Visual Analysis of Multi-Field Data

Jorik Blaas

About the cover

The image on the cover depicts a tree with the subjects of my work and pet-projects. The words in the background were randomly picked from this thesis, and were placed using a fiber tracking algorithm in order to form the skeleton of the tree. Thanks to Dominique for the idea, and to Edwin in helping me with the execution.

FSC
Mixed Sources
Productgroep uit goed beheerde bossen, gecontroleerde bronnen en gerecycled materiaal.
Cert no. CU-COC-803902
www.fsc.org

Visual Analysis of Multi-Field Data

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof.ir. K.C.A.M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op donderdag 24 juni 2010 om 10:00 uur

door

Jorik BLAAS

informatica ingenieur
geboren te Amsterdam.

Dit proefschrift is goedgekeurd door de promotor:
Prof. dr. ir. F.W. Jansen

Copromotor:
Ir. F.H. Post

Samenstelling promotiecommissie:

Rector Magnificus	voorzitter
Prof.dr.ir. F.W. Jansen	Technische Universiteit Delft, promotor
Ir. F.H. Post	Technische Universiteit Delft, copromotor
Prof.dr.ir. M.J.T. Reinders	Technische Universiteit Delft
Prof.dr.ir. J.J. van Wijk	Technische Universiteit Eindhoven
Prof.dr. H. Hauser	University of Bergen, Norway
Prof.dr. A. Ynnerman	Linköpings Universitet, Sweden
Dr. C.P. Botha	Technische Universiteit Delft

Advanced School for Computing and Imaging

This work was carried out in the ASCI graduate school.
ASCI dissertation series number 198.

NWO
Netherlands Organisation for Scientific Research

This project was supported by the Netherlands Organization for Scientific Research (NWO) as project number 643.100.503 “Multi-Field Medical Visualization”.

ISBN 978-90-8559-042-2

Preface

The research described in this thesis was performed at the Computer Graphics and CAD/CAM research group of the Delft University of Technology. This research was part of a larger project, supported by the Netherlands Organisation for Scientific Research (NWO) within the I-Science Programme VIEW, section Interactive Data Visualization, Programme "Interactive, Multi-Field Data Visualization Techniques for Medical Applications".

Jorik Blaas

Acknowledgements

First of all, I'd like to thank my family, Paul, Lucie and Jasper, for having always supported me, and for making it possible to find my way in life.

This work wouldn't have come to fruition, without all the people involved in the Computer Graphics and CAD/CAM group. In particular I'd like to thank the following people.

Charl Botha, thesis advisor and close friend, with whom I spent a good number of evenings chasing deadlines and an equally good number not chasing them.

Frits Post, my co-promotor, for always being interested in new ideas, and being able to relate them to existing ones.

Erik Jansen, my promotor, for his quick and thorough review of my thesis, and for having me in his group.

Thanks to all these colleagues of the computer graphics group who are so much more than colleagues: Stef Busking, Gerwin de Haan, Peter Krekel, Lingxiao Zhao, Eric Griffith, François Malan, Xi Zhang, Peter Kok, Peter Schaafsma, Rik Sonderkamp, Rafa Bidarra, Tim Tutenel and Ruben Smelik.

Our dynamic tech support duo Bart and Ruud, who kept all our hardware and software running smoothly, and Toos for keeping the group connected.

This work was part of a larger collaboration with TU Eindhoven, through which I have met many bright and enthusiastic researchers: Anna Vilanova, Vesna Prčkovska, Tim Peeters, Paulo Rodrigues, Danny Holten and Ralph Brecheisen.

I would like to thank some of the people I have met throughout my PhD in particular. Some of them I met at conferences, while others just happened to show up at the right place at the right time.

Gordon Kindlmann, for always being interested and enthusiastic about new ideas, and for providing a solid basis on which I could build my DTI work.

At the SDSC, Amit Chourasia, all-round friendly guy, and yielder of unimaginable computer power.

At material science, Joris Dik, for always coming up with an novel angle to attack timeless problems.

Jelle Feringa, for being one of the very few people who use a computer exactly how it is supposed to be used, and being genuinely happy about that.

My colleagues from Swansea, Bob Laramee, Ed Grundy and Mark Jones, for a fun and fruitful collaboration on a subject I never imagined I would work on.

Bart Peters and Matthan Caan of the AMC, for being my ever-helpful link to clinical practice in DTI research.

Paul de Bruin, for MAME, LaTeX, MetaPost, good conversations and excellent cooking.

I would like to thank all of my friends for advice, relaxation and many animated discussions. In particular: Niels, Tijs, Tristan, Edwin, Ivo, Jan David, Henrik, Huibert-Jan, Renze, Marijn, Sicco, Steve, Kuno, Nuey, Wouter, and all of those I failed to mention.

Special thanks to Phill, for putting exceptional ideas to work.

Above all thanks to my girlfriend, Dominique, who has supported me wondrously and without whom this planet would be far less wonderful.

Contents

Introduction

1

1.1 Preliminaries

One of the most impressive feats of the human brain is the visual system. Even in extreme conditions we can recognize objects in a fraction of a second. The goal of visualization is to exploit this capability to its fullest extent in order to turn raw data into insight.

The ability of the user to comprehend and interpret the data depends fundamentally on the visual representation chosen. The visual interpretation of such representations becomes particularly demanding when the number of data points is large, or when the data is of high complexity. In these cases a static representation of the underlying data is not effective. To cope with this, the user can **explore** the data interactively by manipulating the visualizations.

One of the most important aspects of interactive exploration is the concept of **selection**. By reducing the visualization to the objects of interest, the overall visual complexity is reduced. Focus and context are particularly effective at visualizing a subset of the data within the context of the general structure.

A key factor to an effective **interaction** experience is low latency and high interaction speed. If a system is too slow in responding to the user's requests, it both strains the user's attention span as well as decreases the coupling between an action and the result. Especially when coping with huge datasets, the choice of algorithms and datastructures can make or break system performance.

One of the most important questions that has to be answered in order to make an effective visualization is what to visualize, and how to visualize it.

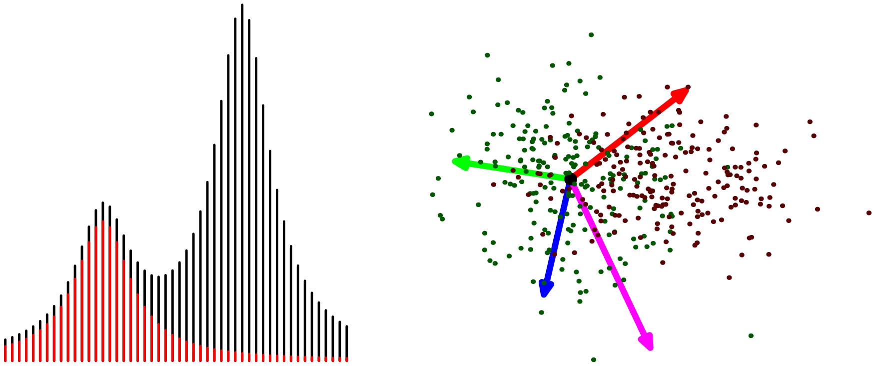
In the medical field, data complexity has been growing consistently over the past few years. Not only does the size of the data grow, but so does the need to visualize beyond traditional boundaries. While we have very effective visualization methods for a single patient with a single scalar data volume, questions from the researchers interpreting their corpus of data are always one step ahead of what we can do. The true challenges lie in visualizing multiple scans, at multiple points in time, and even across multiple patients and multiple modalities, as punctuated by Munzner [Mun09b].

The need for visualization of multi-variate data has spawned a number of issues that need to be resolved. Data consisting of single scalars can often be represented intuitively, but as both data complexity and size grow, this is no longer straightforward. The visualization of multiple scalar fields is not just solved by blending colors together, and the visualization of vector fields encompasses a large body of specialized techniques. More complex data is often reduced to a simpler form, and then visualized, but information is lost in the process. Each type of data has its own set of visual representations that work well for a specific task, and mix-and-matching representations is highly non-trivial. There are two situations in which a simple direct visualization is no longer an option, and data reduction becomes necessary.

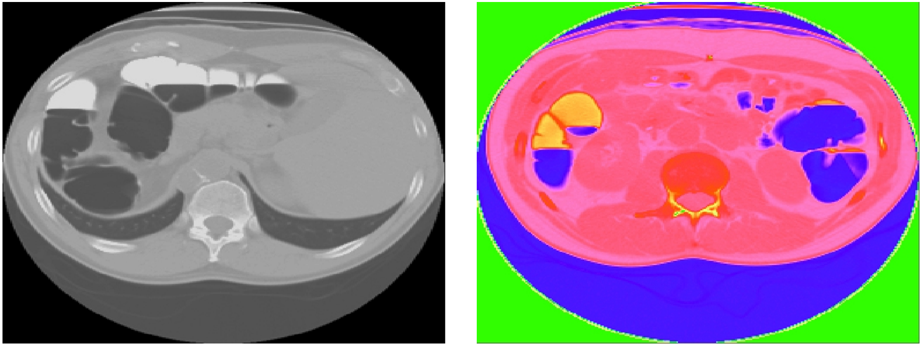
The first type of data reduction lies in the reduction of the number of dimensions of the samples. This often entails the reduction of high-dimensional samples to scalar values by means of projection. For example, in spectrometry the measured spectra are reduced to elemental scalars indicating the presence of known elements. Once the data is reduced to a manageable number of scalars, more classic techniques can be used to create a meaningful visualization. In the case of Diffusion Tensor Imaging, measurements can be interpreted as values in a tensor domain. In this case dimensional reduction can be as simple as a metric that describes the shape of the tensor. The way in which these reductions are performed is often domain-specific, but as we demonstrate, there are also techniques from pattern recognition and machine learning that perform these reductions in a domain-independent way.

The second case is the reduction of the number of samples, or data points. When dealing with datasets that have tens of millions of points, we are often not particularly interested in visualizing each individual sample. We are far more interested in the distribution of the samples and in finding statistical clusters and outliers. If we can identify a large group of points with similar values, we can reduce them to a single entity, consequently reducing the number of points many-fold.

Both forms of reductions are mostly independent of the spatial location of the samples. The entities in the reduced spaces often no longer have an inherent spatial component. To maintain context when exploring these reduced domains, we also



Histogram Scatter plot



Slice-Viewer Color Slice-Viewer

Figure 1.1: Analysis of CT colonoscopy data using a multiple linked view system (see Chapter 4). The four components used are (from top-left to bottom-right): histogram, scatter plot, slice-viewer and pseudo-color slice-viewer. For color plate, see Figure A.1 on page 178.

need to present a spatial context. When we visualize the values separate from the spatial location, we need to maintain a form of linkage between the different representations. This coupling can be maintained by the use of multiple linked view interaction, as illustrated by Roberts [Rob07].

An example of a multiple linked view system is shown in Figure 1.1. An important step in virtual colonoscopy is the segmentation of the colon wall surface. To aid this segmentation two spatial views (bottom) are complemented by two feature space views (top). To make the correspondence between the different visual representations intuitively clear, selections are shared between all views (for more details, refer to Chapter 4). Spatial views are used to make a localized selection, while feature views are used to select voxels based on their values.

The use of multiple representations at the same time, with linked views, is a powerful basis for exploring data. For example, in diffusion tensor imaging, full brain tractographies contain a wealth of information, but the set of fibers that we actually want to see may be only a very small subset of the entire dataset (see Chapter 2). Filtering these fibers can be done using spatial queries, but also using queries that relate to the maximum fMRI activation along a tract. In this scenario, we not only have the points that make up the fiber, in which each has a tensor valued measurement and several scalar measurements, but also the fibers themselves, which connect an arbitrary number of points together into a curve. For clinical questions, sometimes we only want to visualize the fibers that run through a specific area, followed by a filtering step where we only show the fibers that go through one specific activation area (see Chapter 3).

The wide range of parameter settings that are used for the visualization makes it difficult to pick the right values. Transfer functions, for example, where careful tuning of the transfer function can not only shift the focus of the visualization, but even make or break its effectiveness. Just as the transfer function maps a scalar value to a color, higher dimensional data points can be mapped to lower dimensional ones. These projections from high to low dimensional spaces have a huge number of variants, each with their own parameters.

This wealth of parameters yields an infinite number of possible views of the data, which makes finding the right views of the data no easy task for the user. We can aid this process by assisting the user in finding viewing parameters by using pattern recognition techniques.

This is where multiple linked views can move beyond just linking selections in different views. Projection parameters can be determined based on the current selection, dynamically adapting the parameters to the values found in the current selection set. This selection-specific projection yields a new view of the data which complements the existing views. The linkage between the multiple coordinated views makes sure the correspondence between the samples is never lost.

Using these dynamically generated views, the user can focus on a parameter set or projection that maximizes the contrast in the current selection set, or it can be used to maximally differentiate the samples found in two selected spatial areas.

Once such a projection is found, derived metrics can be introduced on the samples, such as the distance in the projected space. These derived metrics are then a source for further processing. For example, a ready-made selection can be extended to include all points that share similar measurements by using a distance metric in the projected space. When multiple different regions are required, a nearest-neighbor or other type of trained classifier can be used.

The aforementioned techniques form a basis to transform the problem of exploring multi-variate data into a problem of finding a set of independent representations and exploring these with a suitable interaction method.

1.2 Context of This Work

1.2.1 What is multi-variate data

In essence, *multi-variate* means there are multiple variables or attributes for each data point. Some data sources produce inherently multi-variate data, such as color photography, where each point yields scalars for red, green and blue. Questionnaires are another classical source of multi-variate data. Each respondent to a questionnaire is a data point and each answer is represented as a single variable. Currently, modern medical imaging devices are also becoming a source of huge multi-variate datasets. Combined datasets from multiple MRI scans, Diffusion Tensor Imaging and CT scans can be interpreted as multi-modal multi-variate datasets. The fact that each voxel in the dataset is now a multi-variate measurement leads to an explosion in the size of data, as well as the visualization complexity, compared to classical data sources.

As data complexity increases, our ability to understand the data decreases. Lee et al. [LCG$^+$02] indicated that working with multi-variate datasets is becoming ever more important, especially now that we are moving outside of the domain of task-specific solutions and starting to focus more on reconfigurable high-level reusable components.

As noted by Bürger et al. [BH07], most of the automated data processing and analysis algorithms are only applicable once we know what to look for. In the overview that they provide on the state-of-the-art in visualization of multi-variate scientific data, the key element is a global move towards a mix-and-match type of pipeline, where relevant visual representations can be chosen and combined in a flexible way, moving further away from problem-specific solutions developed in the past. The presented structure helps to identify and categorize these techniques, so that each representation can be selected on its merits, leading to a well-combined visualization, tuned for the available data and the task at hand, as different tasks need different representations.

1.2.2 What multi-variate data can do, and why we can't see it

Each part of a multi-variate dataset contains its own information. In the case of multi-modal medical data, each imaging modality has its own focus and characteristics. For example, a CT scan captures bones, while an MRI can measure soft tissue, perfusion or even free water diffusion characteristics.

Showing all, or even a part of the aspects of a multi-variate dataset combined is not a simple task. For a single-scalar volume a natural representation might be to simply take the greyscale values, or to look at an isosurface. When we move to multi-variate data, the mapping from data to visual representation is often not as straightforward. Given even one of the simplest forms of multi-variate data, a dual-scalar volume, it becomes clear that there are a wealth of options on how to visually display both scalars.

For a 2-scalar volume, the obvious approach would be to map each scalar to a color channel (red and green). This approach, however, leads to visuals suffering from false colors, as the human visual system interprets the blended scalars as a set of red, green and yellow colors, making it difficult to interpret the image as two separate channels.

In color photography, the camera produces inherently multi-scalar data, usually in three channels: red, green and blue. When trying to perform a basic image processing task, such as separating a number of colored objects (see Figure 1.2 on the next page), it becomes immediately clear why we need to look at multi-scalar data as a whole. Even though each bean has a unique color, there is no straightforward projection that maps each unique color to a single scalar. When we examine the separate image channels, it becomes apparent that in each channel objects exist that have the same intensity within that channel, while they have a different color overall. Part of the data is thrown away in the projection, effectively reducing the number of discernible colors.

If we try to separate the objects by introducing an edge detection filter, the same observation holds. When edges are defined within a single scalar color channel, they can never capture anything more than the variations within that channel. Only when an edge is defined by the change in the multi-scalar space, taking into account the distance in the color space, do the edges between all the beans become apparent.

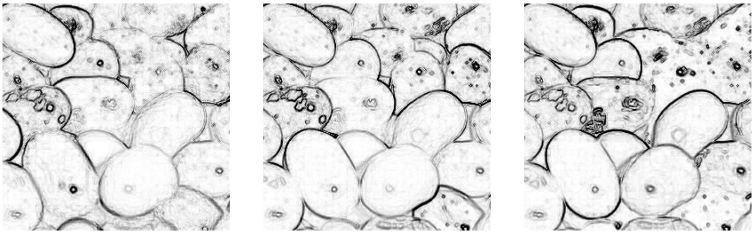
This demonstrates how derived features based on multiple scalars (the color channels) can produce single-scalar data (edge strength) that captures something that is invisible when looking only at a single scalar channel in the original.

1.2.3 Leaving the physical space

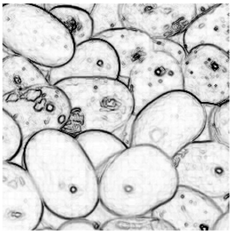
While medical data often has an inherent spatial component, the data itself may also be interpreted without considering its spatial location. The simplest example of this is a histogram. A histogram of an image can give us a lot of information about the

(a) Color image

(b) Red channel (c) Green channel (d) Blue channel



(e) Red edges (f) Green edges (g) Blue edges



(h) Combined edges

Figure 1.2: Edge detection on a color photograph (a) of jellybeans. When visualizing each channel (red, green and blue) independently (b,c,d), some of the beans have the exact same intensity. Looking at edges detected in the three images (e,f,g), it is clear that some of the edges are only present in a single color channel. When all color channels are used to compose an edge image (h), all beans are clearly delineated, even though we are visualizing the edge strength as a single scalar channel. For color plate, see Figure A.2 on page 179.

values that occur within the dataset. For multi-variate data we can generate joint-histograms of any pair of attributes, focusing on the correlation between the two values.

It is often useful to consider datapoints to be sampled within a specific domain, or space. Volumetric measurements are by definition sampled within a three dimensional domain. However, the measured values are also inside a domain. Often this is simply the scalar domain. By looking solely at the distribution of the datavalues, we can completely ignore the spatial domain, yet it can tell us oft-occuring values and help us detect outliers. In some cases, such as in Diffusion Tensor MRI, the measurements are tensors, resulting in a domain which is far less trivial to explore.

Often we are not interested in the spatial structure of our data, but only in the data values, or in the distribution thereof. In these cases, we can apply techniques from the field of information visualization. Usually these are not designed to cope with a huge number of data points, so the techniques have to be screened carefully for their possible application to datasets that are orders of magnitude larger. If we can succesfully combine these non-spatial techniques with the spatial techniques from scientific visualization, we can get a best-of-both-worlds scenario, where exploration of spatial locality of the data goes hand in hand with the exploration of the distribution of values.

This is tightly coupled with one of the key issues, as presented in the NIH/NSF Visualization Research Challenges Report [JMM$^+$05], which states: “Visualization tools and methods should provide tighter integration with other analytic tools and techniques, such as statistics, data mining and image processing, in order to facilitate analysis from both qualitative and quantitative perspectives.”

1.3 Themes of This Work

The global goal of our work is to develop methods and structures to explore large and complex datasets. The size and complexity of the datasets, both in the number of samples as well as in the dimensionality and structure of the data, lead to an interesting compromise between flexibility and speed. To cope with the complexity of the data, we need new views and new representations, we need to have different levels of abstraction, and we need algorithms to aid the user in the exploration.

There are four main themes to this work, which are detailed in the following sections. These themes are ordered in a top-down fashion, starting with abstract concepts and slowly progressing towards implementation details. The first theme is the handling of large data for exploratory visualization. After that, we describe the use of *feature spaces* as a key component of visual exploration. With these two subjects in the back of our mind, we progress towards the implementation by looking at *data fusion* and the issues that arise when combining multiple sources of data. The final subject we discuss is performance optimization to keep the visualization interactive.

1.3.1 Exploratory large data handling; Issues of scale

When datasets grow large, bottlenecks shift and a whole range of other difficulties arises. The issues of scale that arise can be roughly split into two parts: the technical part and the human part.

On the technical side, difficulties mostly arise in maintaining interactive update rates. All parts of the pipeline – in particular storage, processing and display – have to be designed in a way that allows for interactive updates. This is described in more detail in Section 1.3.4.

On the human side, the intrinsic scalability of visual representations is put to the test. A classical scatter plot works well for thousands of points, but when billions of points are to be drawn the graph quickly loses its effectiveness due to overdraw and clutter. In the case of scatter plots, modifications can be made to alleviate these issues, but other representations do not scale as well. An ellipsoid is an effective representation of (the eigensystem of) a single tensor, but an entire 512-cubed volume of such tensors is inherently cluttered.

This section briefly describes our approach when dealing with large datasets, aimed at maintaining interactivity with visual representations that scale.

Representations at several levels of abstraction

The human visual system is brilliant at focusing on a particular aspect of a picture, but this ability only scales well if the data is represented in a meaningful and sufficiently concise way. Part of the task of reducing the mental load is to choose a visualization at the right level of abstraction, so that the focus lies on the data relevant to the task. The most common example of filtering for visual representations is that of the transfer functions used in volume rendering, which focus on a certain set of materials or material transitions.

A first step towards reducing visual complexity is to filter the data either by removing parts of it, or by aggregating the data into higher level visual structures.

In Diffusion Tensor Imaging, the neural pathways in the brain are reconstructed from a tensor-valued field of diffusion measurements. These neural pathways are commonly visualized as curved lines which follow the local diffusion direction. While the generated curves correspond to meaningful physiological features and are of great use to a clinician, a part of the data is actually thrown away. The full shape of the tensor at each point is no longer visible, making it impossible to judge the occurrence of more complicated phenomena such as fiber crossings.

While the high level *aggregate* visualization provides the clinician with a good anatomical overview, it also obscures some information from the view. Therefore, the need for a combined visualization arises. Similar to the use of isosurfaces and direct volume renderings, a combined visualization exists for neurological pathways. In addition to the use of streamlines to display the fiber bundles, glyphs can be used only at points where that representation adds extra information not conveyed in the streamline.

The art of removing the right parts of the data rapidly becomes more difficult when the data consists of billions of points, each with multiple scalars or even tensors.

1.3.2 Feature space

Many intrinsic attributes of the data are not visible in a spatial context. These need views of a more abstract nature, which we call feature space views.

One of the most important visual representations of data in statistical data visualization is the scatter plot, as popularized by Cleveland [Cle93]. As we demonstrate, these techniques from information visualization also have their merit in medical visualization.

Feature space views

Terminology

A scatter plot visualizes the attributes (measured properties) of a number of entities. Each entity is represented by a point, and the choice of axes determines which attributes are used to position the point. We can see this in a *feature space* view, as only the measured values influence the visualization, while the spatial location of where the entity resides is ignored.

As an example, consider a map of the Netherlands. The map is split into a number of smaller areas, and each one of them has a number of attributes. For each area we know the average income, the number of inhabitants and surface area.

When we need to investigate the distribution of income over the Netherlands, a color-coded map probably suffices. We can color each area based on the average income attribute, making the spatial distribution of income visually clear.

However, if we are interested in how income relates to population density, it is simpler to move away from the physical space and use a feature space visualization.

In this case we can use a scatter plot with the axes *population density* and *average income*. This shows the correlations between the two attributes, at the cost of losing the spatial context.

In volumetric datasets, we can apply a similar methodology. Each voxel now corresponds to an entity, just like the areas on the map. Similar to the map, the voxel has a natural geometric location at which it is measured.

Linking and projections

Linkage between feature space views and spatial views is very important in order to localize the effects and to understand the data. We can link values to locations, and vice versa. For example, if we create a sorted table of the population densities for all areas, then the location of the top-10 areas can be visualized by selecting them in the table, and propagating the selection to the map, highlighting the matching areas.

When the number of attributes is small, the feature space is of low dimensionality, and selecting the right attributes for a scatter plot is fairly straightforward. For slightly larger numbers, a scatter plot matrix can be of assistance, but this does not scale well.

An important insight here is that the scatter plot is simply showing a meaningful projection of the high dimensional feature space. We do not necessarily have to choose one attribute for each axis, but can use more complex projections. An arbitrary linear combination of attributes can in many cases aid in finding clusters in high dimensional data. The difficulty lies in helping the user determine the right projection for the task at hand.

We experimented with the use of techniques borrowed from pattern recognition to assist in finding these projections. Since we have a set of dynamically linked views, selections in physical space can be transferred to selections in higher dimensional feature space. A great advantage of having the user build selections in physical space is that it is inherently clear which anatomical structures are being selected. Once an initial selection has been made, the projection of the feature space can be adjusted in such a way that the variance is maximized (PCP), or that the difference between two selections is maximized (LDA).

Aggregation

Continuing the metaphor of the map of the Netherlands, aggregation is a powerful tool for managing the size of data. Let us presume a hundred areas can be grouped together as a province. If we interpret the data on a provincial level, the number of data points is reduced tremendously. The most straightforward data source at provincial level is to take the data available at area level, and average each attribute. However, since many values are reduced to a single value, we can also apply statistical measures such as the variance and introduce these as an attribute at province level. A more complex class of attributes that we can introduce are not just based on the underlying attributes, but can be based on geometry, such as the total connectivity of the underlying areas or the length of the bordering region.

1.3.3 Fast and flexible data fusion

Multiple data sources are difficult to combine, as the working memory of the person examining the data is limited. This makes it cumbersome to transfer information mentally between one view and another. A common approach, that we have successfully adopted, is to transform the data into a common physical frame of reference so that measurements can be joined into a single view, reducing the mental load for the user. For some sources of data the frame of reference is intrinsically the same, while for others a registration step is necessary.

For example, for surgical planning purposes, an MRI T1 weighted scan is often made of the entire head, while a T2 weighted scan is only made of the brain. Each of these scans contains valuable information, but they are made in different frames of reference.

Getting the data into a common frame of reference

Resampling both scans into a common frame of reference is an oft-used solution, but has a few drawbacks. Firstly, information is lost in the resampling process. To combat this loss, the resulting volume has to be sampled at a resolution at least as high as the finest input. If the resolution and field of view of the scans differs greatly, this results in an unnecessary increase in the amount of data, as low-resolution data volumes are resampled to a higher resolution.

Secondly, not only is data interpolated when resampling, but a number of rendering algorithms require interpolation as well. This means the data is effectively interpolated twice, introducing errors in the final image. This can be combatted by choosing a sufficiently high resolution for the resampled volume at the cost of increased memory usage.

Resampling on the fly

As fusion of data is a very common operation in our project and the data sets we use are fairly large, we have adopted an approach that eliminates the need for volumetric resampling altogether.

Instead of resampling the volumes in a discrete fashion, we simply treat each volume as a continuous function that can be sampled at any given point, along with a transformation that positions it in the world. When the value at a point in world-space is required, we can always transform the coordinates back into object space and sample the corresponding point in the original.

For example, when a slice viewer intersects multiple volumes, a dense grid of points on the surface of the slice are sampled by linking them to the position in the original data volume. The interpolation is subsequently performed in the original space.

An added advantage is that moving and scaling volumes is effectively a free operation. Even during interactive viewing, moving a volume is as computationally expensive as moving the sliceplane itself.

This behavior is abstracted in the concept of a probe, which can retrieve a value from the underlying volume when given an arbitrary location.

A simple algebra for compositing multiple dense fields

Since each underlying dataset is abstracted as a probe, we now have the ability to combine fields together using a pseudo-probe class. This newly implemented probe simply contains a reference to two (or more) probes; when called to retrieve a value it will retrieve that of both probes at the specified point, perform a specified mathematical operation to combine the two values into one and return the resulting value.

These operations are flexible in the way that they can support all basic arithmetic operators, including multiplication, addition and division. More complex functions can be decomposed into a set of basic probes, allowing for a high degree of freedom. Operations such as color mapping can also be seen as a pseudo-probe, which simply has a reference to a source probe, whose resulting value it then returns through through a color map.

Another advantage of using a probe abstraction is that transformations in coordinate systems or even non-rigid transforms can be implemented within a pseudo-probe class by simply changing the location at which the underlying field is sampled.

Joining multiple visual representations

Once the transforms that combine the fields into a common frame have been established, values can be combined as described above, and additionally it allows for the mixing of datasets at a higher level. Instead of fusing data values at the continuous field level, we can also fuse at a geometric level.

From basic volume visualization we know that there are several representations to visualize (part of) a volumetric dataset. For solid boundaries an isosurface rendering can be performed, while in other cases a volume rendering based on a color and opacity transfer-function works best. The isosurface may be lit, so that the shape of the boundary surface is clearly delineated, while the volume rendering can be tuned to draw more attention to the density or distribution of the underlying materials.

Since each visual representation has its own merits, we can often gain insight from picking a good set of visual representations for the task at hand and joining them together into a single space.

As an example, in Diffusion Tensor Imaging, the underlying tensor fields can be visually represented in numerous ways. Each data sample can be represented by an ellipsoid or superquadric glyph [Kin04], but from a neurological point of view the structure of the fiber bundles (that cause the tensor to be oriented) is often more important than the shape of the local tensor. To visualize the bundles directly, a tractography is often used, consisting of a set of streamlines that locally follow the most likely fiber bundle directions. This type of indirect visualization often has a fairly high information content, as it was designed to yield visual cues that represent the underlying data in such a way that it connects well with the mental framework of the viewer by abstracting away the unnecessary details.

Fusion of multiple representations often gives the user more context. A simple addition such as a transparent isosurface of the cortical surface, or a transparent rendering of the skin can provide strong hints without distracting from the core of the visualization.

A visualization is never just the sum of its parts. When both the focus and the context is right, joined visualizations can give both a high level overview as well as a representation of the details, providing a better overview than just the components in separate views. On the other hand, clutter quickly becomes an issue when multiple representations are just thrown together.

Combining multiple representations can be a difficult task, but as demonstrated for the case of pre-operative planning of brain tumor resections (see Chapter 3), a well-chosen set of primitives proves to be a powerful tool.

1.3.4 Performance optimization

The issues that arise when scaling to larger datasets are not only a problem for humans, as discussed in Section 1.3.1. Larger datasets often require rethinking the way data is stored, both in memory as well as on disk. Algorithms that have a high constant overhead cost but a low asymptotic complexity suddenly become a viable choice when the data is orders of magnitude larger. Throughout our work, several approaches and techniques are applied to speed up interactive response times:

- Tuned data structures
- Pre-processing
- Optimized processing on the GPU

The selection of a good data structure and accompanying processing algorithms is a task which is often overlooked. Especially with scalability in mind, the algorithmic complexity can quickly warrant switching to a different algorithm when the dataset size grows large.

For additional gains in speed, we can apply cache optimization by making sure memory accesses have high coherency.

Linked to the choice of data-structures is the choice of a storage format. When dealing with large datasets in particular, it pays off to investigate IO bottlenecks and check whether they can be alleviated by using compression or data quantization. Specifically, when processing steps are not parametrized and when the resulting data is significantly smaller than the source data, it pays off to pre-compute the results in a pre-processing step.

In the face of IO bottlenecks, there is a constant trade-off between on-demand processing and pre-processing. Pre-processing has the advantage that expensive data handling steps are performed off-line and incur no extra latency to the user, while on-demand processing has the advantage of not taking up any extra storage space. Especially when the processing step is parametrized, a change in settings could invalidate the entire pre-processed dataset. In these cases a mixed approach, where on-demand processing is performed when a pre-computed version of the data is not available, using a disk cache is often a good middle road.

When using on-demand processing, parts of the processing can often be performed in idle time as the user often needs some time to interpret the presented data and determine the next action. In a similar fashion, incremental or streaming processing can greatly improve the interaction round-trip time, as feedback is provided to the user as soon as the first results are available.

While many operations that we encountered were IO-bound, some operations have a high computational complexity, such as fiber tracking. In these cases it may be worthwhile to offload part of the processing to the graphics card (or GPU). With the advance of accelerated rendering hardware in mainstream PC's, an initially largely unused computational resource has become available. In the previous years, an ever increasing number of applications and algorithms has been ported to function on the GPU, often providing a tenfold performance increase over classical techniques [LHG$^+$06]. Not all algorithms are suitable for mapping onto the architecture that a graphics card imposes, as they have to be parallelizable and preferably have a regular memory access structure.

Especially at the end of the visualization pipeline, in rendering, it is worthwhile to organize the data structures and shaders in a way that minimizes CPU to GPU transfers. If user interaction causes a large part of the data on the GPU to be invalidated, then it often makes sense to move that part of the processing pipeline to run on the GPU to reduce the overall interaction latency. As an example, in our interactive selection technique for neural fiber bundle selection (see Chapter 2), only the flags that determine whether a fiber is selected are updated to the GPU, while the geometry itself can stay stored in GPU memory unchanged. The reduction in the amount of data transferred alone nearly doubled the rendering speed.

