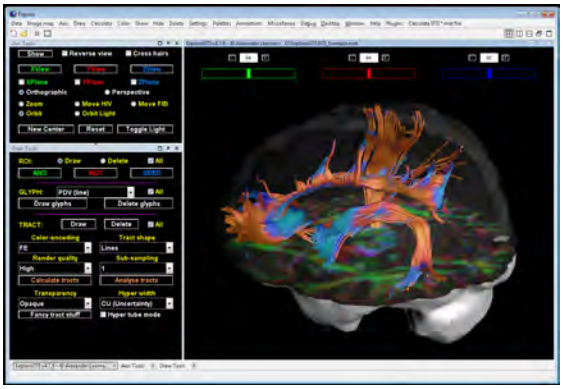
Table 2.1: Requirements for practicable fiber processing in radiology based on patterns of medical image processing.



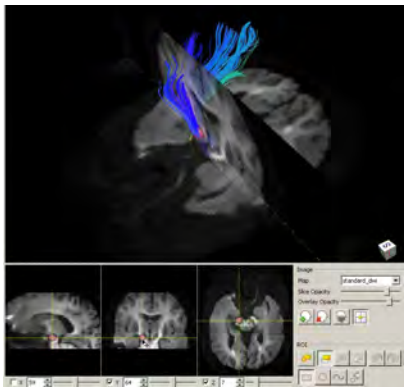
(a) MedINRIA application



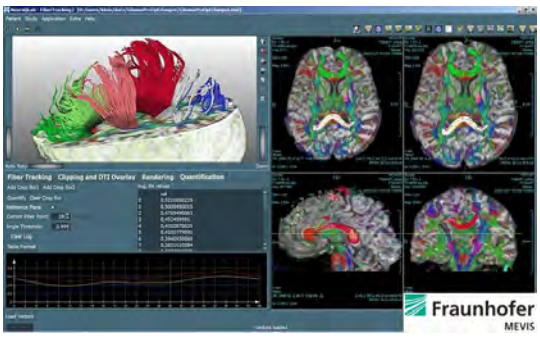
(b) Interaction cube of (a) to select fibers



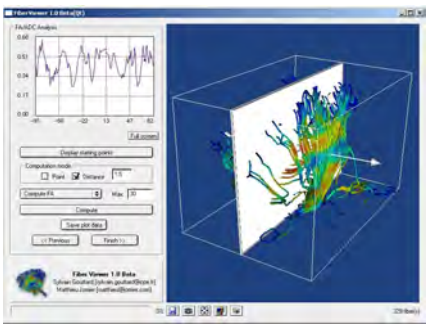
(c) ExploreDTI application



(d) TrackVis application



(e) MeVisLab application



(f) Fiberviewer application

Figure 2.2: This figure shows current applications for fiber visualization and processing. Only MedINRIA (a) and MeVisLab (e) support 2D visualization of fibers.

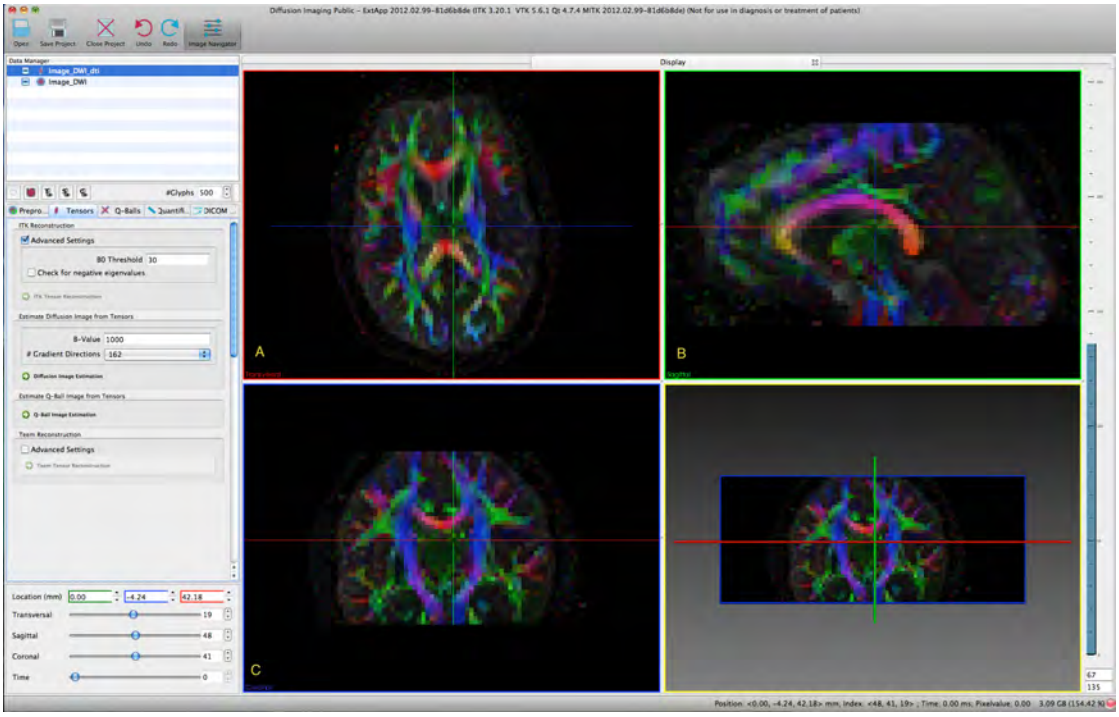


Figure 2.3: MITK application; (top left) datamanager hosting DWI and DTI image; (middle left) diffusion imaging plugin with several tabs; (left bottom) slice navigator; (center) multi window view including 2D transversal (A), sagittal (B), coronal (C) and 3D representation of DTI (color image); (right) window/level modification bar

CHAPTER 3

Methods

In this chapter methods and approaches for fiber visualization and processing based on the requirements defined in the previous chapter are introduced. These requirements can be classified into groups of the application level, providing the main software infrastructure for development, data structures managing fiber data and techniques to visualize and process fibers. Two events, namely the DTI-tractography challenge in 2011 at the *International Conference on Medical Image Computing and Computer Assisted Intervention* (MICCAI) [66] and illustrating all fiber structures of the human brain for the book *Diffusion Tensor Imaging: Introduction and Atlas* [67] published by Springer are used to evaluate the practical usage of the outcome of this work. For authoring the book, fiber processing and visualization needed to be practicable for the authors whereas at MICCAI tractography and fiber processing tasks needed to be accomplished on-site with time limits.

3.1 Software design and application-level support

Since MITK is based on ITK, VTK and CTK, the process of fiber visualization and interaction could draw support from previously existing mechanisms. Two fiber tracking algorithms were included, which were implemented as ITK filters. The stochastic tractography was realized by Ngo [38] which uses a local and probabilistic approach whereas Gibbs fiber tracking takes a global and probabilistic approach [35]. These filters were either be executed by a program or explicitly initiated by e.g. mouse interaction of the user. According GUIs were implemented using Qt. One of the aims of this work was to develop a data structure which provided methods for fiber processing and fiber interaction. Initially, it was unclear which toolkit provided the optimal data type and processing mechanisms for fiber structures. Since MITK is based on ITK and VTK, data types for both approaches were implemented to develop the optimal solution which fits best into the workflow of fiber processing tasks. Accordingly, two data types were introduced, the ITK based `mitkFiberBundle` (FB) and the VTK based `mitkFiberBundleX` (FBX). Both fiber bundle data types were incorporated into the MITK factory mechanism which

provided extensions to support loading, saving, and drag and drop of fiber structures. FB used an XML based output format *.fib and FBX the widely used VTK polydata format (ASCII) *.vtk. To recognize the stored `vtkPolyData` file as FBX, the file ending was renamed to *.vfib in the exporting process. A more detailed description of the data types is available in the following sections. Figure 3.1 gives an illustration of how components for fiber visualization and interaction are organized in MITK.

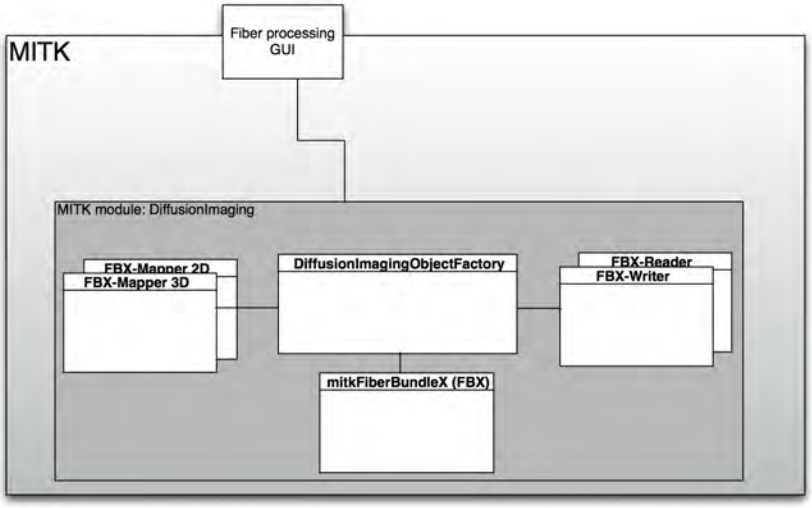


Figure 3.1: This figure illustrates how FBX interacts with other essential components for fiber visualization and interaction. FBX as well as the corresponding mapping, import and export mechanisms are part of the diffusion imaging module in MITK. GUI plugins represent an interface to interact with these components.

ITK based fiber bundle

ITK provided a data type namely `itkDTITubeSpatialObject` which implements methods for managing single fibers. This class was also capable of tubal structures, hence a radius can be defined for each point of the fiber. Its API provided a method, `AddSpatialObject()`, which allows combining several single fibers. In terms of processing fibers, e.g. extracting points at a certain position to identify which points are crossing a defined ROI, no adequate methods were provided by `itkDTITubeSpatialObject`. Therefore, an appropriate functionality for this task was realized in the `mitkFiberBundle` class. The ROI was passed to the method `extractFibersByPF()` which calculated the intersection points of the ROI and each fiber. Planar figures (PF) represented ROIs for fiber processing. Since a fiber consists of a point set, a ROI could either cross a point directly or is placed in between two adjacent points. Geometric calculations for this purpose were implemented in methods `calculateCrossingPoint()` and `checkForGap()`. All points were stored as index coordinates of the input image where fiber tracking was executed. Additional fiber process-

ing mechanisms are described in sections 3.2 and 3.3. `mitkFiberBundle` was registered to the `mitkDiffusionImagingObjectFactory` class which manages all diffusion imaging related data types and connects them to the visualization pipeline as well as to the appropriate reading and writing mechanisms for fiber import and export. An overview of the FB data type can be found in figure 3.2(b) showing the class diagram.

Initializing mitkFiberBundle

Several interfaces were available to transfer tractography to the FB data type. The stochastic tractography algorithm provided an `itkVectorContainer` containing `itkSlowPolyLine-ParametricPath` as output. After initializing a new FB object, this output could be passed to the method `additkStochTractContainer(output)`. In comparison, Gibbs fiber tracking used `itkPoints` stored in an `itkVectorContainer` as output which could then be passed to the `addTractContainer(output)` interface. An example is shown in the following:

```
        mitk::FiberBundle::Pointer fb = mitk::FiberBundle::New();
fb->additkStochTractContainer( stochasticOutput ); fb->initFiberGroup();
```

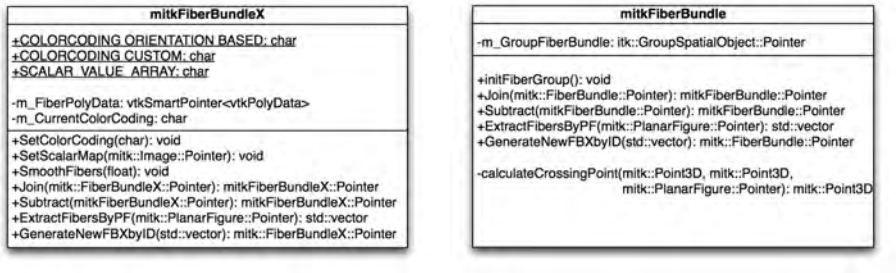
VTK based fiber bundle

Alternatively, `mitkFiberBundleX` was based on VTK which places the fiber data structure closer to the visualization side in MITK. VTK itself offered no explicit data structure for fiber tracking results but accomplished a more generic solution known as `vtkPolyData`. This data type was able to host vertices, lines, polygons, and triangle strips. Additional values of tractography could be added to each single fiber. These values then could be used for coloring fibers. Fibers itself were represented as `vtkPolyLine` containing a point set. Points were stored as world coordinates to minimize additional calculations in the workflow between tractography output, fiber processing and fiber visualization. FBX provided methods to calculate several color codings and since the fiber structures were stored in VTK polydata format, processing of fibers, i.e. polydata, was relatively straight forward due to the rich repertoire of filters. By default the orientation based color coding was computed. This strategy kept the FBX compact and provided a better overview. For visualizing and interacting with fiber structures, no additional computations of fiber structures were required since FBX hosted all necessary formats for rendering and efficient fiber processing. Any tractography application could implement FBX as an output format if VTK was available. Methods to add additional information to fibers such as fractional anisotropy (FA) maps, scalar values and color information as well as methods to extract, join and subtract fibers were successfully implemented. All fiber processing mechanisms are described in sections 3.2 and 3.3. Another advantage was the VTK polydata output format, which allows importing tractography results into other VTK supporting platforms. FBX was also registered to the `mitkDiffusionImagingObjectFactory` class where corresponding visualization and import/export mechanisms were set. Figure 3.2(a) shows the UML of `mitkFiberBundleX` in detail.

Initializing FiberBundleX

Since FBX is VTK based, all tractography outputs needed to be converted into `vtkPolyData` once. A valid polydata consists of a point set, `vtkPoints` and lines, `vtkPolyLine`. The polyline data structure hosted all related point IDs from the point set. A FBX object was initialized by passing a valid `vtkPolyData` to the constructor. An example is shown below:

```
mitk::FiberBundleX::Pointer fbx =
mitk::FiberBundleX::New( outputPolyData )
```



(a) UML class diagram of mitkFiberBundleX (b) UML class diagram of mitkFiberBundle

Figure 3.2: This figure shows only important variables and methods for fiber processing.

Qt based views and perspectives

In MITK, GUIs were designed using the Qt software framework. Such interfaces were named views or bundles, which represent a plugin including methods to call and execute computational tasks. Multiple views could be accumulated to produce a so called perspective. Tools for fiber processing were available in the *Fiber Processing view* which was present in the *Tractography* perspective. This view supported multithreading to separate graphical interaction and computational tasks. Therefore, the GUI was not blocked during calculations required for fiber processing. Design and Qt specific interaction patterns relied on the publicly available MITK style guide [68].

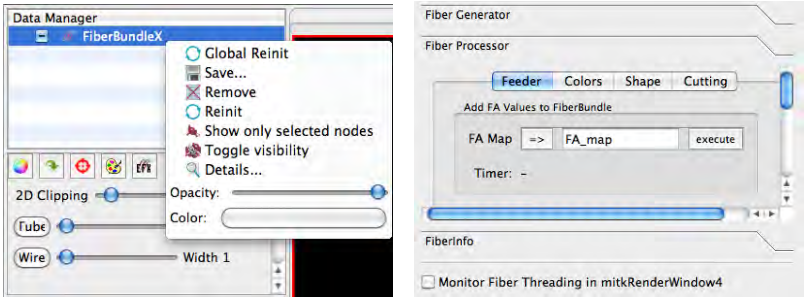
3.2 Visualization of fiber tracts

Current tractography approaches can compute fiber structures of the human brain containing more than hundred thousand fibers. Therefore, efficient techniques to visualize this enormous amount of information in 2D and 3D space are necessary to provide adequate orientation and meaningful handling of fibers. Figure 3.4 shows the activity diagram for visualizing fibers. Judder free navigation through fiber structures was required to guarantee convenient user interaction. Also, accurate representation of fibers within the current context was mandatory e.g. if opacity of fibers is set to 50%, underlying pixel from images such as DWI, CT, etc., should be

visible accordingly. MITK uses VTK render windows for drawing 3D and 2D data. VTK provided a generic data structure `vtkPolyData` which hosted graphical primitives and extensions of them. To render an object in a VTK render window, three major steps were required:

- A polydata object was passed to the `vtkMapper` object which transformed the object for correct geometric representation in space.
- The mapper then was passed to the `vtkActor` which provided several properties for rendering, e.g. color and representation modes.
- In the final step, the actor was passed to the `vtkRenderer` which actually painted the requested object into the render window.

The visualization of both, FB and FBX were managed by `mitkFiberBundleMapper2D/3D` and `mitkFiberBundleXMapper2D/3D` respectively. These formated the fiber structures and passed them to the MITK rendering pipeline. Mappers of the ITK based fiber structure needed to transform the `itkDTITubeSpatialObject` into `vtkPolyData`, and point sets were also translated from the index coordinates to world coordinates. In contrast, FBX already possessed the requested format for the rendering process. However, for both data types the finalized `vtkPolyData` was passed on to the `vtkPolyDataMapper`, which mapped fiber data to graphics primitives. Options for fiber representation such as color coding, opacity and shading effects were set with the GUI by the user. According to this information, mappers then activated the requested representation options. Figure 3.3(a) shows a FBX data node in MITK with its options for visualization. The workflow from calculating fiber tracts to fiber visualization is illustrated in Figure 3.5, where both ITK and VTK based data structures are compared.



(a) Options to modify and edit properties of the fiber set.

(b) Add FA map to selected fiber set.

Figure 3.3: GUI elements for (a) modifying and (b) adding additional values to any selected fiber bundle.

3D visualization

In MITK, visualizing fibers in 3D did not require much effort as fiber data were three dimensional and VTK implemented a complete 3D rendering pipeline. However, before passing fiber data to the rendering mechanism, the requested representation options for visualizing

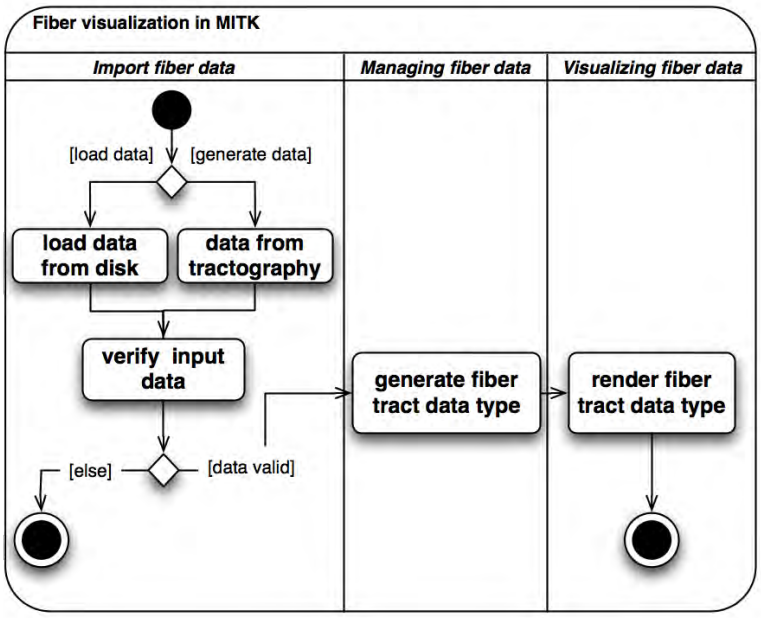


Figure 3.4: This figure illustrates an activity diagram starting from importing and generating tractography data until visualizing fibers.

fiber structures needed to be applied. In general, mappers finalized fiber data for the rendering pipeline. MITK mappers were initialized by the MITK factory mechanism. If a FB or FBX data object was activated `mitkDiffusionImagingObjectFactory` initializes the according mappers. Representation modes and parameters for color, opacity etc. were passed to the VTK rendering pipeline by applying parameters to `vtkOpenGLPolyDataMapper` and `vtkOpenGLActor`. The actor was picked up from the MITK rendering mechanism, which was visualized in the 3D render window. In the following two sub-sections 3D mappers for FB and FBX are described.

mitkFiberBundleMapper3D

Within the whole fiber processing workflow this mapper defined the visualization part of fiber structures. When a `mitkFiberBundle` object was passed on to this mapper, all points from the `itkDTITubeSpatialObject` were transformed into world coordinates. In MITK, the `mitkGeometry3D` class provided a method for this purpose. Each single point of the fiber was passed to the method `IndexToWorld(point3D)` and returned the appropriate world coordinates. Based on the transformed point set, `vtkPolyLines` were generated. In the next step these line were smoothed by the `vtkKochanekSpline` filter. Based on the smoothed point set, individual colors were applied to each point depending on its position in space. If a original fiber structure also hosted additional values for each point, such as FA or *Generalized Fractional Anisotropy* (GFA), these values were then applied to each point. In the next step

explicit properties were set defining the resulting representation modes. These options included color coding and several representation modes such as point, line or tube based representation.

mitkFiberBundleXMapper3D

Based on settings of the FBX object, properties of color, opacity and representation were added to the rendering pipeline. The implementation of the `mitkFiberBundleXMapper3D` supported the initialization of several 3D render windows. In other words, initializing e.g. two 3D render windows simultaneously where one shows FA based color coding and the other draws fiber structures from a different angle were supported. This option was managed by the `mitkLocalDataStorage`, which hosted the required data for each 3D render window.

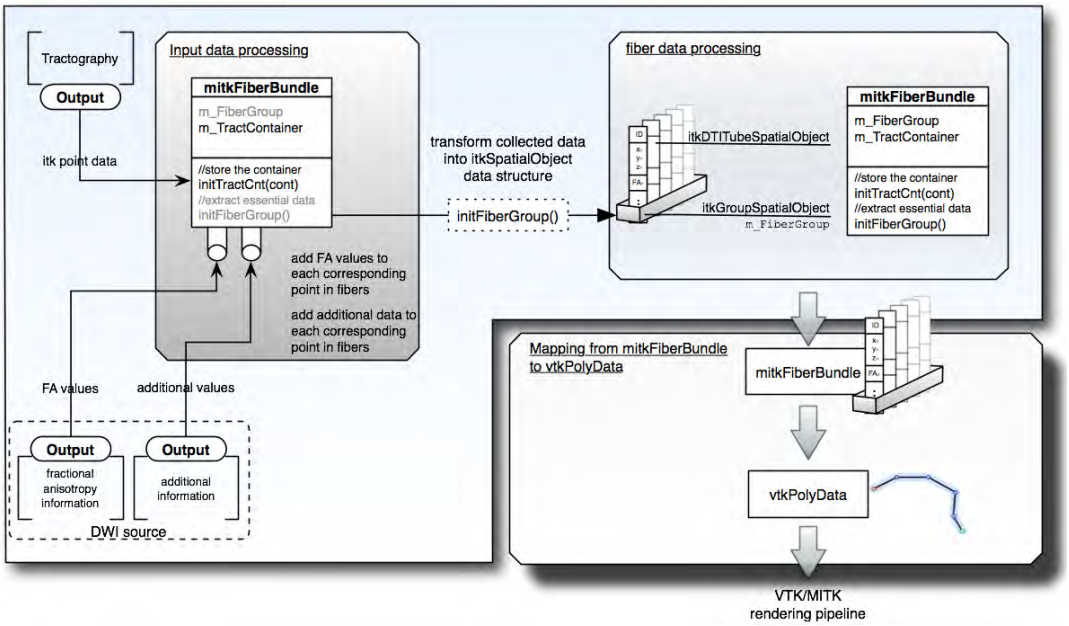
2D visualization

In general, medical images such as CT or MRI consist of a set of 2D slices. Hence, transferring them into 3D space requires computational effort. In comparison, fiber structures are defined as 3D structures, therefore a 2D representation is more challenging since 3D data needs to be projected to 2D. This can either be realized by slicing through the given 3D data and returning the intersecting data or by limiting the depth of the view to a slice. In radiology, state of the art medical applications provide 3D representation as well as 2D representation of normal images in horizontal, sagittal and coronal views. It is common to use crosshair navigation across 2D images, whereas in 3D the crosshair is transformed into three planes. The center of the crosshair indicates the point of interest and its position indicates the slices along horizontal, sagittal and coronal orientation. Methods for slicing fiber structures were introduced into `mitkFiberBundleMapper2D` and `FiberBundleXMapper2D`.

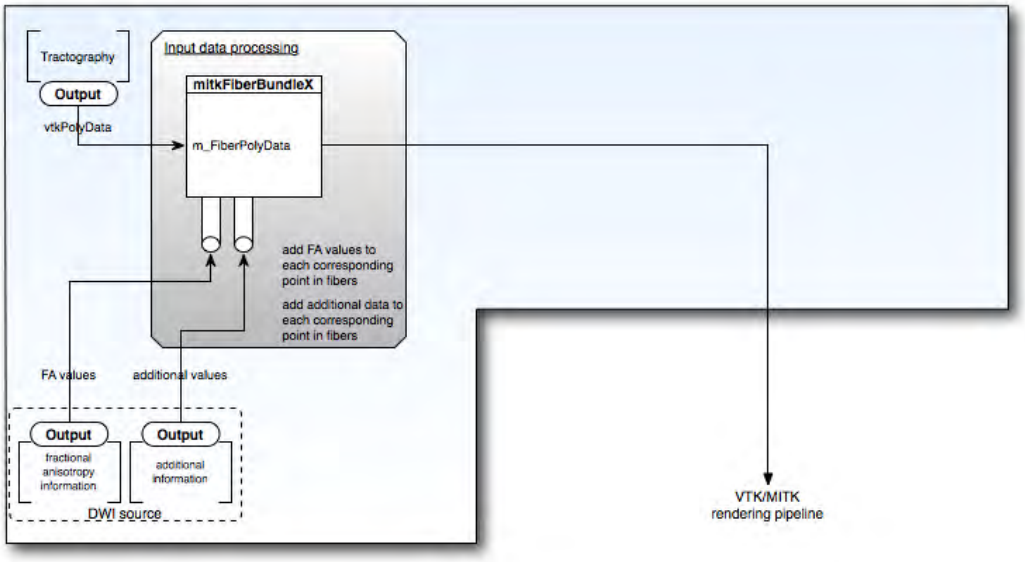
mitkFiberBundleMapper2D

This class prepared fiber structures for 2D visualization. In general, MITK used three render widows for 2D visualization to draw images in coronal, sagittal and horizontal view. Therefore fibers needed to be sliced accordingly to the plane of view. The position of the planes was defined by the MITK crosshair. When representing different slices of the same data set, the initialized mapper must be able to manage individual data for each render window. This was performed by `mitkLocalDataStorage` class which was reimplemented in `mitkFiberBundle-Mapper2D`. In practice, applications in radiology allow adjusting the thickness of a slice, so called thick slab. This feature was performed by this mapper which projects multiple slices of an image on a single plane. The following two sub-sections introduce two approaches for visualizing fibers in 2D.

vtkCutter filter The `vtkCutter` provided methods to cut fibers at any position and returned all intersecting points. Required inputs were `vtkPolyData` and a `vtkPlane` for slicing. A code example is listed below. To realize thick slab, several cuts along the plane normal were performed and merged together. The output was passed on to the `vtkOpenGLPolyDataMapper` and `vtkOpenGLActor`, which were managed by the `mitkLocalDataStorage` object.



(a) Workflow of FB. After adding all necessary tractography data and additional values, the data structure is transformed to the standardized ITK data types and to VTK polydata.



(b) Workflow of FBX. After adding all necessary tractography data and additional values, the data structure is directly passed to the rendering pipeline.

Figure 3.5: Schematic illustration of the workflow using (a) FB and (b) FBX data structures.

```
          //create plane
 vtkPlane* plane = vtkPlane::New();
   plane->SetOrigin(0.0, 0.0, 0.0);
   plane->SetNormal(0.0, 0.0, 1.0);

    // cutting the fiber structure
vtkCutter* cutter = vtkCutter::New();
   cutter->SetInput(fiberPolyData);
    cutter->SetCutFunction(plane);
         cutter->Update();
```

Adjusting clipping range of vtkCamera Another approach to visualize fibers in 2D is to modify the clipping range of the `vtkCamera` object in the rendered scene. To support thick slab, the clipping range of `vtkCamera` could be adjusted. Only elements, i.e. fibers between the set range along the normal direction of the camera were drawn. Pixels outside the range were not rendered. This technique manipulated the shader mechanism of the VTK rendering pipeline, which is performed by the GPU. An example code listed below shows how the clipping range can be set to draw only fibers between the range of 40 and 41. However, there are some limitations using this technique especially when several images are added to the rendering scene which is mandatory in the case of MITK. Due to an open bug in VTK (id 7823) which was submitted in 2008, adjusting the clipping range to a small value affects the visualization of images. Imported 2D images are not visible anymore.

```
renderer->GetVtkCamera()->SetClippingRange(40.0, 41.0)
```

mitkFiberBundleXMapper2D

This mapper was initialized when a `mitkFiberBundleX` data structure was passed on to the MITK factory mechanism. Again, the `mitkLocalDataStorage` hosted individual mappers and actors for polydata to provide the requested data for each render window. This mapper adopted a GPU based approach for rendering fibers in 2D. In VTK, custom shaders could be added to a `vtkOpenGLActor` object, which was executed during the rendering process. For each view, the shader decided as to which pixel should be passed to the render window and which pixel is dropped. Furthermore, all informations available for the 3D representation, for example, coloring of fibers was also available for the 2D representation, since 2D and 3D mappers are representing the same fiber set. Alternatively, if no shader was set, the fiber structure in 2D was drawn identical to the 3D representation. Only the viewing angles for horizontal, sagittal and horizontal view varied. In detail, the shader was written in *OpenGL Shading Language* (GLSL) and modified the fragment shading process in the shading pipeline. Inputs including the current viewing plane position and plane normal were passed to the shader defining the requested slice. Also parameters for slice thickness could be set by the user. One also has the option to enable fading of fibers, thus coloring the end of the fibers dark or transparent. This mode actually brought an illusion of depth into 2D.

```
//initialize mapper and add the fiber structure
vtkMapper* mapper = vtkMapper::New();
mapper->SetInput( FiberBundleX->GetFiberPolyData());

//initialize actor and add mapper and shader
vtkActor* actor = vtkActor::New(); actor->SetMapper(mapper);
actor->GetProperty()->LoadMaterial(myFiberShader.xml);
actor->GetProperty()->ShadingOn();
```

Coloring of fibers

An intuitive representation of fibers consists of both, shape and color. For example, coloring fibers according to their orientation support intuitive navigation through complex fiber structures. Also anatomical structures can be colored explicitly to illustrate certain areas of the human brain. In general, appropriate color codings support better understanding of the context one is dealing with. Colors can indicate physical orientation in space and help to identify regions of interest. Besides these, illustrating functional information is often performed with no direct reference to tractography results. An option to intuitively connect functional information with fiber structures can be achieved by coloring fibers according to functional data provided by DWI.

Scalar based coloring

Tractography and diffusion data provide additional information such as FA, GFA and orientation of tensors, just to mention a few. FA maps are commonly used in the research field of DTI, but as a matter of fact, these maps are represented as 2D images. Instead of loading maps and fibers separately, it would be more efficient if functional data is stored together with fiber structures. Fibers of high and low FA can be identified easily since the coloring indicates the according values. As an alternative, opacity levels of fibers can also be used to visualize functional data. Figure 3.7(a) shows the GUI elements for coloring fiber structures. Methods to generate and apply the desired representation were implemented in `mitkFiberBundleMapper3D` and `mitkFiberBundleX`. Since FBX already used VTK data, coloring could be directly applied to the data structure, whereas for the FB data structure, this data could only be performed in the mapper. Since both approaches passed VTK data on to the rendering pipeline, the following two sections apply for both of them. They only differ in the class where the actual calculation of colors s performed. VTK supports RGBA coloring so each point in a fiber could be set with one or more color values. RGBA values for each point were stored in a separate `vtkUnsignedCharArray`. To activate this array for the rendering process, only the name of the array needed to be passed on to the `vtkPolyDataMapper` hosting the fiber structure (see following section for an example code for orientation based coloring which is based on the same principle). To apply scalar values for color representation, an appropriate array, for example `vtkDoubleArray` containing values for each point needed to be added to the fiber structure. When applying a scalar array for coloring, additionally a lookup table needed to be set which translated the scalar value into RGBA. This was achieved by the `vtkLookUpTable` object.