1.4 Short Overview of the Chapters

The following sections contain summaries for the main chapters. For continuity, these summaries are reproduced in the chapter headings.

Chapter 2 : Fast and reproducible fiber bundle selection

The first chapter presents an approach to handling brain-scans in neuroimaging. A diffusion tensor map of the brain can be made by using a special MRI imaging sequence, which measures water diffusion in living neural tissue. These diffusion tensor maps have various representations, ranging from scalar reductions and volume renderings to fiber tractography maps. No single representation is best, and therefore we present a careful mix of filtered representations and interaction, which uses spatial box selection of the fibers as a basis for selecting anatomical structures in the tractography. We demonstrate that even for a large number of fibers these queries can be performed interactively, thanks to a fast spatial data structure.

Chapter 3 : Interactive visualization of fused fMRI and DTI

Chapter 3 presents an extension to the neuroimaging work, this time focusing in particular on pre-operative planning for the surgical removal of brain tumors. For this application, the main goal is to assess which operation path would minimize functional damage. To map functional damage, not only the tumor itself is segmented from functional scans, but specific motor areas of the brain are delineated through functional MRI. With knowledge of these areas, the tractography can be used to find tracts which are vital for the motor tasks, so that damaging these can be avoided during surgery.

Chapter 4 : Visualization of multi-field medical data

After experimenting with single-view applications for diffusion tensor imaging, we expanded our views to include feature space representations. In Chapter 4, we introduce the multiple linked view metaphor for looking at volumetric multi-scalar data. Each voxel is not only represented as a pixel in a slice view at its physical position, but it is also represented by a point in feature space, positioned by its tuple of measured scalars. We show how two-way interaction can be accomplished by sharing selections and projections between feature space and physical views. A selection can be used in combination with pattern recognition techniques to find projections of the feature space data, which in turn can be used for interactive brushing to generate new selections. Feature spaces generated by filtering of the volume data especially, such as LH-spaces, turn out to be extremely useful in interactively segmenting multi-material boundaries.

Chapter 5 : Visual exploration across patients, modalities and timepoints

While the previous chapter focused on the visualization of a single multi-scalar volumetric dataset, the same techniques can also be extended to function at a higher level, allowing the handling of far larger datasets. Chapter 5 describes the extension towards the study of groups of patients over time. We use a linked hierarchy of features and derived features over different domains. At the top of this hierarchy, a data-point corresponds with a single patient, while at the bottom of a data-point corresponds with a single voxel. The values accross these domains are represented as points in a feature space that is parametrized by values aggregated over the lower levels of the hierarchy. A strong coupling allows us to map points inside the domain of individual patients to voxels, and vice versa. Similar to the example describing the map of the Netherlands, values can be transferred from one domain to the other by aggregation or sampling. Since for each of the twenty subjects six full MRI scans were taken, special care had to be taken to allow loading all datasets at the same time. We implemented a disk-based on-demand caching approach to overcome the limitations of the computer's working memory.

Chapter 6 : Extensions of parallel coordinates

While the previous chapters mostly focus on using scatter plots and histograms as alternate representations, we soon found that other techniques from the field of information visualization could also be adapted to handle large spatio-temporal datasets.

Building on the concept of multiple linked different representations, we present a volumetric temporal multi-scalar dataset as a combination of physical and feature space views. The feature space is explored using a parallel coordinates plot with interactive selection, while the physical space is represented by slice views and a three dimensional isosurface view. Selections are shared between all views, as to tightly couple the measured values to their locations. Spatial context clearly helps to understand the spatial selections, and vice versa.

Chapter 7 : Smooth graphs

In Chapter 7, we apply the multiple linked view metaphor to a different type of problem, that of the interpretation of biological sensor data. Sensors attached to marine animals gathered acceleration, pressure, temperature and other characteristics over the course of a day at a rate of 20 Hertz. The data samples are subsequently labeled into multiple categories by means of a trained classifier. Since the interest of the biologists lies in interpreting the global behavioral patterns of animals, we developed a graph-centric exploration method that focuses on recurring sequences of states. We introduce a new smooth-graph technique, that complements state transition graphs with information on the temporal context in a visual way.

BioExplorer shows well how techniques from information visualization can be used to tackle the analysis of highly complex temporal multi-scalar data by means of linked *abstract* representations.

1.4.1 Paper contributions

Chapter 2 presents an interactive DTI fiber selection technique that is an order of magnitude faster than existing techniques, leading to highly-reproducible results.

Chapter 3 describes a balanced toolset for surgical planning which combines functional, structural and tractography data to provide an effective visualization. Chapter 4 presents a dynamic coordinated view approach that integrates pattern recognition and high-dimensional feature space visualization to interactively explore (medical) multi-field data. Chapter 5 details a framework and a demonstrator application for longitudinal medical studies, providing interactive exploration of the trends in the data over patients, modalities and timepoints simultaneously. Chapter 6 examines the scalability of parallel coordinates for the exploration of multi-field volumetric data, resulting in a highly interactive multiple linked view demonstrator. In Chapter 7, we design a visual method for representing these higher-order state transitions that visually disambiguates the higher-order transitions, providing an intuitive representation for state transition sequences.

The following table describes the area of contribution of the main chapters. The darker the squares, the more relation to the subject.

Theme	Ch. 2	Ch. 3	Ch. 4	Ch. 5	Ch. 6	Ch. 7
Data Fusion	▓	█	□	░	□	□
Aggregation	□	▒	▒	█	□	▓
Scalability	░	░	▒	█	█	▓
Complex-Valued Data	▓	█	▓	█	▒	□
Spatial	█	█	▒	▓	▓	□
Feature Space	□	□	█	█	▒	▒
Multi-Patient	□	□	░	█	□	□
Temporal	□	□	▒	█	█	▓
Exploratory	█	▓	█	█	█	█
Pattern Recognition	□	□	▓	█	□	▒
Image Processing	□	□	▓	█	□	□
Selection	█	█	▓	█	▓	█
Interactivity	█	█	█	▓	█	█

Fast and Reproducible Fiber Bundle Selection in DTI Visualization

2

Published as: Jorik Blaas, Charl P Botha, Bart Peters, Frans M Vos and Frits H Post, *Fast and Reproducible Fibre Bundle Selection in DTI Visualization*, in IEEE Visualization 2005: 8.

This chapter presents an approach to handling brain-scans in neuroimaging. A diffusion tensor map of the brain can be made by using a special MRI imaging sequence, which measures water diffusion in living neural tissue. These diffusion tensor maps have various representations, ranging from scalar reductions and volume renderings to fiber tractography maps. No single representation is best, and therefore we present a careful mix of filtered representations and interaction, which uses spatial box selection of the fibers as a basis for selecting anatomical structures in the tractography. We demonstrate that even for a large number of fibers these queries can be performed interactively, thanks to a fast spatial data structure. Clinical applications and results are published separately[1][2][3]

---

[1] Bart D Peters, Lieuwe de Haan, Nienke Dekker, Jorik Blaas, Hiske E Becker, Peter M Dingemans, Erik M Akkerman, Charles B Majoie, Therèse van Amelsvoort, Gerard J den Heeten, Don H Linszen, *White matter fibertracking in first-episode schizophrenia, schizoaffective patients and subjects at ultra-high risk of psychosis*, Neuropsychobiology 2008;58(1): 19-28.

[2] Bart D Peters, Peter M Dingemans, Nienke Dekker, Jorik Blaas, Erik M Akkerman, Therèse van Amelsvoort, Carhles B Majoie, Gerard J den Heeten, Don H Linszen, and Liewe de Haan, *White matter connectivity and psychosis in ultra-high-risk subjects: A diffusion tensor fiber tracking study*, Psychiatry Research: Neuroimaging, December 2009.

[3] Bart D Peters, Jorik Blaas, Liewe de Haan, *Diffusion Tensor Imaging in the Early Phase of Schizophrenia: What Have We Learned?*, accepted for publication in Journal of Psychiatric Research, May 2010.

Abstract

Diffusion Tensor Imaging (DTI) is an MRI-based technique for quantifying water diffusion in living tissue. In the white matter of the brain, water diffuses more rapidly along the neuronal axons than in the perpendicular direction. By exploiting this phenomenon, DTI can be used to determine trajectories of fiber bundles, or neuronal connections between regions, in the brain. The resulting bundles can be visualized.

However, the resulting visualizations can be complex and difficult to interpret. An effective approach is to pre-determine trajectories from a large number of positions throughout the white matter (full brain fiber tracking) and to offer facilities to aid the user in selecting fiber bundles of interest.

Two factors are crucial for the use and acceptance of this technique in clinical studies: Firstly, the selection of the bundles by brain experts should be interactive, supported by real-time visualization of the trajectories registered with anatomical MRI scans. Secondly, the fiber selections should be reproducible, so that different experts achieve the same results.

In this chapter we present a practical technique for the interactive selection of fiber-bundles using multiple convex objects that is an order of magnitude faster than similar techniques published earlier. We also present the results of a clinical study with ten subjects that show that our selection approach is highly reproducible for fractional anisotropy (FA) calculated over the selected fiber bundles.

2.1 Introduction

In recent years, MR diffusion tensor imaging (DTI) has emerged as a new technique for the quantification of water diffusion in living tissue [WMB$^+$99, WMM$^+$02]. Diffusion is commonly represented for every voxel by a second-order symmetric positive definite tensor. These tensors, derived from a sequence of diffusion-weighted scans, each sensitive to a different diffusion direction, describe the free motion of water molecules in tissue. This makes the technique very suitable for imaging fibrous tissue, as there are large differences in diffusion speed parallel with the fiber direction, and perpendicular to it. The anisotropy is reflected in the tensor data; the amount of anisotropy and the main directions of diffusion can be made explicit by eigen analysis of each tensor.

DTI is now widely used as a research tool for studying the micro-structure of white matter tissue in the brain. Examples of applications are studies to differentiate the global nerve structures between brains malformed by disorders and healthy brains, quantification of the strength and integrity of nerve tracts (e.g. for stroke patients), and studies to determine the connectivity structure between different parts of the brain.

Based on the diffusion tensors, the directional structure of the underlying tissue can be determined and visualized by tracing paths that follow white matter tracts. This operation is usually called fiber tracking, or white matter tractography in brain applications [MCCvZ99, BPP$^+$00]. From a number of seed points placed in a region of interest, large numbers of fibers can be tracked. The fibers generated can be used for visualization, but the resulting images are complex and difficult to interpret. An effective approach is to pre-determine trajectories from a large number of positions throughout the white matter (full brain fiber tracking) and to offer facilities to aid the user in selecting fiber bundles of interest connecting specified regions in the brain, or passing through a specified region.

Two factors are crucial for the use and acceptance of such a technique in clinical studies: Firstly, the selection of the bundles by brain experts should be interactive, supported by real-time visualization of the trajectories, registered with anatomical MRI scans. Secondly, the fiber selections should be reproducible, so that different experts achieve the same results.

Therefore, interactive techniques are needed that allow brain experts to select the anatomically correct fiber bundles. If a number of fibers can be collected that meet the same requirements, such as connecting the same regions or passing along a certain location, the selected set can be considered to correspond to a fiber bundle. If such fiber bundles can be extracted in a reproducible way by different expert operators, then they can be used for quantification of diffusion along specific bundles to compare their characteristics between different patients, or for one patient at different times.

In this chapter we describe a practical technique for fast interactive selection of fiber bundles with multiple convex objects that are defined by half-spaces. The selection technique is about an order of magnitude faster than a similar selection technique presented earlier [ASM$^+$04], and thus selection remains interactive for far larger numbers of fibers. In addition, we present a specific selection approach based on three selection boxes and then show the validity of this approach with a group of ten subjects and two operators. The result is that this approach yields a high inter-operator reproducibility for fractional anisotropy, or FA, calculated over the selected fiber bundles.

The rest of this chapter is structured as follows: Section 2.2 contains a summary of existing work on the visualization of DTI data. In section 2.3 we explain how to perform real-time fiber selection with multiple convex selection objects. We present the reproducibility study in section 2.4. In section 2.5, we give a brief overview of our DTI visualization software tool, called DTII. We present a summary and our conclusions and indicate possible avenues for future research in section 2.6.

2.2 Related Work

Many techniques have been developed for the visualization of diffusion tensor data. For example, each tensor can be visualized with various kinds of glyphs that reflect the properties of the tensors that they represent [WMB$^+$99, Kin04]. Direct volume rendering techniques can be applied where the color and opacity transfer functions are determined by anisotropy metrics that are derived from the tensor [KWH00].

White matter tractography [MCCvZ99, BPP$^+$00], or fiber tracking, attempts to calculate and visualize the probable trajectories of neural pathways in the white matter of the brain. These pathways are formed by bundles of neuronal axons. This is probably one of the most popular techniques for visualizing DTI data.

Until recently, most fiber tracking implementations have focused on allowing the user to select volumes of interest and then computing the fibers that pass through these regions. However, recent work has investigated the merits of pre-calculating a large number of fibers throughout the brain and analyzing the results.

For example, fibers can be automatically clustered and visualized [ZL05, BKP$^+$04]. It has also been demonstrated that pre-calculating a large number of fibers throughout the brain and enabling flexible interaction with the fibers is an effective approach [ASM$^+$04]. Fiber bundles can be interactively selected by manipulating a number of selection boxes or ellipsoids and combining the selected fibers with logical operations. This chapter emphasizes the importance of interactivity as an important component of this approach.

In this work, we show that such a selection technique leads to highly reproducible FA measurements over fiber bundles and can therefore be used as a measurement tool in pre-clinical studies. In addition, we improve on the interactivity aspects by presenting a technique whereby fiber selection with multiple convex selection objects can be performed extremely rapidly.

2.3 Real-time Fiber Selection

As explained above, interactive selection of pre-calculated fiber bundles with multiple selection objects is an important component of effective visualization of DTI data. However, it is crucial that this selection can be done in real-time while the user interacts with the system.

In this section, we present a technique for the real-time selection of fiber bundles based on their intersection with an arbitrary number of convex objects. The final selection of fibers is the result of an arbitrary binary relation between the different selection objects. For example, one could request all fibers that intersect with objects A and B but not with C.

2.3.1 Overview

Fibers are represented as collections of short line segments. The simplest way to check which fibers intersect with a given selection object, is to check all line-segments of all fibers against the selection object. However, this would be quite slow, since there is a large number of line segments: an average dataset contains 45000 fibers with on average 60 line segments each resulting in a total of 2.7 million points.

Instead, we chose to implement this test as follows:

1. During pre-processing, all points that make up the fibers are inserted into a spatial data-structure.
2. All points inside the given selection object are enumerated.
3. All fibers that have at least one point inside the selection object are marked.

Instead of intersecting the polylines that represent the fibers with the selection object, we check simply for containment of the vertices that define these polylines. Since the length of a line-segment is small compared to the size of a selection object, this does not significantly affect the results. The spatial data-structure that we use is the kd-tree. This can be used to enumerate all points inside any convex polyhedron. Each point is linked to its fiber, so that visited points can quickly be mapped to visited fibers.

In the following subsections, we explain how the kd-tree is built and how it should be traversed, and we show how the technique can be applied to combinations of multiple selection objects. We conclude with some performance figures for the three-box selection method described previously.

2.3.2 The kd-tree

The kd-tree is an n-dimensional tree of spatial subdivisions [Ooi87]. Each subdivision splits a rectangular sub-space into two halves, which preferably contain an equal number of points. The splitting planes are axis oriented, and their orientation

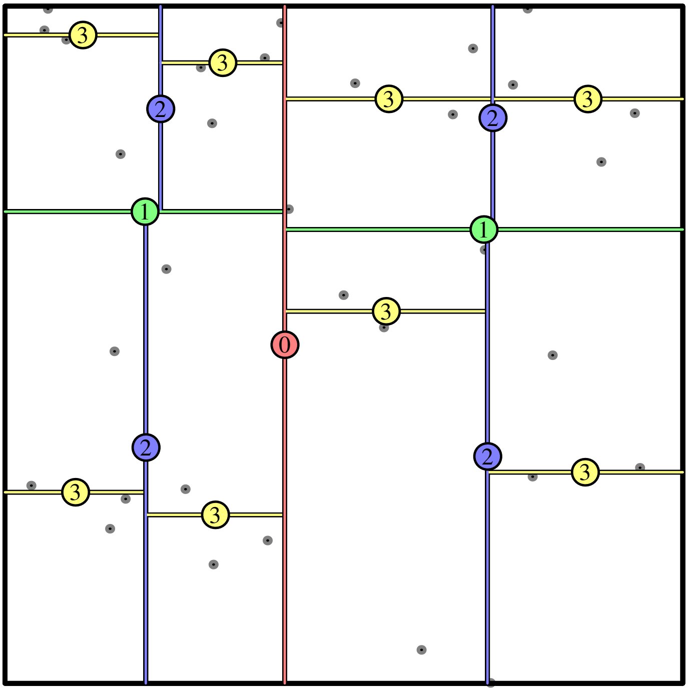


Figure 2.1: A two-dimensional kd-tree of depth 4 containing 32 points.

cycles with each level of recursion. For a two dimensional tree, this means that on level 1 the tree splits on the x-axis, on level 2 on the y-axis, and on level 3 on the x-axis again.

In contrast to other spatial subdivision schemes like the octree, we can freely choose the coordinate of the splitting plane, which gives us the freedom to balance the tree. If we choose each splitting location exactly at the median of the points sorted along the splitting direction, then each split space contains the same number of points, thus producing a balanced tree structure.

The fiber selection uses a 3 dimensional version of this tree, but for illustration purposes we examine the two dimensional case.

To illustrate how this splitting technique works, 32 two dimensional points and their corresponding kd-tree are shown in Figure 2.1. Each split is marked with its depth, such that level 0 and 2 split along the x-axis, and level 1 and 3 along the y-axis. As we can see, each leaf node contains 2 points, but since the distribution of the nodes varies, not all leaf-nodes have the same dimensions.

2.3.3 Traversal

Once the kd-tree has been constructed, we can use it to determine which fiber vertices are contained in an arbitrary convex polyhedron P. We describe the polyhedron P as the intersection of a set of half-spaces $H_1 \ldots H_n$:

$$P = \bigcap_{i=1}^{n} H_i$$

To indicate in which half-spaces a given point x is contained, we use a binary vector $\mathbf{v}$ of length n.

$$v_i = \begin{cases} 1 & x \in H_i \\ 0 & x \notin H_i \end{cases} \quad i = 1 \ldots n$$

These binary vectors form a way of classifying areas, as can be seen in Figure 2.2 on the next page.

The general idea of our approach is to recursively traverse the kd-tree in order to determine which nodes of the kd-tree are contained in P, while visiting as few nodes as possible.

A pseudo-code version of the traversal procedure is shown in Algorithm 1. Refer to this during the following explanation.

We start by examining the root node of the tree, which contains all points. Each visited node is classified respectively to the given polyhedron P. A node can be either completely inside of P, completely outside of P, or partially inside (see Figure 2.3 on the following page).

A node is classified into one of the three categories by examining all corners of the bounding box of that node. In our case, this three-dimensional box has 8 corners, denoted by $\mathbf{b}_1 \ldots \mathbf{b}_8$. The collection of all eight corners is denoted B.

If all corners of the bounding box are inside of P then, as a result of the convexity of P, the entire box must be contained in the polyhedron. In this case, we can simply mark all points contained in this node as “inside”, and no further testing is necessary.

If this is not the case, then we establish whether the bounding box is completely outside of the polyhedron. We do this by checking whether there is any half-space in P for which all points are outside. If this is the case, then we can conclude that

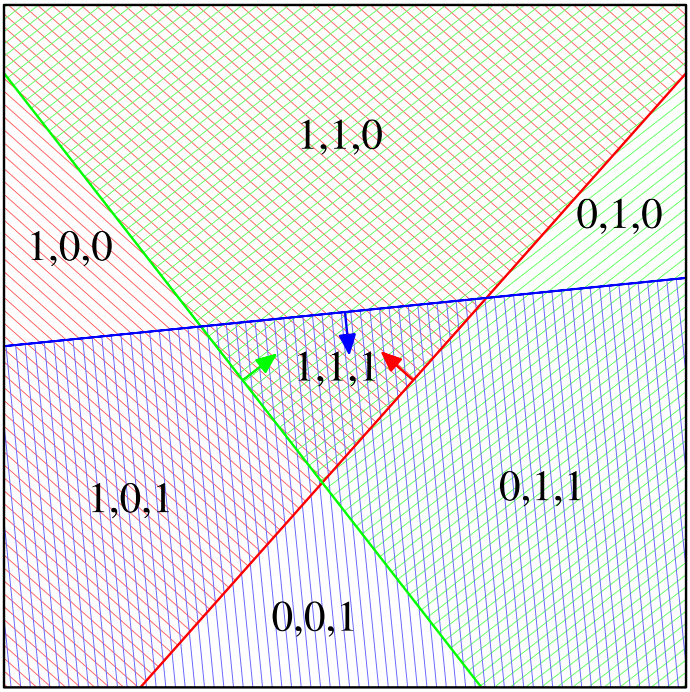


Figure 2.2: In two dimensions, the intersection of three half-planes yields a convex polygon. The three half-planes are hatched in red, green and blue respectively. Each area is marked with its bit vector, so that (111) *forms the polygon. In three dimensions, the polygons become polyhedra, and the half-planes become half-spaces.*

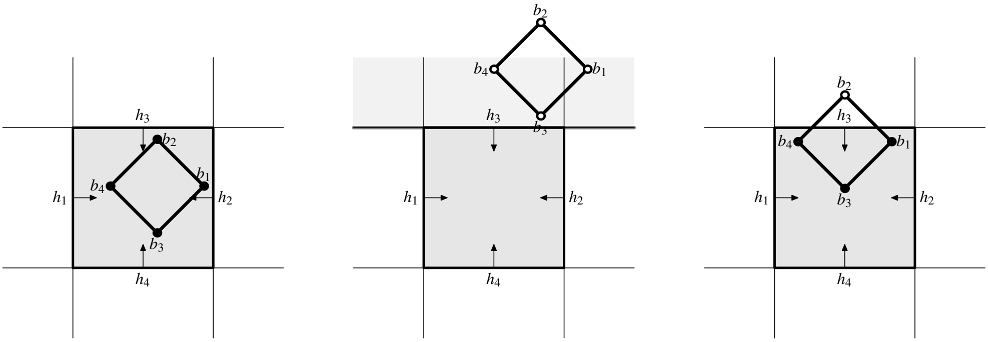


Figure 2.3: The three classes of containment for a box with of 4 corners $b_1 \ldots b_4$*, with respect to a polyhedron defined by the intersection of four half-spaces* $h_1 \ldots h_4$*. On the left, the box is completely contained. In the middle, all points are outside the half-space* h_3*, so the box is completely outside the polyhedron. In the rightmost figure neither is true, and further recursion is necessary.*

none of the points inside the bounding box can be inside P, and no further recursion is necessary.

When neither of the two previous cases are true, then it is possible that the points are partly inside and partly outside convex polyhedron P. This is where we exploit the kd-tree's spatial subdivisioning by recursively examining both child-nodes. This recursion continues until all points contained in the kd-tree have been classified.

Algorithm 1 The recursive kd-tree search algorithm.

```
function recurse
if leafnode then
  visit()
else
  $allInside \Leftarrow 1$
  for all h $\in H$ do                              // iterate the h over all half-spaces
    $allOutside \Leftarrow 1$
    for all x $\in B$ do                      // iterate the x over all corners of the box
      if $x \in$ h then                                 // x is inside the half-space h
        $allOutside \Leftarrow 0$
      else                                                             // x is outside
        $allInside \Leftarrow 0$
      end if
    end for
    if $allOutside = 1$ then                                // all points were outside h
      return                                         // completely outside: visit no points
    end if
  end for
  if $allInside = 1$ then                           // all points were inside all half-spaces
    visit_all()                                       // completely inside: visit all points
    return
  end if
  recurse(left)
  recurse(right)
end if
```

The main kd-tree search algorithm. The auxiliary function *visit* tests all points contained in the leaf node against the polyhedron, and flags all points that are inside. The *visit_all* function does the same but assumes all points are always inside.

In short, the procedure we described is recursive and starts by examining the root node. It examines the surrounding box for the node and efficiently decides how the corners of the box relate to the convex search area. This relation can be classified in three cases: fully inside, fully outside, neither. Only in the last case further recursion is necessary. Figure 2.4 shows a graphical representation of all the visited nodes of the kd-tree after the cingulum has been selected.

2.3.4 Marking the fibers

Now that we have found all the points, we still need to find all the corresponding fiber bundles. Since each point only belongs to a single fiber we store a simple fiber-reference with each point (see Figure 2.5 on page 32). Using this pointer, we can quickly flag all fibers that contain visited points.

2.3.5 Multiple convex regions

We can expand this idea to work with multiple regions by changing the fiber flag to a bit-vector. For each convex region we reserve a single bit.

We initially set all bit-vectors to zero, and then for each region flag all fibers that have points inside the region. After processing all regions, each fiber has a bit-vector corresponding to which regions it intersects. This can be easily used to select all fibers that correspond to any given logical combination of selection objects.

A useful selection approach is a logical AND of three boxes, so that only the fibers that cross all three boxes are visible. This allows us to select most standard fiber tracts.

Sometimes these selection methods erroneously mark extra fibers. To get rid of these fibers we introduce extra boxes that work in a negated fashion: any fibers that pass through any of these boxes are filtered out. This allows for easy manual pruning of a selection set.

2.3.6 Performance

The performance of any selection method of this type is dependent on the size of the dataset, the number of fibers in the data, the size of the selection objects and the current position of the selection objects. We measured the performance of the implemented techniques by making use of a dataset with 47500 fibers and we used three selection objects as explained in section 2.4. By moving the selection boxes to different parts of the brain, we measured a range of selection speeds.

In all cases, the system remained completely responsive, and the fiber selection was updated in real-time during intensive interaction with the selection objects. On a first generation Pentium-M at 1.6GHz, our implementation processed between 1 and 2 million fibers per second. On a Pentium 4 at 3GHz, it processed between

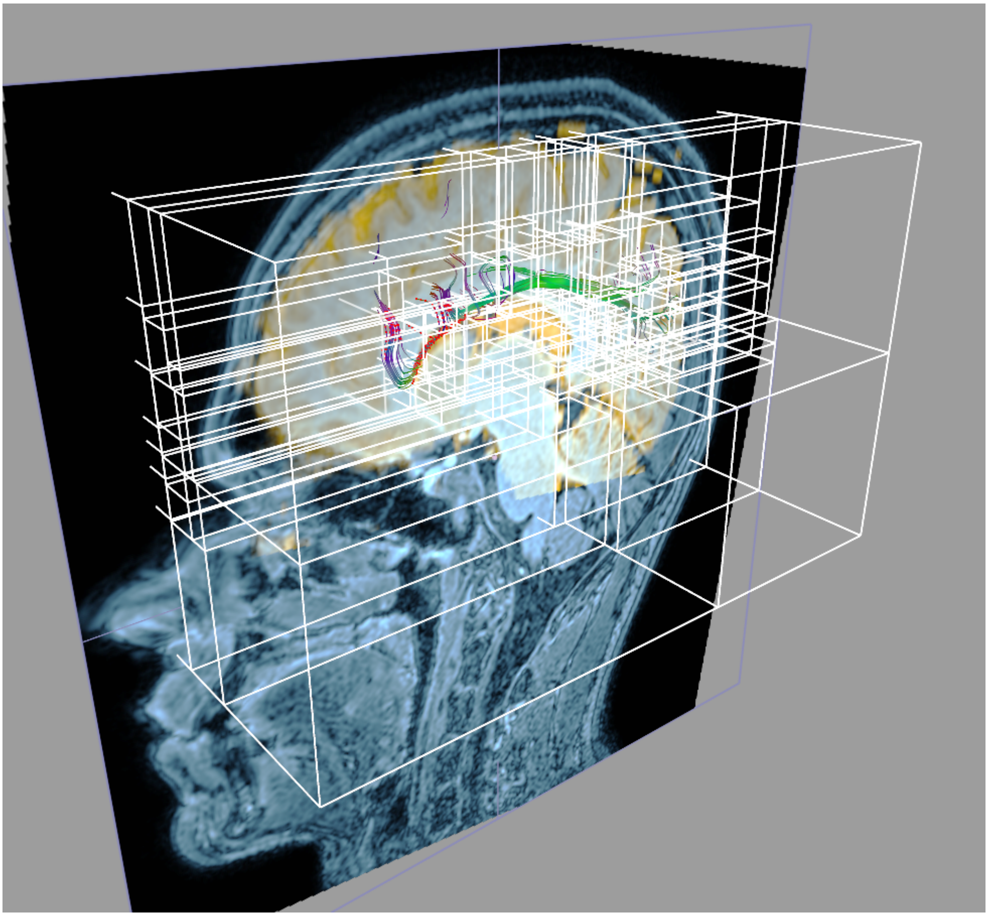


Figure 2.4: The visited nodes of the kd-tree for the selection of the cingulum.

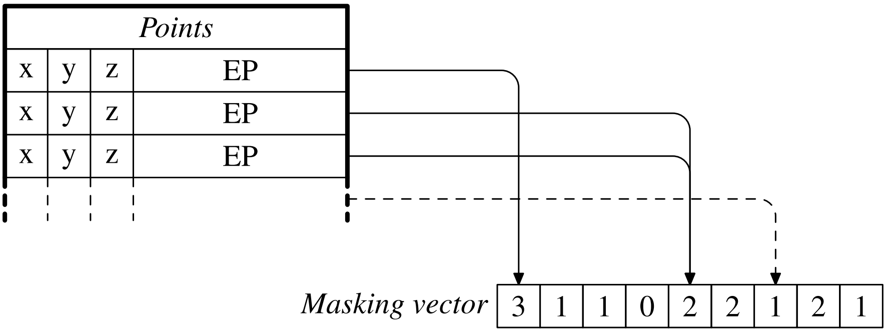


Figure 2.5: A number of points in a leaf-node, each with a reference to a fiber mask entry.

1.5 and 3.5 million fibers per second. This includes selection *and* rendering of the selected fiber bundles.

This is at least an order of magnitude faster than the figures published in [ASM$^+$04]: on a "1.6GHz Pentium laptop PC" they were able to intersect a single VOI with between 80000 and 220000 pathways per second. We attribute this improvement to the fact that our selection and rendering implementations were purpose-designed for highly-interactive DTI, whereas general-purpose visualization and collision-detection libraries were used in [ASM$^+$04].

2.4 Validation of Three-box Selection Method

In a DTI-based study of healthy and schizophrenic subjects, a selection method was required whereby sections of fiber bundles could be easily and reproducibly selected. Parameters derived from DTI, such as for example the FA, or fractional anisotropy, can then be calculated over these sections and be compared for different test subjects.

Based on the techniques presented in section 2.3, we chose an approach based on three bounding boxes. All three bounding boxes can be freely positioned, rotated and resized within the DTII user interface. One or more slice-views of the data can be used as context during this navigation. All fibers intersecting with any one of the three cubes are pre-selected. One of the three bounding boxes also functions as the section selector: only sections of the pre-selected fibers that are contained within this box are added to the final selection. Figure 2.6 shows such a fiber selection in progress. Unselected fibers are shown with a different shading to serve as context for the selected fibers. The box in the middle is the section selector in this case: only the fiber sections contained in this box are actually selected. These are marked with

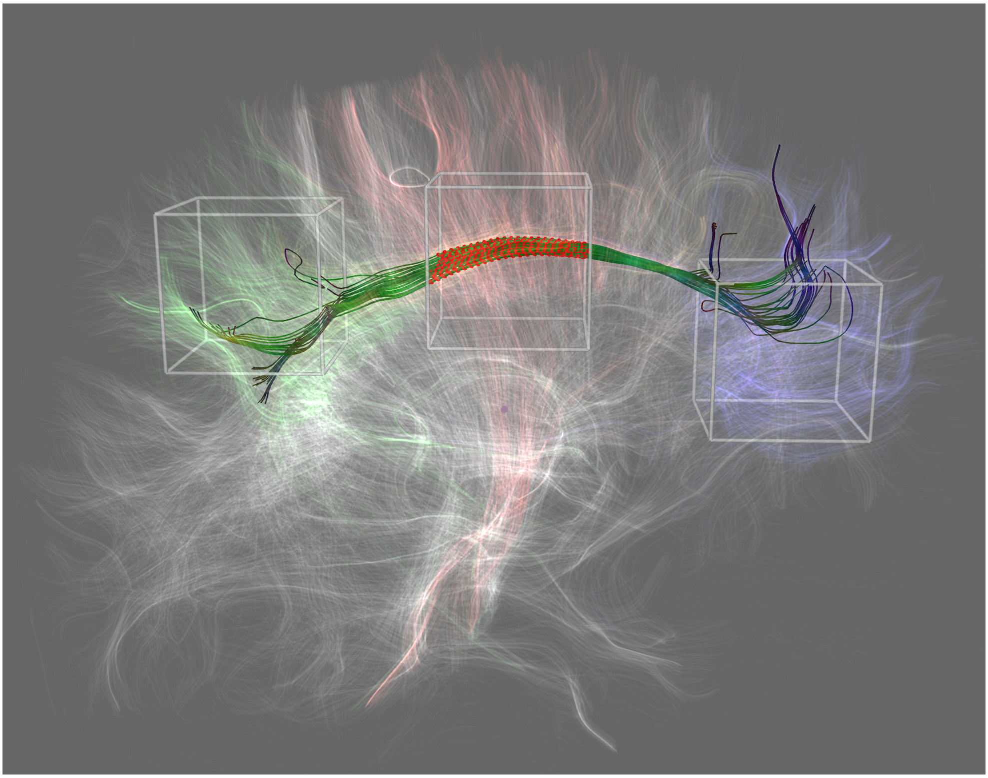


Figure 2.6: The three-box selection method in progress. A pre-selection is made of all fibers that intersect all three boxes. Out of this pre-selection, only fiber sections that are contained within the center box are selected for further processing. For color plate, see Figure A.3 on page 180.

red dots. The selection is performed and shown in real-time throughout the user's interaction with the system.