FA based coloring

For the `mitkFiberBundle` data structure, FA values were already set when initializing an instance of `itkDTITubeSpatialObjects`, but such values were randomly generated to test if FA based color coding and applying FA values as opacity levels delivered the desired outcome. Since the FBX approach was VTK based and `vtkPolyLines` do not offer explicit methods for adding FA values, FA maps could be passed to the fibers and each point stored its corresponding value in the `FA_VALUE_ARRAY`. Implementation of this method `setFAMap(FAmap)` could also be used as a template for other values. Figure 3.3(b) shows the GUI for adding a FA map to the fiber structure. `vtkPolyData` offered convenient options to apply scalars such as FA values as color values. No extra computations were necessary; only a `vtkLookUpTable` object needed to be added to the rendering mechanism. The code is listed in the following.

```
vtkLookupTable* m_lut = vtkLookupTable::New();
                m_lut->Build();
m_FiberVtkPolyDataMapper->SetLookupTable(m_lut);
  m_FiberVtkPolyDataMapper->SelectColorArray(
      mitk::FiberBundleX::FA_VALUE_ARRAY )
```

Another way of representing FA values was using them as opacity levels. Parts of fibers with high FA showed 100% opacity whereas sections with low FA were drawn with less opacity. Since FA values were normalized, they can directly be applied to the alpha part in RGBA tupels. The example below shows how the alpha value is replaced by the corresponding FA value.

```
double faValue = FAValArray->GetValue( pointID );
           faValue = faValue * 255.0;
ColorArray->SetComponent( pointID, 3, faValue );
```

Orientation based coloring

This color coding illustrates the physical orientation of fiber segments in 3D space. Depending on the relative position of adjacent points, the connecting line in between was colored individually. In between two adjacent points which are located along the x-axis, the connecting line was colored in red. Similarly, points along the y- and z-axis were colored in green and blue, respectively. Segments of multiple directions were colored proportionally. A closer look at how the colors for the points were calculated is illustrated in figure 3.6. To pass orientation based color coding on to the rendering mechanism, properties for this color array needed to be set in the *vtkPolyDataMapper*. The command for this procedure is listed below.

```
  m_FiberVtkPolyDataMapper->SelectColorArray(
mitk::FiberBundleX::COLORCODING_ORIENTATION_BASED )
```

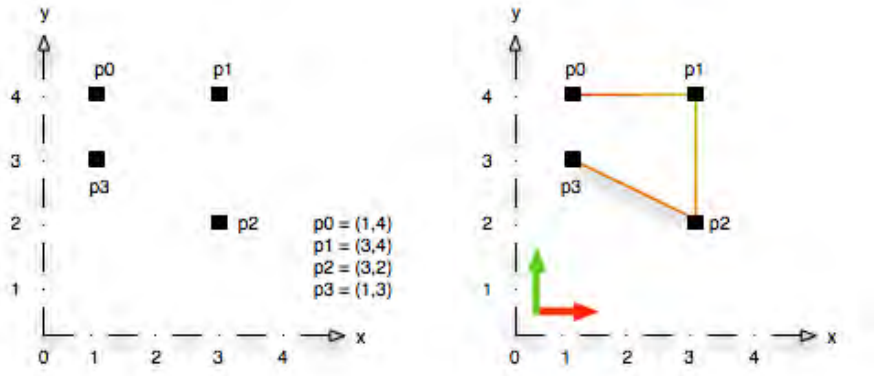


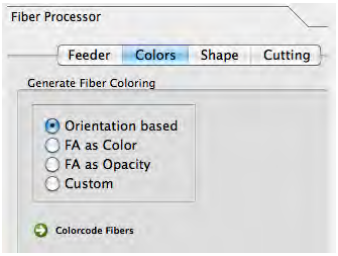
Figure 3.6: This figure illustrates how orientation based color coding is generated. This fiber is defined by 4 points and the arrows in the lower left corner indicate the color corresponding to the orientation.

Smoothening of fibers

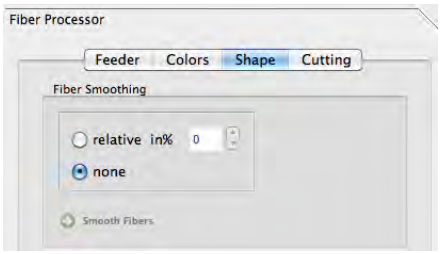
Fiber structures in MITK were also provided with an option for fiber smoothening. This feature was implemented to generate smooth structures if a fiber tracking output contains unrealistic sharp edges. Therefore this option was for visualization purpose only and needed to be applied explicitly by the user. However, the smoothed fibers replaced the original data therefore this feature could also be interpreted as fiber processing, but its main purpose was for visualization. There exist many algorithms for interpolating and smoothening point based lines. Some approaches, e.g. B-Splines are calculated without including the original points in the resulting line, which leads to an inaccurate representation of fiber structures, since the fiber points actually represent the tract. VTK provided the filter `vtkKochanekSpline` generating smooth structures. Methods tuning the interpolation resolution were provided by the `vtkParametricFunctionSource` object, which generated interpolated fibers, meaning that the original point set from tractography was extended by the interpolated ones. A code example for fiber interpolation with a resolution value of 100 is listed below. This means that the new fiber is sampled to 100 points. Interpolation of the ITK and VTK based fiber structure was performed in the `mitkFiberBundleMapper3D` and `FiberBundleX` objects, respectively. The GUI elements for fiber smoothing are shown in figure 3.7(b).

```
        //initialize splines for x, y and z direction
    vtkKochanekSpline* xSpline = vtkKochanekSpline::New();
    vtkKochanekSpline* ySpline = vtkKochanekSpline::New();
    vtkKochanekSpline* zSpline = vtkKochanekSpline::New();

        //prepare structures for interpolation
vtkParametricSpline* spline = vtkParametricSpline::New();
                spline->SetXSpline(xSpline);
```

```
        spline->SetYSpline(ySpline);
        spline->SetZSpline(zSpline);
    spline->SetPoints(pnts_original_fiber);

//set interpolation resolution and generate smoothed data
    vtkParametricFunctionSource* functionSource =
        vtkParametricFunctionSource::New();
    functionSource->SetParametricFunction(spline);
        functionSource->SetUResolution(100);
        functionSource->SetVResolution(100);
        functionSource->SetWResolution(100);
            functionSource->Update();
```



(a) Options for coloring fibers.



(b) Interpolation resolution for fiber smoothing.

Figure 3.7: This figure shows GUI elements to modify (a) colors and (b) shape of fibers.

3.3 Fiber interaction and processing

In addition to fiber visualization, the exploration of fiber structures requires mechanisms to dynamically interact with fiber data. A practicable fiber processing application requires intuitive and high-performant interaction mechanisms. Therefore flexible navigation through fiber data and a clear organization of processed data support to keep the focus on the given task and to achieve the desired result efficiently. MITK comes equipped with interaction mechanisms including zooming, panning and rotating. But fiber specific tasks such as extracting, merging and subtracting needs to be developed explicitly. Figure 3.8 shows the activity diagram of fiber processing including visualization and fiber interaction.

Extracting fibers using ROIs

A common way to extract fibers is by manually setting a ROI which is intuitive and allows precise placement. ROIs, i.e. polygon shapes, are usually set in 2D views and support interactive adjustment of size and shape. In general, shapes are not only visualized in 2D views but also in 3D space to achieve consistency and better overview of the context. MITK provided a set

of ROIs, namely `mitkPlanarFigure`, including circle and polygon based figures. Planar figures were 2D shapes allowing interactive adjustments using control points after placing them. Mechanisms to apply planar figures and use them as ROIs for selecting and extracting fibers were developed. Figure 3.10(b) shows the GUI elements for setting ROIs and figure 3.10(e) shows an ROI in the transversal 2D view.

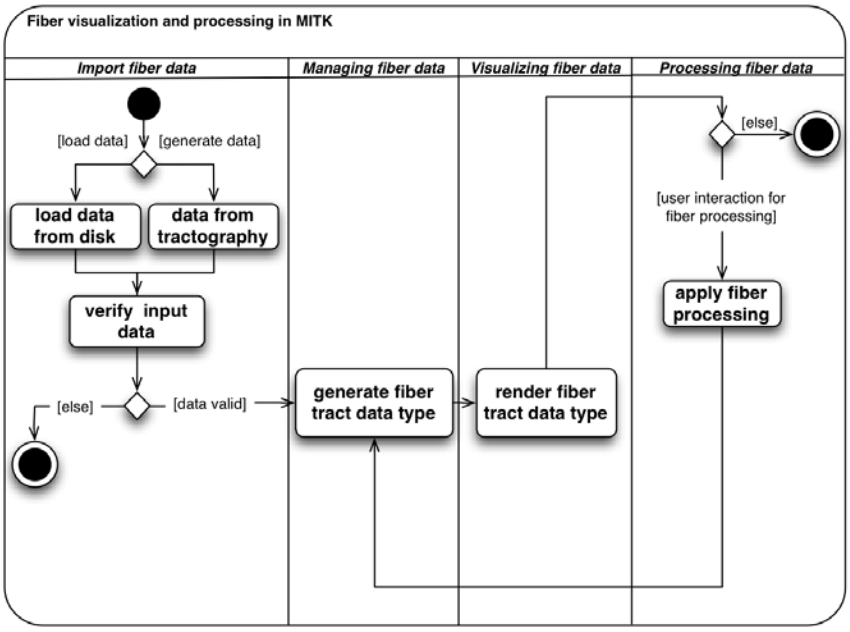


Figure 3.8: This figure illustrates an activity diagram starting from importing and generating tractography data until visualizing and processing fibers.

Fiber extraction for FiberBundle

For the ITK based approach, a custom implementation to detect fibers inside an ROI was developed. Unfortunately, the `itkDTITubeSpatialObject` class did not provide any method for retrieving information about crossing or intersecting with e.g. a plane. To overcome this issue, each fiber was checked for intersecting the plane of the ROI. Therefore it had to be assessed if the line is parallel to the plane, on the plane or actually crossing the plane. ROIs could be placed anywhere and their coordinates usually did not match with the coordinate points of fibers. For example, the intersection point between an ROI and a fiber was not available in the fibers' point set. On this account, interpolation techniques were implemented to identify all fibers crossing the ROI. However first, all fibers and their crossing points which matched the plane equation of the ROI were calculated. In the next step, crossing points inside an ROI were identified. When using a circular ROI, crossing points were identified by measuring their distance to the center of the circle. If the distance lies within the circle radius, then the fiber intersected with the circle. A schematic description of the necessary steps is illustrated in fig-

ure 3.9 For polygonal ROIs a different approach provided by VTK was implemented. The `vtkPolygon` class provided a method to check if a given point was inside the polygon or not. Therefore this ROI was converted into a `vtkPolygon` object and by calling the method `isInsidePolygon(crossingPoint)`, fibers crossing the ROI were identified.

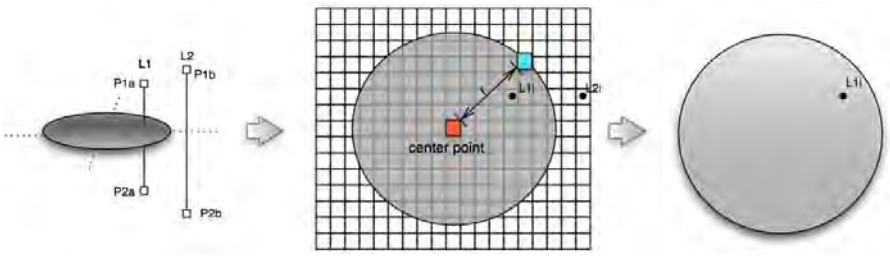


Figure 3.9: This figure illustrates the steps needed to calculate intersection points inside a circle ROI. The left image shows 2 lines $L1(P1a, P2a)$ and $L2(P1b, P2b)$ where both lines intersect with the plane of the ROI. The image in the middle shows the crossing point $L1i$ of $L1$ and $L2i$ of $L2$. Only $L1i$ is within the radius r of the circle (right).

Fiber extraction for FiberBundleX

Fiber extraction for the VTK based FBX was also possible. But in this case no custom implementation was necessary since VTK provided appropriate filters dealing with `vtkPolyData`. For calculating crossing points between a fiber and the plane of the ROI, two filters were available which are listed below.

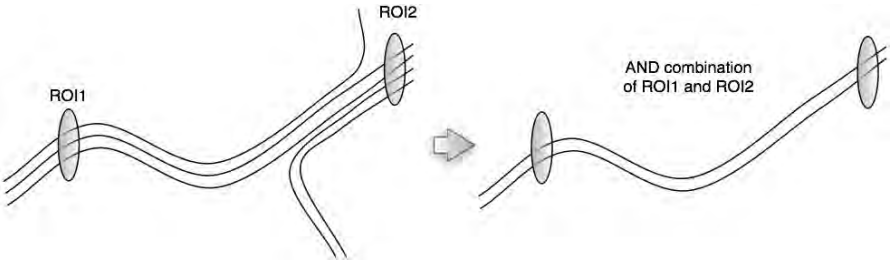
vtkCutter The `vtkCutter` sliced through the `vtkPolyData` object and returned all crossing points between the fibers and the cutting plane. Unfortunately, the output of the cutter allowed no reconciliation between crossing points and their related fibers. Therefore no fibers could be identified.

vtkClipper This filter also sliced through the polydata object but the output included information to identify which crossing point belonged to which fiber.

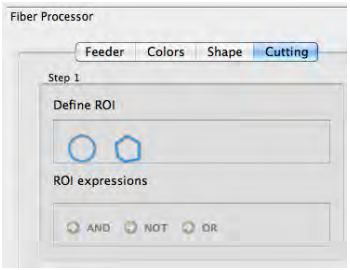
Combining ROIs

Fiber structures of the human brain are complex and simple ROIs are not sufficient to extract the required fibers. To achieve this, multiple ROIs were combined to boolean expressions including *AND, OR, NOT* operations. These combinations were managed by the `mitkPlanarFigure-Composite` data type which includes all defined ROIs. Hence only one data object was passed to the extraction mechanism instead of passing on all ROIs explicitly. Figure 3.10(a) illustrates an AND expression of two ROIs. Figure 3.10(d) shows several combinations of ROIs. Image 3.10(b) shows the GUI elements for ROIs and ROI expressions. An example of how to initialize an expression of ROI is listed below.

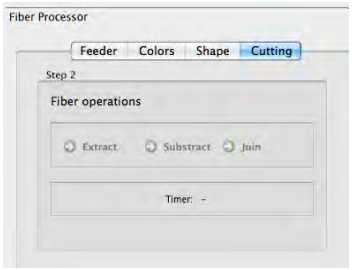
```
mitk::PlanarFigureComposite::Pointer PFCAnd =
    mitk::PlanarFigureComposite::New();
PFCAnd->setOperationType(mitk::PFCOMPOSITION_AND_OPERATION);
    PFCAnd->addPlanarFigure( ROI1 );
    PFCAnd->addPlanarFigure( ROI2 );
    PFCAnd->addPlanarFigure( ROI3 );
```



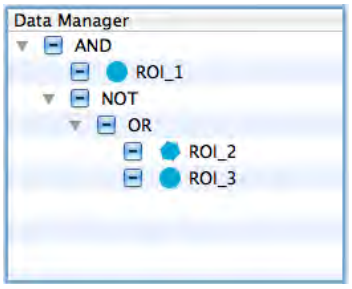
(a) AND expression of ROIs (left) and the related result (right).



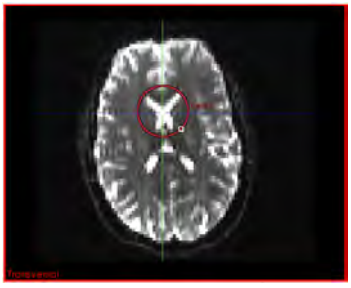
(b) Set and define ROIs.



(c) Executing fiber operations including ROIs.



(d) Combination of ROIs.



(e) Circle ROI drawn on DWI image in 2D view.

Figure 3.10: GUI elements of ROI operations and 2D representation.

Bringing ROIs into 3D

In MITK, ROIs defined by `mitkPlanarFigures` were visualized only in 2D views. However, from the users point of view a 3D representation would be helpful for navigation and

orientation. Unfortunately planar figures of the 2D view were not incorporated into the 3D view. To overcome this, mechanisms for visualizing circular and polygonal ROIs in 3D were implemented. Converting ROIs into VTK objects and passing them on to the 3D rendering pipeline was realized in `mitkCirclePlanarFigureMapper3D` and `mitkPolygonPlanarFigureMapper3D`. Adjustments of the ROIs made in the 2D view were automatically updated in the 3D view too. Additionally, ROIs in 3D were enhanced with color and opacity properties which can be modified by the user.

Joining and subtracting fibers

Mechanisms to investigate and compare fiber structures are needed in the research field of tractography. Merging and subtracting fibers are alternate methods to extracting fiber structures. Some tasks can be applied more efficiently using this approach. In both approaches, `mitkFiberBundle` and `mitkFiberBundleX`, all selected fibers were copied to a new fiber bundle when merging fiber structures. When subtracting fibers, only those which do not occur in the other fiber bundle were copied into a new one. Figure 3.10(c) shows the corresponding GUI elements.

Monitoring fiber processing

For usability reasons, a program usually provides feedback to the user when it is busy, e.g. while executing an algorithm. MITK provided an interface with a Qt progress bar which allowed the user to monitor the current progress of the computation. Unfortunately, this interface did not work properly on all platforms, therefore, the `mitkFiberBundleXThreadMonitor` class was introduced. Since the fiber processing framework was capable of multitasking, threads executing fiber processing tasks and their current status were monitored by the user. When executing a thread, signals were sent which call methods before and after initialization. In the mentioned methods, the fiber monitor was updated to the current status. This feedback was directly fed into the 3D render window which allows fancy and appealing visual effects. The fiber monitoring infrastructure was implemented in the `mitkFiberProcessingView` class. Monitoring could be activated in the GUI via a checkbox, which is shown in figure 3.3(b).

3.4 Evaluation

The evaluation part was divided into quantitative and qualitative sections to assess the success of the applications described above. One of the aims was to test this application with radiologists and domain experts, and determine its practicability. This was accomplished by two real-life scenarios namely the DTI-tractography challenge MICCAI 2011 where the corticospinal tract of the human brain had to be tracked based on clinical data. This tasks is a highly relevant neurosurgically challenge in practice. The second scenario was the visualization of all fiber structures of the human brain published in the DTI-Atlas book. In the context of these two real-life applications the fiber processing framework was evaluated on both practical usage, and a sufficient feature set supporting these tasks. However, in this work, the evaluation of the

addressed problems and features along with their performance and efficiency are also considered .

Qualitative evaluation

The aim of the qualitative evaluation was to assess whether the implemented tool for fiber processing and fiber visualization meet the expectations of the end-users. End-users are domain experts, radiologists and neuroscientists. In order to test applicability and translational relevance of this work, scenarios were generated to evaluate and assess the framework. The aim was to figure out, which of the realized options and modes were useful and supportive for the target group. Results were obtained by personal interviews of the user after each experiment. All experiments are listed below:

Representation of fiber structures

Tractography results can be represented in different ways, from vectors and matrices including coordinates and direction information up to tubal shaped fiber tracts in 3D. Evaluation of the shape and type of point, line and tube based representations in 3D with and without line smoothening as well as fiber visualization in 2D were performed.

Coloring of fiber structures

When using colors, complex data can be visualized intuitively and efficiently. Color codings such as orientation based, FA based color coding and FA as opacity level as well as custom coloring of fibers were examined.

Fiber processing tools

Tools and computations for fiber processing are essential to work with fibers. The selected options for fiber processing include extracting, joining and merging of fibers. They were evaluated if they are sufficient to perform common tasks in this field.

Usability

In this case, usability focuses on stability of fiber processing and the appearance of the GUI compared to the remaining MITK plugins including interaction workflows and data handling. Also subjective impressions about latency and their effect on the user during fiber processing was evaluated.

Quantitative evaluation

In this section, a description of the methods and tasks of fiber processing and assessment of their error rate, performance and efficiency is given. First, the output of fiber processing mechanisms needed to be validated especially in case of fiber extraction. In reference to this, artificial fibers with varying length but similar directions were generated. Additionally, fiber extraction was

performed on real data based on the Gibbs tracking approach and evaluated subsequently. In the next step, computational tasks were measured using the `itkTimeProbe`. Time measurements were repeated ten times for calculating a mean value. The fiber processing framework supported multithreading. The start command for time measuring was set right before the computational task within the executing thread and was stopped right after the computation. Therefore, all thread related pre- and post-processing were not included. The next measurements enumerated user interactions and inputs to accomplish a given task or workflow. In the final evaluation study, the fiber processing infrastructure was tested with results of common tractography approaches; this included the local and stochastic approach of Ngo [38], Gibbs global and probabilistic, and FACT streamline approach. The FACT output was computed using the external tractography application from MedINRIA. This data was then imported into MITK. All computations and measurements of the quantitative evaluation were performed on a computer with specifications listed in table 3.1. The fiber processing framework was tested within the MITK *ExtApp* application, available as an installer built (release mode). An overview of all tasks for quantitative evaluation is listed below.

Checking for correct fiber extraction, join and subtraction: The purpose of these tests was to validate if the applied fiber processing method performed as expected especially when combining ROIs to boolean expressions. Therefore, a small set of fibers (Fiber phantom, see table 3.2) were used which allowed a fast visual inspection of the computationally extracted fibers. Following ROIs and combinations of ROIs were evaluated:

- Circular ROIs
- Polygonal ROIs
- AND planar figure composite
- OR planar figure composite
- NOT planar figure composite
- Join of extracted fibers
- Subtraction of fibers

Measuring processing and executing time: These tests were performed on fiber sets FS_A, FS_B and FS_C (see table 3.2). The aim was to to give an assessment of the performances of the following:

- Loading and initializing fiber data structures
- Rendering performance of representation modes
- Changing color modes
- Using interaction tools

Enumerate the number of inputs given by the user: For the following tasks fiber set FS_B (see table 3.2) was used.

- Extracting the cingulum

- Extracting the corticospinal tract

Compatibility of fiber processing: The purpose of this evaluation was to examine the compatibility of the fiber processing tools to common tractography approaches including external applications. In addition to Gibbs fiber tracking, fiber processing was also applied on following tractography approaches:

- Ngo's stochastic tracking: Extract a fiber set using a ROI and apply white coloring (using FS_D in table 3.2)
- FACT, performed on MedINRIA, see FS_F in table 3.2: Subtract any fibers but the corpus callosum from the original fiber set.

Specifications of hardware and data

Machine	iMac (Mid 2010)
CPU	Quad Core-i7 (1.92GHz)
RAM	8 GB
Graphic Card	ATI Radeon 5070
Storage Media	HDD
Operating System	OSX 10.6

Table 3.1: Operating system and hardware specifications of the testing machine.

MRI	MRI of human head. Single-shot EPI seqence on 3T MR scanner; Three repetitions of 64 different gradient directions; 2.5 mm isotropic resolution; Two shells at b-values of 1000 mm/s^2 and 3500 mm/s^2.
DWI post-processing	Eddy currents and head motion correction using FSL flirt [69] (affine registration of the baseline and diffusion weighted volumes to the first baseline volume); Gradient directions were corrected according to the transformation.
Diffusion model 1	Q-Ball reconstruction of acquired DWI data using default parameters provided by MITK. Based on this data, Gibbs tractography was applied.
Diffusion model 2	Tensor reconstruction of acquired DWI data using default paramters provided by MITK. Based on this data, stochastic tracking was applied.
Tractography algorithm 1	Gibbs tractography using default parameters provided by MITK.
Tractography algorithm 2	Stochastic tracking using default parameters provided by MITK.
Fiberset_A (FS_A)	Gibbs tractography result containing 639 fibers and 6,491 points in total.
Fiberset_B (FS_B)	Gibbs tractography result containing 35,109 fibers and 427,867 points in total.
Fiberset_C (FS_C)	Randomly generated fibers using `vtkPointSource()`; 149,978 fibers and 1,800,000 points in total.
Fiberset_D (FS_D)	Stochastic tractography of corpus callosum and spinal cord containing 986 fibers and 89,102 points in total.
Fiberset_E (FS_E)	Gibbs tractography result containing 35,109 fibers and 427,867 points in total.
Fiber phantom	Gibbs tractography result based on DWI data set of a phantom provided by physics department at DKFZ. Fiber set contains 223 fibers and 3,333 points in total.
Fiberset_F (FS_F)	FACT tractography performed by MedINRIA application. Fiber set contains 27,756 fibers and 3,743,101 points in total.

Table 3.2: Applied acquisition and post-processing methods of DWI data as well as the performed fiber tracking algorithm with its parameters.

CHAPTER 4

Results

In this section the results of the implemented framework are presented. This involved the integration of visualization of fibers in 2D and 3D as well as interaction and fiber processing mechanisms into MITK. Sixty seven experiments were performed to evaluate quantitative and qualitative aspects of the described work. Further, the practicability of the fiber processing application was evaluated based on the two mentioned real-life scenarios, DTI-tractography challenge MICCAI 2011 and DTI-Atlas book.

4.1 Software design and application-level support

Data structures, fiber processing algorithms and user interfaces for efficient visualization and interaction were successfully incorporated into the MITK platform. The source code is available in the open source part of MITK. The fiber processing framework was incorporated into the executable application *MITK Diffusion Imaging* (MITK-DI release date: 02-11-2011). This application as well as source code are available online for Windows, Linux and OSX platforms. GUI and GUI elements were designed in accordance with usability guidelines defined by the MITK community in order to provide the optimal look & feel as well as usability patterns. Furthermore, the fiber processing framework supported multithreading to increase performance and improve the interaction experience for the end-users.

Qualitative Evaluation

The qualitative evaluation refers to the interviews and feedback of the end-users. *The look & feel of the fiber processing matches with the remaining MITK environment. Also the workflow design, from data selection to applying an algorithm on fibers is similar to the remaining image processing tools incorporated in MITK. When fiber processing tasks are performed, the application provided feedback about its current status. Feedback from a progress bar would be more intuitive than a status message in the 3D render window.* Alternatively, some of the interviewees suggested no feedback for computation at all. *No subjective impressions like a frozen or crashed*

application are recognized during fiber processing. During the final testing of the MITK-DI release, all fiber processing tasks terminated successfully without any noticeable crashes whereas during the real-life scenarios some crashes of the whole MITK application occurred which were not of limiting relevance. Figure 4.1 shows the MITK application including GUI elements of the fiber visualization and interaction framework.

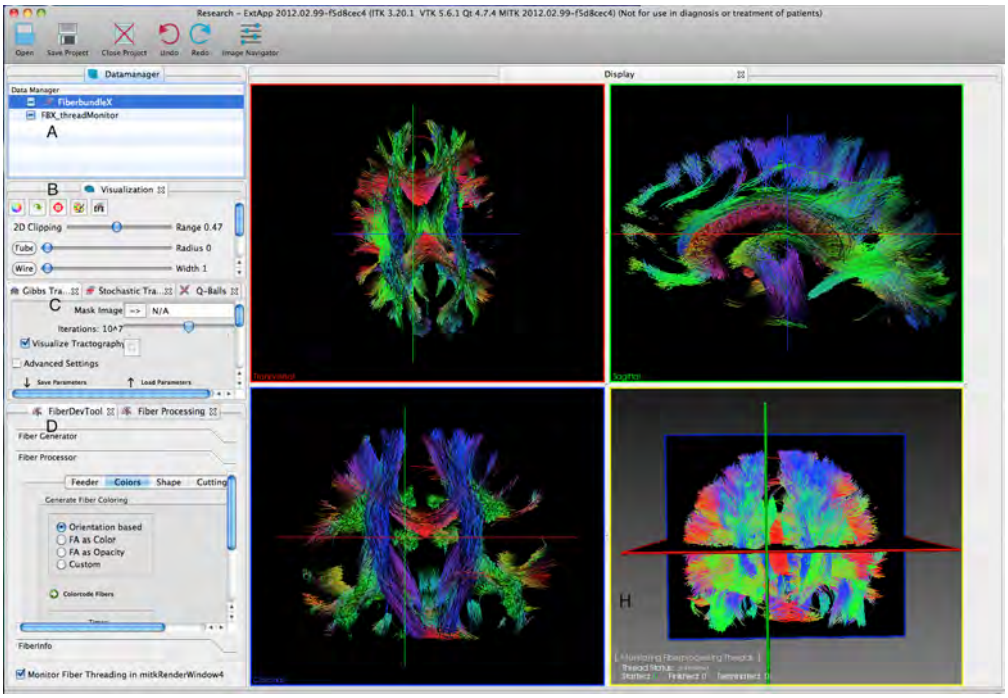


Figure 4.1: This figure shows the MITK application including the datamanager (A), options for visualizing fibers in 2D and 3D (B), parameter settings for Gibbs tractography (C), options for coloring the fiber bundle(D). Feedback about the current status of fiber processing is shown in the lower left corner of the 3D render window (H). Fiber set FS_B with orientation based color coding is visualized.

Quantitative Evaluation

Time measurements of a basic I/O workflow and computation of the fiber data structures were performed with data sets FS_A, FS_B and FS_C. The workflow was divided into three separate sections, reading the fibers from the hard disk, initializing the appropriate data structure of fibers, and finally, preparing and processing the fiber structures for the visualization pipeline. Table 4.1 shows the results for both data types, FB and FBX. As expected, both data types, FB and FBX, showed no difference when visualizing them. Figure 4.2 shows the fiber sets used for evaluating the performance in table 4.1.

Fiber set	Data type	Reading from HDD	Initializing data type	Preprocessing for rendering
FS_A	FB	0.07 (0.00)	0.36 (0.00)	0.17 (0.00)
	FBX	0.03 (0.00)	0.01 (0.00)	-
FS_B	FB	4.57 (0.14)	20.59 (0.26)	11.28 (0.66)
	FBX	2.23 (0.01)	0.19 (0.00)	-
FS_C	FB	20.89 (0.82)	89.70 (0.78)	122.72 (16.48)
	FBX	9.35 (0.02)	1.12 (0.01)	-

Table 4.1: Computation of data structures in seconds (lower = better). Measurements are represented by the mean value and standard deviation of 10 replicates for each experiment. The standard deviation is listed in brackets next to the mean value. Precision was round up to two positions after decimal point. FB (mitkFiberBundle), FBX (mitkFiberBundleX).

4.2 Visualization of fiber tracts

Fibers were visualized in 2D and 3D space which supported representation of points, lines, tubes and smoothed structures. When fiber data was loaded into MITK, visualization of the fibers in the corresponding renderwindows and orientation based coloring were automatically applied.

Qualitative evaluation

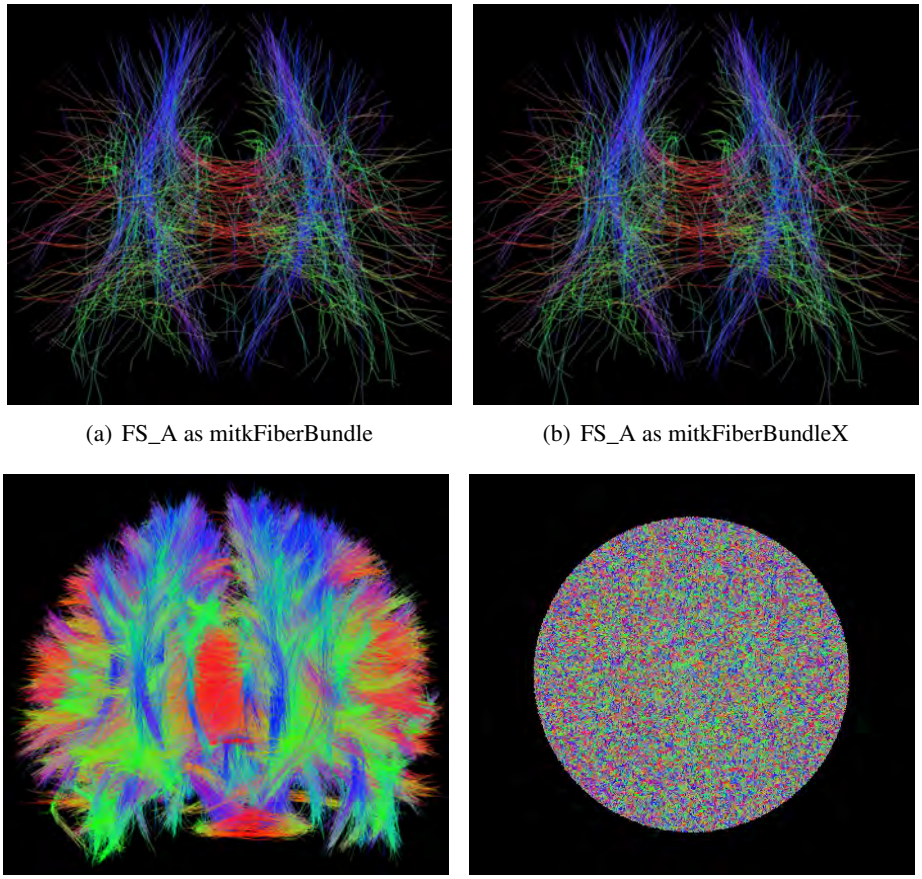
The qualitative evaluation of this section describes different representation modes for fiber visualization and reflects the feedback of end-users. In addition to that, color coding such as orientation based coloring as well as scalar based coloring were tested and evaluated.

Representation of fiber structures

For this evaluation, fiber set FS_B was used. The following four paragraphs focus on fiber representation in 3D, followed by an explicit paragraph summarizing the feedback for 2D fiber visualization.

Points: With respect to point based representation, all end-users were of the same opinion. *A point based representation does not give the best overview of neural pathways, although this mode represents exactly the results of fiber tracking algorithms because they provide a sorted list of points. However, such point clouds in 3D assist interpretation of density of fiber structures.* An example is shown in figure 4.3(a).

Lines: This mode provided an intuitive overview of fibers which was also applied to generate images for the DTI-Atlas book. *It is easy to track fibers using navigation mechanisms such as zooming, panning and rotation. Its representation can give rise to sharply edged tracts that do not match with anatomical expectation of human fibers. A realistic 3D impression is not given because light and shade was missing. However, when navigating through the fibers using zooming and rotation this drawback is compensated.* An example is shown in figure 4.3(b).



(a) FS_A as mitkFiberBundle (b) FS_A as mitkFiberBundleX

(c) FS_B (d) FS_C

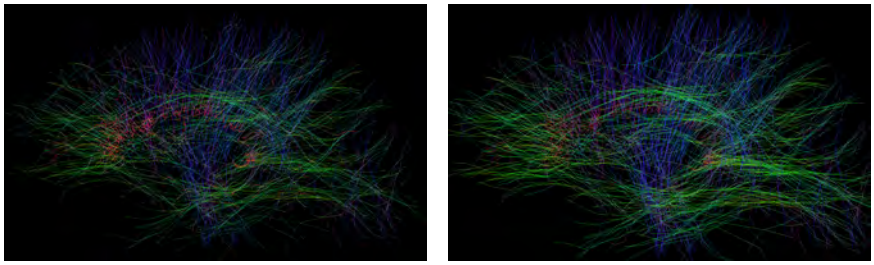
Figure 4.2: This figure shows the fiber sets used for evaluating the performance of the data structure in terms of I/O within MITK. Subfigures (a) and (b) show fiber set FS_A containing 639 fibers, where (a) is a FB data structure and (b) is FBX. From the visualization point of view, both data types look the same. Subfigure (c) shows fiber set FS_B including 35,109 fibers and (d) shows fiber set FS_C hosting 149,978 fibers. All structures show orientation based color coding.