DTI datasets were acquired of five healthy and five schizophrenic subjects on a Philips Intera 3 Tesla MRI scanner. The datasets were acquired along 16 different diffusion directions. Each direction was scanned twice and averaged to improve the signal to noise ratio. The scan time was approximately 6 minutes per patient and the final resolution of the DTI dataset was $2 \times 2 \times 2$ mm.

Two users, both researchers in the field of neuropsychiatry, applied the method described above to select the fiber bundles corresponding with the left and the right cingulum in all ten datasets. In other words, 40 independent selections were performed. On average, each selection took less than two minutes.

	Left Cingulum		Right Cingulum	
Percentile	Obs. 1	Obs. 2	Obs. 1	Obs. 2
25	4359.25	4370.25	4969.00	5039.25
50	4801.00	4742.00	5505.50	5513.00
75	5196.75	5329.75	5689.50	5760.00
IQR	837.50	959.50	720.50	720.75

Table 2.1: 25, 50 *and* 75 *percentiles and the interquartile ranges (IQR) for the FA measurements by each of the observers on the left and the right cingulum. The FA measurements have been scaled to the range [0, 10000].*

The average FA, or fractional anisotropy, was calculated over the selected fiber sections, resulting in four FA measurements per subject. The fiber tracking was performed anew by each user. We made use of FA due to the specific clinical questions that were being asked and due to its popularity in clinical DTI literature. In the future, we will continue our research by investigating other metrics, both existing and newly developed.

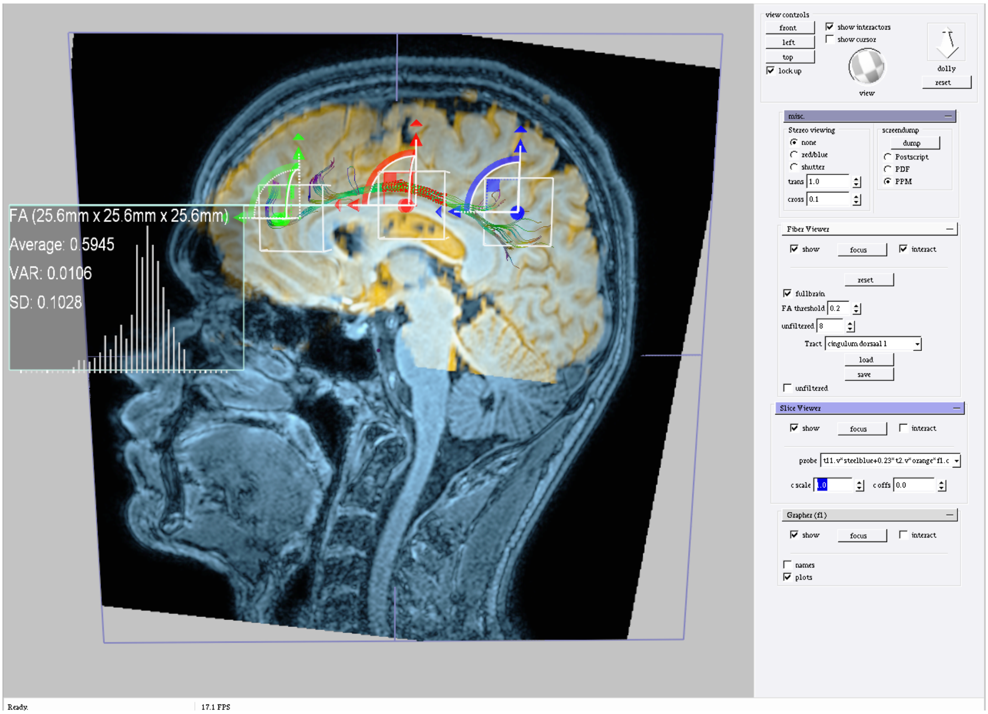
Due to the relatively small sample size, we chose to make use of non-parametric statistics to analyze our data.

Table 2.1 shows the 25th, 50th and 75th percentiles as well as the interquartile ranges (IQR) for the measurements by each of the observers on the left and on the right cingulum. The interquartile ranges show that the FA measurement varied significantly over different patients.

For the left cingulum, the non-parametric Spearman correlation between the two observers was 0.903 and for the right cingulum it was 0.976, both with a two-tailed significance of less than 0.001. This shows that the inter-observer correlation is sufficiently high for the reproducible selection of fiber bundles and subsequent measurement of FA over these selections.

2.5 The DTII System

The DTII platform was designed to provide functionality for the interactive exploration of DTI data. It was developed in cooperation with the Academic Medical Centre in Amsterdam. The software performs full brain tractography as a preprocessing step and allows the user to perform interactive selections on the precalculated fibers by using the techniques presented in section 2.3. In addition, other MRI data can be combined with the DTI visualization to provide context. It supports fusion of anatomical MRI, functional MRI (fMRI) and DTI data. This data is


FA (25.6mm x 25.6mm x 25.6mm)
Average: 0.5945
VAR: 0.0106
SD: 0.1028

☑ show interactors
☐ show cursor
☑ lock up

Stereo viewing: ● none ○ red/blue ○ shutter
trans 1.0
cross 0.1

screendump: ○ Postscript ○ PDF ● PPM

Fiber Viewer: ☑ show ☑ interact
☑ fullbrain
FA threshold 0.2
unfiltered 8
Tract cingulum dorsaal 1
☐ unfiltered

Slice Viewer: ☑ show ☐ interact
probe t11.v*steelblue+0.23*t2.v*orange*f1.c
c scale 1.0
c offs 0.0

Grapher (f1): ☑ show ☐ interact
☐ names
☑ plots

Ready.
17.1 FPS

Figure 2.7: The main user interface of DTII. *A T1-weighted and T2-weighted anatomical scan are combined using color fusion (blue/orange). This is used as a background context against which the fiber selection of the cingulum is performed using the three-box method.*

usually acquired in a single scanning sequence, but with proper registration multiple scans can also be combined. Figure 2.7 shows the main user interface of the software. This tool was used in the reproducibility study documented in section 2.4.

DTII employs the TEEM library for all data handling and fiber tracking. TEEM is an open-source image processing and visualization library that also supports tensor processing [Kin10]. Via this toolkit, DTII offers two types of fiber tracking: fourth-order integration based streamlines on the primary eigenvectors of the diffusion tensors or tensor lines [WKL99].

All visualization methods in DTII support progressive updates in order to maintain high frame-rates even during complex manipulations. DTII was written in C++ and is highly portable, depending only on a few external libraries. Currently, versions for Windows and Linux are in use. It is complemented by a conversion utility which can convert both MRI and DTI DICOM series to the native DTII data format. This facilitates deployment in a medical environment.

The user interface is customizable so that only the components relevant to the task at hand are displayed. This greatly eases repetitive selection processes, as the user is no longer distracted by superfluous user interface elements.

Feedback from clinical users has been positive.

2.6 Conclusions and Future Work

This chapter presents two contributions to the field of DTI visualization. Firstly, we presented a technique that enables the real-time selection of fiber bundles with multiple convex selection objects. Sets of fibers that intersect with the convex selection objects can be combined with logical operations. The kd-tree datastructure combined with the halfspace representations of the selection objects is very efficient: selections with three objects in a full brain dataset with 2.7 million fiber points works very well in interactive mode. We showed that this technique performs an order of magnitude faster than similar previous techniques.

Secondly, we have tested the interactive selection of DTI-derived fibers with three manipulable boxes on 10 subjects. For each of these 10 subjects, two users independently selected the fiber bundles corresponding to the left and the right cingulum. The average fractional anisotropy (FA) was calculated over these fiber bundles. Our analysis of the test data shows both a significant variation of the FA over the different subjects and a significantly high correlation between the measurements by the two observers. This inter-observer reproducibility indicates that this type of approach can be used as a measurement tool in pre-clinical studies.

The interactive bundle selection may be combined with semi-automatic methods such as similarity clustering. We will continue developing the DTII tool and plan to continue working on its integration in a clinical setting. We will also perform more similar studies on test subjects to determine the reproducibility of other fiber selection based metrics. Ultimately, we wish to develop reliable and reproducible connectivity metrics that can be used to measure the “strength” of a neural connection between two regions of interest. In this way, diagnostic test sets can be developed to examine the main connection pathways in the brain to diagnose brain disorders. Visualization will play an important role in this type of diagnosis.

Our selection method is based on multiple convex polyhedra. At the moment there is no requirement for concave selection objects, but if this requirement arises, we plan to extend our approach by integrating methods for decomposing concave polyhedrons into multiple convex polyhedra.

Acknowledgements

DTII makes use of the `teem` toolkit, available at `http://teem.sourceforge.net`.

The involvement of Charl P. Botha and Frits H. Post in this work took place in the context of the Virtual Laboratory for e-Science project (`http://www.vl-e.nl/`). This project is supported by a BSIK grant from the Dutch Ministry of Education, Culture and Science (OC&W) and is part of the ICT innovation program of the Ministry of Economic Affairs (EZ).

Interactive Visualization of Fused fMRI and DTI for Planning Brain Tumor Resections

3

Published as: Jorik Blaas, Charl P. Botha, Charles Majoie, Aart Nederveen, Frans M. Vos and Frits H. Post: *Interactive Visualization of Fused fMRI and DTI for Planning Brain Tumor Resections*, accepted for the SPIE Conference on Medical Imaging 2007.

This chapter presents an extension to the neuroimaging work, this time focusing in particular on pre-operative planning for the surgical removal of brain tumors. For this application, the main goal is to assess which operation path would minimize functional damage. To map functional damage, not only the tumor itself is segmented from functional scans, but specific motor areas of the brain are delineated through functional MRI. With knowledge of these areas, the tractography can be used to find tracts which are vital for the motor tasks, so that damaging these can be avoided during surgery.

Abstract

The surgical removal of brain tumors can lead to functional impairment. Therefore it is crucial to minimize the damage to important functional areas during surgery. These areas can be mapped before surgery by using functional MRI. However, functional impairment is not only caused by damage to these areas themselves. It is also caused by damage to the fiber bundles that connect these areas with the rest of the brain. Diffusion Tensor Images (DTI) can add information about these connecting fiber bundles. In this chapter we present interactive visualization techniques that combine DTI, fMRI and structural MRI to assist the planning of brain tumor surgery. Using a fusion of these datasets, we can extract the fiber bundles that pass through an offset region around the tumor, as can be seen in Figure 3.1 on the facing page. These bundles can then be explored by filtering on distance to the tumor, or by selecting a specific functional area. This approach enables the surgeon to combine all this information in a highly interactive environment in order to explore the pre-operative situation.

3.1 Introduction

Brain tumors are a major health risk. An estimated 43,800 new cases of primary non-malignant and malignant brain and central nervous system tumors are expected to be diagnosed in the United States in 2005 [CBT06]. The surgical removal of brain tumors is a critical operation, as it may lead to functional impairment, but also to recurrent surgery if a tumor is not completely removed. Therefore, good surgical planning is crucial to minimize damage to important functional areas. Support from MRI imaging has become standard practice, by presenting patient-specific structural (anatomical) information to the neurosurgeon. More recently, different MRI imaging modalities are being combined to provide a good overview of the potentially affected functions.

An anatomical MRI scan can show both the tumor and important brain structures. In planning tumor resection, it is of prime importance to determine how the tumor is related to the surrounding anatomical structures.

Functional MRI is based on the increase in blood flow to the local vasculature that accompanies neural activity in the brain. In fMRI, the increased oxygen utilization associated with this activity can be detected using an MR pulse sequence such as BOLD (Blood Oxygenation Level Dependent). The effects are measured using rapid image acquisition (1 frame per 1 – 4 sec.), with a spatial resolution of 1.5 – 3 millimeters. The human subject performs certain tasks, such as moving a hand or foot,

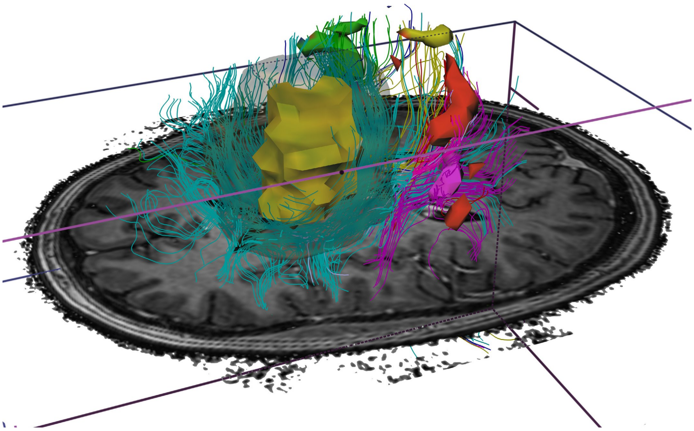


Figure 3.1: Combined visualization of a brain tumor, the white matter tractography and the functional areas associated with motor tasks. The tumor is surrounded by a safety area, which is used to filter and color the fibers in the tractography. Both the tumor and the activation areas directly influence the color of the fibers. For color plate, see Figure A.4 on page 181.

and the response can be shown (after a delay) as zones of increased neural activity, or activation zones. Functional MRI can be of great help to provide a mapping of the brain zones activated by certain motor activities. However, functional impairment can not only be caused by direct damage to these functional zones, but also by damage to the neural pathways connecting these zones with other parts of the brain.

Diffusion Tensor Imaging (DTI) has recently emerged as a technique for the quantification of water diffusion in living tissue [WMB$^+$99, WMM$^+$02]. Diffusion is usually represented by a second-order symmetric tensor for each voxel. These tensors describe the free motion of water molecules in living tissue and can be derived from a sequence of diffusion-weighted scans, each sensitive to a different diffusion direction. This makes DTI usable for imaging fibrous tissue, such as neural brain fibers, as there are large differences in diffusion speed parallel with and perpendicular to the fiber. By eigen analysis of the tensors, the anisotropy and the main directions of diffusion can be extracted. Thus, DTI can be used to visualize the connection structure of neural fibers in the living human brain, by tracing paths following the strongest diffusion directions. This technique is called fiber tracking, or white matter tractography. For this reason, DTI is now also used for surgical planning of tumor resection, to add information about these connecting nerve structures, which may be damaged by the resection.

It is clear that each of the three MRI techniques provide relevant information for tumor resection. The problem is how to combine the information from these three modalities: structural, functional, and diffusion MRI, and present it to the neurosurgeon in such a way that it can be helpful for planning the resection.

In this chapter we present an ordered set of visualization tools to examine structure and function of the resection area around a tumor, and possible access paths for surgery. The main basis for this visualization set is (1) multimodal imaging (structural, functional, and diffusion MRI), (2) identifying the functional activation zones and neural connections, and (3) interactive examination of the large multimodal datasets. Combined visualization of the three types of information leads to the insight for the neurosurgeon, to support the critical decisions in planning an operation.

The set of techniques we developed is the result of preliminary work in surgical planning for tumor resection. For many neurosurgeons, the information presented is quite new and unusual, and together we are searching for an effective way to present, explore, and utilize this information. At this stage it is not possible to perform rigorous clinical tests to measure the effectiveness, or to formalize the planning information to be used for intra-operative navigation. The main purpose at this point is to allow the neurosurgeon to get a clear image of several aspects of the surgery area.

Our main contributions in this work are the interactive selection and probing techniques that can be used to combine the information from the three different modalities, each providing a different type of information for the planning. We present an ordered toolset, which means that it is a balanced, goal-directed set of visualization tools, which fully covers its area of functionality, and not an open-ended, unstructured collection of tools for free exploration. For this purpose, our existing interactive platform for DTI data visualization DTII [BBP+05] was extended.

This chapter is structured as follows: after a discussion of related work, we present the multi-modal visualization toolset. It is based on a simple model defining the main visualization components, and describing the functions by combining these components pairwise and all three. Then we present a discussion, with conclusions and directions of future work.

3.2 Related Work

fMRI and DTI are exciting research fields and the number of publications is growing fast. Both fields provide excellent additional information for planning brain surgery. In the past few years a number of techniques have been proposed to combine structural MRI together with different MRI based techniques

This combination of techniques has been researched in a number of different settings. Specifically, the connectivity of the primary motor cortex has been studied by Guye et al. who concluded that MRI-correlated DTI is a promising tool to study the structural basis of functional networks [GPS$^+$03].

In the particular field of surgical planning, recent developments have focussed on combining anatomical information about the tumor with fMRI and DTI. Mori et al. have shown that fMRI and tractography can be used together for pre-surgical mapping [HPS$^+$03]. This combination of functional MRI and DTI was tested on a larger patient group by Talos et al. [TOW$^+$03]

These approaches however usually adhere to a fairly rigid processing pipeline, where user interaction is limited to viewing the end results. Interactive exploration of the surroundings of the tumor and the activation areas is severely limited. Our approach focusses on presenting all the information in a comprehensive interactive framework, aimed at exploring all available data to the fullest extent.

3.3 Data Acquisition and Preparation

All datasets were acquired on a Philips Intera 3 Tesla MRI scanner. For the structural MRI we used a T1 weighted scan with a $0.5 \times 0.5 \times 1$ mm resolution.

A Functional MRI scan was then performed to map the activation of important motor tasks. The activation patterns for hand and foot motion were imaged on both the left and right side. A repeated 60 second sequence of alternating motion and rest was used. The resolution of the functional scan was $1.7 \times 1.7 \times 3.3$ mm.

The DTI datasets were acquired along 16 different diffusion directions. Each direction was scanned twice and averaged to improve the signal to noise ratio. The scan time was approximately 6 minutes per patient and the final resolution of the DTI dataset was $2 \times 2 \times 2$ mm.

3.3.1 Preprocessing

As the data from fMRI and DTI scans need complex preprocessing, we have used third party tools to perform the required steps on these data. The functional MRI was processed using FEAT, the FMRI Expert Analysis Tool, which is provided as part of the FSL package [SJW$^+$04].

To make the diffusion weighted images from the DTI scan suitable for fiber tracking, a tensor estimation step was performed using the TEEM toolkit [WKL99].

3.3.2 Registration

Since not all scans could be made in the same frame of reference, a registration step was necessary before the datasets could be fused.

We used a semi-automatic registration approach, where the initial transform is automatically determined, after which the alignment can be fine-tuned by the user while viewing a realtime fusion of both datasets.

We have chosen to align both the functional and the structural scan to the DTI. To ease the visual alignment process we presented the user with a fused B0 volume reconstructed from the DTI data together with the scan to be aligned. This approach turned out to provide valuable feedback to the user, and has made the alignment task relatively easy.

3.4 Visualization Components

After preprocessing, the data volumes are loaded into the DTII framework. This tool facilitates the interactive inspection of MRI data by providing a number of visual components that can be used to inspect any aspect of the input datasets. The fusion of multiple datasets, even with custom blending and compositing functions, is handled by the platform in a highly responsive way.

DTII employs the TEEM library for all data handling and fiber tracking. TEEM is an open-source image processing and visualization library that also supports tensor processing. Via this toolkit, DTII offers two types of fiber tracking: fourth-order integration based streamlines on the primary eigenvectors of the diffusion tensors or tensor lines [WKL99].

All visualization methods in DTII support progressive updates in order to maintain high frame-rates even during complex manipulations. DTII was written in C++ and is highly portable, depending only on a few external libraries. Currently, versions for Windows and Linux are in use. It is complemented by a conversion utility which can convert both MRI and DTI DICOM series to the native DTII data format. This facilitates deployment in a medical environment.

The next sections describe the components present in DTII.

3.4.1 The slice viewer

The SliceViewer is a well known component in data visualization, it shows a single cutting plane through a data volume, as illustrated in Figure 3.2 on the next page.

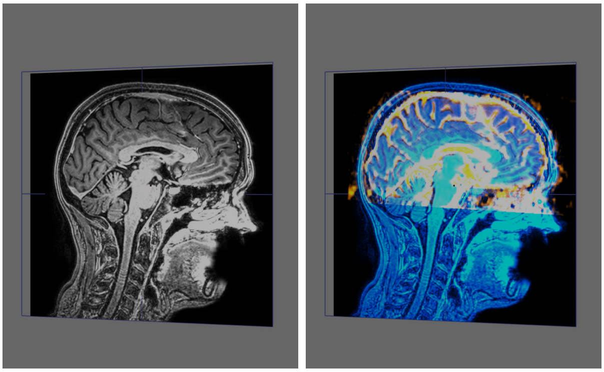


Figure 3.2: The sliceviewer, showing an structural T1 MRI (left) and the fusion of a structural T1 with a B0 reconstructed from a DTI scan.

The sliceviewer in DTII works in the same way, except that it can operate on an arbitrary combination of volumes. We can use this to superimpose activation areas on structural scans, but also to assist the alignment process by showing a red/green fusion of two datasets.

3.4.2 Tumor neighborhood isosurface rendering

For rendering the geometry of the tumor, we use an ordinary polygon renderer in combination with a marching cubes surfacer. However, apart from the geometry of the tumor, the surgeon also needs information about the area surrounding the tumor. Safety zones are normally used in tumor surgery, and for this reason we have chosen to make it possible to visualize a neighborhood of the tumor by specifying an offset distance. We use an order-independent renderer to show multiple transparent isosurfaces at the same time, so that both the geometry of the tumor and that of the surrounding safety margin are visible together. The tumor and an offset surface at a user-specified distance are shown in Figure 3.3 on the following page.

3.4.3 Point-based activation area rendering

The functional activation areas are commonly stored as volumes with statistical z-values, indicating how significantly the measured signal matches the model pattern. We have such a volume for each task. A higher value generally means that the activation in that voxel is more likely to be part of that task. The activation areas are extracted by thresholding these statistical values.

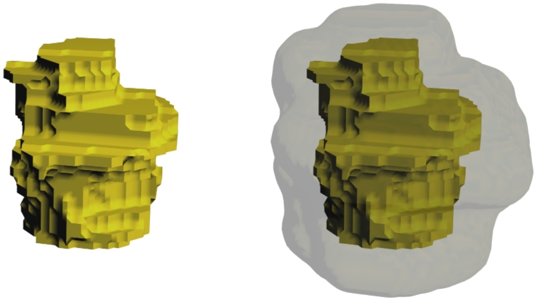


Figure 3.3: The segmented tumor displayed separately (left), and with a surface at a fixed distance from the tumor (right).

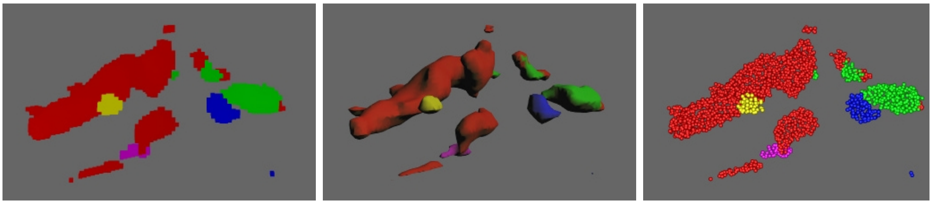


Figure 3.4: The activation areas, displayed using three different rendering modes. From left to right: Shell rendering, Polygon rendering from marching cubes, Point based rendering. The colors correspond with the different tasks.

As shown in Figure 3.4, we have a number of methods for rendering these areas. The first method is a simple shell renderer, that uses blocks to mark all voxels that belong to a specific area. The shell renderer provides an unshaded surface representation, which is sometimes preferred when complex coloring modes are used. The second method is the marching cubes renderer that is also used to show the tumor surface. While its shaded surface provides a good impression of the shape of the areas, it is sometimes hard to discern the surface when used in combination with other complex visualization components. Also, the surface is often considered a bit too clean for the uncertain and noisy nature of functional MRI. To solve these problems we have a third rendering mode, which uses a point based renderer to display the activation areas. The point based renderer uses a large number small spheres that lie on the isosurface. This makes the areas easily perceptible, while maintaining a sense of the uncertainty associated with functional MRI scans. For all of these rendering methods, the DTII platform allows the user to interactively modify the threshold values as needed and get direct feedback on the resulting changes.

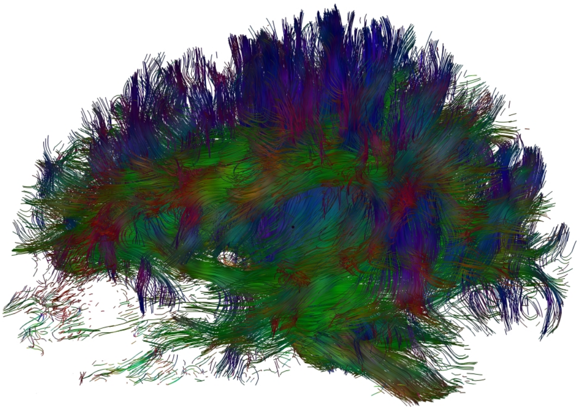


Figure 3.5: A full brain tractography consisting of seventeen thousand pathways, color coded by the local diffusion direction. A green color means that the diffusion is the largest in the anterior-posterior direction, red in the left-right direction and blue in the up-down direction.

3.4.4 Tractography display

After preprocesing the Diffusion Tensor Imaging data, we use fiber tracking to create a full brain white matter tractography (as seen in Figure 3.5). The fiber tracts can be displayed with a large set of fully customizable color modes. The local tensor shape can determine the color to indicate the anisotropy or direction of the diffusion. Optionally, the color can also be determined using data from another scan, such as secondary fields like tumor-distance or activation maps. The colors can either be calculated locally for each point on the fiber, or they can be automatically aggregated over each fiber, as to store the maximum or minimum color along the fiber path. This tractography generally contains twenty to thirthy thousand fiber tracts. To assist the exploration of this complex structure, we provide filters for selective visualization of the tracts.

We present two basic filtering modes, to help the user select the tracts he is interested in.

1. Filtering by arbitrary convex selection objects. This first filtering mode is meant for interactive exploration of larger sets of fibers. We use an interactive filtering technique that provides the user with a set of movable boxes [BBP+05]. All the boxes can be freely positioned, rotated and resized within the DTII user interface. The fibers are tested against each box for intersections. The most commonly used filter is to only display the tracts that pass through all of the

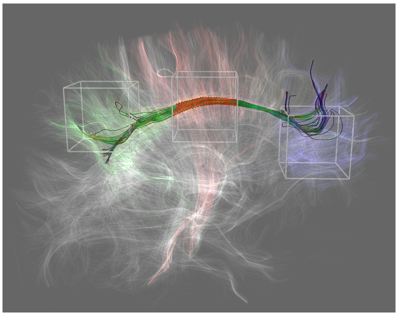


Figure 3.6: A full brain tractography filtered by using three selection boxes. Only the fibers passing trough all of the boxes are selected and therefore rendered opaque. The other fibers are rendered transparent, in a color that represents which boxes they pass through. Each box has a unique color to make it easy to spot fibers that pass through any combination of boxes. For color plate, see Figure A.3 on page 180.

boxes, as seen in Figure 3.6 on the next page. However, the technique is expandable to any logical combination, so that extra boxes may be used in a negated fashion to prune away fibers from the selected set.

2. Filtering by aggregation of arbitrary volume data over the fiber. Apart from the box filters described in the previous section, we also have filters that use the geometry of the fiber tracts to locally retrieve a scalar from a secondary volume. Much like the aggregated color over the fiber, the measured values are then combined over each tract, and the aggregated value is compared with a threshold to determine the visibility of the tract. This can for instance be used to quickly remove all tracts where the minimum fractional anisotropy is below a specified value.

3.5 Multi-domain Visualization Framework

In this section, we present our conceptual framework and implementation for the visualization of and interaction with structural MRI, fMRI and DTI datasets. The goal of the visualization is to assist interactive exploration of the information contained within the datasets before a tumor resection procedure. During this exploration, the surgeon attempts to gather information about the various interactions between tumor location, fMRI activation areas and fiber bundles in the white matter of the brain.

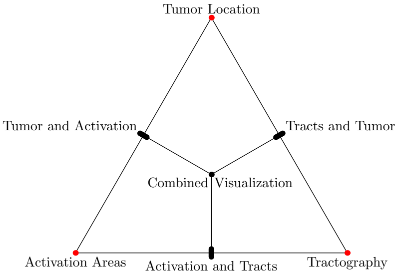


Figure 3.7: The triangular space representing the possible combinations of the three data domains. The corners represent the tumor location, the activation areas and the tractography.

Combining all datasets in a useful manner and presenting them in a single interactive visualization poses quite a challenge. Our framework helps in solving this problem.

The framework is illustrated in Figure 3.7. The three vertices represent the information contained in each of the three datasets involved in the exploration: The structural MRI is used primarily for locating the tumor, the fMRI for the activation areas and DTI for the tractography. We have termed the information contained in the datasets and represented by the vertices of the triangle, such as the fMRI activation areas, domains.

Each of the three edges represents interaction between two different domains. For example, the edge on the right represents interaction between the tumor location domain and the tractography domain. Increasing the tumor distance threshold as explained in Section 3.4.2 directly interacts with the current fiber selection and a number of fibers are activated, deactivated or colored differently. For each edge, there are a number of different interaction possibilities.

In the following sections, we describe four visualization approaches. Each of the first three address one of the edges of the triangle. The final approach combines the interactions represented by all three edges, and can thus be seen as situated in the interior of the triangle.

3.5.1 Tumor and activation

In the first approach, the interaction between tumor locations and fMRI activation areas is explored. The tumor distance parameter can be adjusted, thus increasing the volume of the offset surface surrounding it. Activation areas that are intersected by this volume are updated in real-time. This allows the surgeon to experiment with

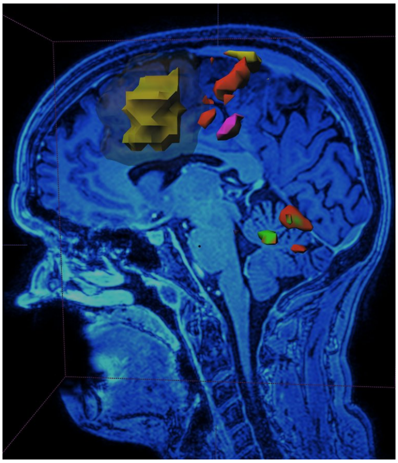


Figure 3.8: Tumor neighborhood and Activation areas

the safety zone surrounding the tumor, and to explore the degrees of freedom that should be available during the resection procedure.

The combined visualization of the activation areas and the tumor assists the highest level of pre-surgical planning. The information presented allows the surgeon to pinpoint the location of the motor areas in the brain, and use this to estimate a surgical path to the tumor that avoids these areas. Figure 3.8 shows an example of this visualization mode.

3.5.2 Tracts and tumor

This visualization enables the operating clinician to explore the interaction between the DTI fiber tracts and segmented tumor. As in all cases, an offset surface surrounding the tumor, representing a safety zone, can be adjusted by modifying a distance threshold. Fibers that do not intersect with the tumor and its safety zone can be deactivated. With the functionality described in Section 3.4.4, tumors that do intersect the safety zone can be colored in different ways:

1. A single color can be assigned to each visualized fiber based on the shortest distance at which it passes the actual tumor. This visualizes, per fiber, how great the risk is to that fiber during a resection. Fibers that enter the interior of the tumor are all assigned the color representing zero distance. Recall that, due to the nature of current DTI techniques, each visualized fiber actually represents the likely path a whole bundle of real axonal fibers follow.

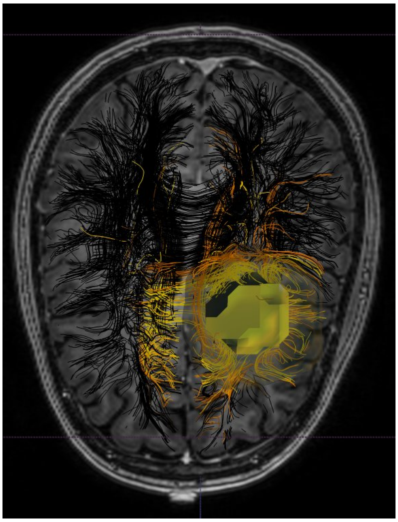


Figure 3.9: Tumor neighborhood and Tractography

2. Fibers that intersect the safety volume can be colored according to the distance of each point to the tumor. In other words, the color across a single fiber changes depending on the distance of that point of the fiber from the tumor.