Tubes: *The tubal shape gives the most realistic impression of fibers even without applying any navigation techniques compared to lines. Tubes actually represent a surface including light and shading. So far a tubal representation is not mandatory since it was mainly used for generating good looking images in 3D. However, there are tasks where the radius of a tube can support describing parameters of fibers and diffusion [70, 71] and tubes would be a nice feature to have.* An example is shown in figure 4.3(c). For the DTI-Atlas book, images of fiber structures represented as tubes were also published.

Smoothed lines and tubes: *Smoothed fiber structures provide a more natural representation of*

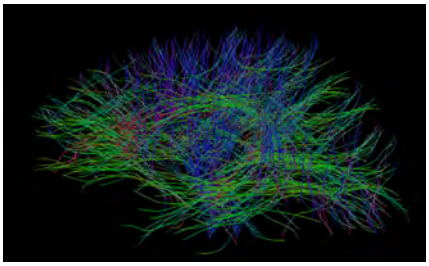
fiber paths appearing in nature. In some cases fiber smoothing provided a better understanding of the fiber orientation, especially in areas of high fiber density. (See figures (e) and (f)). End-users from the computational field made no difference between smoothed and non smoothed fibers except for statistical purposes. *For fiber processing tasks both smoothed and non smoothed fibers can be applied without noticing any difference in the results. In terms of statistical analysis of fiber structures, in general, smoothed fibers shows a longer path and more points compared to original non smoothed fibers.* This is due to the interpolation algorithm which added points in between points from the original data. *However, this a useful option for generating naturally looking fiber structures*. An example for smoothed lines is shown in figure 4.3(d).

Visualizing fibers in 2D: Visualizing fibers in 2D by default was a new experience to most interviewed end-users since other free available applications for fiber visualization require explicit commands of the user. *Fiber structures in 2D and 3D appears consistently allowing intuitive and accurate navigation through fiber data.* It is to be noted that consistent coloring of fibers was applied by the GPU based approach for 2D visualization (see figure 4.5). *Thickness of the 2D slice could be adjusted dynamically, which is a common feature for normal images in radiology.* The results of the CPU based approach using `vtkCutter` is illustrated in figure 4.5(a). At DTI-tractography challenge MICCAI 2011 fiber visualization in 2D lead to accurate ROI placement for fiber extraction.

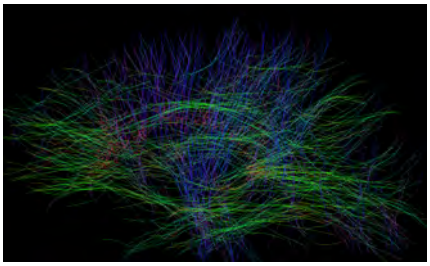


(a) Point based representation

(b) Fibers represented as lines



(c) Fibers represented as tubes

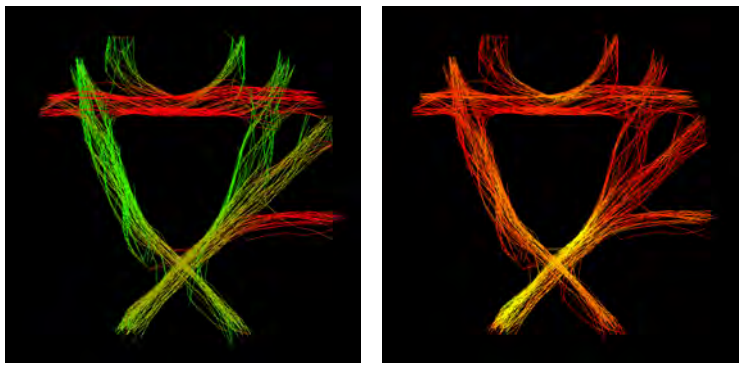


(d) Smoothed fiber structures

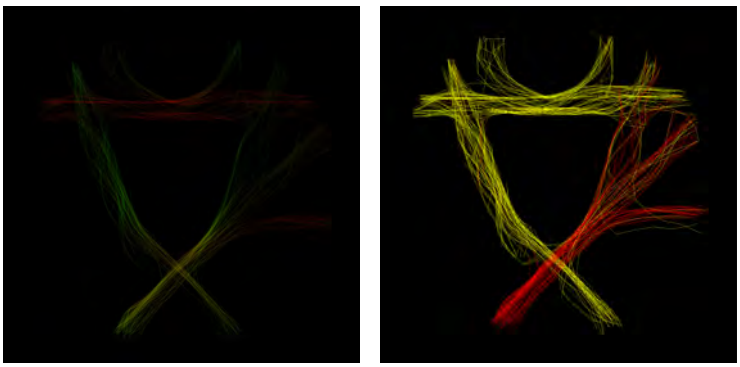
Figure 4.3: This figure shows all implemented representation modes in 3D. Data set FS_A with orientation based color coding is shown. Subfigure (a) represents fiber structures as points, while (b) and (c) represent lines and tubes. In (d) the fibers are smoothed from (b).

Coloring of fibers

In the following sections different color codings for fiber structures are evaluated based on the feedback of interviewed end-users.



(a) Orientation based color coding (b) FA based color coding



(c) FA as opacity levels (d) Individual color coding

Figure 4.4: This image shows different variations of color coding applied to the fiber set of the phantom. (a) shows orientation based color coding, (b) represents FA values as a color scheme, where yellow and red represent high and low FA values, respectively. (c) illustrates FA values as opacity in combination with orientation based color coding and subfigure (d) shows individual colors applied by the user.

Orientation based color coding: *This is a basic color coding also available in other software applications for fiber visualization.* Applications and frameworks were introduced in chapter two. *Due to the correlation of colors and the orientation of fiber segments, visual identification of fibers along similar directions could be accomplished, even on areas of of complex fiber constellation.* An example for orientation based color coding is shown in figure 4.4(a). This coloring was applied to generate images for the DTI-Atlas book and it also supported navigation through the fiber set at DTI-tractography challenge MICCAI 2011.

FA based color coding: *When FA based coloring is performed, fiber segments of high FA shared the same color helping to identify similar levels of diffusion. This helps explaining tractography results since FA coloring visualizes directional uncertainty in combination with the respective fibers. In practice, this technique is applied to evaluate diffusion parameters of fiber phantoms.* See figure 4.4(b) showing a FA color coded phantom which provides information about its physical characteristics. One interviewee, a radiologist, still prefers using FA maps instead of the corresponding coloring fibers.

FA as opacity level: *This is a new and practicable feature, because it combines both advantages of FA and orientation based coloring.* This feature also captured the interest of radiologists, who preferred FA maps to FA coloring. This mode is shown in figure 4.4(c).

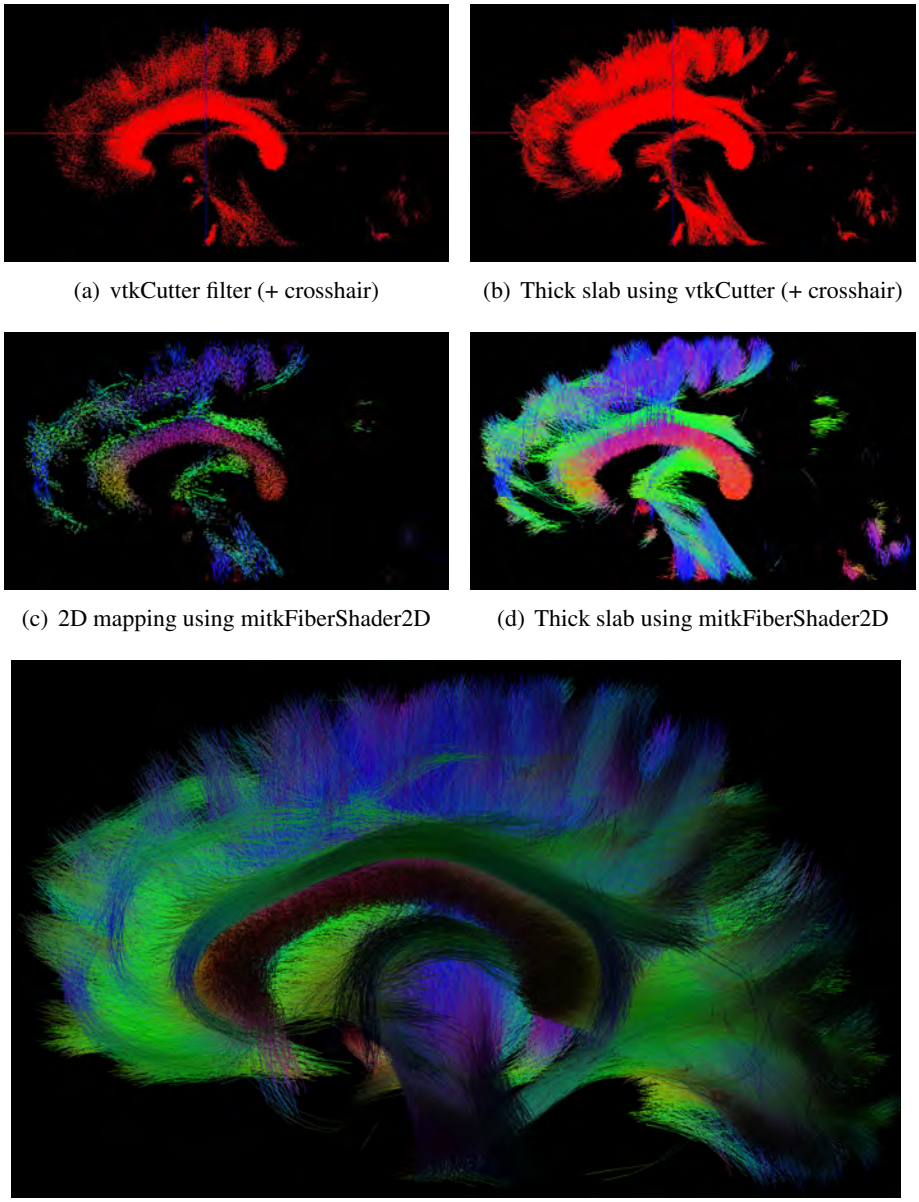
Individual coloring: *This coloring allows individual coloring of individual fibers. As opposed to orientation based coloring, fibers can now be illustrated according to their anatomical affiliation even if they run through areas of complex fiber constellation. Anatomical structures such as the corpus callosum, cingulum etc. are colored individually.* This technique was also applied to generate images of specific areas of the human brain for the DTI-Atlas book. Also physical correlations were displayed intuitively by grouping individual structures. This coloring is illustrated in figures 4.4(d) and 4.9.

Quantitative evaluation

Performance of representation modes in 2D Both the CPU and GPU based approach for visualizing fibers is evaluated in this section. Since the OGSL based `mitkFiberShader2D` shader was executed on the GPU, the calculation time for each frame was measured. Table 4.2 shows the performance of each method. It should be noted that time and frame measurements represent the values for a single 2D render window.

Mapping method	Fiber set	Computation time for 1 slice (CPU)	Computation time of frame (GPU)
`vtkCutter`	FS_A	0.0017 (0.00)	4.8E-05 (0.00)
	FS_B	0.8358 (0.02)	4.5E-05 (0.00)
	FS_C	5.9417 (0.27)	4.9E-05 (0.00)
`mitkFiberShader2D`	FS_A	-	0.0008 (0.00)
	FS_B	-	0.0116 (0.00)
	FS_C	-	0.1361 (0.01)

Table 4.2: Performance of representation modes in 2D. This table shows the computation time for performing a cut through the data set (CPU based cutter approach) and for calculating the 2D visualization on the GPU (shader approach). Lower computation time = better. Measurements are represented as the mean value and standard deviation of 10 replicates for each experiment. The standard deviation is listed in brackets next to the mean value.



(a) vtkCutter filter (+ crosshair)

(b) Thick slab using vtkCutter (+ crosshair)

(c) 2D mapping using mitkFiberShader2D

(d) Thick slab using mitkFiberShader2D

(e) mitkFiberShader2D in combination with thick slab and fading effects

Figure 4.5: This figure shows the outcome of several techniques for 2D visualization including the `vtkCutter` filter, illustrated in (a) and (b). This filter has no information about the original color or adjacent points, therefore the result is colored with a single color only. (b) and (d) show thick slab of 2 cm. (c) to (e) illustrate the results when using the `mitkFiberShader2D` shading mechanism including depth and fading effects.

Performance of representation modes in 3D

These measurements were performed during fiber rotation in the 3D render window using mouse interaction. Table 4.3 shows the results for point, line and tube based representation modes using the method `GetLastRenderTimeInSeconds()` provided by `vtkRenderer`, which determines the calculation time for rendering one frame. Figure 4.2 shows the data sets applied for this evaluation.

Fiber set	Representation mode	Calculation time of frame
FS_A	Points	0.00 (0.00)
	Lines	0.00 (0.00)
	Tubes	0.03 (0.00)
FS_B	Points	0.00 (0.00)
	Lines	0.04 (0.02)
	Tubes	199.13 (46.52)
FS_C	Points	0.00 (0.00)
	Lines	0.11 (0.01)
	Tubes	>200*

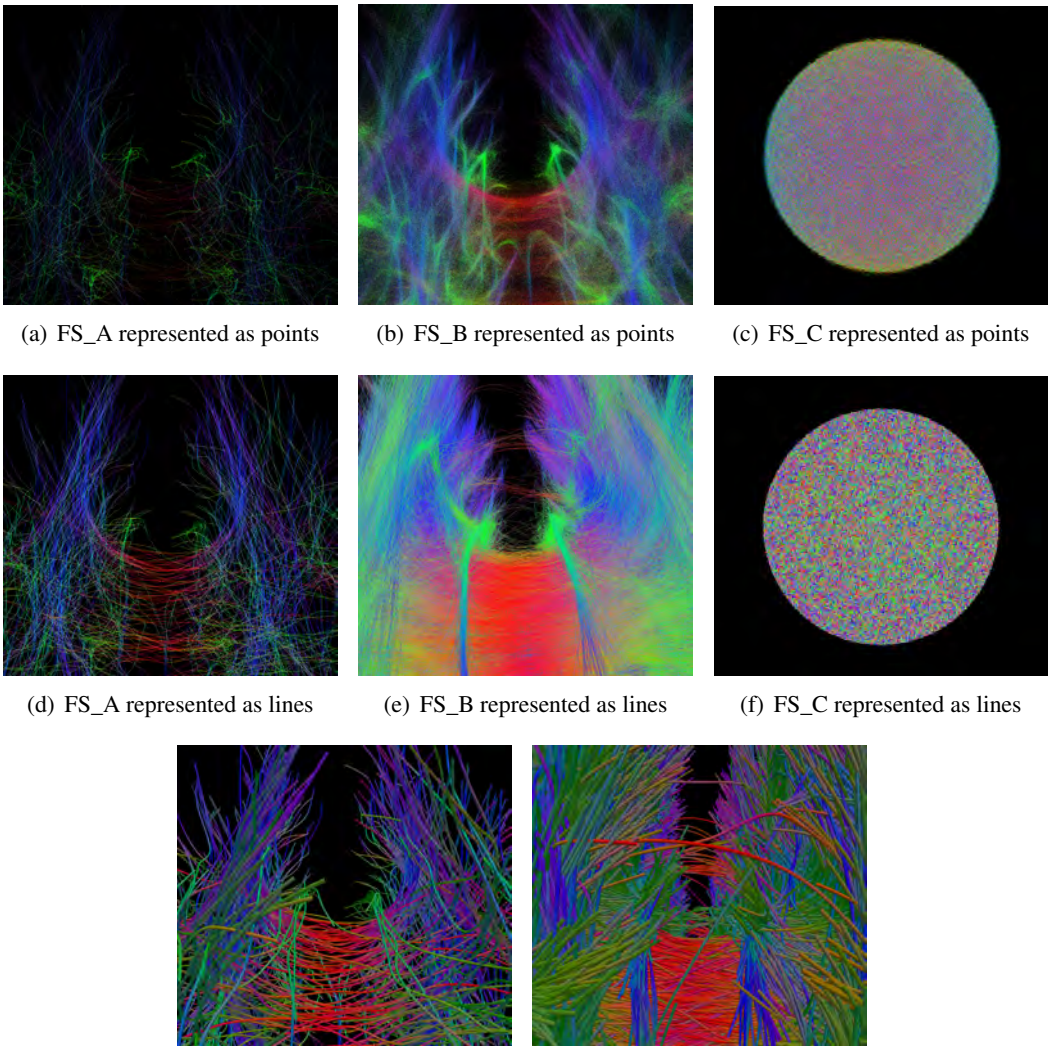
Table 4.3: Performance of representation modes in 3D measured in seconds. This table shows the frame rate for common representation modes in 3D. Lower calculation time = better. Measurements are represented by the mean value and standard deviation of 10 replicates for each experiment. The standard deviation is listed in brackets next to the mean value. *After 120 seconds of initializing this mode, memory started swapping out. Therefore it is assumed that rendering more than 35,109 fibers would take more than 200 seconds.

Computation of fiber smoothing

In this section the computational performance of fiber sets FS_A - FS_C are evaluated. Table 4.4 lists the execution time for applying fiber smoothing to the given fiber bundle. The smoothing parameter was set to 10 points per centimeter. Figure 4.7 shows the processed fiber structures.