Figure 3.9 shows an example of the per-fiber coloring. In this case, the full tractography is shown, but fibers that do not intersect the safety volume are not colored. They can also be completely deactivated, i.e. removed from the visualization. Fibers that are more brightly colored pass closer to the tumor. In Figure 3.10 on the next page two additional examples are given for comparison: on the left, color across the fiber reflects the distance of that point from the tumor and on the right each fiber is colored according to its shortest distance.

Other filtering techniques described in Section 3.4.4 can be applied at the same time. For example, anatomical fiber tracts can be filtered using selection objects, whilst the intersection of these tracts with the tumor safety volume and their coloring are updated in real time.

This visualization aids insight into how the tumor has deformed its surrounding tissue by allowing the user to visualize interactively the full tractography in the tumor’s vicinity.

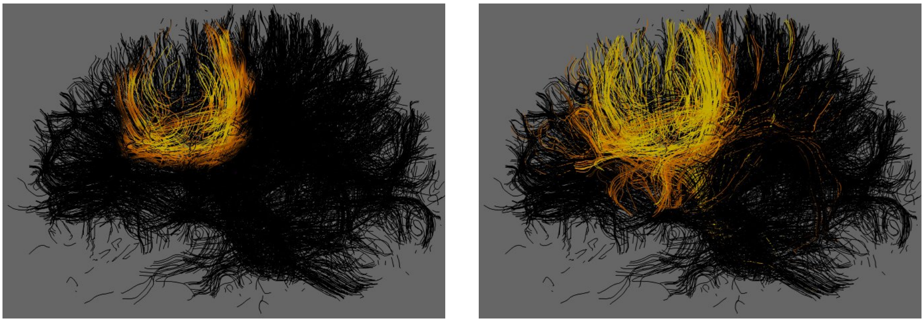


Figure 3.10: The tractography, using the distance to the tumor for coloring. On the left local coloring is used, while on the right a single color per fiber is determined by taking the minimum distance.

3.5.3 Activation and tractography

The third edge of the triangle concerns the interaction between the fMRI activation and the tractography domains. This mode shows the explicit relationship between fiber bundles and fMRI activation areas.

As shown in Figure 3.11 on the facing page, each fMRI activation area is automatically colored with a distinct color and can be rendered using several different rendering methods as explained in Section 3.4.3. Fibers that intersect with an activation area are assigned the same color as that activation area. If fibers intersect more than one activation area, the colors of those areas are blended and assigned to the fiber. The specific colors that have been chosen, along with spatial cues, make it a generally straightforward task to judge through which activation areas a fiber has passed.

3.5.4 Combined visualization

Combinations of the three visualization modes described above can be used during exploration of fused MRI, fMRI and DTI datasets. In Figure 3.12 on the next page the tumor and its safety zone are shown along with a number of fMRI activation areas. Fibers that pass through the tumor safety zone have been colored turquoise, whereas tumors passing through activation areas but not through in the vicinity of the tumor have been colored the same as the activation areas.

Throughout the exploration process, parameters can be changed and visualization modes can be modified. The visualization remains completely interactive and supplies continuous feedback on all changes. This combined visualization mode is situated in the interior of the multi-domain visualization triangle.

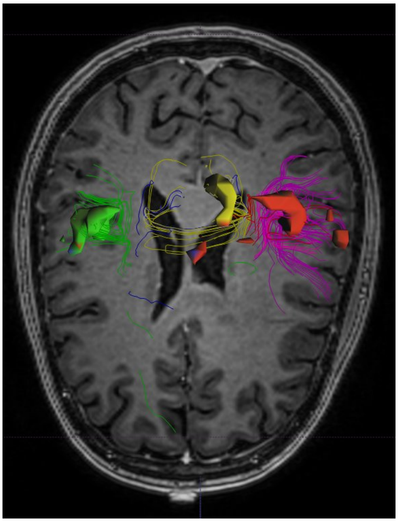


Figure 3.11: Activation areas and Tractography

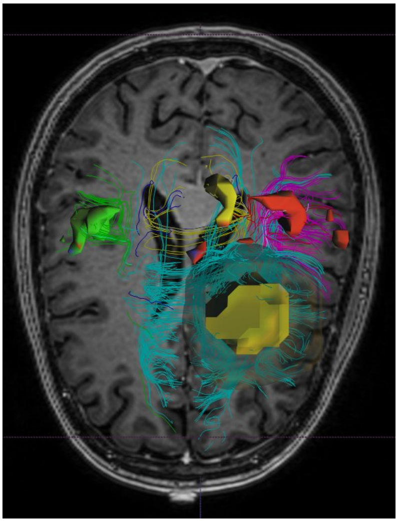


Figure 3.12: Tumor, Activation and filtered tractography

3.6 Discussion

The visualization options described in the previous section are all highly interactive, despite the size of the data. An essential design strategy to achieve this is to combine pre-generation of all possible important information with fast selection techniques. The pre-generated information includes the full-brain tractography, the segmented tumor and the activation zones, while the fast selection techniques (similar to those described in [BBP$^+$05]), filter this information to answer specific queries. Another design aspect is the strong integration of the information by representing everything in a single data space, so the relational queries (such as distances between fibers and tumor, and the fibers passing through functional activation zones) can be easily answered.

Experience has shown that visualization helps to explore the surgery zone and extract the most important information such as the vital functions to be spared and the crucial connections from the zone immediately adjacent to the tumor. Thus, the visualization tools can very well play a role that is supplementary to traditional surgical planning. The use of this information has given surgeons high expectations, even if clinical experience is still very limited and little test data is available. Although in this study real patient data has been used, we are still in a stage of exploring possibly useful visualization tools. So far, the mainly qualitative results have gained great interest and support from the end users.

However, quantitative information is also needed, for example for rigorous clinical testing, comparison between patient populations, and temporal comparisons (such as pre- and post-operative information). In a further stage even more accurate and reliable quantitative information is needed in calculating the risk and damage for a specific surgical approach. Another use for accurate quantitative information is the generation of planning information for use in an intra-operative optical guidance system (such as BrainLab).

To be able to generate both qualitative and quantitative information, a number of medical and technical problems have to be addressed:

- Accurate and (semi-)automatic segmentation of the tumor
- Modeling an access path for resection
- Investigation of the accuracy of fiber tracking near certain tumors (where the anisotropy is much lower compared with healthy brain)
- Investigation of the effects of damage to neural fibers, and how this can be measured by DTI

- Development of good indicators for functional impairment
- Good registration of all modalities, for individual patients and patient groups for statistical analysis
- Real-time deformation to keep pre-operative images registered to the intra-operative visualization

The ultimate test for this type of surgical planning system would be a comparison with strong indicators such as reduced time for the operations, less revision surgery or recurrence of the tumor, less functional damage, and overall improved results of resections. For the time being, we can only develop tools that may eventually be integrated in such a comprehensive surgical planning system.

3.7 Conclusions and Future Work

In this chapter we have presented an ordered toolset of visualization techniques for the interactive and simultaneous exploration of anatomical MRI, fMRI and DTI datasets with the goal of providing information on the resection area around a brain tumor. We have also presented a conceptual framework for this multi-domain visualization exercise.

Our toolset consists of a number of basic components that can be combined within the multi-domain visualization framework to create effective and interactive visualizations of tractography, fMRI activation areas and tumor locations. What differentiates our approach from other work in the field is the fact that it allows simultaneous visualization in all three domains and, very importantly, makes possible rich interaction between these three domains. For example, where other approaches require a tractography to be performed from a pre-selected region of interest before fusion with other modalities, our approach fuses the full brain tractography with the other domains and allows the user to filter information in the fully fused visualization.

We plan to continue our research into which visualization domain interactions are most applicable to and effective for specific clinical questions. We also intend to use this flexible framework to gain direct feedback on the usability of innovative multi-domain techniques in a clinical setting.

Acknowledgements

DTII makes use of the `teem` toolkit, available at `http://teem.sourceforge.net`.

Interactive Visualization of Multi-Field Medical Data using Linked Physical and Feature Space Views

4

Published as: Jorik Blaas, Charl P Botha, Frits H Post: *Interactive visualization of multi-field data using dynamically linked physical and feature space views*, in EuroVis: Joint Eurographics - IEEE VGTC Symposium on Visualization, 2007: 123-130.

After experimenting with single-view applications for diffusion tensor imaging, we expanded our views to include feature space representations. In this chapter, we introduce the multiple linked view metaphor for looking at volumetric multi-scalar data. Each voxel is not only represented as a pixel in a slice view at its physical position, but it is also represented by a point in feature space, positioned by its tuple of measured scalars. We show how two-way interaction can be accomplished by sharing selections and projections between feature space and physical views. A selection can be used in combination with pattern recognition techniques to find projections of the feature space data, which in turn can be used for interactive brushing to generate new selections. Feature spaces generated by filtering of the volume data especially, such as LH-spaces, turn out to be extremely useful in interactively segmenting multi-material boundaries.

Abstract

Multi-field datasets contain multiple parameters defined over the same spatiotemporal domain. In medicine, such multi-field data is being used more often every day, and there is an urgent need for exploratory visualization approaches that are able to deal effectively with the data-analysis. In this chapter, we present a highly interactive, coordinated view-based visualization approach that has been developed especially for dealing with multi-field medical data. It can show any number of views of the physical domain and also of the abstract high-dimensional feature space. The approach has been optimized for interactive use with very large datasets. It is based on intuitive interaction techniques, and integrates analysis techniques from pattern classification to guide the exploration process. We provide some details about the implementation, and we demonstrate the utility of our approach with two real medical use cases.

4.1 Introduction

Multi-field datasets contain multiple variables defined over the same physical domain. As this type of data is being used more often every day in clinical research and practice, it is becoming increasingly important to have techniques with which multi-field data can be effectively explored and visualized. Some examples of medical multi-field data are:

Multi-modality data PET-CT is a multi-modality type of dataset that combines CT (computerized tomography) with PET (Positron Emission Tomography) data. The CT data supplies anatomical information as context, while the PET yields functional data, for example on the precise location of tumors and their metastases.

Registered data Different datasets can be registered in order to extract more information from the combination. This can be done for different modality datasets of the same patient, datasets acquired at different time points or datasets of different patients.

DTI Diffusion Tensor Data is inherently multi-field data in that it contains at least one symmetric second-order tensor per data point. In the basic single-tensor model, there are six unique tensor elements per data point.

Data with derived features This is an important class of multi-field medical data where extra features have been calculated for each data point. For example, based on a CT dataset, the gradient or curvature volumes can be calculated. The purpose of these extra features is often to be used in subsequent segmentation, quantification or visualization steps.

Visualization and image processing techniques could be used by clinicians to make effective use of multi-field data for diagnosis, therapy planning and surgical guidance. However, before this stage can be reached with a particular type of multi-field data and application, visualization and image processing scientists have to develop and refine the necessary techniques for routine clinical use. This is a challenging task, as the necessary exploration of multi-field medical data is complex and there is very little supporting infrastructure. In fact, there is currently no unified visualization approach that satisfactorily addresses this problem.

The approach that we present in this chapter was developed for these scientists and has the main goal of facilitating the crucial exploratory stage of the clinical algorithm development pipeline. Our approach facilitates numerous often-occurring tasks during the exploration of multi-field datasets, for example:

- Investigating combinations of features for finding spatial separations of point sets in physical space (segmentation)
- Classification of feature point sets (pattern classification)
- Finding clusters, patterns, and relations in the data
- Selective visualization based on feature value ranges
- Investigation of possible features and their suitability for segmentation or as pattern classifiers

Our approach is based on the use of interactive dynamic coordinated views. This is based on related work concerning multiple coupled views and multi-field (non-medical) data. Our solution is set apart by three major factors. Firstly, it has been optimized for highly interactive use with specifically multi-field medical datasets. In contrast to problems in information visualization, multi-field medical datasets might have a smaller number of features or variables per data point, but can be very large. The inherent physical component of medical data, i.e. the fact that it is continuously defined over a spatial domain, requires a different approach. Secondly, our approach supports arbitrary coordinated projections of the high-dimensional multi-field feature spaces. Existing solutions support selecting discrete axes or features from feature space to act as the spanning vectors of two-dimensional scatter plots. Through an intuitive interaction technique, our framework supports *arbitrary* projections of feature space. In other words, each axis of the resultant coordinated scatter plot can represent a user-defined combination of features. Finally, we have integrated a number of analysis techniques that are commonly used in pattern recognition, such as clustering and transformation algorithms, linear discriminant analysis, and principal component analysis. These techniques can be applied interactively to result in a more effective exploration.

The rest of this article is structured as follows. We discuss related work in Section 4.2, and in Section 4.3 our approach is explained in detail. Section 4.4 gives design considerations for implementation, and some performance figures. In Section 4.5, two real medical applications are described, demonstrating the exploration process. We present conclusions and future directions in Section 4.6.

4.2 Related Work

XmdvTool [War94] combines a number of techniques for the visualization of multivariate data. These techniques include scatter plots, glyphs, parallel coordinates, hierarchical techniques, and importantly, linked N-dimensional brushing. Linked brushing refers to the functionality that selections made in any one view are reflected in all other views on the same data.

The Grand Tour [Asi85] is a method for automatically selecting a continuous sequence of 2D projections of N-D data that are asymptotically arbitrarily close to the set of all possible projections. Practically, this allows one to visualize a high-dimensional dataset with a single animation consisting of a sequence of 2D projections [BA86].

XGobi [SCB98], as well as its more modern successor GGobi [SLBC03], also supports linking of views, various types of scatter plots, 3D rotation views, parallel coordinate views and linked brushing. It distinguishes itself from XmdvTool in that it supports a number of extensions of the Grand Tour method described above.

Where XmdvTool and the Gobi systems are applications, Orca is a pipeline-based programming framework for building visualization applications for multi-variate data [SRL$^+$00]. The framework supports multiple linked views, linked brushing and Grand Tour like automated generation of sequences of 2D views. The focus was more on the architecture and design of the programming framework, and not so much on the application of the methods themselves.

WEAVE [GRW$^+$00] is to our knowledge the first published work explicitly integrating linked 3D scientific visualizations with multidimensional statistical representations. All views are coordinated and the statistical representations support linked brushing. The 3D visualizations do not support brushing or selection.

The SimVis software system [DGH03] builds on the ideas presented in WEAVE in that it combines scientific visualization views with information visualization views.

It extends this work by presenting a feature definition language (FDL) that formalizes the subsets of data that can be selected by for instance linked brushing.

WEAVE and SimVis depart from the multi-variate information visualization tradition by integrating scientific visualization views, in the form of volume visualizations. However, neither of these systems support linked brushing or selection in the scientific visualization view. Also, whilst Grand Tour-like techniques generate sequences of arbitrary 2D projections of N-D data, all other work discussed in this section that support the interactive definition of 2D projections of N-D data do so only by the selection of two discrete variables from the multi-variate data.

Our work continues in the tradition of WEAVE and SimVis by combining scientific visualization (physical space) and information visualization views, but makes the following new contributions: it supports the interactive visual definition of arbitrary 2D projections of N-D datasets, it also allows linked brushing and selection in the scientific visualization view and, importantly, it allows features selected in information visualization to be directly visualized in the physical view. In addition, we have integrated techniques from pattern analysis that can be interactively applied to enhance the data exploration process.

4.3 Methods

Our approach employs dynamic multiple-linked-view techniques in an environment where tight coupling enhances the users perception and exploration of the data.

There are four main architectural components in our approach. Firstly the datamodel presents a clear and straightforward manner of representing structured volumetric multi-scalar data. Secondly the view component contains the visual components used for user interaction. Thirdly, a coupling component describes the bindings between the views. Finally, a processing component contains the data-processing methods that provide high-level control.

4.3.1 The data model

While the medical data we are focusing on are mostly defined on regular grids, our data model supports arbitrarily positioned measurements. The basic element in our model is a data point, which is the set of measurements that share a location.

Each data point contains the physical coordinates at which it was defined, and a feature space part which represent the measured or derived values. Each data point can be part of a single selection set, which is identified by a numerical identifier.

Two feature types: measured and derived

There are two types of features in our data model. Firstly, the values directly sampled on the physical domain are called the measured features. Since the exploration process later on uses each data point as a separate entity, there is no locality involved that relates for example the value of a point to the value of its neighbors. However, in segmentation this type of local information can be very helpful (see also Section 4.5.1).

For this reason our data model adds derived features. The values for these derived features are computed using locally available measured features. Examples of these computations are gradient computation, denoising and morphological operations.

The main advantage of introducing these derived features alongside the original data is that any combination of derived and measured features can be studied during exploration.

4.3.2 Views

The components in this part of the architecture present the user with a direct visualization of the data, with which the user interacts to explore the data. The user is presented multiple linked but independent views of the same data. Each of the views shows either a physical space or a feature space representation of the dataset.

The physical-space view is based on the actual physical locations of the data points, so that neighboring points in these views represent physical proximity. Usually this mapping is very straightforward, such as a slice view or another form of projection.

The feature space view complements this by showing only the feature values, unrelated to their physical location. Points close to each other in these spaces share the same characteristics, but can be positioned anywhere in the physical volume.

High-dimensional projection of feature space points

Since feature spaces are often high-dimensional, a reduction to lower dimensions is often needed for display. Any two-dimensional display reduces the dimensionality of the feature points. We have chosen to let our feature space view show a projection of the feature values. The high dimensional points are projected onto a plane that can be arbitrarily positioned in feature space.

These extra degrees of freedom over feature selection techniques allow the user to study any linear combination of features as a separate feature. Figure 4.1 shows how an anti-diagonally positioned plane is used to study the difference between two feature values.

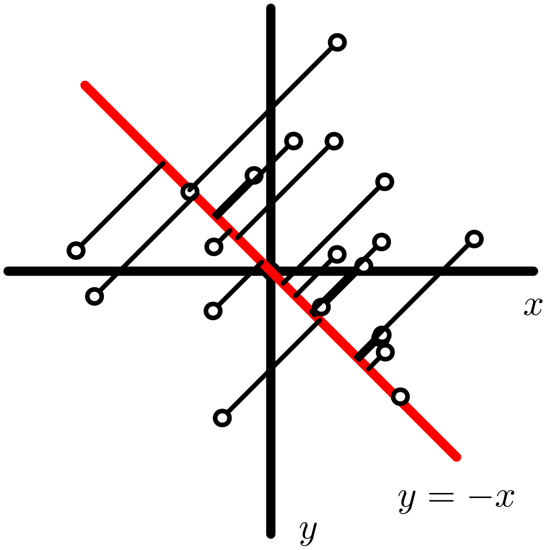


Figure 4.1: A projection plane that maps (x, y) to $x - y$, which can be used to study the difference between two feature values directly.

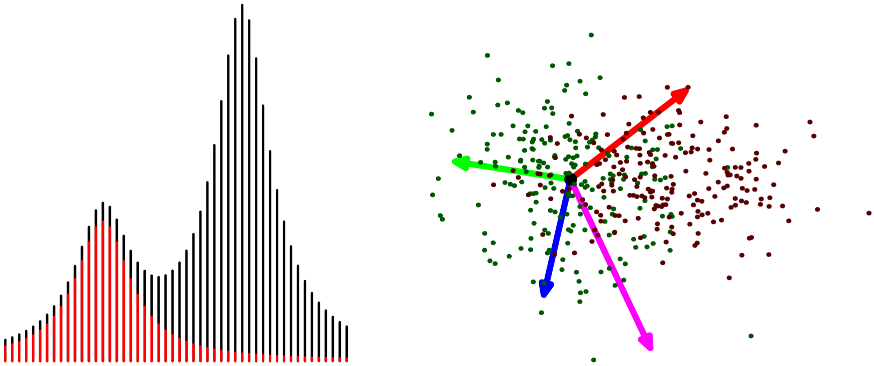
Compared to the feature selection techniques that are most often used, this provides a number of advantages:

1. The relation between several features can be studied at the same time
2. Interactive manipulation of the projection plane provides extra insight into the higher dimensional structure of the feature space
3. Differences between features can be used in the same way as the original features, merely by repositioning the hyper-plane

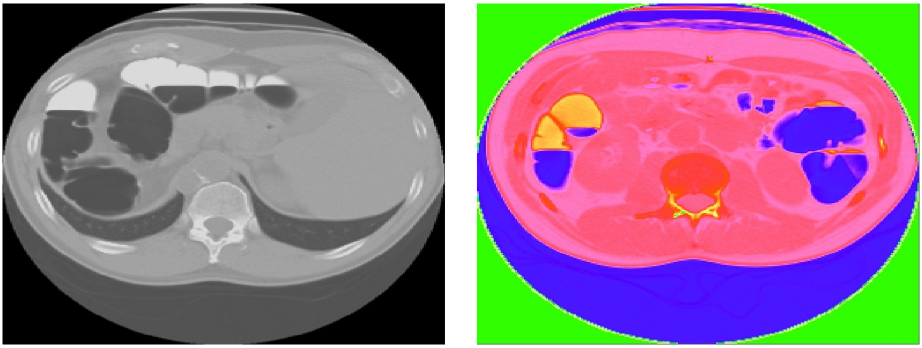
Usually the interest lies in finding clusters, outliers or separation planes. For these measures a projection to one or two dimensions often suffices. Once the data has been reduced to two dimensions, we can employ a scatter plot, or we can use histograms for one dimensional projections.

Main view components

While feature space views provide information about the similarity of points, it is of prime importance to be able to locate these points in the physical space. Our



Histogram Scatter plot



Slice-Viewer Color Slice-Viewer

Figure 4.2: The four view components: histogram, scatter plot, slice-viewer and pseudo-color slice-viewer. For color plate, see Figure A.1 on page 178.

approach currently includes two physical-space view components and two feature space view components (see Figure 4.2).

The basic physical-space view is the slice-viewer. The slicer shows 2D orthogonal planes in physical space, on which the feature space values are represented as colors or grayscales. To map feature space values to a color, a projection of the feature values is needed. This projection can be as simple as selecting a single feature and using that as a grey value, but it can also involve an arbitrary projection that maps a set of features onto RGB colors.

The scatter plot component has some improvements over the scatterplot normally used, in order to deal with the large number of points visible at the same time, and the high dimensionality of the data. Each data-point is rendered as a single point. The scatter plot uses additive rendering to give a good impression of the density where points overlap. Where often in high-dimensional data-analysis the scatter plot is used to visualize only two selected features, we allow arbitrary combinations of features to be visualized.

Dynamic axis manipulation

Since high-dimensional projections are difficult to control, we have chosen to use a straightforward direct control mechanism. We describe the projection-plane as being spanned by two (high dimensional) vectors in feature space. Instead of modifying these vectors directly, we transform the two n-dimensional vectors into n two-dimensional vectors, so that each can be displayed and manipulated in the view's coordinate system.

This provides the user with a single two-dimensional vector for each feature (see Figure 4.3). These vectors are represented as visual axes in the scatter plot, and can be manipulated dynamically through pointing and dragging.

Since each feature has its own manipulable axis, there is a direct link between the movement of the axis and the values of the points for that feature: The larger the value the more the points move around when the axis is manipulated. In this way, it is easy to pick up a feature and see how the currently visible points are distributed along this axis.

Selection and filtering

While a view of all data points may give a good overview, filtering is required to be able to focus on a specific subset. For example, the removal of air in a medical dataset may be necessary to see all the details in the rest of the data.

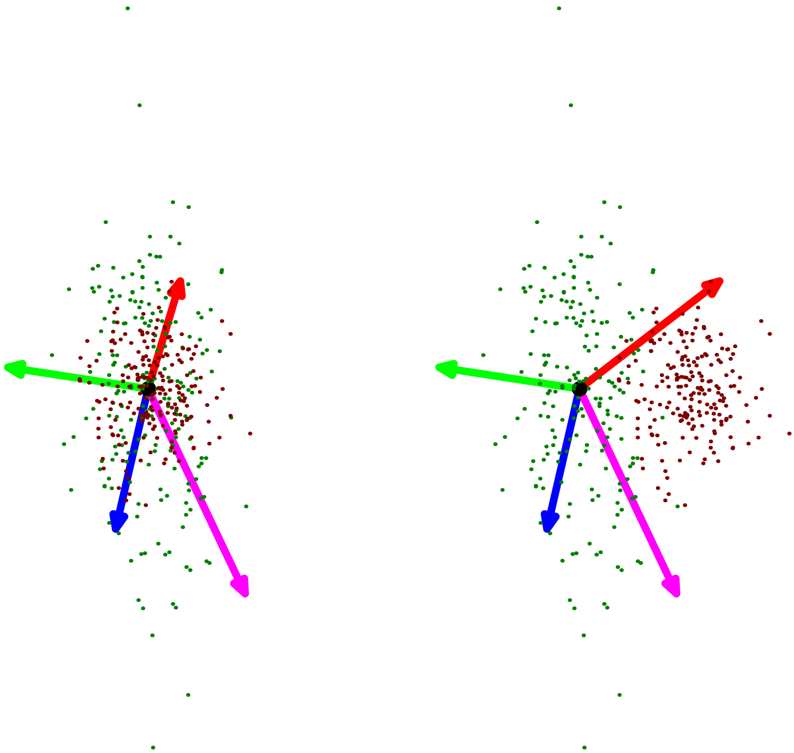


Figure 4.3: Two clusters of points separated by a single feature. The separation can easily be found by moving the axis (red) corresponding to the separating feature. After moving the axis (right), the separating clusters can be distinguished. For color plate, see Figure A.5 on page 182.

To support making these selections, we currently use multiple rectangular brush areas, which can be used in physical space as well as in feature space. In this way making a selection of a region of interest works in exactly the same way as marking all points with the same properties (such as air).

To filter out uninteresting points, any selection can be hidden and consequently removed from further processing steps. While a single selection set is often useful enough, more advanced segmentation/exploration methods may require multiple selection sets to be active simultaneously.

For this purpose, we provide the possibility to mark and store selections with colors. Consequently points in the dataset can be divided into groups, each visible with a unique color. Each point belongs to a single group. With our selection system the visibility of selected sets of points can also be toggled, so that only the region of interest is visible and clutter is reduced.

4.3.3 Coupling

While each of the view components presented in the previous section can by itself be used to explore the dataset, the combination of multiple views is much more powerful. For that reason, we have chosen to use multiple simultaneous viewing components, between which a strong coupling is kept. Selections made in one viewport are immediately propagated to other viewports, and also projections can be linked between views. This section explains in more detail how the views are coupled and how this enhances the data exploration process.

Tightly coupled views

Our approach maximizes the link between the visible views in a highly interactive setting. The dynamic propagation of selection sets and filters to the other views gives direct feedback of the links between the visible points. The use of color coded, additively rendered, points adds to this experience, since the different selection sets can be distinguished in any view.

Since the brushing feedback is interactive, we can quickly investigate whether a group of points visible in one projection is also a group in a different projection, by simply making a selection in one view and by watching the resulting changes in the other views.

Each view by itself presents a complete representation of the active dataset. The filters introduced earlier are always applied to all views at the same time, and all views share the same colored selections. This helps to maintain a sense of uniformity, especially when many viewing components are active simultaneously.

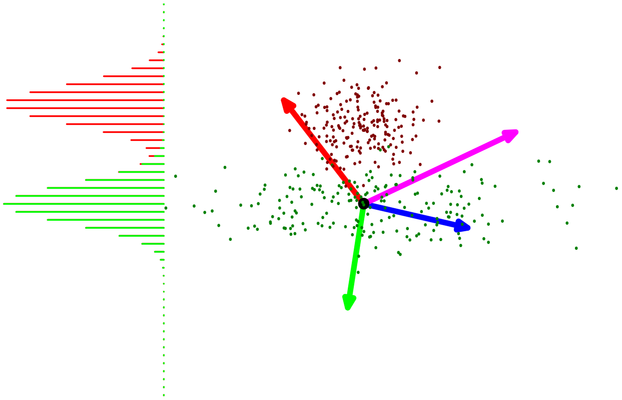


Figure 4.4: A histogram linked to a scatter plot. The histogram uses the Y coordinates of the projected points in the scatter plot as a source. For color plate, see Figure A.6 on page 182.

Linking of projections

While the selection sets are always shared between all views, some other properties can be explicitly linked between a pair of views. For example, the histogram view can be directly linked to the axes of a scatter plot view, so that the histogram uses the projected values. To examine how well two classes can be linearly separated by a one-dimensional separator, we can set up a scatter plot showing both groups of feature-points that we wish to separate and then link the y-axis of the scatter plot to our histogram view. When linked, the histogram view shows a 1D histogram of the projected points, keeping selection colors intact (see Figure 4.4).

This type of linking of projections is also applicable to the color slice viewer. Since each physical point in the slice viewer has a feature value, the same projection can be used to produce color values. For a single axis link, the color slice viewer shows the projected one dimensional features using a pre-set colormap. To use a higher dimensional projection, each axis can be linked to a color channel of the image.

Changes in projection are, just as changes in the selections, directly propagated into the linked views. Hence, feedback from the other views can directly be used in the interaction loop for finding a suitable projection that separates the points in the desired manner.

4.3.4 Processing

While the direct manipulation of the scatter plot's projections is effective to gain insight in high-dimensional spaces, it is often a lot easier to use statistical analysis techniques in order to find proper projections. This component contains all methods that use calculation on the (selected) data to influence the way it is displayed by the visual components.

Using selections to create new projections

Exploring high dimensional feature spaces can be very difficult, even when direct links between feature and physical space are used to put the data into context.

An often recurring task is finding a projection that best shows a selected point set, or that best shows the differentiating features between two sets of points. In the next sections we discuss both situations, and how the processing methods available in our system can assist in these tasks.

In the first case, the high dimensional feature space often shows quite a bit of correlation, which can get in the way of finding intricate details in our area of interest. In these cases, manually finding the projection that keeps most of the important information intact may be tedious and difficult. Therefore we provide the user with an automated way of finding these projections, by the use of principal component analysis (PCA).

In the second case, where the interest lies in separating two sets of points (air and tissue for example), we are often able to make two selection sets typical for each 'material' by brushing in the slice viewer. Once a selection has been made for each of the two materials, linear discriminant analysis (LDA) is applied to find the optimal separating plane between the two sets. This plane provides us with a single axis for which the spread between the two groups of projected points is maximized.

In all cases the axes we find can be directly applied to any view, and to keep a sense of coordination, we change the current projection to the new projection in a smoothly animated fashion.

For some situations brushing can be very tedious, especially when the clusters in feature space have non-linearly separable shapes. Therefore a third processing method is available in the form of k-nearest-neighbor (kNN) classification. This processing method aims to expand the currently painted selections to points that are similar, or nearby in feature space. It takes as input a partially marked set of points, then for each unmarked point it finds the closest point in feature space and copies its selection ID.

This type of processing method is extremely valuable in a paint-by-example setting. An initial set of selections is made for a number of important tissue types, after which the kNN process expands the selection to the whole dataset, resulting in a segmented volume.

4.4 System/Implementation

We have implemented the above methods in a prototype system, while keeping in mind the following goals:

1. Clear visual presentation of all components
2. Interactive display and manipulation at high frame rates
3. Good scalability up to datasets that contain tens of millions of points

The prototype system has been implemented in C++ using OpenGL for rendering, for maximum flexibility and speed.

4.4.1 Arbitrary number of views and components

The implemented system allows an arbitrary number of view components to be active at the same time. Figure 4.5 shows the layout of the screen during an exploration session. To simplify the user interface we have chosen a regular gridded layout. Each cell in the grid can contain a single viewing component, such as slice-viewers or scatter plots.

The configuration of these components can be stored and loaded for task-specific exploration scenarios. The coupling between views can be modified at run-time, but a pre-set combination of couplings is initially present. Most parts of the user interface are stateless, so there is no hidden internal state hindering the user experience.

4.4.2 GPU-efficient data structures

To achieve optimal interactivity, even when processing datasets containing millions of points, efficient data-structures are needed throughout the processing pipeline. To exploit modern graphics hardware, our system uses GPU-friendly datastructures. The feature points are stored on the graphics card in vertex-buffers as well as in main memory. Since the high-dimensional projections used throughout the view components are all written as GPU shaders, almost no communication is needed between GPU and CPU, which usually forms a bottleneck in these kinds of applications. This allows us to transform and draw over 80 million points per second for 4D datasets on a 2GHz AMD64 system with an ATI x1900 videoboard.

The state of the selection groups has to be kept synchronized between the two storages, for which a simple but effective update mechanism was implemented. All

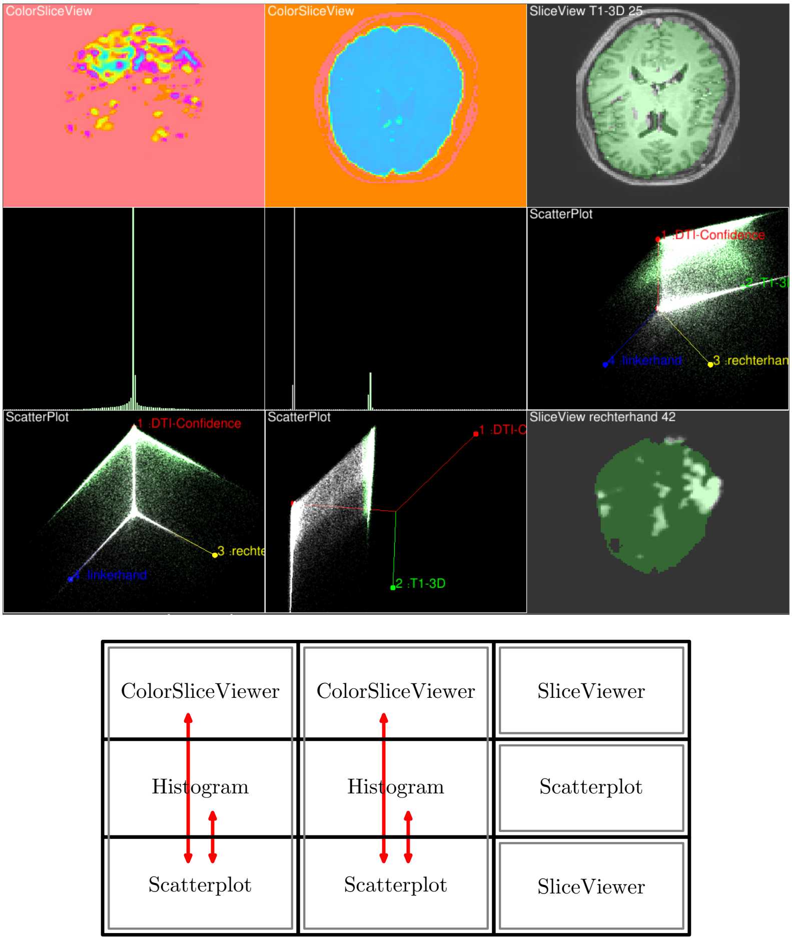


Figure 4.5: Top: The interface during a user session. Bottom: The component grid, where red arrows indicate the coupling between components. For color plate, see Figure A.7 on page 183.

Number of features	**MPoints/sec.**	**Frames/sec.**
4	80.2	61.2
8	44.2	33.7
12	41.7	31.8

Table 4.1: Rendering performance for the scatter plot with 1.3 million feature-points for a varying number of features.

changes are propagated from the main memory to the vertex buffers in a batch-wise fashion, updating as few points as possible for a given optimal batch size.

4.5 Applications

To show the applicability of the exploration techniques we have used the presented methods to perform a number of often required exploration tasks on medical data.

4.5.1 Material classification in CT virtual colonoscopy

In virtual colonoscopy, an important processing step is to find the surface of the colon wall, so that polyps can be detected. To gain more contrast in differentiating the wall, scans are made with a contrast liquid and extra CO_2 gas is added inside the colon. Even with these measures, segmenting the colon surface can be tricky because of partial volume effects that occur on the boundary between tissue and air, or tissue and contrast medium.

The following sections describe how our exploration methods can be applied to this type of problem. The data exploration in this case involves two basic steps. Firstly, we describe the exploration process for finding a linear separation by brushing. Secondly we show how the additional derived features can be used to find a better segmentation.

We start with a simple scalar CT dataset, which we load into a 2×2 gridded user interface as shown in figure 4.6.

Separating tissue, air and contrast medium

Since the main interest lies in the colon wall, we build an initial segmentation that includes the three main materials. Using the brushing interface of the slice-viewer, we mark parts of air, contrast medium and tissue, as seen in Figure 4.7(a).

Once the selection has been made, we want to expand this selection to similar points. The histogram at this point shows fairly well separated peaks, suggesting that a simple classification/separation can be found. By selecting the kNN classification process (see Section 4.3.4), each unmarked point is assigned the selection ID of the closest marked point in feature space. The result of this processing step is a fully marked dataset, as seen in Figure 4.7(b).

This initial segmentation however has misclassified the boundary. The boundary between air and contrast medium was wrongly classified as tissue. This is caused

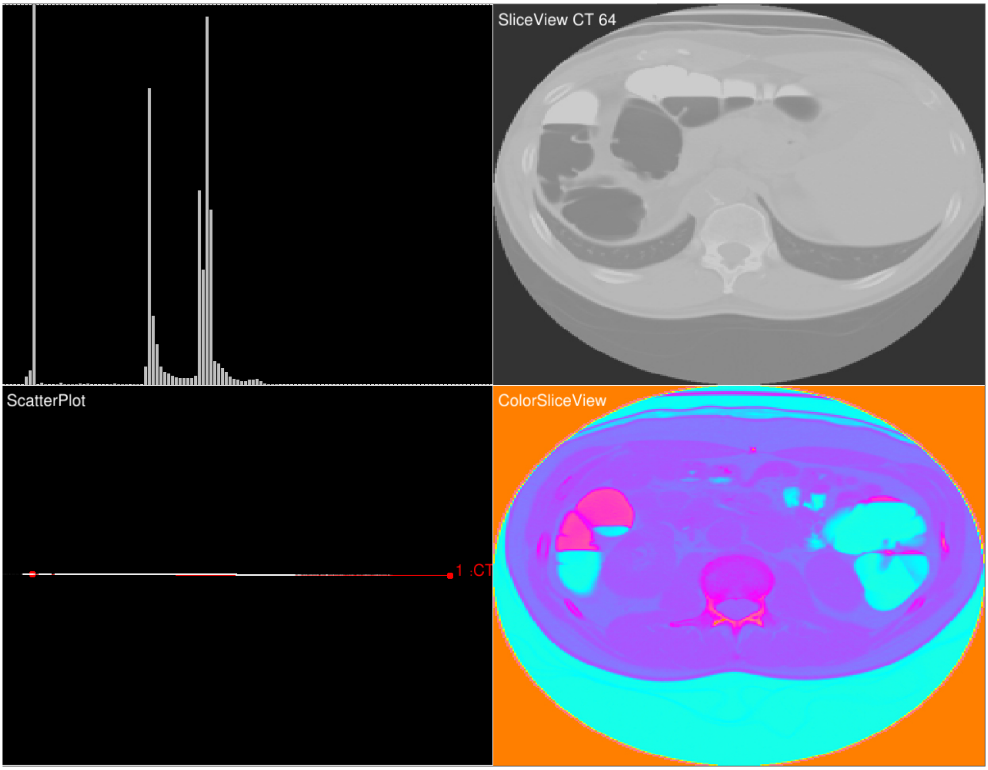
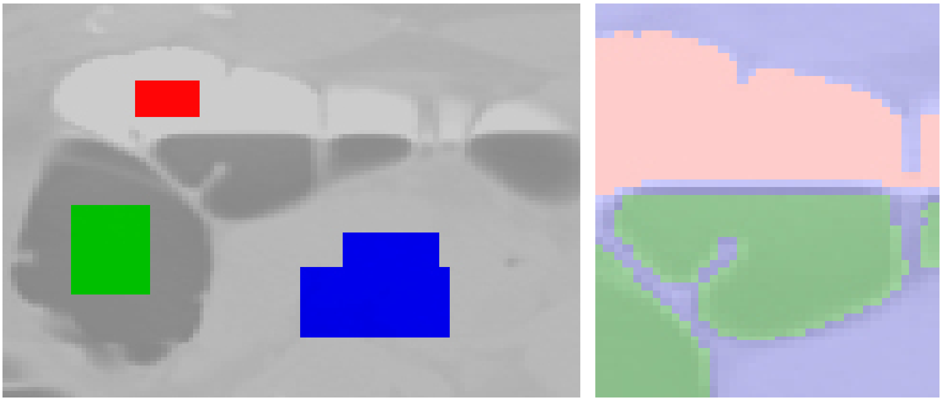


Figure 4.6: The gridded 2×2 *layout of the user interface with the CT dataset loaded. From top left to bottom right: histogram, slice view, scatter plot, color slice view*



(a) Initial Selection (b) Expanded Selection

Figure 4.7: (a) shows the initial selection created through brushing in the slice-viewer. Red represents contrast liquid, green represents gas and blue marks tissue. (b) shows a detail of the expanded selection through application of the kNN classification process, note that the boundary is not correctly classified. For color plate, see Figure A.8 on page 184.

by the partial volume effect, whereby the voxels on the boundary have the same scalar value as surrounding tissue.

Introduction of derived features

Since direct classification of the voxels at the air-contrast boundary is hampered by the partial volume effect, further steps are taken to introduce derived features (see Section 4.3.1). Derived features are computed based on physical-space neighborhoods, so a local gradient or minimum/maximum can at this point be added to the dataset.

We choose to add LH features [SVSG06]. The LH values represent the local minima (L) and maxima (H) by following the local gradient direction. In this way, the boundary between air and contrast medium is identified by an L value corresponding to air, and an H value corresponding to contrast medium.

By examining the scatter plot for the newly added features (see Figure 4.8), we already see a number of clusters emerging. Exploring these interactively by brushing in the scatter plot and watching the slice-views change, we note that (as expected) transitions between materials are presented by clusters in LH-space.

To test how well our LH space helps us to separate the materials, we start by brushing a new selection set which explicitly includes the boundary surface (Figure 4.9).

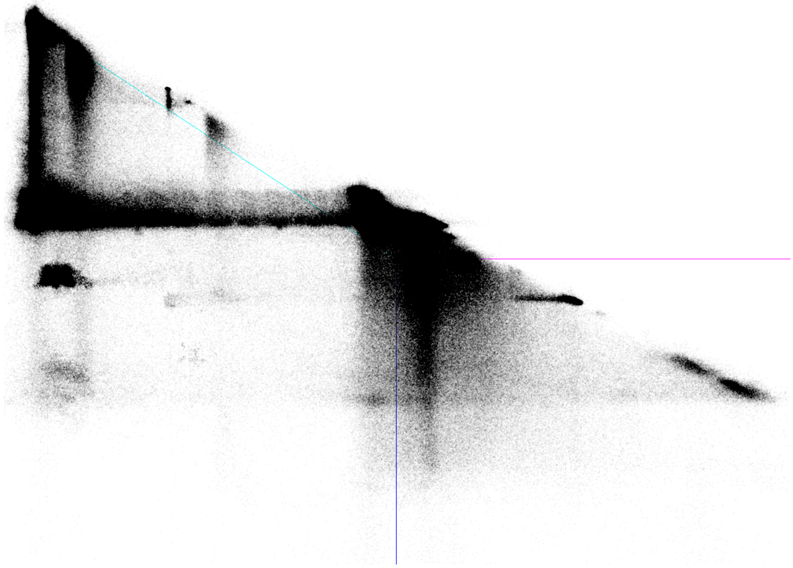


Figure 4.8: Scatter plot showing the distribution of points in LH-space. Horizontal axis is L, vertical is H.

We expand the selection through kNN and note that the two extra dimensions (L and H) in our feature space help to find a proper segmentation, as shown in Figure 4.9. Note how the materials are correctly classified and the colon surface is correctly segmented; the boundary no longer poses a problem.

4.5.2 Functional MRI combined with structural MRI

A number of different scanning techniques are currently employed in brain imaging through MRI. The different techniques provide different types of 3D volumes, and are often used together. We focus on neurosurgical planning for the resection of brain tumors, where it is of prime importance to prevent damage to the important functional areas. For this purpose, multiple scans are made. First a structural MRI to provide a whole-brain scan as a reference image; second, multiple functional MRI scans map the activation areas for the main motor tasks, such as hand and foot movement.

For each of the motor tasks, the functional MRI scan is preprocessed to provide a volume containing the Z values, which describe the correlation of each voxel to the specific task. In total, we have five measured features: one structural MRI and four functional MRIs (for hand and foot motion on each side).

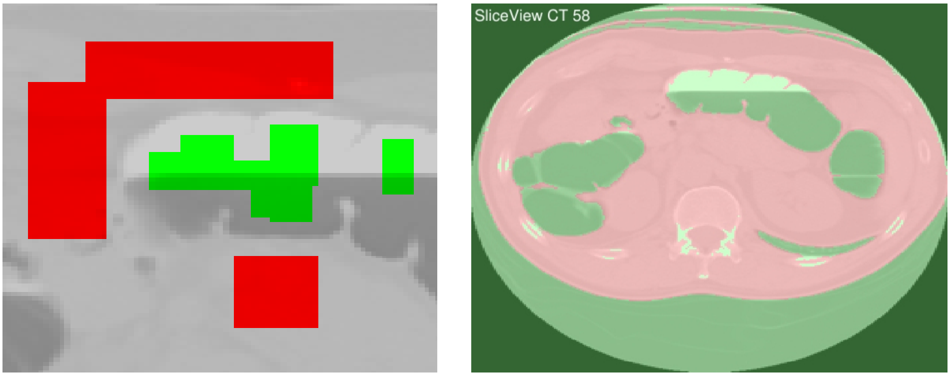


Figure 4.9: Expanded selection through application of the kNN classification process on the feature space that was expanded with the derived LH feature. For color plate, see Figure A.9 on page 185.

Mapping the motor tasks

To find out the positioning of the location of the major activation areas, we used histogram brushing on the fMRI features to threshold the Z values. In this way, each motor task area is marked with a unique color. The sliceviewer can then be used to inspect these areas as they are overlayed on the structural MRI.

Adding a new derived feature: tumor distance

In planning the surgery, it is often important to use a safety margin around the tumor and see if any functional activation areas are within this range. For this, a semi-manual segmentation of the tumor is made beforehand, which we can use to introduce a new derived feature. We apply a distance field processing step, generating a distance map for the tumor, in which each voxel stores the minimum distance to the tumor. This volume can consequently be added as a new derived feature.

Using brushing and filtering on this distance feature, we can now focus on the area inside the safety margin around the tumor. Since filtering does not interfere with the selections assigned previously, the activation areas inside this area can be directly studied.

4.6 Conclusions and Future Work

With this work, we have made the following contributions:

- We presented a dynamic coordinated view approach for the interactive visualization of (medical) multi-field data.
- We showed how scatter plots can be created with arbitrary projections of high-dimensional feature space.
- We demonstrated how techniques from pattern analysis can be interactively applied to facilitate a visualization-based multi-field exploration process.
- Using two case studies, we demonstrated the application of our approach to real medical data.

The techniques we have described can be used for a wide variety of tasks. Especially a quick evaluation of the relevance of certain features for a given problem is extremely useful in practice. Manual exploration of 2D feature space projections is augmented by the PCA, LDA, and kNN analysis, and in this way the user is guided quickly toward important views of a high-dimensional feature space. Throughout, two facilities are essential: the dynamic feedback displayed in multiple views, and the direct axis manipulation. Of course the system as it is now is only intended for users with a background in visual data analysis. For clinical use, a task-oriented user interface layer is required, which will require a major design effort.

There are many possible directions for future research. The current system can be extended with more components for analysis and viewing, such as volume rendering and parallel axis views. Also, more support for time-varying data can be added, such as adding local change metrics as new derived features.

For better support of the interaction and exploration process, hierarchical selection techniques could be added. This could be combined with a logging mechanism to document the analysis process and facilitate backtracking.

Visual Data Exploration Across Multiple Patients, Image Modalities and Timepoints

5

While the previous chapter focused on the visualization of a single multi-scalar volumetric dataset, the same techniques can also be extended to function at a higher level, allowing the handling of far larger datasets. This chapter describes the extension towards the study of groups of patients over time. We use a linked hierarchy of features and derived features over different domains. At the top of this hierarchy, a data-point corresponds with a single patient, while at the bottom of a data-point corresponds with a single voxel. The values accross these domains are represented as points in a feature space that is parametrized by values aggregated over the lower levels of the hierarchy. A strong coupling allows us to map points inside the domain of individual patients to voxels, and vice versa. Similar to the example describing the map of the Netherlands, values can be transferred from one domain to the other by aggregation or sampling. Since for each of the twenty subjects six full MRI scans were taken, special care had to be taken to allow loading all datasets at the same time. We implemented a disk-based on-demand caching approach to overcome the limitations of the computer's working memory.

Abstract

In this chapter we present a conceptual framework for the interactive exploration of large and complex multi-field medical data. Using a powerful domain abstraction, data can be explored simultaneously in multiple domains, providing dynamic linkage between views on per-patient, per-group or per-voxel data. A domain is the space on which a feature is measured. A feature is defined as a function that maps a point in the domain to a scalar value. Features can be explored using a spatial viewing component (volume rendering or slice viewing), or using scatter plots and histograms. Selections, made by interactive brushing, can be linked across different views in a dynamic fashion with an intuitive drag-and-drop interface. We show how linkage between views is used to explore the features defined across different domains. We first introduce the key concepts and methods, and then we demonstrate the utility of the presented methods by working out a use case with a dataset of a longitudinal study of multiple sclerosis, including multiple modalities of multiple patients over multiple timepoints.

5.1 Introduction

In longitudinal medical studies, such as retrospective group studies, large amounts of multi-modal data are acquired of multiple patients at multiple time points. The goals of these studies are to study disease evolution and to find new image-based biomarkers for the early detection of disease. More specifically, researchers examine the multi-modal data for time-dependent patterns, as well as for differences and similarities in groups of patients, that can yield information concerning disease evolution.

Currently an indirect approach is being used: Pre-defined image-derived parameters are calculated and correlations between these parameters and disease outcomes are sought. Feedback is only available once a parameter has been calculated for all available data and the relevant correlation metrics have been studied.

A more direct approach that enables the researcher to explore this type of data with interactive visualization techniques, guided by continuous feedback and the exploratory application of pattern analysis techniques, has the potential to expose patterns in the data more effectively. However, there are currently no visualization approaches that have been designed for this type of data. In addition, current interactive systems do not scale to the magnitude and heterogeneity of longitudinal study data.

In this chapter, we present a generic framework for processing longitudinal data, as well as the specifics of an implementation that enables the interactive visualization of these large and heterogeneous datasets. The framework consists of a set of concepts and accompanying abstractions that can be used to define the processing that is required to do visualization for group studies. Analysis techniques can be interactively applied to any combination of patients, timepoints or modalities. Additionally, our implementation scales up to the magnitude and heterogeneity of longitudinal study data, enabling the interactive exploration of a complete study dataset consisting of a number of patients, modalities and timepoints.

Our data processing pipeline can be interactively modified and the user can quickly focus on specific features or results using linked views. This provides an insightful way of generating new hypotheses. The direct feedback also helps to find errors and correlations in the data and the intermediate results, while a layered exploration process helps to visually pin-point the source of such findings.

The main contributions of this chapter are:

- A conceptual framework for describing all processing steps required during the interactive visualization of longitudinal study (and other multi-field) medical data.
- An integrated approach and demonstrator implementation, based on the terminology above, for the high-level visual exploration of longitudinal study data. Visualization and processing techniques can be interactively applied to any combination of patients, modalities and timepoints.

5.2 Related Work

The medical data we are studying consists of multiple patients, modalities and timepoints. This chapter is concerned with interactive visualization techniques that facilitate the finding of medically meaningful patterns in such data. Discovered patterns represent structural and functional changes, both across patients and for each individual patient over time, that are important for studying disease evolution.

Bürger and Hauser give a comprehensive and state-of-the-art overview of visualization techniques for multi-variate scientific data [BH07]. Our approach combines real-time multi-modal volume rendering and linked projections of highdimensional scatter plots and other statistical representations. In the WEAVE system of Gresh et al. [GRW$^+$00], this combination was investigated for the first time. A number of aspects, especially with regards to the specification of this kind of combined visualization, were formalized by Doleisch et al [DGH03].

Blaas et al. augmented this approach with techniques from pattern analysis and applied it to multi-field medical data [BBP07]. However, this work focused on single timepoint data. Similar techniques were later applied to dynamic 3-D perfusion data and yielded promising results [ODH$^+$07]. Fang et al. employed a different approach, based on the similarity between position-dependent time-activity curves, to visualize dynamic 3-D medical data [FMHC07]. In both these cases, the focus was on studying the dynamic behaviour of perfusion in a single patient. We are interested in investigating group behaviour over time.

With regards to the application field, the visualization techniques for time-varying MRI data presented by Tory et al. in 2001 are relevant [TMA01]. However, their techniques dealt with the focused visualization of per-patient structure shape changes, whereas our approach deals more with interactively visualizing and processing a complete longitudinal study dataset consisting of a number of patients.

5.3 Methods

While visualization of volume data has become increasingly popular, it is often considered separately from the visualization of statistical results and aggregated values. The concepts presented in the following sections abstract the data in such a way that this separation is no longer necessary, while still providing a wide range of exploratory operations.

5.3.1 Preliminaries

We define *multi-field data* as consisting of multiple variables defined over the same physical domain. This implies that the data is already registered, which in some cases requires extra pre-processing steps. Imaging techniques which produce inherently multi-field data, such as dual-energy CT and Diffusion Tensor MRI, are becoming increasingly common. Moreover, this type of data is often acquired over multiple time points, generating large and complex datasets.

Research questions that commonly occur in these studies include finding trends over time, correlations between patient groups, or even bio-markers that can be precursors to disease. Our high-level conceptual framework is designed to structure the underlying data in such a way that visual exploration of extremes, trends and correlations becomes possible.

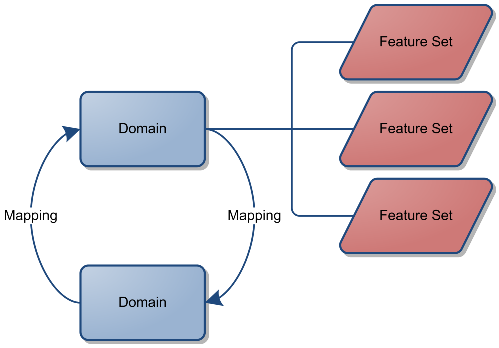


Figure 5.1: Diagram showing the interaction between Domains, Mappings and Features. Each domain can have an arbitrary number of features. The mapping links two domains by defining how their points correspond.

5.3.2 Concepts

The concepts presented in this section consists of three important parts: *domains*, *features* and *mappings*. In short, a domain defines the structure of a space, a feature defines the values that are defined on that space and a mapping defines how points in one domain correspond with points in another domain (see Figure 5.1). With these three concepts as a basis, we can define *tags*, *aggregations* and *filters*. Tags describe the metadata of features, so that groups can be formed. As a layer above mappings, aggregations define how to reduce sets data points. Filters are used to make a selection of points based on their values. The following sections explain these concepts and their relation in more detail.

Domain

The medical imaging data that our framework focuses on is usually defined over a 3D spatial domain, such as images acquired by MRI and CT scans. However, in the process of analysing such data, different domains are often used as an intermediate representation. For example, a histogram of CT values does not contain any of the spatial characteristics of the original data.

We have chosen to abstract these underlying spaces using a *Domain* concept. A *Domain* is the space on which a value is measured. For common imaging this is a regularly sampled rectangular volume, but our concept of a domain also extends towards non-metric spaces (see Figure 5.2). Different types of domains have different properties. For example, the bins of a histogram form a non-metric domain that still has a concept of a neighbourhood.

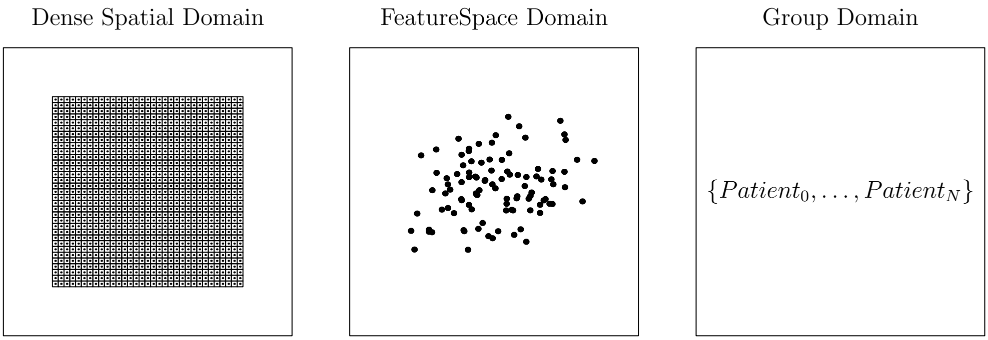


Figure 5.2: Three examples of domains that represent the underlying structure on which values are measured: a densely sampled spatial domain, a feature space domain and a group domain.

The group of domains that we most often see in the physical world is what we call the *spatial* domain. If a domain has a densely sampled regular structure then it is classified as a spatial domain. The properties and the type of a domain determine which processing operations can be applied. For example, most image processing operations require the domain to provide a well defined neighbourhood for each point in the domain.

Features and sets of features

While the domain defines the spatial structure of the measurements, a Feature defines the actual values. A feature can be seen as a function that maps each point in the domain to a scalar value. This is analogous to the convention used in pattern recognition for feature space points.

While a single feature only describes a single measurement per voxel, our multifield datasets generally consist of multiple features on the same domain. For example, a T1 weighted MRI, a T2 weighted MRI and a CT scan can be interpreted as a set of three features defined on the same spatial domain.

Tags and meta-data

To ease the ordering and exploration of large sets of features, the features can be tagged and grouped together. At loading time, each feature added to the domain is marked with a set of *Tags*. The *Tags* represent the metadata, such as timepoint numbers, modalities and patient IDs. The advantage of using tagged features, is that grouping operations can easily extract a subset of features that correspond to a specific modality or patient. Also, tags can be added during analysis to denote the source of newly generated features, directly linking them to the patient or processing step they belong to.

Mappings and aggregations

One of our goals is to be able to use features defined on one domain to create new features defined on a different domain. For example measurements performed in a patient-specific domain can be linked to a normalized atlas space or an aggregated population space.

For this purpose one needs to know how the domains are related. A *Mapping* describes how points from one source domain map onto another target domain. A mapping can be one-to-one or one-to many, where each point in the source domain can be linked to any number of points in the target domain.

When a mapping is one-to-one, it can be used directly to transfer a feature defined on the source domain to a feature defined on the target domain. However, most mappings are not one-to-one, as they are designed to reduce the number of points in the domain. In the case where many points in the source domain map to a single point in the target domain, it is not obvious how the value of the target feature should be determined. In those cases, an *aggregation* function is necessary to define how the feature values on the set of points that map to the target location can be reduced to a single value. We first provide a few examples of mappings, before further elaborating on aggregations.

For instance a deformable registration operation between two images that were acquired at different timepoints can be described by defining a mapping that connects points from the source (fixed) coordinate system to the target (moving) coordinate system. Once the mapping is calculated, basic registration would correspond to transferring the original feature to the new domain. However, the same mapping can be used to transfer other features, such as different modalities, to perform co-registration.

Another example of a many-to-one mapping is the calculation of a histogram (see Figure 5.3). Each point in a two dimensional source domain is linked to a single histogram bin based on its value. The fact that the mapping connects all the voxels directly to their respective bins makes it easy to link a selection in the histogram to the corresponding voxels in the spatial domain.

When multiple locations in the source domain map to a single location in the target domain, the feature values of these points have to be combined into a single value. These reductions of many values to one can be described as *Aggregations*. An aggregation is a function that reduces a list of values to a single value. Common

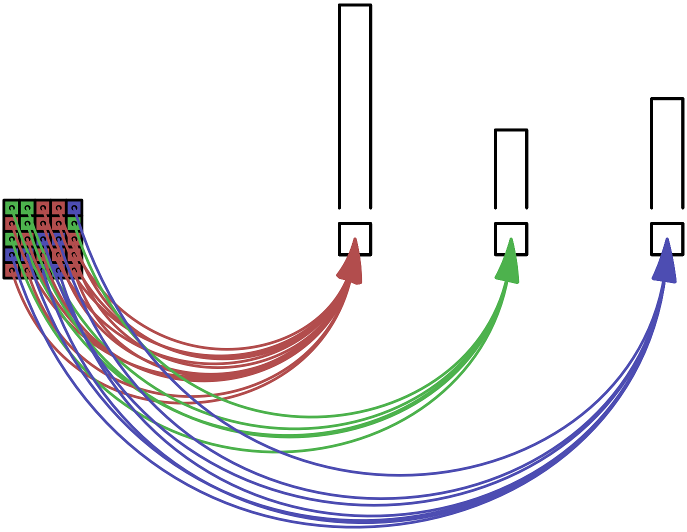


Figure 5.3: Histogram mapping. Points in a two dimensional spatial domain are mapped onto a histogram domain with 3 value ranges. Each point in the spatial domain is linked to a single histogram bin based on its value.

aggregation functions (implemented in the framework) are *sum*, *average*, *variance* and *maximum*.

Mappings and aggregations are powerful tools, which can be used to perform a range of tasks. For example, a volumetric dataset that is registered to an atlas, can easily be averaged over each anatomical area by constructing a mapping from the spatial domain to an atlas domain, in which each point represents a single anatomical area (i.e. tissue, organ, etc.).

Since mappings can be concatenated, it is possible to first aggregate over a segmentation mask, and then aggregate over time, followed by aggregation over patients. We show in Section 5.4.3 how mappings are used in practice.

Filters

Image processing operations are commonly used in pre-processing, but they also have merit during the interactive exploration. A *Filter* is the generic class of operations that uses a number of features as input, and generates one or more new features as output. The generated features are all defined on the same domain as the input features.

The set of filters can be further subdivided by looking at the constraints that a filter imposes on the underlying domain. Common image processing operations, such as blurring or sharpening, need a underlying spatial structure, therefore they can only operate on a *Spatial Domain*. These filters are called *Spatial Filters*. Other operations, such as histogram normalization of the data values are more generic. Such methods are not constrained to the spatial domain, while they process each point in the domain independently from other points, hence they do not need such an underlying structure.

The advantage of this approach is that classes of filters can be transparently applied to data defined on different domains. For example, morphological filters can be applied transparently to volumetric data as well as to data values defined on a two dimensional surface embedded in a three dimensional space, since both domains share the same concept of point neighbourhoods. This makes it easy to find local minima on a surface in exactly the same way as in a volume.

5.4 Implementation Framework

5.4.1 Framework description

The methods presented in the previous section are implemented in a data exploration framework.

The framework presents a user interface in which linked views are used to explore the features defined across different domains. Features can be explored either using a spatial viewing component (a volume renderer or slice viewer), or by using non-spatial views, such as scatter plots and histograms.

Selections can be linked across different views in a dynamic fashion. Links between views can be created through an intuitive drag-and-drop interface. For example, the selections made in a histogram can be directly linked so that the selected voxels are also visible in the slice viewer.

Selections are also treated as features on the same domain as the data. Since the selection is made in the same domain, the feature can just be a binary value for each point in the domain to indicate whether or not the point is included in the selection. Just as other features, these selections can be stored and compared for later use, or they can even be used as input for filter operations (see Section 5.3.2).

5.4.2 Scalability

The size of medical data sets has grown rapidly over the past years. Especially in studies where multiple timepoints, multiple patients and multiple modalities have to be explored concurrently, maintaining interactivity is a difficult issue.

Throughout the framework a data storage backend is employed to reduce memory usage. Often used data is stored in an in-memory cache, while data that is not directly needed is written to a disk-based page cache. This makes it possible to load a large number of datasets, totalling tens of gigabytes in size, while using only a fixed amount of internal memory.

5.4.3 Multi-domain granularity

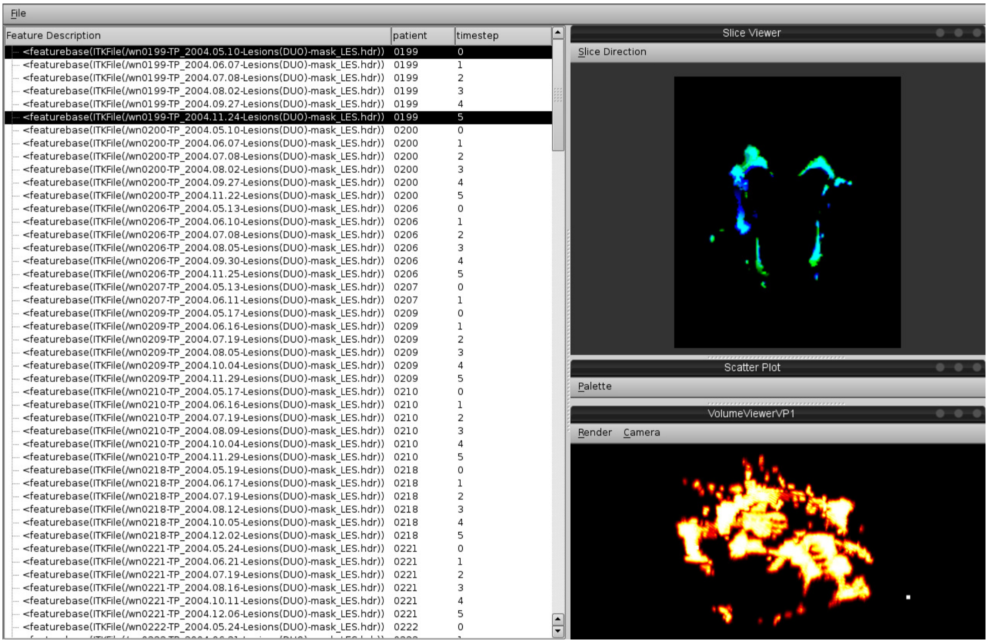
The exploratory interface is built in such a way that multiple views over different domains can be used simultaneously. In this way an overview of aggregated data values can be directly linked to the underlying datasets and voxels contributing to the results.