Fiber set (number of points)	Total points after smoothening	Execution time
FS_A (6,491)	42,037	0.12 (0.00)
FS_B (427,867)	2,810,138	7.23 (0.02)
FS_C (1,800,000)	20,438,638	38.97 (0.08)

Table 4.4: This table shows the execution time in seconds for fiber smoothing of the fiber sets FS_A - FS_C. Lower execution time = better. Measurements are represented by the mean value and standard deviation of 10 replicates for each experiment. The standard deviation is listed in brackets next to the mean value.

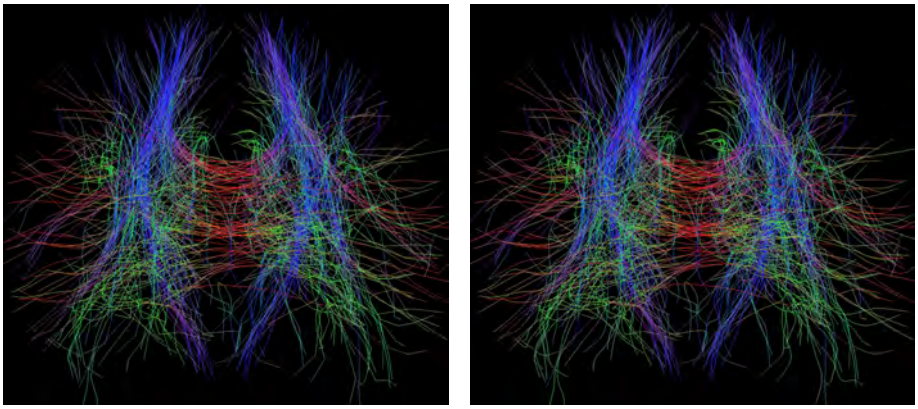


(a) FS_A represented as points (b) FS_B represented as points (c) FS_C represented as points

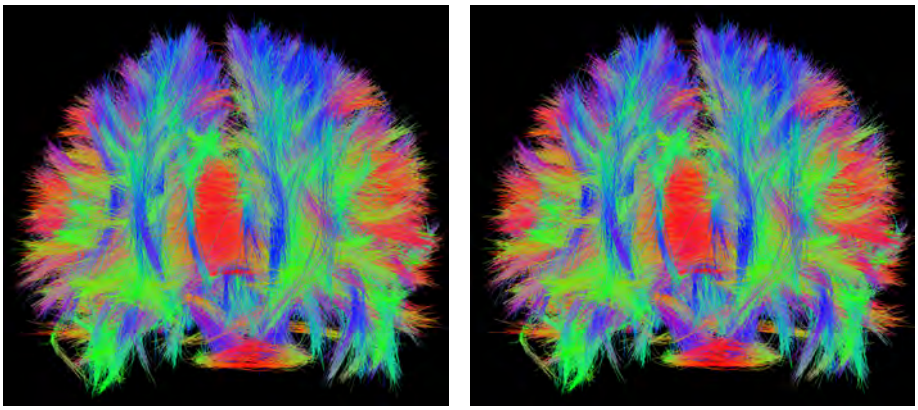
(d) FS_A represented as lines (e) FS_B represented as lines (f) FS_C represented as lines

(g) FS_A represented as tubes (h) FS_B represented as tubes

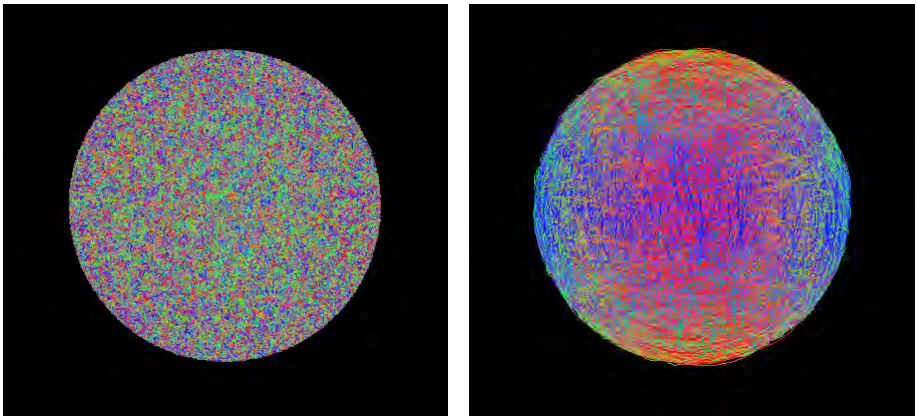
Figure 4.6: This figure shows the processed datasets for evaluating the performance of all implemented representation modes. Fiber sets FS_A ((a), (d), (g)), FS_B ((b), (e), (h)) and FS_C ((c), (f)) were used for this experiment. Subfigure (a) to (c) show the applied data sets in point based representation, whereas (d) to (f), and (g) and (h) illustrate line and tube based visualization, respectively. Due to memory and performance limitations of the testing machine, tubal representation of FS_C could not be accomplished. All fiber structures are shown with orientation based coloring.



(a) FS_A (b) Smoothed FS_A



(c) FS_B (d) Smoothed FS_B



(e) FS_C (f) Smoothed FS_C

Figure 4.7: This figure shows the fiber structures for evaluating the performance of smoothing where the left column represents the original data and the right column the smoothed structures. All fiber sets show orientation based coloring.

Computation of fiber coloring

In this section the results of all implemented color modes tested on fiber bundles are summarized. In table 4.5 the execution time for applying the requested color coding to the given fiber bundle is listed. The data sets used for this evaluation are shown in the left column of figure 4.7.

Fiber set	Coloring	Execution time
FS_A	Orientation based	0.00 (0.00)
	FA coloring	0.00 (0.00)
	FA as opacity	0.00 (0.00)
	Individual	0.00 (0.00)
FS_B	Orientation based	0.05 (0.00)
	FA coloring	0.12 (0.00)
	FA as opacity	0.01 (0.00)
	Individual	0.00 (0.00)
FS_C	Orientation based	0.23 (0.00)
	FA coloring	0.48 (0.00)
	FA as opacity	0.03 (0.00)
	Individual	0.00 (0.00)

Table 4.5: This table shows the execution time in seconds for different color codings and varying number of fibers. Lower execution time = better. Measurements are represented by the mean value and standard deviation of 10 replicates for each experiment. The standard deviation is listed in brackets next to the mean value.

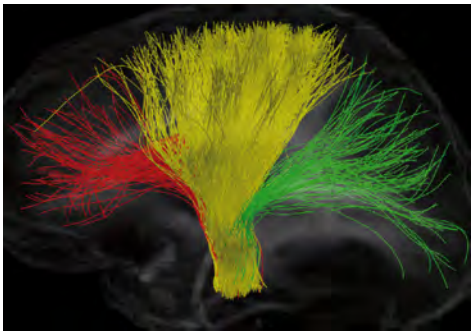
4.3 Fiber interaction and processing

Besides visualization of fibers, interaction with them is one of the main goals of the fiber processing framework. All interaction mechanisms (see chapter 3) were successfully implemented and the results from the user's perspective are described below. For extracting fibers, ROIs were drawn on the 2D views and visualized also in 3D. Accurate and exact placement of ROIs could be achieved with previously existing interaction mechanisms in MITK, namely zooming, panning and swiveling of 2D views. ROIs were combined as shown in figure 4.9(d) where the corticospinal tract was extracted. With this technique all fiber structures could be extracted. The joining and subtraction operation is shown in figure 4.10(j).

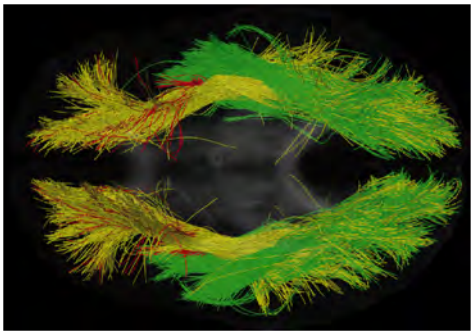
Qualitative evaluation

The fiber processing framework provided tools for extracting, joining and adding functional values to fiber structures. Extracting fibers defined a major task for both the DTI-Atlas book and DTI-tractography challenge MICCAI 2011. Although fiber visualization in 2D could not be accomplished, ROIs could still be placed in the 2D views and were synchronously visualized in the 3D view. Therefore, precise fiber extraction could still be performed. Combination of ROIs

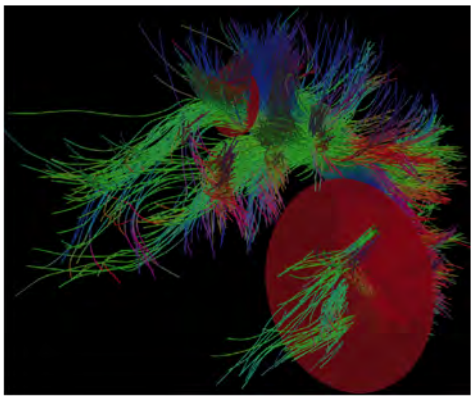
to boolean expressions was also performed for efficient extraction of the desired fiber structure. For example the superior longitudinal fasciculus was extracted using an *AND* combination of ROIs, see figure 4.8(c) and 4.8(d). In combination with options for coloring, all requirements to accomplish the desired tasks of the editors of the DTI-Atlas book were fulfilled. Figures 4.9 and 4.8 show several cases demonstrating how certain structures of the human brain were extracted and visualized. These images were also published in the DTI-Atlas book. The principle challenge at MICCAI 2011, with respect to fiber post-processing, was precise yet at the same time fast extraction of fiber data from the corticospinal tract. Fiber visualization in 2D, optimized extraction mechanisms and real-time visualization of the tractrography data ensured that this challenge was met. Visualization of complex and neurosurgically relevant areas of the brain were achieved by using combination of minimal number of ROIs. From the perspective of radiologists and surgeons this simplified methodology augments its usability and practicability. For example the extraction of the corticospinal tract could be accomplished with just 3 ROIs as shown in figure 4.9(d).



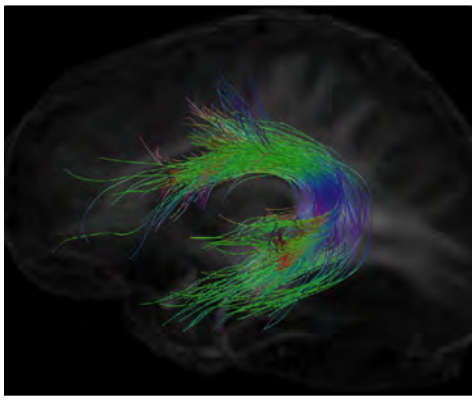
(a) Fronto-pontine tract (red), cortico-spinal tract (yellow), parieto-occipital-pontine tract (green)



(b) Uncinate fascicles (red), inferior fronto-occipital-fasciculus (yellow), inferior temporal cortex (green)



(c) AND combination of ROIs



(d) Extraction of arcuate fasciculus

Figure 4.8: Results of quantitative evaluation of fiber processing. (a), (b) extracted fibers of individual coloring, (d) result of ROI extraction (c) and orientation based coloring of fibers.

Quantitative evaluation

Validation of fiber extraction, joining and subtracting

This section evaluates whether the applied fiber processing mechanisms performed as expected by visual (see figure 4.10) and computational validation (see table 4.6).

Fiber processing	ROI ID	Calculated result	Expected result
Circle ROI	1	72	72
	2	44	44
	3	23	23
Polygon ROI	4	20	20
	5	27	27
AND combination	1, 3	19	19
OR combination	1, 3	76	76
NOT combination	4	203	203
Join	1+2	116	116
Subtraction	(1+2)-2	72	72

Table 4.6: Comparing the expected and observed results of fiber extraction, joining and subtraction based on the fiber phantom. The result columns list the number of fibers.

Tracking interaction steps of the user

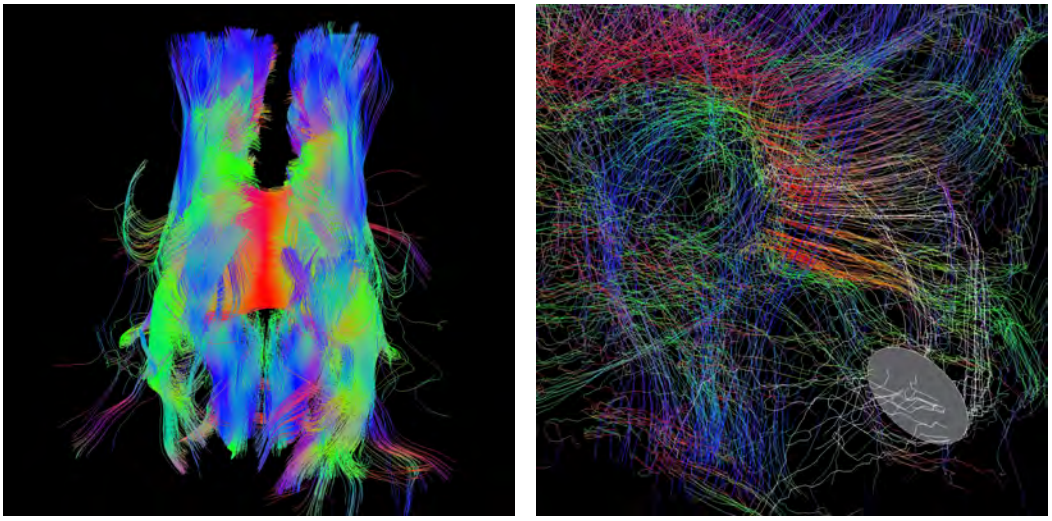
For this part, the user had to accomplish several tasks, for e.g., extracting certain regions of the brain. FS_B was the input for this experiment which was converted into FB and FBX (see table 4.7). To accomplish the given tasks, interaction mechanisms such as zooming, panning, swiveling and rotation were performed. The resulting extraction of cingulum contained 40 fibers and the extracted corticospinal tract contained 960 fibers. Figure 4.9 shows the extracted structures.

Task	Type	Number of ROIs	Execution time
Extracting	FB	2	100.73 (0.85)
Cingulum	FBX	2	0.82 (0.01)
Extracting	FB	3	88.90 (0.62)
Corticospinal Tract	FBX	3	1.24 (0.01)

Table 4.7: Measuring number of ROIs, as well as execution time in seconds to extract specific areas of the fiber set with both FB (mitkFiberBundle) and FBX (mitkFiberBundleX). Fiber set FS_B was used for this evaluation. Measurements are represented by the mean value and standard deviation of 10 replicates for each experiment. The standard deviation is listed in brackets next to the mean value.

Performing fiber processing on common tractography approaches

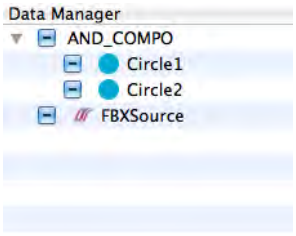
This section evaluates the usage of the implemented fiber processing tools on outputs of various tractography approaches including FACT, local and stochastic tracking, and *Gibbs* tracking. All fiber processing methods (coloring, extraction, join and subtraction) were applied to fiber sets FS_B, FS_D and FS_F. Examples of the processed fiber structures are shown in figure 4.11.



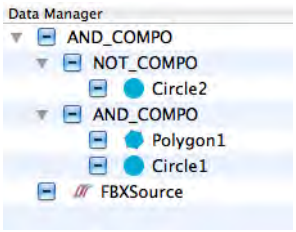
(a) Extracted corpus callosum of FS_F

(b) Stochastic tractography, individual coloring of fibers intersecting with ROI

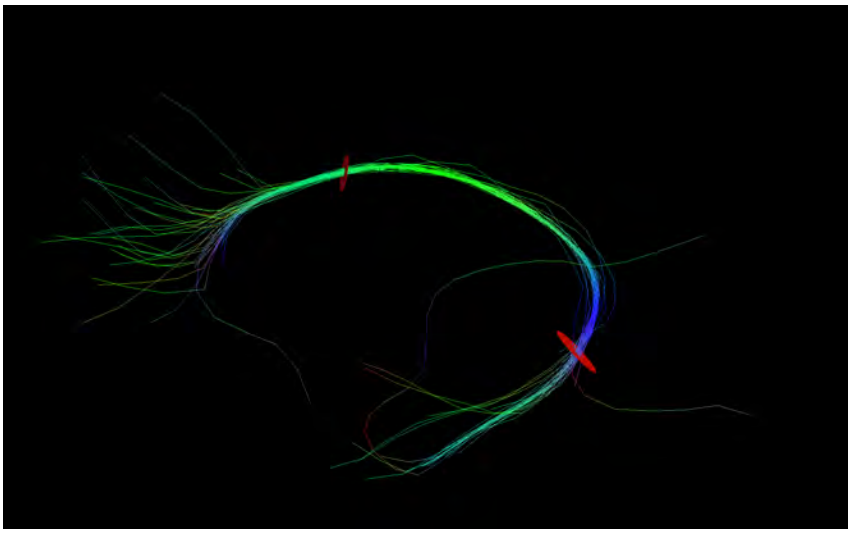
Figure 4.11: This figure shows several fiber interaction and visualization mechanisms applied on results of different tractography approaches. FS_F was imported from MedINRIA and processed using the fiber framework. (a) shows the extraction of the corpus callosum and orientation based coloring. In (b) orientation based coloring as well as individual coloring of the ROI selection were applied on FS_D.