Selections can be made not only across views, but also across domains. This makes it easier to find the source of deviating statistical numbers in the aggregated tables. For example, the volume of a segmentation mask can be introduced to the patient domain by an aggregation mapping which counts the number of voxels for each patient. When we find an outlier in the patient domain, we can directly select the outlier and go back to the spatial domain to see the relevant segmentation mask. This makes it easier to assess data quality, and to establish a mental model of the image processing pipeline that is needed to solve the problem.

5.4.4 Implementation

The entire framework is implemented in C++, using Trolltech's Qt for the user interface. The image processing and registration components make use of the Insight Toolkit [ISNC05] (ITK), while the visualization components are based on the Visualization Toolkit (VTK). For the virtual memory management we use STXXL [DKS05], which provides virtual memory based storage containers and the accompanying algorithms to access them efficiently.

The user interface (see Figure 5.4) is based on a central feature list, surrounded by movable panels that contain the viewing components. The main tools for inspecting features depend on the type of domain we are inspecting. Features defined on a group domain can best be inspected in tabular format, while spatial features can be best explored using the slice viewer or the volume viewer. All views allow for



Feature Description	patient	timestep
<featurebase(ITKFile(/wn0199-TP_2004.05.10-Lesions(DUO)-mask_LES.hdr))	0199	0
<featurebase(ITKFile(/wn0199-TP_2004.06.07-Lesions(DUO)-mask_LES.hdr))	0199	1
<featurebase(ITKFile(/wn0199-TP_2004.07.08-Lesions(DUO)-mask_LES.hdr))	0199	2
<featurebase(ITKFile(/wn0199-TP_2004.08.02-Lesions(DUO)-mask_LES.hdr))	0199	3
<featurebase(ITKFile(/wn0199-TP_2004.09.27-Lesions(DUO)-mask_LES.hdr))	0199	4
<featurebase(ITKFile(/wn0199-TP_2004.11.24-Lesions(DUO)-mask_LES.hdr))	0199	5
<featurebase(ITKFile(/wn0200-TP_2004.05.10-Lesions(DUO)-mask_LES.hdr))	0200	0
<featurebase(ITKFile(/wn0200-TP_2004.06.07-Lesions(DUO)-mask_LES.hdr))	0200	1
<featurebase(ITKFile(/wn0200-TP_2004.07.08-Lesions(DUO)-mask_LES.hdr))	0200	2
<featurebase(ITKFile(/wn0200-TP_2004.08.02-Lesions(DUO)-mask_LES.hdr))	0200	3
<featurebase(ITKFile(/wn0200-TP_2004.09.27-Lesions(DUO)-mask_LES.hdr))	0200	4
<featurebase(ITKFile(/wn0200-TP_2004.11.22-Lesions(DUO)-mask_LES.hdr))	0200	5
<featurebase(ITKFile(/wn0206-TP_2004.05.13-Lesions(DUO)-mask_LES.hdr))	0206	0
<featurebase(ITKFile(/wn0206-TP_2004.06.10-Lesions(DUO)-mask_LES.hdr))	0206	1
<featurebase(ITKFile(/wn0206-TP_2004.07.08-Lesions(DUO)-mask_LES.hdr))	0206	2
<featurebase(ITKFile(/wn0206-TP_2004.08.05-Lesions(DUO)-mask_LES.hdr))	0206	3
<featurebase(ITKFile(/wn0206-TP_2004.09.30-Lesions(DUO)-mask_LES.hdr))	0206	4
<featurebase(ITKFile(/wn0206-TP_2004.11.25-Lesions(DUO)-mask_LES.hdr))	0206	5
<featurebase(ITKFile(/wn0207-TP_2004.05.13-Lesions(DUO)-mask_LES.hdr))	0207	0
<featurebase(ITKFile(/wn0207-TP_2004.06.11-Lesions(DUO)-mask_LES.hdr))	0207	1
<featurebase(ITKFile(/wn0209-TP_2004.05.17-Lesions(DUO)-mask_LES.hdr))	0209	0
<featurebase(ITKFile(/wn0209-TP_2004.06.16-Lesions(DUO)-mask_LES.hdr))	0209	1
<featurebase(ITKFile(/wn0209-TP_2004.07.19-Lesions(DUO)-mask_LES.hdr))	0209	2
<featurebase(ITKFile(/wn0209-TP_2004.08.05-Lesions(DUO)-mask_LES.hdr))	0209	3
<featurebase(ITKFile(/wn0209-TP_2004.10.04-Lesions(DUO)-mask_LES.hdr))	0209	4
<featurebase(ITKFile(/wn0209-TP_2004.11.29-Lesions(DUO)-mask_LES.hdr))	0209	5
<featurebase(ITKFile(/wn0210-TP_2004.05.17-Lesions(DUO)-mask_LES.hdr))	0210	0
<featurebase(ITKFile(/wn0210-TP_2004.06.16-Lesions(DUO)-mask_LES.hdr))	0210	1
<featurebase(ITKFile(/wn0210-TP_2004.07.19-Lesions(DUO)-mask_LES.hdr))	0210	2
<featurebase(ITKFile(/wn0210-TP_2004.08.09-Lesions(DUO)-mask_LES.hdr))	0210	3
<featurebase(ITKFile(/wn0210-TP_2004.10.04-Lesions(DUO)-mask_LES.hdr))	0210	4
<featurebase(ITKFile(/wn0210-TP_2004.11.29-Lesions(DUO)-mask_LES.hdr))	0210	5
<featurebase(ITKFile(/wn0218-TP_2004.05.19-Lesions(DUO)-mask_LES.hdr))	0218	0
<featurebase(ITKFile(/wn0218-TP_2004.06.17-Lesions(DUO)-mask_LES.hdr))	0218	1
<featurebase(ITKFile(/wn0218-TP_2004.07.19-Lesions(DUO)-mask_LES.hdr))	0218	2
<featurebase(ITKFile(/wn0218-TP_2004.08.12-Lesions(DUO)-mask_LES.hdr))	0218	3
<featurebase(ITKFile(/wn0218-TP_2004.10.05-Lesions(DUO)-mask_LES.hdr))	0218	4
<featurebase(ITKFile(/wn0218-TP_2004.12.02-Lesions(DUO)-mask_LES.hdr))	0218	5
<featurebase(ITKFile(/wn0221-TP_2004.05.24-Lesions(DUO)-mask_LES.hdr))	0221	0
<featurebase(ITKFile(/wn0221-TP_2004.06.21-Lesions(DUO)-mask_LES.hdr))	0221	1
<featurebase(ITKFile(/wn0221-TP_2004.07.19-Lesions(DUO)-mask_LES.hdr))	0221	2
<featurebase(ITKFile(/wn0221-TP_2004.08.16-Lesions(DUO)-mask_LES.hdr))	0221	3
<featurebase(ITKFile(/wn0221-TP_2004.10.11-Lesions(DUO)-mask_LES.hdr))	0221	4
<featurebase(ITKFile(/wn0221-TP_2004.12.06-Lesions(DUO)-mask_LES.hdr))	0221	5
<featurebase(ITKFile(/wn0222-TP_2004.05.24-Lesions(DUO)-mask_LES.hdr))	0222	0

Figure 5.4: The user interface. On the left a list showing all the features in the spatial domain. On the right three panels are visible, each containing a viewing component. From top to bottom it contains a slice viewer, a minimized scatter plot and a volume viewer. Data can be opened by dragging the features onto the views.

making selections, which can be directly linked between the different components (see also Section 5.4.1).

5.5 An Informal Case Study

In this section, we demonstrate our framework by using the proposed methods to interactively explore a real longitudinal study dataset.

5.5.1 Defining the datasets

The dataset we have chosen to demonstrate the presented techniques consists of MRI brain scans of 37 multiple sclerosis patients followed over time. The scans were acquired with a 3.0 Tesla scanner (Philips Medical Systems, The Netherlands). Each patient was scanned 6 times. During each session, transaxial Dual TSE (repetition time, 3000 milliseconds; echo times: 26.7 and 120 milliseconds), FLAIR (repetition time: 11000 milliseconds; echo time: 100 milliseconds) with 51 contiguous slices of 3mm, and high resolution 3D T1-weighted (HR3D-T1; repetition time: 7.5 milliseconds; echo time: 3.4 milliseconds) with 160 contiguous slices of 1 mm of the brain were obtained. The resolution after normalisation and registration is 181 x 217 $\times$ 179 voxels.

With acquisition consisting of three different types of scans, this brings the dataset size to a total of 666 scans. This results in a total data set size of 4.6 gigabytes.

Before the data is loaded a set of metadata tags is applied (see Section 5.3.2) to ease further processing. Each timepoint, patient, and modality gets a unique key:

wn0351-TP-2005-06-11-T1-3D.hdr	patient=0351	scan=0	modality=T1
wn0351-TP-2005-07-16-T1-3D.hdr	patient=0351	scan=1	modality=T1
wn0118-TP-2005-06-14-T1-3D.hdr	patient=0118	scan=0	modality=T1
⋮			

We have three main goals in analyzing this data:

- Data quality assessment for a single patient.
- Data quality assessment for all patients.
- Analysis of disease progression for a single patient.

The following sections show how these goals can be accomplished.

5.5.2 Inspection of a single patient

Initially we are interested in making a quick assessment on whether the data was normalized correctly and whether it looks as expected. We examine a single patient through the use of a filter for **patient=0351**.

The data view now shows the 18 scans available for exploration (three modalities for six timepoints). Since patient data is spatial, we spawn a slice viewer and a volume viewer and drag the features of interest into these views.

The slice viewer uses color maps to display a set of features defined on the volumetric domain, allowing slice-by-slice movement through the domain. The slice viewer performs automatic image fusion when multiple features are selected, so that comparative visualization between timepoints and modalities becomes possible. A combination of two features, a baseline (timepoint 0) and follow-up (time point 1) is shown in Figure 5.5

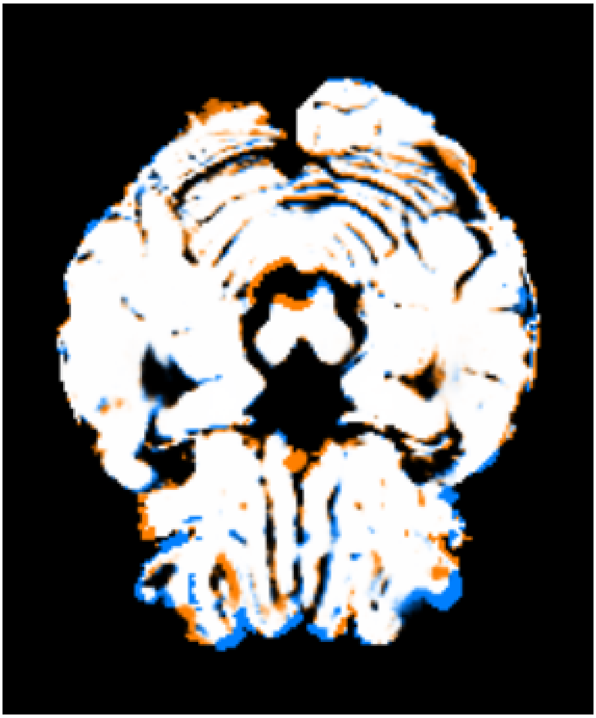


Figure 5.5: The slice viewer shows a blue-orange image fusion of two features. One feature represents the base-line image (blue) while the other feature corresponds to an image taken six months later (orange). Areas where only orange or blue is seen correspond to changes, either due to changes in the brain, or for example due to misalignment during registration. For color plate, see Figure A.10 on page 185.

By comparing the slices and volumes across timepoints and modalities for a single patient, we get a better overview on how the data looks, and whether the data is formatted as expected.

5.5.3 High level examination of an entire dataset

Once we have assessed the structure of the data by inspecting a single patient, we want to explore the data of all the patients. However, visually analysing each separate scan is a tedious process. Therefore, we decide to reduce the amount of data in such a way that it can be inspected more easily. Since we aim to find gross errors in the data, we make sure that the reduced data is still sensitive to the errors that we expect.

Since the errors we expect to find at this point (missing data, normalization issues) are global, they can be safely assumed to still appear when we keep only the average and variance of all voxels.

We establish a mapping that reduces the voxel values of the scan to a set of values specific to each patient and timepoint. Then, instead of a feature in the spatial domain, we set up a domain in which each patient+timepoint combination is a single point. This gives us a domain with 222 points (37 patients times 6 timepoints). We create a mapping that maps all voxels of a single timepoint and patient combination to a single point in this grouped domain.

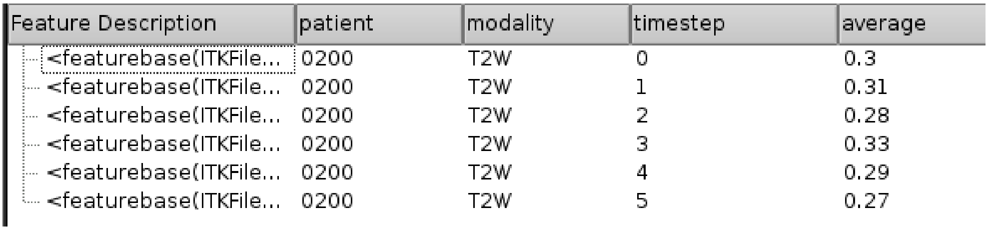
Changing our aggregation function between *average* and *variance* provides us with the two features defined on the new domain.

Since the aggregation of values through a mapping is flexible, we can create a mapping that introduces both the average value as well as the variance for each modality available. Each of these results is represented as a separate feature, and is inserted into the newly created domain.

To examine the data in the new domain we create a scatter plot that shows each patient/timepoint combination as a single dot. Each of the points in this scatter plot has a three dimensional coordinate, corresponding to the features.

We inspect the distribution of points throughout this space, and once we are satisfied that no obvious normalisation errors have occurred in the pre-processing steps, we can further aggregate the data to get per-patient statistics.

Instead of inspecting the newly aggregated features using a scatter plot, they can also be directly integrated into the tabular overview of the data, as shown in Figure 5.6.



Feature Description	patient	modality	timestep	average
<featurebase(ITKFile...	0200	T2W	0	0.3
<featurebase(ITKFile...	0200	T2W	1	0.31
<featurebase(ITKFile...	0200	T2W	2	0.28
<featurebase(ITKFile...	0200	T2W	3	0.33
<featurebase(ITKFile...	0200	T2W	4	0.29
<featurebase(ITKFile...	0200	T2W	5	0.27

Figure 5.6: Aggregated features shown as tags in the list of features. The Average tag corresponds to the average normalized T2 weighted value.

Once we are satisfied that no global errors have been made in the pre-processing and acquisition of the data, we can focus on extracting a specific attribute from the data.

5.5.4 Building of a new feature for a single patient

As an indication for the progress of MS, a lesion count is often used. The hypothesis is that the number of lesions changes rapidly over time in some patients, while it grows steadily in others.

The steps involved in testing this hypothesis are as follows:

- Build a feature that corresponds to lesions
- Aggregate the lesion count per patient
- Detect trends in the lesion count over time for each patient

We start building the lesion feature that we need to test our hypothesis. First, we check whether we can localize the lesions in the scans. In our case, a lesion-mask has already been generated during pre-processing. We can add these mask datasets to the original data and label them with tags in order to link them to the corresponding patients and timepoints. After these lesion-mask features have been added, we can simply threshold the lesion mask variable using a selection tool based on the feature value.

Once we have found the parameters to produce a correct lesion mask for a patient, we can test whether the same settings work for other patients. After making minor adjustments we can generalize the feature metric we have created to obtain aggregated values for each timepoint+patient combination.

These lesion-count values can be aggregated per patient into vectors, which can be shown as in-line graphs, or just as numbers. Each in-line graph represents the

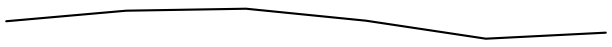
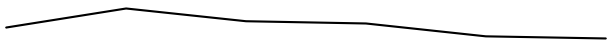
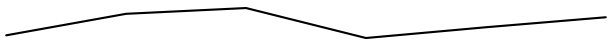
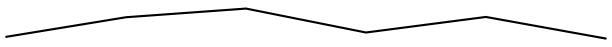
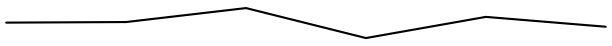
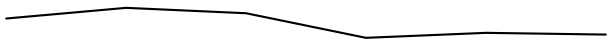
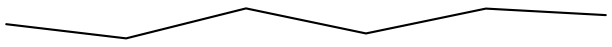
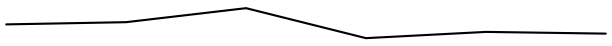
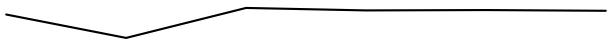
Patient	Lesion count over time
0200	
0206	
0209	
0210	
0218	
0221	
⋮	⋮
0812	
0822	
0910	
1049	
1233	
1241	

Table 5.1: In-line graphs showing the lesion count over time for each patient. Each graph shows from left to right the progression of the amount of lesions over time.

changes of the values over time (see Table 5.1). The line graphs are still linked to their corresponding patient, which provides a direct link back to patient's data.

At this point, we can focus on a specific group of patients, using the visual patterns that we see in the graphs. One such pattern is fast changes in the number of lesions over time, which can be an indicator for regressive multiple sclerosis.

Since we are interested in patients whose values change rapidly, we take the squared sum of the local differences of the lesion count over timepoints. To accomplish this, we reduce the 6 values using a *mapping* that aggreates the squared difference between the lesion count on two succeeding time points, resulting in a single value in the patient domain. At this point we can sort the list of patients by this newly produced value, and see if the in-line graphs produce the patterns we expected; the graphs with the largest feature values are the ones that vary the most over time (see Table 5.2).

5.5.5 Localizing the changes

In the previous section we managed to produce a feature of which we hypothesize that it is related to the progression of the disease. To further verify this hypothesis, we localize where the changes occur in the brain. We first pick a patient to examine, based on the sorted tabular view generated in the previous section. We select the patient with the highest amount of changes over time, and open the according feature set.

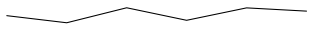
Patient	Squared difference	Lesion count over time
0765	0.294952	
0753	0.29891	
0231	0.651605	
⋮	⋮	⋮
1049	2.65677	
0787	2.85884	
0755	2.94118	

Table 5.2: The list of patients, sorted by the squared amount of changes in lesion count over time. The squared difference corresponds to the amount of rapid changes over time, as can visually verified in the in-line graphs.

This localization can be performed by direct exploration using the slice viewer or volume viewer. As in the first exploration step, we can examine the lesion mask volume for each timepoint, and monitor how the lesions change over time. Visualizing multiple timepoints simultaneously in an effective way is challenging, so we reduce the six time points to a single volume dataset in the same way as we reduced the data in section 5.5.4.

Due to the flexibility of the domains, we can use the exact same reduction as we used before. For each voxel, we take the summed absolute differences in lesion mask over time to produce a new feature. This new feature indicates for each voxel how it has changed over time. By dragging the new feature into the slice viewer, we can inspect the volume and anatomically localize the changes (see Figure 5.7).

5.6 Conclusions and Future Work

We presented a conceptual framework that provides structures and methods to ease the interpretation and analysis of longitudinal multi-field data.

We introduced a prototype implementation, called MULTI-2, of our conceptual framework and demonstrated that it is able to provide interactive exploration on large multimodal data sets.

We have demonstrated a direct two-way coupling between statistical aggregate values and the underlying data, providing a direct means to investigate the data behind the aggregate values.

During the building and testing of hypotheses, our framework provides continous visual feedback.

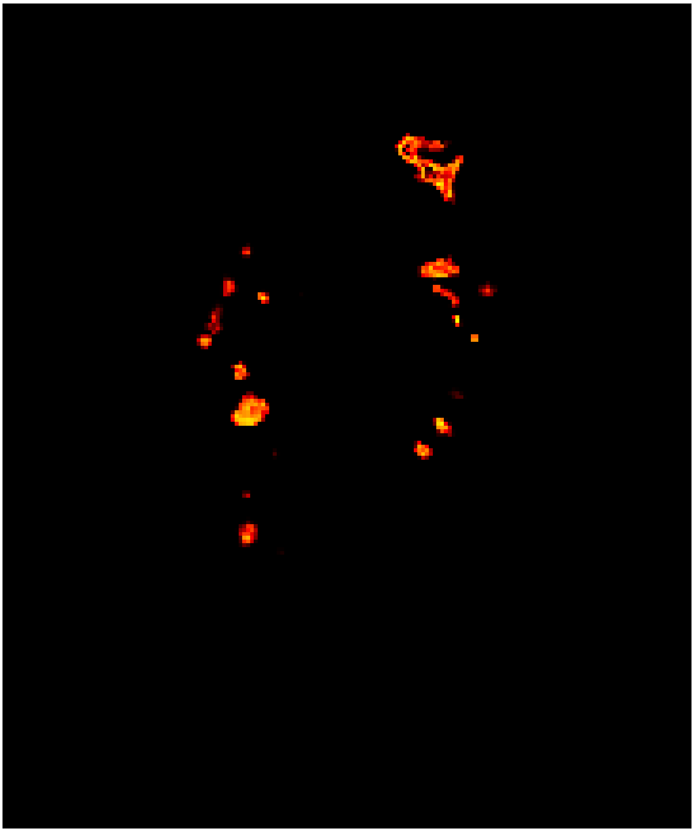


Figure 5.7: A an axial slice viewer showing of a single patient, showing a change-metric using a color-coded scale that indicates how the lesion mask in each point has changed over time. Black indicates zero change while yellow indicates maximal changes.

Our plans are to extend the framework with more analysis techniques, so that trained classifiers and other pattern recognition techniques can be leveraged to create new features. This facilitates the automated testing of correlations and predictions. Also, we intend to investigate the use of other domains, such as line- and surface- domains, so that curvature and other local metrics can be introduced into the feature processing pipeline. Finally, we propose to find a medical test case in which the explicit formulation and testing of a hypothesis can be accomplished.

Acknowledgements

We would like to thank Dr. F. Admiraal-Behloul, previously with LKEB, Departent of Radiology, LUMC, for making available the multiple-sclerosis dataset.

Extensions of Parallel Coordinates for Interactive Exploration of Large Multi-Timepoint Data Sets

6

Published as: Jorik Blaas, Charl P Botha, Frits H Post: *Extensions of Parallel Coordinates for Interactive Exploration of Large Multi-Timepoint Data Sets*, IEEE Trans. Vis. Comput. Graph. 2008 14(6): 1436-1451.

While the previous chapters mostly focus on using scatter plots and histograms as alternate representations, we soon found that other techniques from the field of information visualization could also be adapted to handle large spatio-temporal datasets.

Building on the concept of multiple linked different representations, we present a volumetric temporal multi-scalar dataset as a combination of physical and feature-space views. The feature space is explored using a parallel coordinates plot with interactive selection, while the physical space is represented by slice views and a three dimensional isosurface view. Selections are shared between all views, as to tightly couple the measured values to their locations. Spatial context clearly helps to understand the spatial selections, and vice versa.

Abstract

Parallel coordinate plots (PCPs) are commonly used in information visualization to provide insight into multi-variate data. These plots help to spot correlations between variables. PCPs have been successfully applied to unstructured datasets up to a few millions of points. In this chapter, we present techniques to enhance the usability of PCPs for the exploration of large, multi-timepoint volumetric datasets, containing tens of millions of points per timestep.

The main difficulties that arise when applying PCPs to large numbers of data points are visual clutter and slow performance, making interactive exploration infeasible. Moreover, the spatial context of the volumetric data is usually lost.

We describe techniques for preprocessing using data quantization and compression, and for fast GPU-based rendering of PCPs using joint density distributions for each pair of consecutive variables, resulting in a smooth, continuous visualization. Also, fast brushing techniques are proposed for interactive data selection in multiple linked views, including a 3D spatial volume view.

These techniques have been successfully applied to three large data sets: Hurricane Isabel (Vis'04 contest), the ionization front instability dataset (Vis'08 design contest), and data from a large-eddy simulation of cumulus clouds. With these data, we show how PCPs can be extended to successfully visualize and interactively explore multi-timepoint volumetric datasets with an order of magnitude more data points.

6.1 Introduction

Parallel coordinate plots (PCPs) [Ins85] were developed as a method to create planar graphs of multi-variate data. In parallel coordinates, each N-dimensional data point is transformed into a polyline that intersects N parallel vertical or horizontal axes. Each axis represents a dimension, and the point at which the polyline intersects represents the value of the point on that dimension. Parallel coordinates have received acceptance in statistical data analysis and information visualization as a general method for visualizing arbitrary high-dimensional datasets [WB97]. As each high dimensional data point is represented uniquely by a polyline there is no loss of data due to projections, as is often the case with other methods such as scatter plots. Another important advantage lies in the ability to visualize the geometry of high-dimensional objects, and not just the data.

To apply PCPs to large datasets, the limitation does not lie in the number of dimensions per data point, but in the number of points and associated polylines. Scalability of PCPs is limited by two major problems: visual clutter and reduced performance, hampering the use of PCPs for interactive exploration. A number of extensions have addressed these problems, such as the use of hierarchical and multi-resolution methods, smooth parallel coordinates, 3D PCPs, and techniques to reduce visual clutter (see section 6.2). However, PCPs have been mainly applied to scattered data of up to a few millions of points. One million points is just the lower boundary for volumetric datasets (100^3 voxels). To be usable for multifield volume data, extension of PCPs to the range of tens of millions of points would be necessary.

In this chapter, we present techniques to allow the interactive exploration of multi-timepoint volumetric data in this range. This is achieved by using a combination of data quantization and compression, and the use of data structures to allow very fast computation and GPU-based rendering of joint density distributions, resulting in a quasi-continuous view of the line densities between each consecutive pair of parallel axes. Several facilities for interaction are available, including brushing data selection and data normalization using histogram equalization. Finally, a two-way linking is provided with spatial views of the data. This would make integration possible of PCPs in a full-blown interactive data analysis system with linked related views at the current dataset sizes.

The contributions of this chapter can be summarized as:

- Scalability of PCPs to the range of tens of millions of points for use with realistic multifield volumetric datasets.
- Maintaining high interactivity at this scale.
- Dynamic brushing for data selections.
- Dynamic two-way links between spatial views and PCPs.
- Support for multi-timepoint volumetric datasets.

To demonstrate the feasibility of these techniques, they have been applied to three large time-varying datasets: the contest datasets of Visualization 2004 (Hurricane Isabel) and 2008 (the ionization front instability), and an atmospheric large-eddy simulation of cumulus clouds.

This chapter is structured as follows: related prior work is discussed in section 6.2. The processing pipeline is described in section 6.3, and section 6.4 presents rendering and interaction methods. Section 6.5 gives performance data, and presents the three applications. Finally, section 6.6 draws conclusions and indicates future developments.

6.2 Related Work

In this section, we focus primarily on previous work on parallel coordinates for large, sometimes time-varying, datasets, as this is an important characteristic of our contribution. We conclude by briefly discussing recent work using linked and coordinated views for the visual analysis of large datasets, as we also demonstrate how to combine parallel coordinates on large data with other linked views.

Fua et al. [FWR99] defined large datasets as containing 10^6 to 10^9 data elements or more. They extended XmdvTool [War94] with a special form of parallel coordinates that employed hierarchical clustering. The user could interactively vary the level of detail, making possible a multi-resolutional visualization with smooth transitions to any level of the clustering or the raw data samples themselves.

In the same vein, Johansson et al. introduced the use of self-organizing maps (SOM) in order to cluster data samples and thus reduce large datasets [JTJ04]. The clusters were visualized as variable width bands in parallel coordinates, with the width encoding information about the clusters that they represent. In this case, one could drill-down by expanding a cluster into its constituent samples in a linked view. Specifically, a large dataset is defined as one containing at least 10000 data elements with 16 dimensions.

To increase scalability without increasing visual clutter, line densities were introduced by Miller at al. [MW91]. Artero et al. proposed the use of Interactive Parallel Coordinate Density and Frequency Plots [AdOL04]. Bi-dimensional frequency histograms were calculated for every pair of consecutive parallel axes. For each position in a frequency histogram, a line was drawn between the relevant two axes, with its brightness proportional to the frequency. Maximum intensity compositing was used so that higher frequency samples always had precedence. In this way, large data could be aggregated for a more effective visualization. The method was tested on datasets with up to a million data elements with 50 dimensions.

By transforming each K-means-derived cluster into three high resolution textures, namely an animation, outlier and structure texture, and then compositing all cluster textures onto a polygon, Johansson et al. managed to create cluster visualizations that included information about the internal structure of clusters [JLJC05]. Cluster colors were pre-determined, but opacity was configurable by specifying a transfer function, also non-linearly, mapping from local intensity to opacity. Outliers were determined by inspecting the inter-quartile range on each dimension and visualized by making use of the outlier textures. These techniques were tested on datasets with up to a hundred thousand data elements.

Johansson et al. also investigated temporal parallel coordinates, focussing on visualizing changes over time by adding depth cues and temporal density [JLC07].

Novotny et al. use a method similar to the approach of Artero [AdOL04], where bidimensional histograms, or bin maps are computed [NH06]. Outliers were detected directly in the bin maps and removed for separate rendering. Inspired by image processing techniques, clustering also took place directly on the bin maps. Clusters and outliers were separately rendered to retain visibility of the outliers in the final visualization. The largest dataset tested on consisted of three million data elements over sixteen dimensions.

WEAVE [GRW$^+$00] and SimVis [DGH03] are examples of systems that make use of linked scientific and information visualization views, including parallel coordinates, in order to explore complex datasets. More recently, SimVis was applied to large dynamic datasets [MKO$^+$08], but without significant involvement of traditional parallel coordinates.

Ten Caat et al. focus on a clinical application scenario in which temporal EEG data is explored by students, researchers and experts to assess latencies, amplitudes and symmetries [CMR07]. Their work is a good example of how PCPs can be successfully adapted to improve the assessment of complex medical data.

The framework by Akiba et al. explores the concept of data exploration through linked views in the temporal, spatial and variable domain [AM07].

Existing work on parallel coordinates for large data employs a combination of clustering, binning and other feature extraction, such as outlier detection, in order to cope with large datasets. These techniques reduce both scalability and visual clutter problems.

Our work builds on the bin map idea, but adds a number of refinements in order to show how parallel coordinates can be effectively used for the interactive visualization of even larger multi-timepoint datasets with 25 million data elements per timestep over 10 dimensions. We have developed these extensions also with the idea of integrating our large data parallel coordinate pipeline with current multiple linked view systems.

6.3 Processing Methods

To make PCP rendering and processing fast enough for interactive exploration, an optimized data processing pipeline was adopted. Our on-disk data structure was designed to provide fast access to the data needed during interaction, and it can cope with the data access patterns that arise in multi-timepoint data.

We identified a number of common tasks that are necessary to fully exploit the PCP-based exploration.

- Rendering the parallel coordinate plot
- Selection of a range of points in the plot by defining an attribute range on one of the axes
- Data probing operation at a spatial location
- Spatial display of a selection
- Re-ordering the axes
- Moving between time points

Our processing pipeline, further explained in the following sections, is designed to provide a fast structure to perform the above-mentioned operations.

6.3.1 Processing pipeline

The processing steps (see Figure 6.1 on the facing page) are separated into two categories: preprocessing and interaction. The preprocessing steps are only needed to be ran once, to convert the source dataset into a compact and easily-accessible storage format. Once preprocessing has been performed, the stored data is loaded on-the-fly during the exploration process.

The following sections further explain the details of the methods shown in the pipeline diagram.

6.3.2 Histogram equalization

One of the problems with large datasets is that the distributions of the attribute values are often not very smooth. When such a distribution is highly skewed, a large fraction of the values falls within a small range of values. When such a dataset is linearly rescaled to a range $[\min, \max]$, only a small part of the vertical axis is used in the resulting plot. This leads to a high level of clutter, and makes it difficult to spot the internal correlations that occur inside the dense area.

This is why we optionally perform a non-linear univariate normalization. For each attribute, a histogram is created of the data-values for a single attribute over all voxels and over all timesteps. Once the histogram of each attribute is known, its values can be mapped through histogram equalization to values in the range $[0, 1]$ in such a way that the values have a continuous density in the target domain.

The effect of normalization of both the pressure and the temperature in a single slice of the hurricane Isabel dataset is shown in Figure 6.2 on page 106. Note that the

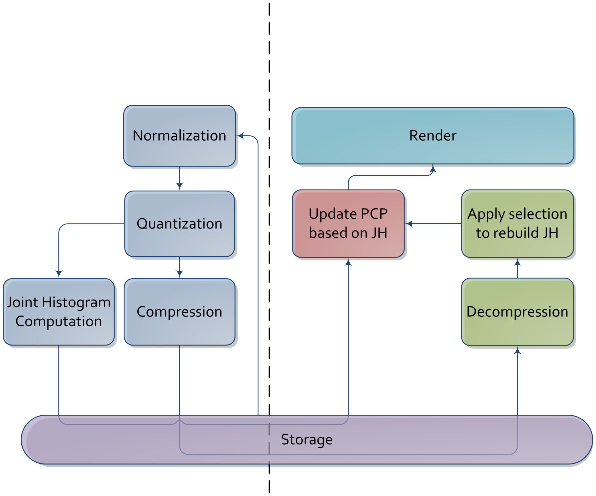


Figure 6.1: The processing pipeline. The methods to the left of the dotted line are performed during preprocessing, while the other methods are continuously performed during user interaction.

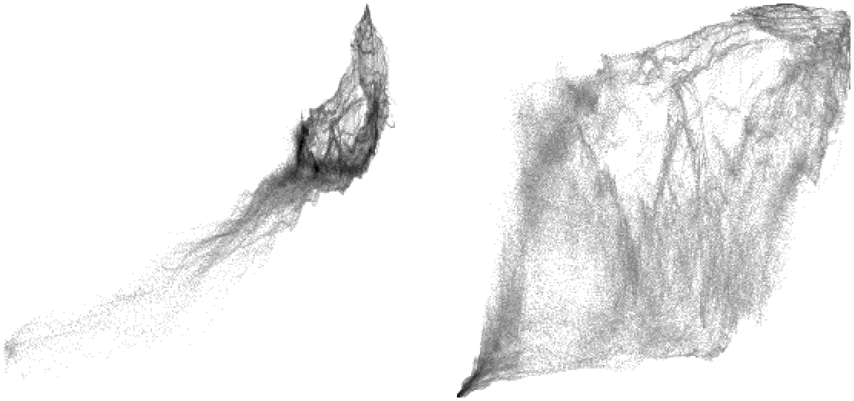
intricate internal patterns that occur in the high density area of the unnormalized data are well visible after normalization.

It is crucial that the histogram equalization is not performed on each timestep individually. If each timestep was normalized individually, then each timestep would have a different normalization mapping, and the correspondence between attributes over time would be lost, making it impossible to compare values between timesteps or to spot changes that occur over time.

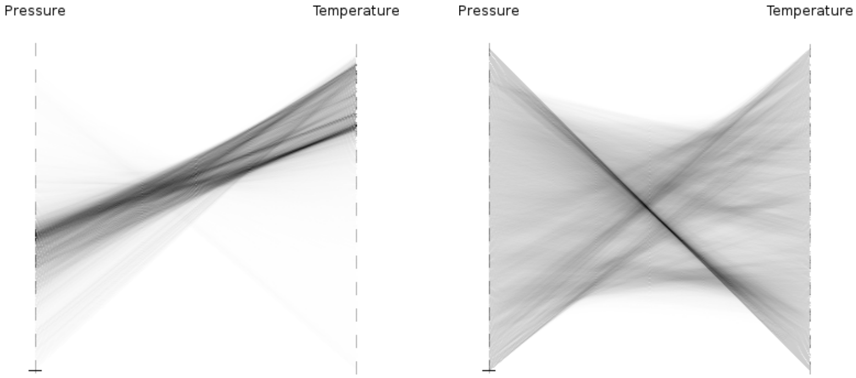
6.3.3 Quantization

Parallel coordinate plots have little to gain from a high precision floating point representation of the data. When data values are rounded to a lower precision representation, the maximum erroneous displacement that a line in the plot has is directly related to the rounding error. Since the axes of a parallel axis plot are generally not higher than 512 pixels, a quantization to 8-bit values yields a maximum displacement of a single pixel, which is adequate for our purpose.

This quantization step reduces the data from four bytes to a single byte per point per attribute, greatly reducing the necessary storage. As we demonstrate later, storing the data in a fixed-point format also improves the compressibility.



(a) not normalized (b) normalized



(c) not normalized (d) normalized

Figure 6.2: The effects of normalizing the data by histogram equalization. A two-dimensional dataset consisting of pressure and temperature data is shown before and after normalization. (a) shows a scatter plot of the data without normalization, (b) shows the scatter plot after the joint histogram has been normalized. (c) and (d) show the parallel coordinate plot corresponding to the data in figure (a) and (b), respectively.

6.3.4 Joint-histogram generation

A parallel coordinate plot without any selections can be quickly generated solely from the joint-histograms of the data. We make use of the binning approach proposed by Artero et al. [AdOL04]. Using this technique, only the joint histogram between each pair of neighboring axes is needed to build the parallel coordinate plot.

Fast exploration of the data over time is made possible by pre-computing joint-histograms of all pairs of axes. For N axes, this costs $N * (N-1) / 2 * bins^2 * \text{sizeof}(uint32)$ space. Whenever a new timestep is selected, $N-1$ reads suffice to quickly produce a new parallel coordinate plot based on the joint histograms between each pair of axes.

6.3.5 Storage and compression

As seen in Figure 6.1 on page 105, two types of stored data are used during interaction: pre-computed joint-histograms and raw compressed data volumes.

For each timestep, the joint-histograms are stored as raw blocks of $bins^2$ unsigned integers. Since the axis order determines which of the joint histograms are needed, we store all $N * (N-1) / 2$ joint histograms in separate files. This makes it easy to load the $N-1$ needed files for any axis order at runtime.

The raw data volumes each represent a single scalar defined over the full volume. Since the data values have been quantized, only one byte per voxel is needed. Since these datasets are fairly large, the disk access still forms a major bottleneck when changing to another timestep. To partly alleviate this bottleneck, a compression step is performed.

We have selected to use the LZO (Lempel-Ziv-Oberhumer) compression, of which a public implementation is available [Obe05]. LZO compression is well suited for realtime decompression, since it has a very high decompression speed while still maintaining a good compression ratio. Depending on the compressed size, the rate at which decompressed data was produced ranged from 110 to 250 MB/sec, measured using a single core of a 2.0 GHz AMD Athlon64 X2 3800+ processor.

To store the compressed data on disk, one file per variable/timestep combination is used. This makes it easy to load the complete volume for a specific timestep, while still keeping the possibility of using a reduced set of variables to speed up processing when necessary.

The volume of the currently loaded timestep is loaded fully in memory. The in-memory structure of the volume is such that the fastest changing axis corresponds to the attribute number, followed by the three spatial axes. This ordering enhances the spatial locality during the histogram creation phase, resulting in faster selection processing.

The following table shows the order in which the variables are stored in memory:

T_0	P_0	LW_0	T_1	P_1	LW_1	...

T_x, P_x and LW_x represent the temperature, pressure and wind-speed at location x. Each cell represents a single memory location, ordered in a left to right fashion.

6.4 Interaction Methods

We implemented a demonstrator to show which type of interaction is possible with the datasets of the intended size.

6.4.1 Rendering

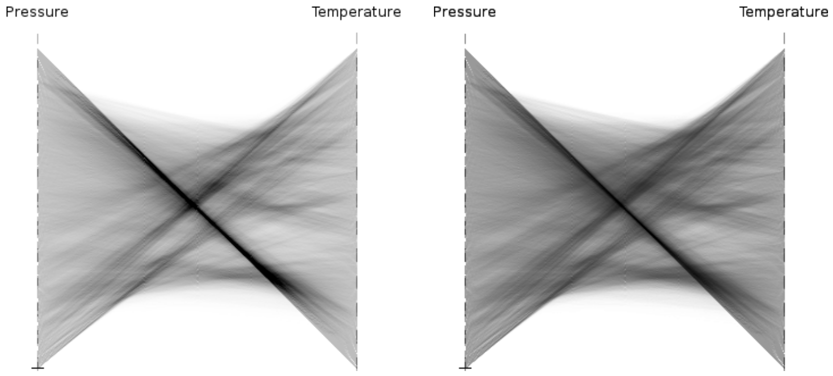
We adopt the rendering approach of Muigg et al. [MKO+08], where the histogram bins form a direct basis for drawing the primitives. Instead of having to draw a line for each data point, only a single primitive is drawn for each histogram bin.

To combine all drawn primitives together, additive blending is used. Since the intensity over the plot varies widely, a high precision floating point framebuffer is used as a render target, so that no clipping of color values is necessary. The added advantage of this technique is that intensities can be converted to color values in a post processing step.

We use a logarithmic intensity scale (see Figure 6.3 on the next page), as to prevent over-saturation of high-density areas in the plot, while keeping a good visual contrast in low intensity areas. The contrast can be modified interactively by the user to emphasize the high or low intensity areas of interest.

6.4.2 Selections

Two commonly used types of selections are implemented in our demonstrator application. Basic selections are made by dragging the mouse over a range of values at any given axis. In addition, compound selections can be made by combining basic selections in an AND or OR-like fashion, so that more complex phenomena can easily be studied.



(a) Linear (b) Logarithmic

Figure 6.3: The effect of using a logarithmic intensity scale. Where the high density center area on the left is completely saturated, the logarithmic intensity map on the right shows no signs of clipping and still has a high contrast in all areas.

Changes in selection criteria are processed by performing a linear scan over the data, splitting it in two sets of points: selected points and unselected points. The two sets of points are treated independently. Each of them is rendered as a parallel coordinate plot, which, after coloring, can be blended to form a clear visual representation of the selected points.

6.4.3 Spatial exploration through linked views

Multiple linked views present a good way of linking the different representations of data. In this case we have chosen for a two-way linkage between the top and the bottom half of the screen (see Figure 6.4).

Whenever a point is selected with the mouse in the slice viewer, a probe determines the data values for the selected voxel and overlays these on the parallel axis plot.

Selections made in the lower part of the screen are similarly linked to the slice viewers. The slice viewers continuously display which pixels are included in the selection. The orange and blue color scheme corresponds to the parallel coordinate plot such that selected pixels are orange and unselected pixels are blue (see Figure 6.5).

To provide further insight on the selected values, the current selection can also double as a colormap definition. Each selection that the user makes uses the data values of that selected variable to produce colors. The colors are determined based

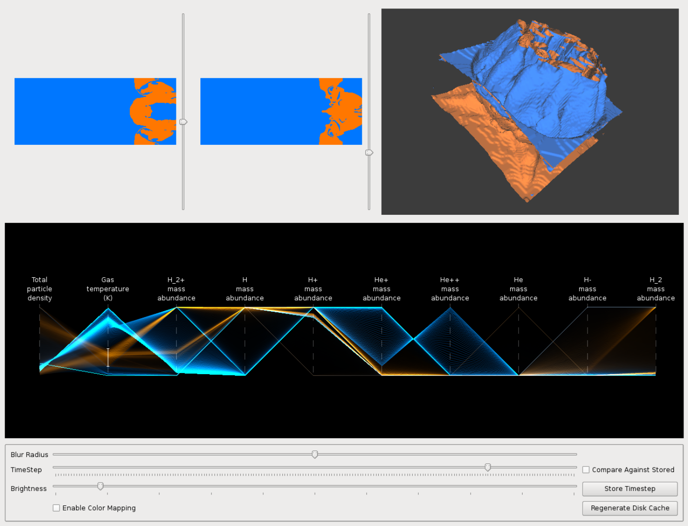


Figure 6.4: The main user interface of the demonstrator application. The top part of the screen contains spatial viewing components (two slice viewers and a 3D isosurface view). The parallel coordinate plot is positioned in the middle, and the control interface is positioned at the bottom. For color plate, see Figure A.11 on page 186.

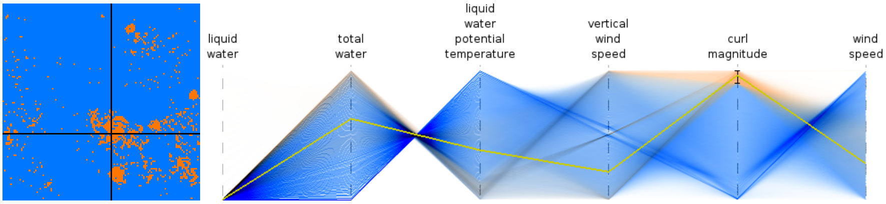


Figure 6.5: A slice viewer in which a point is selected (left) linked to a parallel coordinate plot (right) which displays the selected value as a bright yellow line. For color plate, see Figure A.12 on page 186.

Figure 6.6: Defining a colormap using two selections. Two ranges are selected in the PCP on the left, which correspond to two separate color maps in the slice view on the right. Bright blue pixels correspond to a high gas temperature, and bright green pixels to a high H_2^+ mass abundance. For color plate, see Figure A.13 on page 187.

on the relative position of the sample inside the selected range of values, so that low values correspond to dark colors and high values to bright colors. In this way, each range selection made in the parallel coordinate plot defines a single-color mapping (see Figure 6.6).

6.4.4 Temporal exploration

The PCP at a specific timestep visualizes the distribution of the datapoints, but in multi-timepoint data the changes in the distribution over time also possess key information. The demonstrator application provides a time slider through which the user can navigate through all available timesteps. During the movement of the slider, the PCP is rapidly updated using the joint histograms (Section 6.3.4). When a timestep is selected, the full dataset for that timestep is loaded and decompressed into memory. The typical loading time is 0.5 to 2 seconds, which is acceptable for this style of interaction.

Comparison against baseline

Since changes in volumetric simulation data between adjacent timesteps are often limited to a small number of points, the changes in the total distribution of the values over all points are quite small. To be able to focus on these smaller changes, a baseline distribution on one timepoint can be stored. The PCP can then be used to visualize the difference between the distribution at another selected timepoint and the stored distribution (see Figure 6.7).

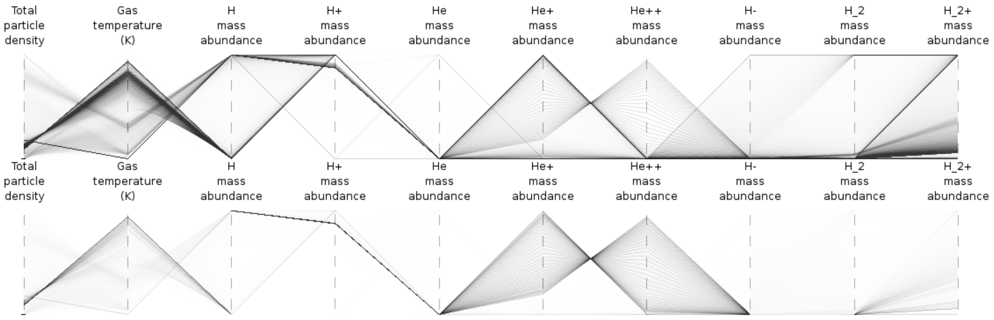


Figure 6.7: Comparison against baseline. The top figure shows the PCP for timepoint 126. The bottom figure focuses on the changes over time by showing the difference between timepoint 126 and a stored baseline at timepoint 125.

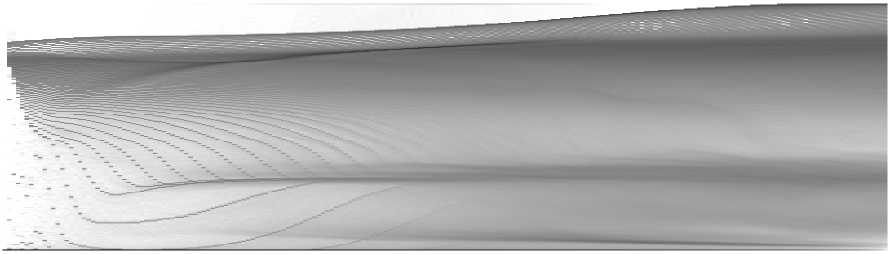


Figure 6.8: Visualization of the change in distribution of the gas temperature over time. The horizontal axis corresponds to time, the vertical axis to gas temperature, while the intensity values in each pixel represent the number of data points within the corresponding temperature range. This makes each column of the plot a histogram of the temperature values for that timestep.

Histograms over time

In the PCP, at each labeled axis, the intensity of the pixels of that column corresponds to the histogram of a distribution of the corresponding variable. The changes in these distributions over time are often indicative for the specific chemical reactions or other events. Inspecting these changes often provides key insights into what is happening in the studied phenomena.

To monitor the changes in the distribution of the values, the PCP can be inspected while moving the timestep slider. In some cases however, this method can be tedious and it can be difficult to pin-point a specific point in time at which the distribution starts changing.

Therefore we propose a second method to inspect these changes. For a single selected axis in the PCP, a temporal view can be opened that displays a plot of the distribution on that specific axis over all timesteps. For each timestep, the intensity profile along the column of the selected axis in the plot is extracted. The extracted profiles are joined together in a single plot (see Figure 6.8 on the facing page).

6.4.5 Axis order

The ordering of the axes is a vital part of any good PCP. The demonstrator application starts with a pre-computed axis ordering, but the user can interactively drag and drop the axes to reorder them if necessary.

An axis can be swapped with another axis, or it can be moved to a position between a pair of adjacent axes. As only the joint-histograms related to the current axis order are contained in memory, new joint-histograms have to be loaded from disk after such a manipulation. In the worst case, four new joint-histograms have to be loaded.

6.5 Results / Applications

To study how well the proposed techniques work on a real-life dataset, we have selected three large volumetric datasets to perform visual exploration on. We have selected the visualization contest datasets of 2004 and 2008, since they are publicly available and both good examples of multi-scalar temporal datasets. The third dataset we selected is an atmospheric simulation with the goal of studying cumulus clouds. Table 6.1 shows the characteristics of the explored datasets. The following sections describe the data and accompanying exploration.

All performance measurements were made on a desktop PC consisting of a 2GHz AMD Athlon64 X2 CPU, 2 gigabytes of RAM and an Nvidia GeForce 7950GT graphics card. Since no multiprocessing was implemented in the demonstrator application, only a single CPU core was used during the benchmarks.

Dataset	Hurricane Isabel	Vis 2008 contest (halved)	Cumulus cloud dataset
Spatial Resolution	500 × 500 × 100	300 × 124 × 124	128 × 128 × 80
Points	25.000.000	4.612.800	1.310.720
Timesteps	48	200	600
Attributes	10	10	6
Original size	44.70 GB	34.37 GB	17.58 GB
Quantized size	11.18 GB	8.59 GB	4.39 GB
Compressed size	2.41 GB	0.78 GB	3.00 GB

Table 6.1: Overview of the used datasets and their size.

6.5.1 Hurricane Isabel dataset

The hurricane Isabel dataset is part of the visualization contest of 2004. It contains a detailed simulation of a hurricane moving over the west Atlantic region.

The hurricane Isabel dataset has the highest spatial resolution of all three datasets we explored. Each timepoint consists of 25 million points with 10 attributes each, resulting in 250 million data values.

Performance

During navigation with the time slider, the loading time for the precomputed joint-histograms is consistently around 0.02 seconds per timestep. Combined with the 0.1 seconds it takes to draw the complete parallel coordinate plot, this results in 8–9 frames per second when moving the time slider. Once a new timestep has been selected, it takes 2.5 seconds to load and decompress the full data volume from disk. The recalculation of the histograms takes another 2.4 seconds. While this recomputation takes a considerable amount of time, it is important to note that this step is only necessary when the selection changes. Once the histograms have been recomputed, the slice view, probe and 3d-rendering run at over 30 frames per second.

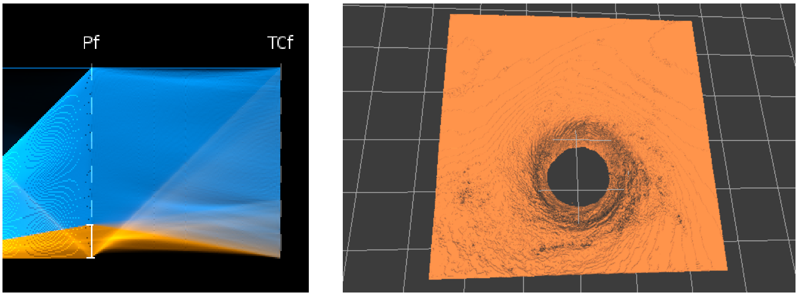
Eye of the hurricane

To get a good overview of the dataset, we select the low pressure area near the eye of the hurricane by applying a range selection on the temperature axis. The 3D view shows the spatial shape of selected area, while the PCP provides us with the information that low pressure areas are in this case highly correlated with low temperature areas (see Figure 6.9 on the next page). The hurricane's movement over time is directly visible when moving the time slider to a different timepoint.

Snow and precipitation

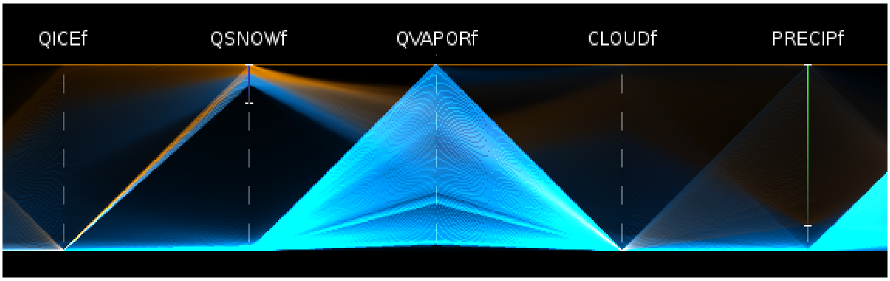
In normal conditions, precipitation leads to low humidity. We explored how the amount of snow is related to the precipitation (see Figure 6.10 on the facing page).

We found no real changes in the distribution over time. While the eye of the hurricane does move spatially, the overall composition does not change significantly, as the distribution of the values within the hurricane is rather constant.

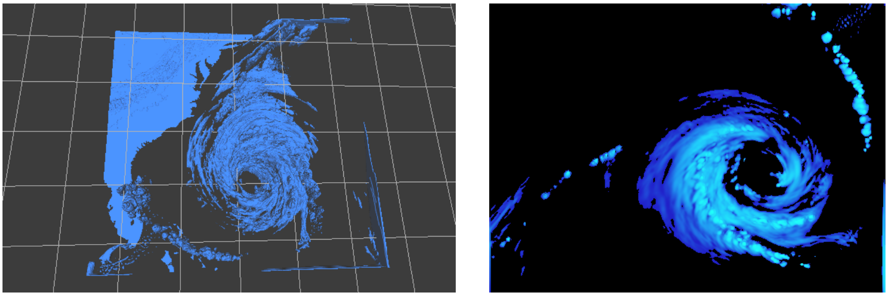


(a) PCP selection (b) 3D view

Figure 6.9: Selection of low pressure areas (shown in orange) reveals the area of low-temperature near the eye of the hurricane. The compact horizontal shape of the orange band in the PCP reveals that low-pressure areas mostly have a low-temperature as well. For color plate, see Figure A.14 on page 187.



(a) PCP selection



(b) 3D view (c) Color-mapped slice view

Figure 6.10: Exploration of the hurricane Isabel dataset. The combination of high precipitation and snow has been selected so that the blue colors in the slice view correspond with snow while green and cyan correspond to areas with high precipitation. For color plate, see Figure A.15 on page 188.

6.5.2 Vis2008 contest dataset

The 2008 visualization contest dataset is a simulation of an ionization front instability [WN08]. There are ten attributes that describe the total particle density, the gas temperature, and the abundance of H, H^+, He, He^+, He^++, H^-, H_2 and H_2^2.

We have down-sampled the data spatially to half the resolution, resulting in a volume size of 4.6 million points. The loading and decompression time of a single timestep was 0.6 seconds, while histogram updates took 0.4 seconds.

One of the most interesting features of this dataset is the fact that the distributions of the attributes change considerably over time. The simulation starts out as an ionization front hits a small spherical bump, causing the front to break in a turbulent matter. This fact can be seen in the PCP, as in the early moments of the simulation it shows only a single line, which quickly spreads out into a wide set of bands.

We were able to track the ionization front by selecting all high temperature points, and monitoring their three dimensional spatial structure over time (Figure 6.11 on the next page). This clearly shows how the initially stable front breaks up in a highly turbulent structure.

We noticed that the presence of H and H^+ shows an interesting correlation (Figure 6.12 on page 118), which we can pinpoint spatially through the use of color mapping on a slice view. After the selection of the attribute ranges, the color map feature was enabled so that high H mass abundance and H^+ mass abundance are represented by bright blue and bright green colors respectively.

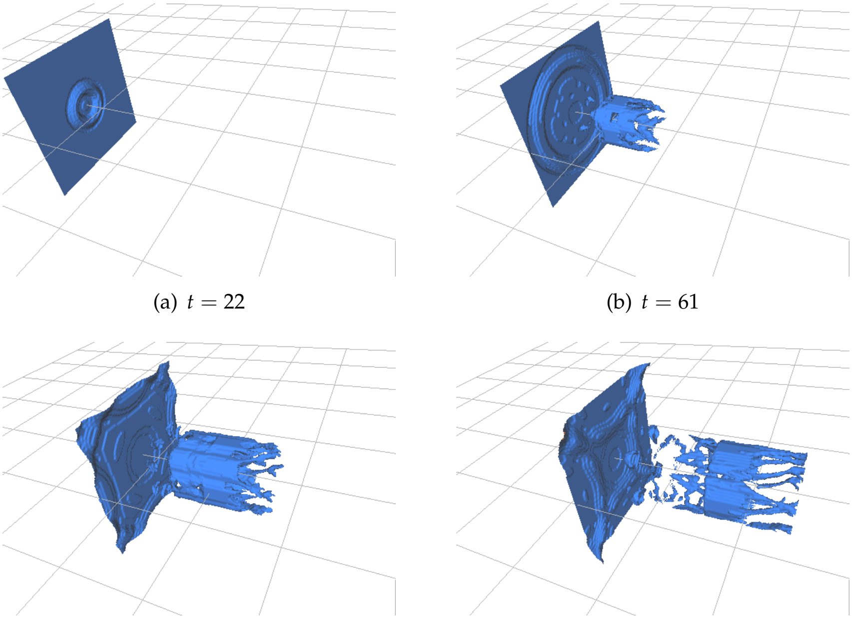
The distribution changes can be partly explained by the fact that the area of effect moves outside the bounds of the simulation.

6.5.3 Cumulus clouds

The cumulus clouds dataset is the result of a Large-Eddy simulation with the aim of studying cloud life-cycle patterns. The dataset contains four attributes, representing the amount of liquid water, the potential temperature, a wind vector and the amount of total water. We have added derived features in a preprocessing step, resulting in two additional features; the wind speed and the vorticity.

Since the spatial resolution of the dataset is quite low, the data does not compress as well as the other datasets. However, as the volume size is relatively small, the load times are still under 0.1 seconds per timestep. A full histogram update takes 0.07 seconds.

Firstly, we verified that the simulation is in a steady state, by examining the histogram over time of the main attributes. The results indicate that the distribution



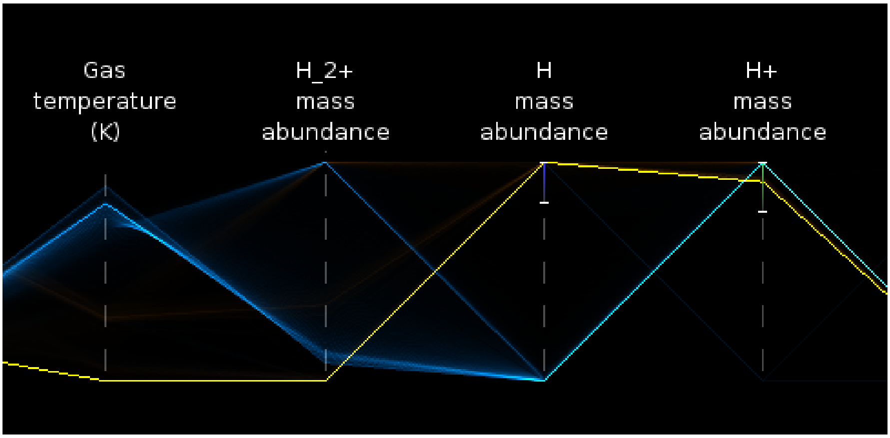
(a) $t = 22$

(b) $t = 61$

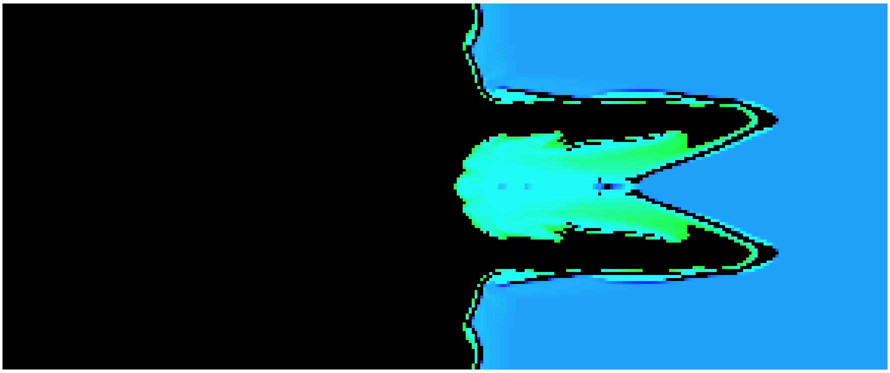
(c) $t = 91$

(d) $t = 118$

Figure 6.11: Isosurface view of the movement of a high temperature front in the 2008 contest dataset.

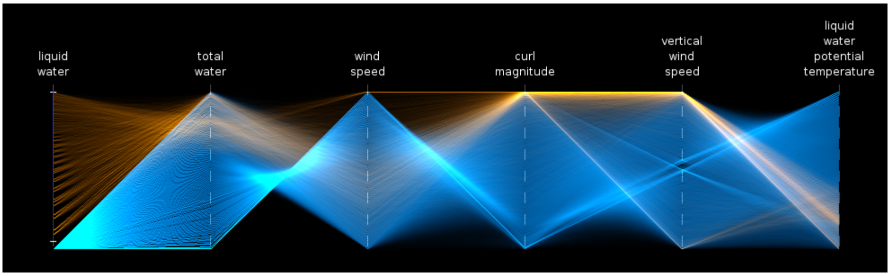


(a) PCP

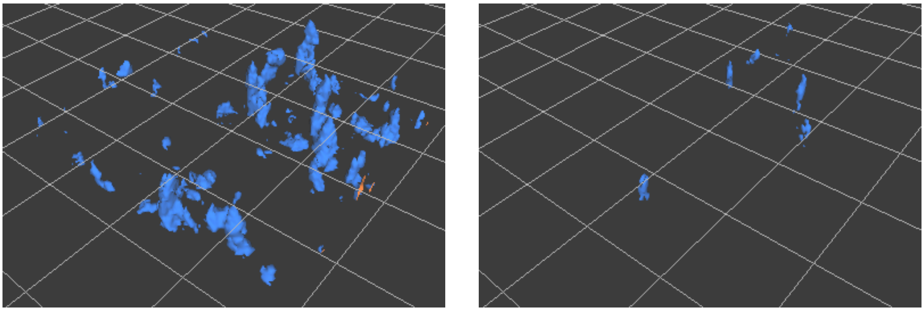


(b) Slice

Figure 6.12: A PCP showing the correlation between H and H^+ mass abundance. The PCP uses two color-mapped selection ranges so that the colors in the slice viewer represent the H mass abundance (blue) and the H^+ mass abundance (green). For color plate, see Figure A.16 on page 189.



(a) PCP selection of clouds



(b) All clouds

(c) High-windspeed clouds

Figure 6.13: Clouds can be selected based on the liquid water attribute (a,b). The orange lines show the distribution of the other attributes within the clouds. When the high wind-speed outlier is selected, only a select subset of the clouds is visible (c). For color plate, see Figure A.17 on page 190.

is almost uniform, and no major changes over time are visible. This corresponds to the expected distribution of a steady state.

A large part of the volume does not contain visible clouds. To visualize the clouds, we performed a selection based on the amount of liquid water present in a voxel (see Figure 6.13 on the previous page). When moving the time-slider, the three dimensional isosurface representation of the clouds corresponded to our expectations. The formation of new clouds is clearly visible.

The PCP shows that our selection produces one interesting outlier when looking at the wind speed. A secondary selection can be used to find where these outliers are located spatially. These high wind-speed parts are characteristic to a specific phase of the cloud life-cycle, and their formation is subject of current research.

6.5.4 Insights gained

Our main goal with these three examples was to show that the techniques presented in this chapter enable the interactive exploration of large time-varying datasets with parallel coordinate plots. With dataset sizes ranging from 1.3 million points with 6 attributes over 600 timesteps to 25 million points with 10 attributes over 48 timesteps, and our system enabling navigation through timesteps at 8 to 9 frames per second, loading of a full timestep to updated linked views and selections at between 0.1 and 5 seconds and finally interaction with linked volume and slice views at 30 frames per second, we think that we have successfully reached our main goal.

With regard to PCP as a suitable visualization method, the fact that all points are explicitly linked over all dimensions is a clear advantage over many other multidimensional visualization techniques. Clusters existing in a subset of the dimensions are explicitly linked and visible in all other dimensions, so that one can study divergence over the non-clustered dimensions. With scatter plots, one has to make use of multiple views with linked brushing to accomplish this, in which case one can only study selected versus non-selected points, as there is no other easy way to link points in different linked views. In parallel coordinates, the explicit linking is independent of the selection and valid for all points.

In all three examples, selection was used to investigate the data. However, due to the explicit linking explained above, relations over all dimensions are visible even before selections are made. For example, in the case of Hurricane Isabel's eye, the correlation between low pressure and low temperature was already visible. Based on this, the selection could be made to further study the relation and to show the eye in the linked views. Generally speaking, the point to polyline mapping characteristic of PCPs facilitates the selection of interesting patterns over all dimensions.