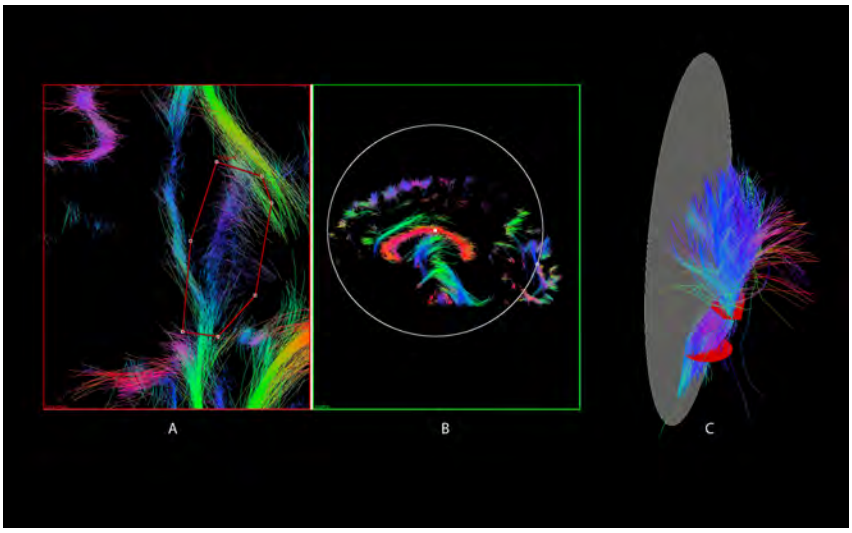
(a) ROIs for cingulum



(b) ROIs for corticospinal tract

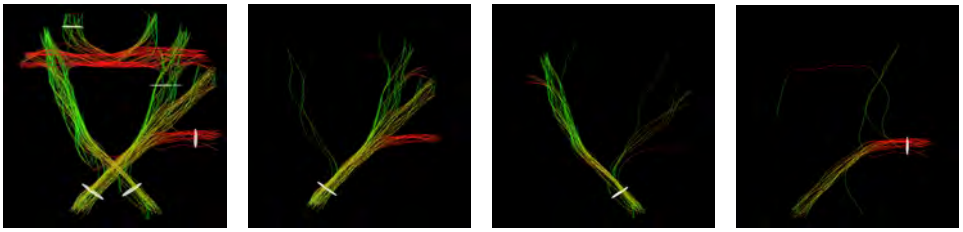


(c) Cingulum, orientation based coloring

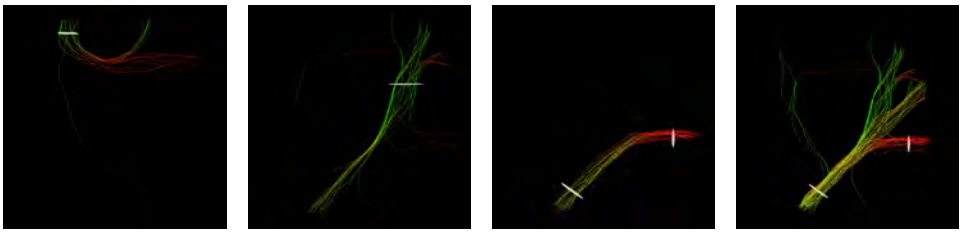


(d) (A,B) ROIs in 2D on FS_B, (C) corticospinal tract in 3D, orientation based coloring

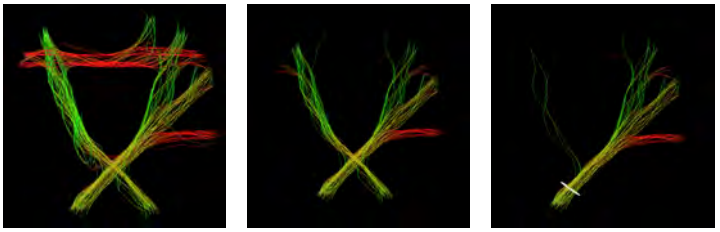
Figure 4.9: This figure shows the boolean concatenation of ROIs in the data manager (a) and (b). The extracted results are shown in (c) and (d) C. (d) A and B show ROI selection in 2D. For this experiment fiber set FS_B was used. For this experiment fiber set FS_B was used.



(a) Original fiber set (b) Circle ROI 1 (c) Circle ROI 2 (d) Circle ROI 3



(e) Polygon ROI 4 (f) Polygon ROI 5 (g) AND of ROI 1 and 3 (h) OR of ROI 1 and 3



(i) NOT of ROI 4 (j) Join of (b) and (c) (k) Subtraction of (j) and (c)

Figure 4.10: This figure shows the data set and ROIs used for evaluating fiber extraction, as well as their joining and subtracting. For this experiment the phantom fiber set was used. All fibers show orientation based color coding. The ROIs were set with the support of 2D plane swiveling.

CHAPTER 5

Discussion

In this work, a framework for fiber processing was introduced to the open source community of MITK. Tools for managing and processing of fiber structures were successfully incorporated. To meet challenging requirements (chapter 2 table 2.1) such as efficient fiber visualization and interaction, a flexible strategy for the development was defined. This included the implementation of two approaches based on ITK and VTK, for the representation of fibers in the framework. These two methods were then contrasted and compared by quantitative evaluation of their performance to determine the more efficient approach. An effective distribution of the superior method was subsequently achieved due to the open-source feature of the framework, thus empowering developers to include the code into their platforms. CTK compatibility allows sharing data structures with other CTK based platforms. Further, due to the modular concept of MITK it is possible to build one's own application around such a framework. Executable installers for Windows, Linux and OSX including this fiber framework are available online in the MITK-Diffusion application. Commonly used data formats for medical imaging are also fully supported. This increased level of compatibility and interoperability are beneficial for a long term distribution of the fiber framework in a large radius. Additionally, to gain acceptance of end-users, features from radiological applications were adopted to increase the application spectrum of fiber processing. Therefore, features from currently available solutions as well as requirements directly from the end-users were included. Another major focus was to introduce innovative techniques for fiber visualization in 2D and 3D, as well as intuitive fiber handling in conjunction with accurate and interactive tools for efficient fiber processing. All interaction and visualization mechanisms were efficiently synchronized between 2D and 3D to keep the medical context consistent and to provide an intuitive and user friendly environment. To develop this in adequate time, several testings with selected end users were applied as well as constant performance testings of the implemented methods and techniques were conducted. In the following paragraphs, the main features of the application along with their advantages and limitations are discussed. An objective comparison to currently established fiber processing applications is also included.

5.1 Software design and application-level support

The fiber processing workflow defines several steps starting from initializing the data type containing fiber information, techniques to interact with and process fibers and to initialize fiber visualization. For efficient fiber processing, each step of the workflow needed to be optimized. Therefore, both ITK and VTK based techniques were assessed in parallel for performance as well as seamless integration into the whole fiber processing workflow. The data type, `mitkFiberBundle`, was implemented based on the ITK approach, whereas `mitkFiberBundleX` was developed using VTK based methods. From the operational point of view, both cases were able to fulfill the given tasks and requirements but in terms of efficiency, FBX showed higher performance and better integration into the workflow. The ITK based fiber structure provides more convenient interfaces to manage fiber data due to the `itkDTITubeSpatialObject` class. In contrast, VTK offers a more generic data type namely `vtkPolyData` which is more flexible and allows adding individual parameters. Overall, the ITK based approach showed several bottlenecks, especially when processing fibers, e.g. extracting or cutting a set of fibers. For this task no appropriate methods are provided by ITK, therefore custom methods needed to be implemented explicitly. In comparison, VTK offers many powerful filters for its polydata format which experience constant contribution from its open source community. At the visualization level, FBX integrates better into the workflow since its data format is already compatible to the visualization pipeline. On the other hand, all data from `itkDTITubeSpatialObject` needed to be processed into the VTK format for visualization. The feature to visualize temporary results of running tractography algorithms could only be accomplished by the VTK based data structure efficiently. To be fair, requirements for the fiber framework were more focused on the visualization and interaction level where VTK offers adequate methods compared to ITK which specializes in traditional image processing. Hence, a modular concept separating fiber visualization from fiber processing using VTK and ITK respectively was not efficient.

5.2 Visualization of fiber tracts

For representing fiber structures in the VTK based render windows of MITK, three major modes including points, lines and tubes were assessed. In the digital world, fibers are composed of adjacent points, while intuitive representation of axons and nerve fibers usually are associated with tubes or strings. Representing only raw data, i.e. points, was insufficient for fiber exploration because it took tremendous effort and concentration to follow directions of fibers when dealing with large data sets. This method was applied for identifying only areas of high fiber density.

3D visualization

So far, tubes in 3D provide the most realistic impression of fibers because VTK defines tubes as surfaces supporting light and shade effects. When dealing with fiber sets $>$ 15,000 fibers, on current mainstream computers, the rendering mechanism of this mode occupies all resources causing juddering during user interaction, e.g. when rotating in 3D. However, to overcome this performance issue, specific fibers can be explicitly extracted and visualized as tubes without

compromising judder free user interaction. On the other hand, line structures can be visualized up to 150.000 lines or even more without any notable performance issues. A difference in color representation was recognized between the two modes due to the missing light and shade effects of line structures. However, by applying rotation, e.g. mouse interaction, the 3D impression could be achieved even without 3D illumination of fibers.

2D visualization

For 2D visualization several techniques to map fiber structures onto 2D planes were tested and evaluated. When dealing with small fiber sets, all mapping mechanisms showed similar results in performance, but for large data sets only the GPU based shader approach was able to accomplish this task efficiently. Instead of explicitly extracting 2D slices of the fiber structure using CPU based methods provided by VTK, the OGSL based shader script, `mitkFiberShader2D` was added to the 2D rendering pipeline which performs the mapping directly on the GPU by dropping all fiber fragments not matching to the 2D viewing plane. Thick slab was realized by adjusting a parameter defining which fragments within the desired range are still visible. When using thick slab, fibers in 2D view show 3D characteristics, therefore lower endings of these fiber parts are rendered with fading effects for depth illustration. Fading is optional and can also be turned off. When activating them in 2D, colors of the fibers are manipulated. Therefore this feature should be used carefully especially when mapping scalar values, e.g. FA, to colors. So far, each fiber representation mode is treated equally by the shader, e.g. tubes in 3D share the same 2D representation as lines and points. In this particular case the context between 2D and 3D is not consistent anymore but serves subjectively for clear representation especially when dealing with large fiber sets. Point based representation can be applied by setting thick slab to the minimal range. This implies that each crossing point of the fiber set is drawn onto the 2D plane even if this point does not occur in 3D. On the other hand, fiber tubes in 2D would be drawn as circles depending on the radius they can overlap which leads to new challenges. However, so far no other platform or solution has been able to offer 2D visualization on such a level of interaction and performance. Especially for radiologists, to our knowledge, workflows with 2D visualization is a major part of their clinical routine. Therefore the provided 2D representation in combination with the interaction tools increased both usability and accuracy of fiber processing to an efficient and practical level.

Coloring of fibers

Colors are a powerful instrument to gain better understanding of the illustrated output and results. Common color codings such as orientation based and individual coloring as well as mapping scalar values to colors, e.g. FA maps were implemented into the framework. For better comprehension of fiber paths, orientation based color coding was applied to fiber structures by default. This provided a better overview of fiber structures in 3D. Furthermore single fibers could also be determined effectively due to mixing colors in areas of fiber crossings. For FA based color coding, opposing arguments from radiologists and scientists were encountered. Scientists, especially tractography developers supported this coloring since this mode was applied to evaluate diffusion parameters. Also unusual sharp edged fibers could be analyzed immedi-

ately based on colors indicating high and low FA. In contrast, radiologists preferred more the combination of orientation based coloring and FA as opacity levels, thus unifying advantages of both options. Since coloring can be individual for different tasks, the framework was designed to allow easy implementation of individual modes within a few steps.

5.3 Fiber interaction and processing

The basic idea of the fiber processing concept was to provide intuitive tools with accurate and interactive characteristics. Other solutions require segmentation images or 3D objects to perform fiber selection and extraction, whereas in the introduced fiber framework shape types such as circles and polygons can be drawn into 2D views to select fibers for extraction or further processing. Such selection tools are also available in many mainstream photo and image processing software kits. Therefore, no time consuming training for applying selections and ROIs are needed. ROI figures allow permanent interaction, e.g. modifications and adjustments, as well as methods for storing and loading are supported to raise the level of reusability. Further, to keep the context between 2D and 3D consistent, each ROI is immediately rendered and updated in 3D as well as in the remaining 2D render windows. From the user perspective, this feature improves flexibility and accuracy by providing methods to monitor and fine tune selections in real time. Fiber extraction could be performed with a few mouse clicks by loading the fiber data, drawing a ROI in 2D and executing the extraction mechanism. A concept for ROI interaction directly in 3D was not developed during this work. This would have required the support and evaluation of several input mechanisms for 3D navigation. The human brain contains a complex web of fiber structures, hence the fiber processing tools were also designed for complex tasks. Extracting fibers that run through several areas of the brain are achieved by combining ROIs to boolean expressions such as *AND, OR* and *NOT* operations. However, when using the ITK based fiber data structure for fiber extraction, the computation of the resulting output was highly inefficient. Due to complexity issues such calculations took up to 15 minutes or even more on huge data sets. Instead of iterating each point and each fiber, efforts to improve the performance were taken, which reduced the computation steps. VTK already provides methods to master this task in a more efficient way. In addition to the performance boost experienced within the visualization workflow, FBX also shows high performance for fiber extraction tasks. Tasks which needed couples of minutes could now be accomplished within few seconds since VTK provides optimized filters calculating intersection points between lines stored in `vtkPolyData` and the plane of the ROI. Information about which crossing point is related to which fiber is provided by the output of the filter. This method as well as the index based lookup mechanisms for crossing point processing were applied to identify fibers inside a ROI efficiently. Protocols to measure the error rate for extraction tasks were applied to ensure that the optimization steps did not cause any unwanted side effects.

5.4 Comparison of performance to previously available fiber processing frameworks

The tested applications for fiber processing were introduced in chapter two. With respect to the defined requirements for fiber processing in practice, e.g. in radiology, visualization, interaction and fiber processing features were missing. Some applications were able to visualize normal images in 2D but not fibers. Only MedINRIA was able to visualize fibers in 2D but the representation was not consistent with the fibers represented in 3D. Additionally their performance in 2D was not judder free with larger datasets (e.g. FS_B containing approx. 36,000 fibers). In contrast, MITK equipped with the fiber processing framework developed during the course of this work, is capable of fast, judder free and consistent 2D fiber visualization, even with large data sets. Features such as thick slab for fibers which are unavailable in MedINRIA have been incorporated into MITK. Accurate methods to interact with fibers in 2D are not provided by others. Instead, 3D objects like spheres and cubes are used to select fibers. This is not practicable in a radiology setup where radiologists are used to drawing segmentations intuitively onto 2D slices. Whereas tools for selection and extraction of fibers using circles or polygons which can be drawn onto the 2D views can now be realized in MITK.

CHAPTER 6

Conclusion

This thesis makes an important contribution to porting tractography into practice. It introduces a framework for fiber processing that is capable of dealing with tasks currently in demand in this area. In combination with previously existing interaction mechanisms of MITK, common tasks in this field can be accomplished efficiently. Different imaging modalities in combination with diffusion imaging data, explicit tractography information and a consistent context between 2D and 3D representation lead to improved or even new applications in this area.

Accurate investigation and processing of results from different tractography algorithms including FACT, stochastic tracking and Gibbs tracking can now be performed within the same framework. Instead of implementing a previously existing tractography algorithm in the framework, fibers generated by any desired tractography method from an external software application can simply be imported into the framework. This is possible, since usually external software applications support a standard `vtkPolyData` export mechanism. Fibers from the FACT algorithm were imported because MITK currently implements only stochastic tracking and Gibbs tracking. This versatility and adaptability allows the comparison of selected structures of the brain from different tractography approaches. Additionally, to evaluate different tractography algorithms, the influence of varying DWI sequences on fiber tracking can also be visually evaluated. Tools of the fiber framework including fiber extraction using ROI selection, applying color coding as well as MITK-wide interaction mechanisms such as zooming, panning, rotation in 3D and swiveling of 2D planes were successfully applied for each tracking approach. The tools provided simple and interactive interfaces to accomplish both accurate and complex tasks in an intuitive manner. Users with MITK experience were able to manage the fiber framework in a short time, hence the same workflow patterns were adopted. However, since usability and intuitive designs are subjective parameters, evaluation of the platform needs to be done on a larger scale to get a clearer picture.

A few of the limitations of this fiber framework are the automatic and misleading change of colors in 2D when shading effects are applied, constraints in 3D visualization when rendering tubes of large data sets, and unavailability of the option to select fibers using spherical ROIs as opposed to other frameworks.

Mechanisms for fiber visualization and fiber processing described in this work were used at the DTI-tractography challenge MICCAI 2011 and to generate images for the DTI-Atlas. Positive results show that the principles of this fiber framework may have high potential in the research field. In the last 10 years, MITK has garnered acceptance in the research field as well as in industry and clinical practice. Therefore such an effective technique for interactive fiber processing implemented into a popular platform such as MITK would augment its translational relevance and its application in a clinical field. From a different point of view, mobile applications have gathered a growing interest in medicine in recent years. Therefore, to maximize the target group, the fiber framework can also be ported to mobile devices. Although a prediction of the degree of acceptance and clinical ramifications can not be given at this juncture, there seems to be an appreciable potential for tractography applications in the clinical field. This can be attributed to the successful collaborations of past decades between radiologists and computer scientists, ultimately benefitting patient care.

Bibliography

[1] P. C. Lauterbur, "Nmr zeugmatographic imaging in medicine.," *J Med Syst*, vol. 6, pp. 591–597, Dec 1982.

[2] D. L. Bihan, E. Breton, D. Lallemand, P. Grenier, E. Cabanis, and M. Laval-Jeantet, "Mr imaging of intravoxel incoherent motions: application to diffusion and perfusion in neurologic disorders.," *Radiology*, vol. 161, pp. 401–407, Nov 1986.

[3] R. Brown, "A brief account of microscopical observations made in the months of june, july, and august, 1827, on the particles contained in the pollen of plants; and on the general existence of active molecules in organic and inorganic bodies," *Edinburgh new Philosophical Journal*, vol. July-September, pp. 358–371, 1828. Additional remarks on active molecules (Brown, 1829) is also included.

[4] H. C. Berg, *Random Walks in Biology*. Princeton University Press, revised ed., sep 1993.

[5] B. Bodini and O. Ciccarelli, *Diffusion MRI in neurological disorders*. Elsevier Academic Press, 2009.

[6] K. H. Fritzsche, *Quantification of Structural Changes in the Brain using Magnetic Resonance Imaging*. PhD thesis, Ruprecht - Karls - Universität Heidelberg, July 2010.

[7] K. H. Fritzsche, A. von Wangenheim, D. D. Abdale, and H.-P. Meinzer, "A computational method for the estimation of atrophic changes in alzheimer's disease and mild cognitive impairment," *Computerized Medical Imaging and Graphics*, vol. 32, pp. 294–303, 2008.

[8] Y. Liu, G. Spulber, K. K. Lehtimäki, M. Könönen, I. Hallikainen, H. Gröhn, M. Kivipelto, M. Hallikainen, R. Vanninen, and H. Soininen, "Diffusion tensor imaging and tract-based spatial statistics in alzheimer's disease and mild cognitive impairment.," *Neurobiol Aging*, Nov 2009.

[9] N. H. Stricker, B. C. Schweinsburg, L. Delano-Wood, C. E. Wierenga, K. J. Bangen, K. Y. Haaland, L. R. Frank, D. P. Salmon, and M. W. Bondi, "Decreased white matter integrity in late-myelinating fiber pathways in alzheimer's disease supports retrogenesis," *NeuroImage*, vol. 45, pp. 10–16, 2009.

[10] B. Bosch, E. M. Arenaza-Urquijo, L. Rami, R. Sala-Llonch, C. Junqué, C. Solé-Padullés, C. Peña-Gómez, N. Bargalló, J. L. Molinuevo, and D. Bartrés-Faz, "Multiple dti index analysis in normal aging, amnestic mci and ad. relationship with neuropsychological performance.," *Neurobiol Aging*, Apr 2010.

[11] K. H. Fritzsche, B. Stieltjes, S. Schlindwein, T. van Bruggen, M. Essig, and H.-P. Meinzer, "Automated mr morphometry to predict alzheimer's disease in mild cognitive impairment," *Int J Comput Assist Radiol Surg*, vol. 5, pp. 623–632, 2010.

[12] P. Belli, M. Costantini, C. Ierardi, E. Bufi, D. Amato, A. Mule', L. Nardone, D. Terribile, and L. Bonomo, "Diffusion-weighted imaging in evaluating the response to neoadjuvant breast cancer treatment.," *Breast J*, vol. 17, no. 6, pp. 610–619, 2011.

[13] D. C. Colvin, M. E. Loveless, M. D. Does, Z. Yue, T. E. Yankeelov, and J. C. Gore, "Earlier detection of tumor treatment response using magnetic resonance diffusion imaging with oscillating gradients.," *Magn Reson Imaging*, vol. 29, pp. 315–323, Apr 2011.

[14] A. Afaq and O. Akin, "Imaging assessment of tumor response: past, present and future.," *Future Oncol*, vol. 7, pp. 669–677, May 2011.

[15] H. Einarsdóttir, M. Karlsson, J. Wejde, and H. C. F. Bauer, "Diffusion-weighted mri of soft tissue tumours.," *Eur Radiol*, vol. 14, pp. 959–963, Jun 2004.

[16] V. J. Wedeen, D. L. Rosene, R. Wang, G. Dai, F. Mortazavi, P. Hagmann, J. H. Kaas, and W.-Y. I. Tseng, "The geometric structure of the brain fiber pathways," *Science*, vol. 335, no. 6076, pp. 1628–1634, 2012.

[17] S. M. Smith, M. Jenkinson, H. Johansen-Berg, D. Rueckert, T. E. Nichols, C. E. Mackay, K. E. Watkins, O. Ciccarelli, M. Z. Cader, P. M. Matthews, and T. E. Behrens, "Tract-based spatial statistics: Voxelwise analysis of multi-subject diffusion data," *NeuroImage*, vol. 31, no. 4, pp. 1487 – 1505, 2006.

[18] P. J. Basser, J. Matteiello, and D. LeBihan, "Estimation of the effective self-diffusion tensor from the nmr spin echo," *J Magn Reson*, vol. 103, pp. 247–254, 1994.

[19] J. Crank, *The mathematics of diffusion*. Clarendon Press, Oxford, 1956.

[20] P. J. Basser and C. Pierpaoli, "A simplified method to measure the diffusion tensor from seven mr images.," *Magn Reson Med*, vol. 39, pp. 928–934, Jun 1998.

[21] M. Perrin, C. Poupon, B. Rieul, P. Leroux, A. Constantinesco, J.-F. Mangin, and D. Lebihan, "Validation of q-ball imaging with a diffusion fibre-crossing phantom on a clinical scanner.," *Philos Trans R Soc Lond B Biol Sci*, vol. 360, pp. 881–891, May 2005.

[22] Y. Cohen and Y. Assaf, "High b-value q-space analyzed diffusion-weighted mrs and mri in neuronal tissues - a technical review.," *NMR Biomed*, vol. 15, no. 7-8, pp. 516–542, 2002.

[23] P. T. Callaghan, A. Coy, D. Macgowan, K. J. Packer, and F. O. Zelaya, "Diffraction-like effects in NMR diffusion studies of fluids in porous solids," *Nature*, vol. 351, pp. 467–469, 1991.

[24] R. Bammer, B. Acar, and M. E. Moseley, "In vivo mr tractography using diffusion imaging.," *Eur J Radiol*, vol. 45, pp. 223–234, Mar 2003.

[25] A. Cherubini, G. Luccichenti, F. Fasano, P. Péran, G. E. Hagberg, E. Giugni, and U. Sabatini, "Imaging nervous pathways with mr tractography.," *Radiol Med*, vol. 111, pp. 268–283, Mar 2006.

[26] H. Johansen-Berg and T. E. Behrens, *Diffusion MRI: From quantitative measurement to in-vivo neuroanatomy*. Academic Press, first edition ed., 2009.

[27] P. Hagmann, L. Jonasson, P. Maeder, J.-P. Thiran, V. J. Wedeen, and R. Meuli, "Understanding diffusion mr imaging techniques: From scalar diffusion-weighted imaging to diffusion tensor imaging and beyond1," *Radiographics*, vol. 26, no. suppl 1, pp. S205–S223, October 2006.

[28] D. K. Jones and M. Cercignani, "Twenty-five pitfalls in the analysis of diffusion mri data," *NMR in Biomedicine*, vol. 23, no. 7, pp. 803–820, 2010.

[29] H. Johansen-Berg and T. E. J. Behrens, "Just pretty pictures? what diffusion tractography can add in clinical neuroscience.," *Curr Opin Neurol*, vol. 19, pp. 379–385, Aug 2006.

[30] P. J. Basser, S. Pajevic, C. Pierpaoli, J. Duda, and A. Aldroubi, "In vivo fiber tractography using dt-mri data.," *Magn Reson Med*, vol. 44, pp. 625–632, Oct 2000.

[31] M. Lazar, D. M. Weinstein, J. S. Tsuruda, K. M. Hasan, K. Arfanakis, M. E. Meyerand, B. Badie, H. A. Rowley, V. Haughton, A. Field, and A. L. Alexander, "White matter tractography using diffusion tensor deflection.," *Hum Brain Mapp*, vol. 18, pp. 306–321, Apr 2003.

[32] C. Gössl, L. Fahrmeir, B. Pütz, L. M. Auer, and D. P. Auer, "Fiber tracking from dti using linear state space models: detectability of the pyramidal tract.," *Neuroimage*, vol. 16, pp. 378–388, Jun 2002.

[33] G. J. M. Parker, C. A. M. Wheeler-Kingshott, and G. J. Barker, "Estimating distributed anatomical connectivity using fast marching methods and diffusion tensor imaging.," *IEEE Trans Med Imaging*, vol. 21, pp. 505–512, May 2002.

[34] E. Pichon, C.-F. Westin, and A. R. Tannenbaum, "A hamilton-jacobi-bellman approach to high angular resolution diffusion tractography.," *Med Image Comput Comput Assist Interv*, vol. 8, no. Pt 1, pp. 180–187, 2005.

[35] M. Reisert, I. Mader, C. Anastasopoulos, M. Weigel, S. Schnell, and V. Kiselev, "Global fiber reconstruction becomes practical.," *Neuroimage*, vol. 54, pp. 955–962, Jan 2011.

[36] W. Chen, Z. Ding, S. Zhang, A. MacKay-Brandt, S. Correia, H. Qu, J. A. Crow, D. F. Tate, Z. Yan, and Q. Peng, "A novel interface for interactive exploration of dti fibers.," *IEEE Trans Vis Comput Graph*, vol. 15, no. 6, pp. 1433–1440, 2009.

[37] I. Wolf, M. Nolden, T. Böttger, I. Wegner, M. Schöbinger, M. Hastenteufel, T. Heimann, H.-P. Meinzer, and M. Vetter, "The mitk approach," in *8th International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI) - Workshop on Open-Source Software*, (Palm Springs, USA), 2005.

[38] T. M. Ngo, P. Golland, and T. M. Ngo, "A stochastic tractography system and applications," 2007.

[39] slicer.org, "Slicer, visualization and medical image computing." [Online]. Available: http://www.slicer.org. [Accessed: May 20, 2011], 1998.

[40] tcl.tk, "Tcl programming language and the tk graphical user interface toolkit." [Online]. Available: http://www.tcl.tk. [Accessed: March 12, 2012].

[41] J. Zhang, H. Ji, N. Kang, and N. Cao, "Fiber tractography in diffusion tensor magnetic resonance imaging: A survey and beyond," 2005.

[42] N. Toussaint, J. Souplet, and P. Fillard, "Medinria: Medical image navigation and research tool by inria," in *Proc. of MICCAI'07 Workshop on Interaction in medical image analysis and visualization*, (Brisbane, Australia), 2007.

[43] S. Mori, B. J. Crain, V. P. Chacko, and P. C. van Zijl, "Three-dimensional tracking of axonal projections in the brain by magnetic resonance imaging.," *Ann Neurol*, vol. 45, pp. 265–269, Feb 1999.

[44] P. Fillard and G. Gerig, "Analysis tool for diffusion tensor mri," in *In Proc. of Medical Image Computing and Computer-Assisted Intervention*, pp. 967–968, Springer, 2003.

[45] MeVisLab.de, "Mevislab, medical image processing and visualization." [Online]. Available: http://www.mevislab.de. [Accessed: May 20, 2011], 2002.

[46] M. Schluter, O. Konrad-Verse, H. K. Hahn, B. Stieltjes, J. Rexilius, and H.-O. Peitgen, "White matter lesion phantom for diffusion tensor data and its application to the assessment of fiber tracking," in *Proc. SPIE 5746, 835*, vol. 5746, 2005.

[47] mristudio.org, "Mristudio, image processing under windows." [Online]. Available: http://www.mristudio.org. [Accessed: May 20, 2011], 2007.

[48] H. Jiang, P. C. M. van Zijl, J. Kim, G. D. Pearlson, and S. Mori, "Dtistudio: resource program for diffusion tensor computation and fiber bundle tracking.," *Comput Methods Programs Biomed*, vol. 81, pp. 106–116, Feb 2006.

[49] P. A. Cook, Y. Bai, S. Nedjati-Gilani, K. K. Seunarine, M. G. Hall, G. J. M. Parker, and D. C. Alexander, *Camino : Open-Source Diffusion-MRI Reconstruction and Processing*, vol. 14, pp. 22858–22858. 2006.

[50] A. Leemans, B. Jeurissen, J. Sijbers, and D. Jones, *ExploreDTI: a graphical toolbox for processing, analyzing, and visualizing diffusion MR data*, vol. 245, p. 3537. 2009.

[51] R. Wang, T. Benner, A. G. Sorensen, and V. J. Wedeen, *Diffusion Toolkit: A Software Package for Diffusion Imaging Data Processing and Tractography*, p. 3720. International Society for Magnetic Resonance in Medicine(ISMRM), 2007.

[52] F. Calamante, J.-D. Tournier, G. D. Jackson, and A. Connelly, "Track-density imaging (tdi): super-resolution white matter imaging using whole-brain track-density mapping.," *Neuroimage*, vol. 53, pp. 1233–1243, Dec 2010.

[53] T. H. J. M. Peeters and A. Vilanova, "Visualization of DTI fibers using hair-rendering techniques,"

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

[55] A. Köhn, J. Klein, F. Weiler, and H.-O. Peitgen, "A gpu-based fiber tracking framework using geometry shaders," *Proceedings of SPIE*, vol. 7261, no. 1, pp. 72611J–72611J–10, 2009.

[56] N. Amenta, S. Levy, T. Munzner, and M. Phillips, "Geomview: a system for geometric visualization," in *SCG '95: Proceedings of the eleventh annual symposium on Computational geometry*, (New York, NY, USA), pp. 412–413, ACM Press, 1995.

[57] A. Henderson, *The ParaView Guide: A Parallel Visualization Application*. Kitware, Nov. 2004.

[58] L. Ibanez, W. Schroeder, L. Ng, and J. Cates, *The ITK Software Guide*. Kitware, Inc. ISBN 1-930934-15-7, http://www.itk.org/ItkSoftwareGuide.pdf, second ed., 2005.

[59] M. Malaterre and al., *GDCM Reference Manual*. http://gdcm.sourceforge.net/gdcm.pdf, first ed., 2008.

[60] libpng.org, "An open, extensible image format with lossless compression." [Online]. Available: http://www.libpng.org. [Accessed: March 12, 2012].

[61] I. Kitware, *VTK User's Guide Version 5*. Kitware, Inc, 2006.

[62] Opengl, D. Shreiner, M. Woo, J. Neider, and T. Davis, *OpenGL(R) Programming Guide : The Official Guide to Learning OpenGL(R), Version 2 (5th Edition)*. Addison-Wesley Professional, Aug. 2005.

[63] J. Blanchette and M. Summerfield, *C++ GUI Programming with Qt 4 (2nd Edition) (Prentice Hall Open Source Software Development Series)*. Prentice Hall, 2 ed., 2 2008.

[64] commontk.org, "Common toolkit ctk." [Online]. Available: http://www.commontk.org. [Accessed: April 12, 2011], 2010.

[65] O. Alliance, "Osgi service platform release 4." [Online]. Available: http://www.osgi.org/Main/HomePage. [Accessed: April 12, 2011], 2007.

[66] S. Pujol, R. Kikinis, A. Golby, G. Gerig, M. Styner, W. Wells, C.-F. Westin, and S. Gouttard in *DTI Tractography for Neurosurgical Planning: A Grand Challenge*, MICCAI, September 2011.

[67] B. Stieltjes, R. M. Brunner, K. H. Fritzsche, and F. Laun, *Diffusion Tensor Imaging: Introduction and Atlas*. Springer, 2012.

[68] MITK.org, "The mitk style guide." [Online]. Available: http://docs.mitk.org/nightly-qt4/StylesAndNotesPage.html. [Accessed: March 12, 2012], 2005.

[69] B. Walter, "Fsl 4.1 manual," 2008.

[70] H. Axer, S. Beck, M. Axer, F. Schuchardt, J. Heepe, A. Flücken, M. Axer, A. Prescher, and O. W. Witte, "Microstructural analysis of human white matter architecture using polarized light imaging: views from neuroanatomy.," *Front Neuroinform*, vol. 5, p. 28, 2011.

[71] Y. Assaf, T. Blumenfeld-Katzir, Y. Yovel, and P. J. Basser, "Axcaliber: a method for measuring axon diameter distribution from diffusion mri.," *Magn Reson Med*, vol. 59, pp. 1347–1354, Jun 2008.