6.6 Conclusions and Future Work

We have demonstrated the use of our interactive tool for exploring large volumetric datasets using PCPs, linked views, and interactive brushing selection with three large time-varying datasets. The results in terms of clarity of visualization and interactive response times were quite encouraging.

We do not want to suggest that PCPs are the only or even the best way to analyze large high-dimensional datasets. Our aim is to show that PCPs can be fruitfully used with full-sized time-varying volumetric datasets, and to make it possible to integrate PCPs in an interactive multiple-linked-views type of environment.

As noted in section 6.3.4, the number of joint-histograms that has to be precomputed scales quadratically with the number of dimension of the dataset. This makes application of the proposed technique difficult when the number of dimensions exceeds about 20, as both pre-processing time and used disk-space grow quadratically. However, even without the pre-computed histograms the data can be explored, as interactive brushing does not depend on the joint-histogram data. The temporal navigation though, is slowed down considerably, as each timestep has to be loaded from disk. We intend to partly alleviate this problem by calculating the joint-histograms in an on-demand fashion, so that only the histograms related to the current axis ordering are computed. Also, we intend to investigate the usage of automatic axis-ordering algorithms to further aid the user in exploring higher dimensional data.

The proposed normalization method worked well in our cases, but it focuses mostly on displaying the relative distributions, and does not allow for quantitative display of data values. To alleviate this, we intend to provide each visible parallel axis with a set of tickmarks that are equally spaced in the original data domain, so that an intuitive mapping can be made between the normalized and the original data values. Non-linear tickmark placement however is not a trivial task, and the ability to mentally transform them to data ranges will probably vary between viewers.

Although the techniques used were designed for efficiency, the current implementation can definitely be further optimized for speed. We intend to do this by avoiding operations on the full data as much as possible, and by applying a streaming data-on-demand strategy. This will allow us to use only a subset of data for previewing, and load the full data in the background. We also want to explore clustering techniques for reducing visual clutter. An interesting addition is to enable partial histogram equalization, so that the user can smoothly change from the original to the normalized data. Finally, we intend to make the PCP tool freely available to the research community.

Acknowledgements

The processing software makes use of the `teem` toolkit, which is available for download at `http://teem.sourceforge.net`.

The authors will like to thank Bill Kuo, Wei Wang, Cindy Bruyere, Tim Scheitlin, and Don Middleton of the U.S. National Center for Atmospheric Research (NCAR) and the U.S. National Science Foundation (NSF) for providing the Weather Research and Forecasting (WRF) Model simulation data of Hurricane Isabel.

We also would like to thank Thijs Heus and Harm Jonker of Delft University of Technology for providing the Large Eddy Simulation datasets.

7

Smooth Graphs for Visual Exploration of Higher-Order State Transitions

Published as: J. Blaas, C. P. Botha, E. Grundy, M. W. Jones, R. S. Laramee, F. H. Post: *Smooth Graphs for Visual Exploration of Higher-Order State Transitions*, IEEE Trans. Vis. Comput. Graph 2009 15(6): 969-976 (2009)

In this chapter, we apply the multiple linked view metaphor to a different type of problem, that of the interpretation of biological sensor data. Sensors attached to marine animals gathered acceleration, pressure, temperature and other characteristics over the course of a day at a rate of 20 Hertz. The data samples are subsequently labeled into multiple categories by means of a trained classifier. Since the interest of the biologists lies in interpreting the global behavioral patterns of animals, we developed a graph-centric exploration method that focuses on recurring sequences of states. We introduce a new smooth-graph technique, that complements state transition graphs with information on the temporal context in a visual way.

BioExplorer shows well how techniques from information visualization can be used to tackle the analysis of highly complex temporal multi-scalar data by means of linked *abstract* representations.

Abstract

In this chapter, we present a new visual way of exploring state sequences in large observational time-series. A key advantage of our method is that it can directly visualize higher-order state transitions. A standard first order state transition is a sequence of two states that are linked by a transition. A higher-order state transition is a sequence of three or more states where the sequence of participating states are linked together by consecutive first order state transitions.

Our method extends the current state-graph exploration methods by employing a two dimensional graph, in which higher-order state transitions are visualized as curved lines. All transitions are bundled into thick splines, so that the thickness of an edge represents the frequency of instances.

The bundling between two states takes into account the state transitions before and after the transition. This is done in such a way that it forms a continuous representation in which any subsequence of the timeseries is represented by a continuous smooth line. The edge bundles in these graphs can be explored interactively through our incremental selection algorithm.

We demonstrate our method with an application in exploring labeled time-series data from a biological survey, where a clustering has assigned a single label to the data at each time-point. In these sequences, a large number of cyclic patterns occur, which in turn are linked to specific activities. We demonstrate how our method helps to find these cycles, and how the interactive selection process helps to find and investigate activities.

7.1 Introduction

One of the common ways to visualize state transition sequences is by using graphs. Each node represents a state, and an oriented edge between two nodes represents a transition between those two states. For the exploration of time-series label data, such a graph can be constructed by examining all succeeding pairs of states and generating a set of edges between the nodes representing them.

While these graphs give a good overview of the transitions between states, one important aspect is lost in the visualization: the context in which these transitions occur is not visible. Figure 7.2 shows how the first-order transition graph may ambiguously represent multiple underlying sequences. The leftmost graph could either correspond to the sequence ABCABCABC..., CDECDECDE..., i.e. multiple repetitions of each of the triangles as shown in the rightmost graph, or the sequence ABCDECABCDECA..., i.e. all states in one long sequence passing multiple

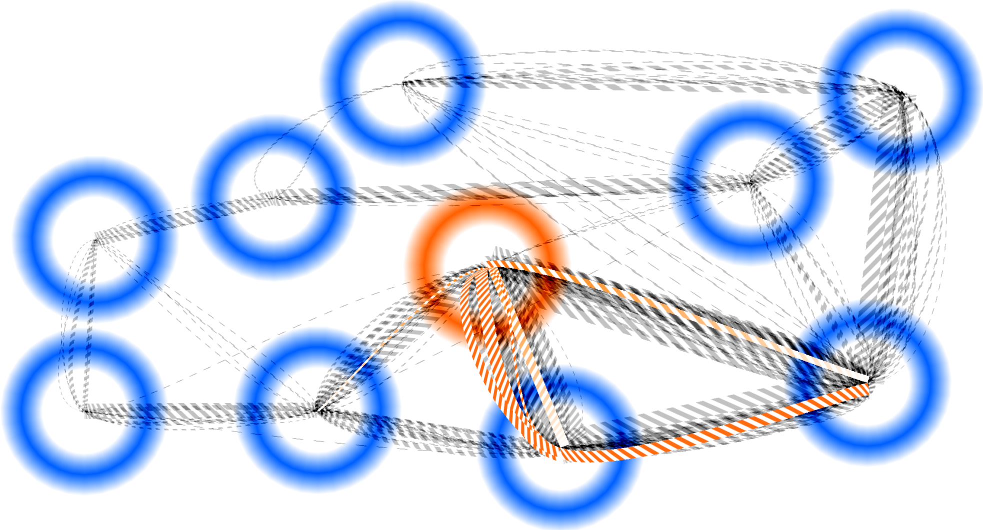
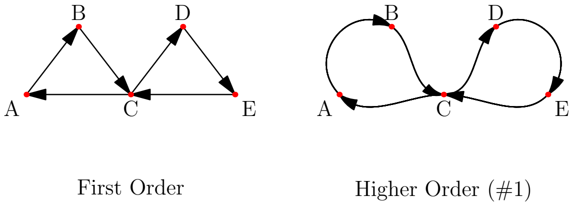


Figure 7.1: A smooth graph representation of a labeled biological time-series. Each ring represents a state, and the edges between states visualize the state transitions. This graph uses smooth curves to explicitly visualize third order transitions, so that each curved edge represents a unique sequence of four successive states. The orange node is part of a selection set, and all transitions matching the current selection are highlighted in orange. For color plate, see Figure A.18 on page 191.



First Order

Higher Order (#1)

Higher Order (#2)

Figure 7.2: A graph representation of a state transition sequence. The leftmost figure uses classical first order transitions, making it impossible to distinguish the sequence ABCABCABC…, CDECDECDE… *from the sequence* ABCDECABCDECA…. *The rightmost figures take into account higher-order transitions to visualize the order in which the state transitions occur.*

times through C as shown in the middle graph. One of our goals is to visually disambiguate these two situations. In the following sections, we elaborate how this can be done by taking into account higher-order state transitions when drawing the edges. The middle and rightmost figure show how our method uses curved edges to emphasize the order in which the transitions occur, which gives each of the two sequences a unique visual representation.

7.1.1 Problem domain

Our method is designed specifically to handle labeled time-series data. In such data, a set of labels is defined, and each point in time is assigned one of these labels. Each of these labels can be seen as a state, and a change of state then forms a transition. One differentiating characteristic of these time-series data is that it contains information on the order in which state transitions occur. This also implies that statistical correlation can exist between the sequence of states taken to arrive at a node, and the transition taken to leave that node.

We want to be able to visualize these oft-occurring sequences of states, and we want emphasize the surrounding conditions in time under which these occur.

For example, the state C in the sequence ABCDECA... (see Figure 7.2), is visited in two different contexts. It can be reached either from state B, or from state E. The subsequent state is determined completely by the previous state, as arriving from B means the next state will be D, and arriving from E the next state will be A.

This correlation between the incoming state and the outgoing state can be put into a broader perspective by looking at so called higher-order transitions.

7.1.2 Higher-order transitions

A standard first order state transition is a sequence of two states that are linked by a transition. A higher-order state transition is a sequence of three or more states where the sequence of participating states are linked together by consecutive first order state transitions. Our definition of a higher-order state transition is derived from that used with higher-order Markov models.

Representing the conditions, or context, under which transitions occur is an important task, as it leads to insight into the temporal linkage between the corresponding states. In other words, the fact that a given state change depends on the previous state change and has an effect on the following state change plays an important role in exploring such phenomena.

We have designed a visual method for representing these higher-order state transitions. In section 7.3, we present a representation that visually disambiguates the higher-order transitions, providing a *smooth graph* representation for visualizing state transition sequences.

A major point of novelty is that the presented method uniquely represents each unique higher-order transition with a curved line in such a way that any chain of subsequent transitions is visualized in a smooth and continuous fashion. This makes it not only easy to pinpoint frequent-occurring transitions, but it also helps to identify the set of state transitions that lead to them by implicitly visualizing the temporal context.

7.2 Related Work

A state transition graph represents a system of states and state changes. Each state is represented by a vertex and each state change by a directed edge. State transition graphs are generally represented by simple node-link diagrams, where each node represents a state and each link a state transition [Pre08].

These node-link diagrams have been extended with 3-D layout algorithms [JJ95] and also with intelligent 3-D positioning of edges and nodes that better shows the hierarchy and organization of the state transitions [vvv02, vvv01]. Taking a different approach to coping with highly complex node-link diagrams, Leuschel et al. presented techniques to reduce the complexity of the underlying state transition graphs by merging nodes [LT05].

Holten [Hol06] proposes *Edge Bundles* for visualizing hierarchical data, where adjacency relations are bundled together using B-spline curves. He also shows that such a method can be combined with any of the existing major tree visualization techniques. Cui et al. [CZQ$^+$08] also cluster edges within graphs and encode additional information by using color and opacity. They also employ animation to show intermediate transitions from the original graph to the edge clustered graph, and to allow different levels of detail. They smooth polylines to ensure visually pleasing paths through the graph. Another example of using curves for graph visualization is given by Eppstein et al. [EGM07] where they visually identify bicliques in a graph by depicting them using Bézier curves such that each curve passes through the central point for the biclique. Although these three methods use smooth lines, they do not use any continuity constraints to represent higher-order sequences as we do here.

Continuity has long been identified as important to human perception, and Field et al. [FHH93] demonstrate the ability of the human visual system to successfully follow continuous paths even against randomly oriented background paths, or with partially obscured paths. Curves have been used in parallel coordinates by Theisel [The00] to show correlations in non-adjacent axes, and by Graham and Kennedy [GK03] to distinguish axis crossing visually through curve continuity.

Ham and Rogowitz [vHR08] carried out a study where participants created their own graph layouts which could then be measured to try to evaluate the importance of various visual features. Through this process they are able to provide recommendations for graph drawing algorithms.

Also related to this work are visualization techniques for biological sensor data including Ware et al. [WAPW06] where accelerometer data was visualized using TrackPlots, which enabled scientists to check the theory that whales roll onto their sides for specific prey capture, and Grundy et al. [GJL$^+$09] where spherical plots, spherical overlays, spherical histograms and a posture state graph are shown to be effective at leading to biological understanding

Pretorius et al. [Pv06, Pre08] presented a unified approach to visualize highly complex state transition graphs, employing node-link diagrams to visualize a hierarchical clustering of the different states, the bar tree to visualize the number of occurrences of states and finally an arc diagram to visualize the actual state transitions. A method was also presented whereby multi-variate graphs could be more explicitly explored: Nodes are arranged in a source group and a target group, and all edges are shown in between [Pv08] and queries can be interactively performed. This method was also applied to state graph visualization. Herman et al. [HMM00] provide a good survey of graph drawing techniques specifically aimed at information visualization.

All of the mentioned techniques have as one of their major goals successfully coping with the high complexity of most state transition graphs. However, none of them take into account higher order state changes, i.e. state changes where the context, the states before and after the current state change, plays an important role. None of the techniques are able to visualize or represent these higher order state changes in any way.

The method that we present in this chapter explicitly represents higher order state changes, thus enabling the exploration of state transition graphs where the context of a state change is important.

7.3 Method

A number of Gestalt principles exist that are applicable to the design of visual representations, as discussed by Ware [War04]. In the design of our visual representation we focus on three aspects in particular: continuity, connectedness and closure. The principle of continuity states that elements that make up a smooth and continuous shape are more likely to be perceived as a visual entity. The connectedness principle entails that a relationship can be expressed by graphical objects by for example connecting them with lines. This principle is one of the prime principles of node-link diagrams, and a common method for visually representing relationships between entities. Closure implies that closed shapes are likely to be perceived as a single entity. We want to make sure that closed shapes likely occur where interesting patterns emerge, as to focus on the cyclic patterns.

7.3.1 Design goals

Based on the principles above, we have formulated two main design goals. Firstly, each higher-order transition should be represented uniquely. Secondly, the visual representation should be continuous and smooth.

Uniqueness

When each higher-order transition is uniquely represented visually, each state in the transition influences the form, so that the combination of all states in the transition together uniquely identifies the shape. Since a first-order transition can occur as part of several higher-order transitions, this requires that edges are duplicated.

Figure 7.3 illustrates how this applies to a real-world example. Marine birds that dive have two different types of diving behavior, they either dive from flight, or they dive while swimming. In the analysis of our labeled time series, distinguishing between a dive that is initiated from flight and one that is initiated from a swimming position is a frequently occurring task. If we take into account second-order transitions, then the DIVE, FLOAT transition is part of the two second-order transitions FLY, DIVE, FLOAT and FLOAT, DIVE, FLOAT.

Instead of having to query the state sequences explicitly for the two patterns, our goal is to incorporate the higher-order information into the visualization, making a visual distinction between the two occurrences.

Continuity

As the transitions that we visualize are all part of a longer sequence, we also want to make this clear in the visual representation. We constrain the visual representation of the edges in such a way that the longer sequence of transitions that links these edges together still forms a continuous and smooth curve. We aim for angular

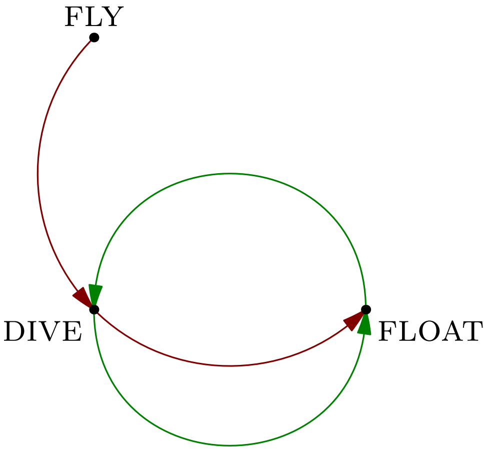


Figure 7.3: Smooth graph representation of a set of two state sequences: FLY, DIVE, FLOAT *in red and* FLOAT,DIVE,FLOAT *in green. Note the duplicate edges between* DIVE *and* FLOAT

smoothness, as this makes it easy for the human eye to track corresponding edges that go through a node, linking the incoming edges to the corresponding outgoing edges in an intuitive fashion.

7.3.2 Techniques

We have designed a set of techniques that fulfill both design goals, while maintaining a representation that is close the classical graph-based layout. The techniques we have selected to build the smooth graph representation are as follows: 2D Graph Layout, Smooth Nth-Order Interpolation, Edge Bundling, Interaction, Directionality Through Texturing. Each is discussed in more detail in the following sections.

2D graph layout

The states and transitions are visualized using a two dimensional graph layout. The main advantage of a two dimensional graph layout is that this allows for an intuitive representation of states (as nodes), while providing us the freedom to use curvature when drawing the edges. Also, the extra degrees of freedom introduced by using curved lines allow us to enforce smoothness and continuity constraints on the edges.

As a first step, we perform a classical 2D graph layout, based only on the first order transitions. From the existing graph layout algorithms in the literature, we have

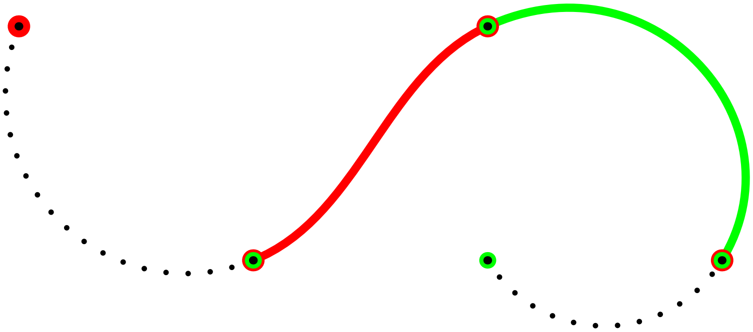


Figure 7.4: Smooth spline interpolation of third order transitions, using a fourth order spline. Each of the colored segments is drawn by fitting a smooth curve through the four points of the same color.

chosen to use the Fruchterman-Reingold force directed layout algorithm [FR91], a simple and robust algorithm that yields satisfactory results.

Once the layout algorithm is done, the locations of the nodes are set, but further refinements of the location may be made during the interaction process by the user. This allows the user to modify the proximity of the nodes so that visual groups can be built during the exploration process, as the knowledge the user has of the underlying phenomena grows.

Smooth Nth-order interpolation

As stated in section 7.3.1, one goal is to visualize each unique higher-order transition individually. Since multiple higher-order transitions may contain the same first order transitions, there may be multiple edges between a pair of nodes that belong to different higher-order transitions.

A good vehicle for maintaining continuity and smoothness throughout all edges is to use interpolating splines. The user decides upon the transition length, N, and then we use the Catmull-Rom spline, so that each Nth-order transition can be represented by a spline of order N+1, as it should smoothly interpolate through N+1 nodes.

The advantage of this method is that the segments for successive transitions form one smoothly connected spline (see Figure 7.4), since the overlap between the higher-order transitions enforces smoothness along the connecting edges. This continuity constraint therefore guarantees that the entire state sequence, or any subsequence thereof, can be followed visually as a continuous line, greatly enhancing the visibility of the order in which states occur.

The only place where lines overlap, is where the higher-order transition is exactly the same, so where not only the from-and-to nodes of an edge are the same, but where the entire sequence of N+1 transitions is the same.

Edge bundling

Since each higher-order transition maps to exactly one edge trajectory, we can bundle edges based on the frequency of the transition. We modulate the thickness of the edges to represent the number of times the transition was taken.

This is an implicit form of edge bundling, as all the edges taking part in a bundle are never explicitly drawn. Each unique state sequence, no matter how many times it occurs, is only drawn once, with thickness representing the number of occurrences. The thickness modulation makes it easy to track the frequently occurring transitions, and to find frequently occurring cycles within the data.

Interaction

To further inspect the state sequence, interactive selection can be used to investigate the graph. To assist the user in finding transitions of interest, we designed a user interface to incrementally build a constrained selection. The user can create such a selection by sequentially clicking on a number of nodes. All edges that connect the selected nodes in order then become part of the selection, and are visually highlighted (see Figure 7.7).

The selection process employs three steps. First, the starting state is selected by clicking. Then visual feedback is given by highlighting all possible continuations of the currently selected sequence. After which the selection criteria can be extended by selecting additional nodes.

By using color to highlight the selected edges, there is no need to make any changes to the layout of the nodes, which improves the visual coherence.

As we demonstrate in section 7.5, the selection can be rapidly linked to correspond with all the ranges in time at which the selected sequence of transitions occurs.

As a secondary form of interaction, the nodes of the graph can be rearranged by the user. This creates direct feedback on which edges contain the moved node in their higher-order transition, as all these spline edges interactively changes shape when the node is moved.

Ware and Bobrow [WB04] have demonstrated that this class of dynamic highlighting, particularly when combined with other highlighting methods (static highlighting through coloration and motion highlighting using crawl edges), is a useful aid for large graph querying.

Directionality through texturing

Classically, arrowheads are often used in graphs to visualize the direction of an edge. In our case, we want to be able to preserve the visual continuity that the edges have throughout nodes. Arrowheads distort the perception of the continuous spline, so we have opted for a basic-yet-effective approach of using texturing to visualize the orientation of the edges (see Figure 7.5).

Each edge is longitudinally split into two halves and each of these gets assigned a thickness, or width, based on the number of times that the transition in the corresponding direction occurs. A cyclically repeating texture is applied to each of the halves, and is animated in the direction of the state change that it represents in order to enhance its representation.

This is reminiscent of the approach presented by Wegenkittl et al. for the visualization of continuous state changes, called trajectories, through the four-dimensional state space of the Wonderland econometric model [WGP96]. They employed spiral textures that were animated in the direction of the state change, mapped to the tube representing that trajectory.

Our texturing scheme proved flexible enough for later customization, so that further experiments can be done to find an optimal visual representation for the directionality. As frequently used in flow visualization, we currently use an intensity gradient along the direction of the line to indicate orientation of the edges.

7.4 Implementation and Performance

We have implemented the proposed method as a functional component within a larger experimental data-exploration system. As the focus of the system lies on the exploration of large time-series data, we have required that all queries and drawing are performed at highly interactive frame-rates, even for larger datasets.

We use tuple hash tables to make searching the higher-order transitions efficient (see Figure 7.6). As the problem is similar to fixed-length string searching, a multitude of methods are available to efficiently count and enumerate the time-points at which a specific sequence of states occur. The hash table uses the Nth-order transitions as keys, and maps those to a sorted array of time-points at which the corresponding sequence occurs.

These hash tables can not only be used to determine the width of each drawn edge (through counting the number of occurrences), but they can also be used to perform the queries needed during the selection process. Once a selection of states has been made, the hash table can be queried to find all sequences that start with the selected

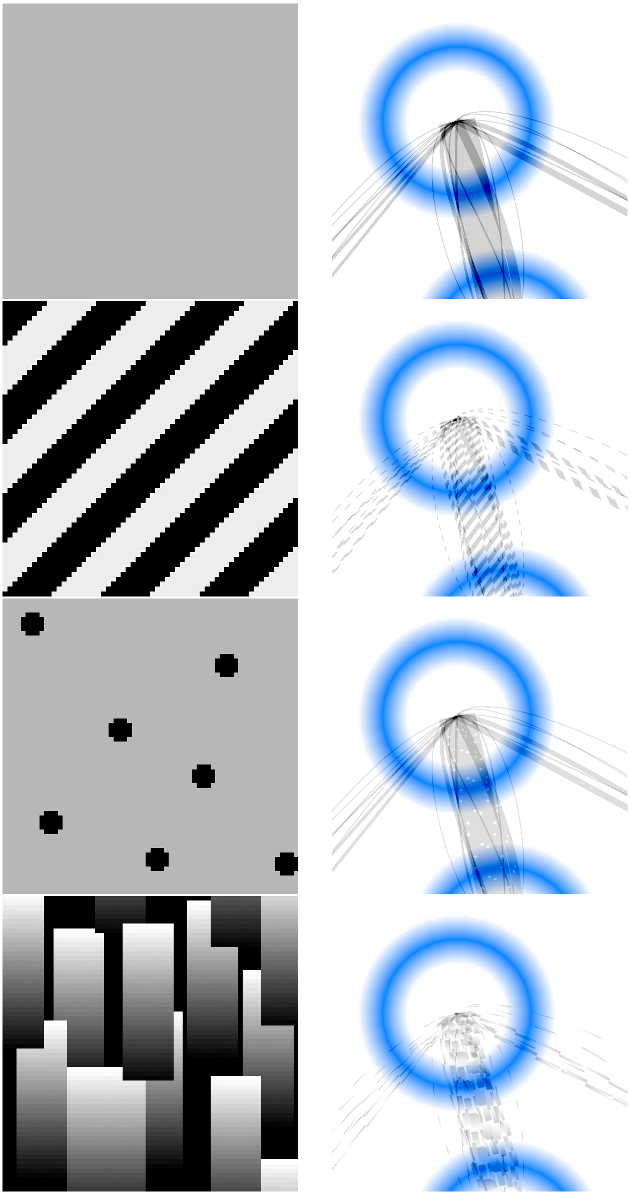


Figure 7.5: Edges drawn with four different textures, from top to bottom: solid, striped, stippled and gradient. The left column shows the texture used, while the right column shows the resulting graph.

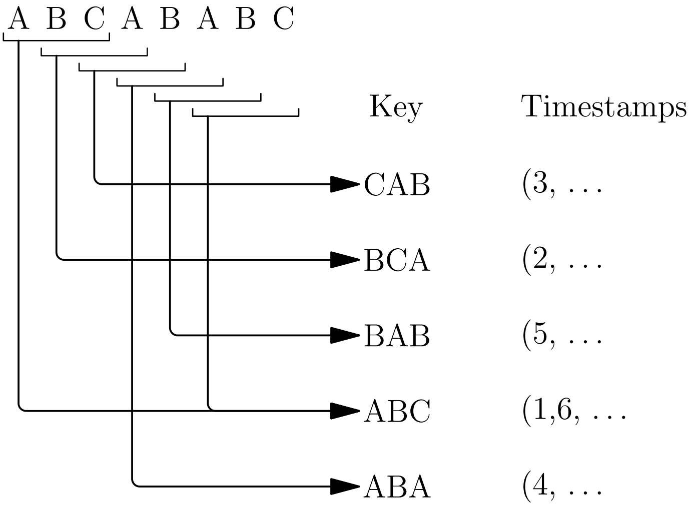


Figure 7.6: Contents of the hash table after storing the sequence ABCABABC

sequence. When a limited range of time-points is to be examined, a binary search in the sorted time-point array suffices to filter the results.

Since each unique higher-order transition only has to be drawn once, the rendering is fast, even for large numbers of samples. Though the drawing of curved lines is a relatively slow operation, the number of edges drawn is sufficiently low that this does not impede overall performance.

In our data-set of one million data points and 10 unique labels, the number of 3rd order edges that had to be drawn was only 238. While in random state sequences the number of unique Nth-order transitions scales exponentially with N, in natural data the number rises fairly slowly (see Table 7.1).

The drawing of both nodes and edges was implemented in OpenGL. The curved edges are drawn using strips of triangles, so that the thick curved edges can be filled efficiently. The total time it takes to draw a graph with 238 edges and 10 nodes is below 5 milliseconds on a modern workstation. This allows for the entire application to be used interactively. Figure 7.8 shows a snapshot of the rendering.

7.5 Application: Biological Sensor Data

The motivation behind the development of this visualization tool, as shown in Figure 7.9 on page 138, was to enable researchers (biologists) to interactively explore

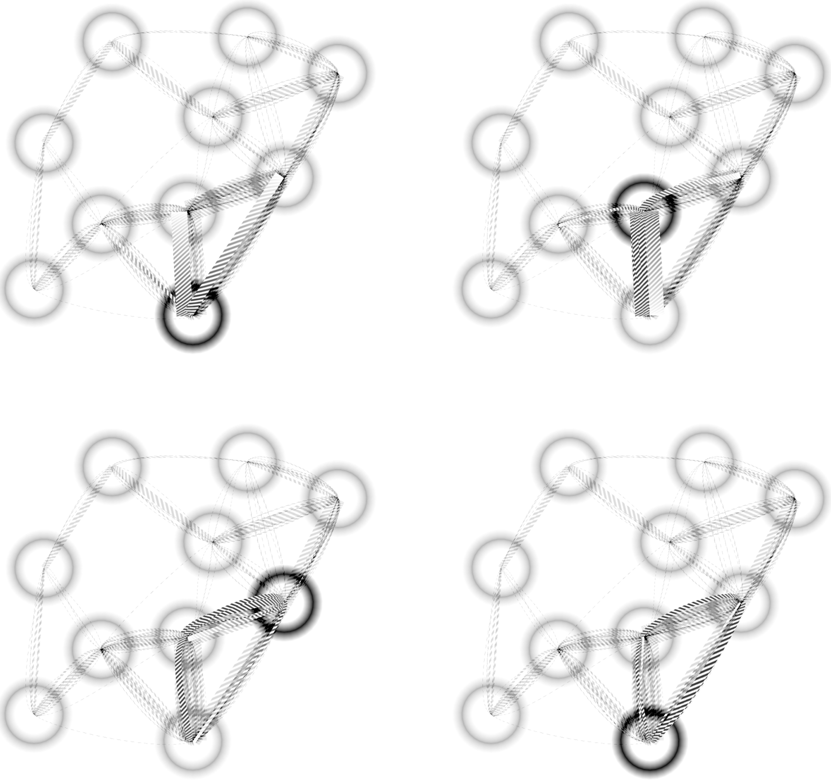


Figure 7.7: The stages of the selection process, from left to right: A starting node is selected and three additional nodes are added to the sequence

Transition Order (N)	Edges (5 states)	Edges (10 states)
1	18	41
2	40	108
3	70	238
4	105	440
5	142	689
6	188	989

Table 7.1: The total number of edges in the graph for a real-world data-set of 1 million samples, as a function of the order of the transition. The original multi-scalar data was clustered with a chosen number of labels. The two columns show the results for 5 and 10 labels respectively.

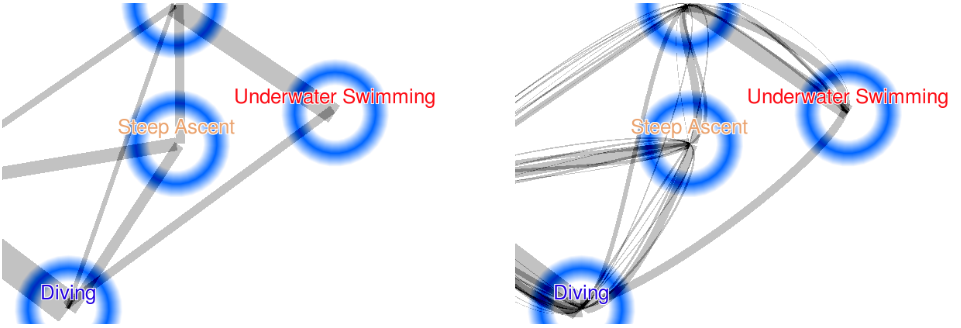


Figure 7.8: Comparison between first-order (left) and third-order (right) state transition visualization. Notice how the edge between diving *and* underwater swimming *consists of just a single line, even in the third-order representation. This directly indicates that this transition only occurs from within a single context, so there can be only a single specific node that precedes the transition, and there can be only a single specific node that succeeds it. Following the smooth edge in both directions, it is clear that the succeeding node is the top one, while the preceding node is apparently positions to the far left of the graph.*

biological sensor data. The multi-attribute data is collected from sensors attached to animal subjects over durations of hours to days at a frequency of 10–20Hz. This results in a large time-span (many hours/days) of small-scale behavior (tenths of seconds for scratching behavior to a couple of minutes for a diving behavior). The tags record data from an accelerometer, a magnetic compass and environmental sensors. Due to the large volume of data and complex inter-relationships at play between the different sensor channels the behavioral data is difficult to analyze, and requires extensive experience to interpret. Currently, tracking data is visualized as primitive, 2D time series plots (the ScalarView in Figure 7.9 on the next page), and analysis is based upon simple summary statistics. Interpreting and drawing conclusions about behavior from accelerometer data requires a great familiarity with both the behavior of the subject (to aid interpretation), and a great wealth of experience of working with accelerometer data of this form. Cognitive integration of experience with a scalar view of the data allow researchers to make statements about the behavior of the animal within its environment.

The cognitive cost can be reduced by employing clustering techniques upon the data, and visualizing the resulting cluster labels as part of the scalar view. This presents a good summary visualization, but for long time series data, it is still difficult to understand the behavior of the animal (for example by identifying the predominant behaviors or determining any unusual behaviors). The color bars above the scalar data in figure 7.9 give one example of visualizing the cluster information.

We have previously shown [GJL$^+$09] that clustering and visualizing the data using

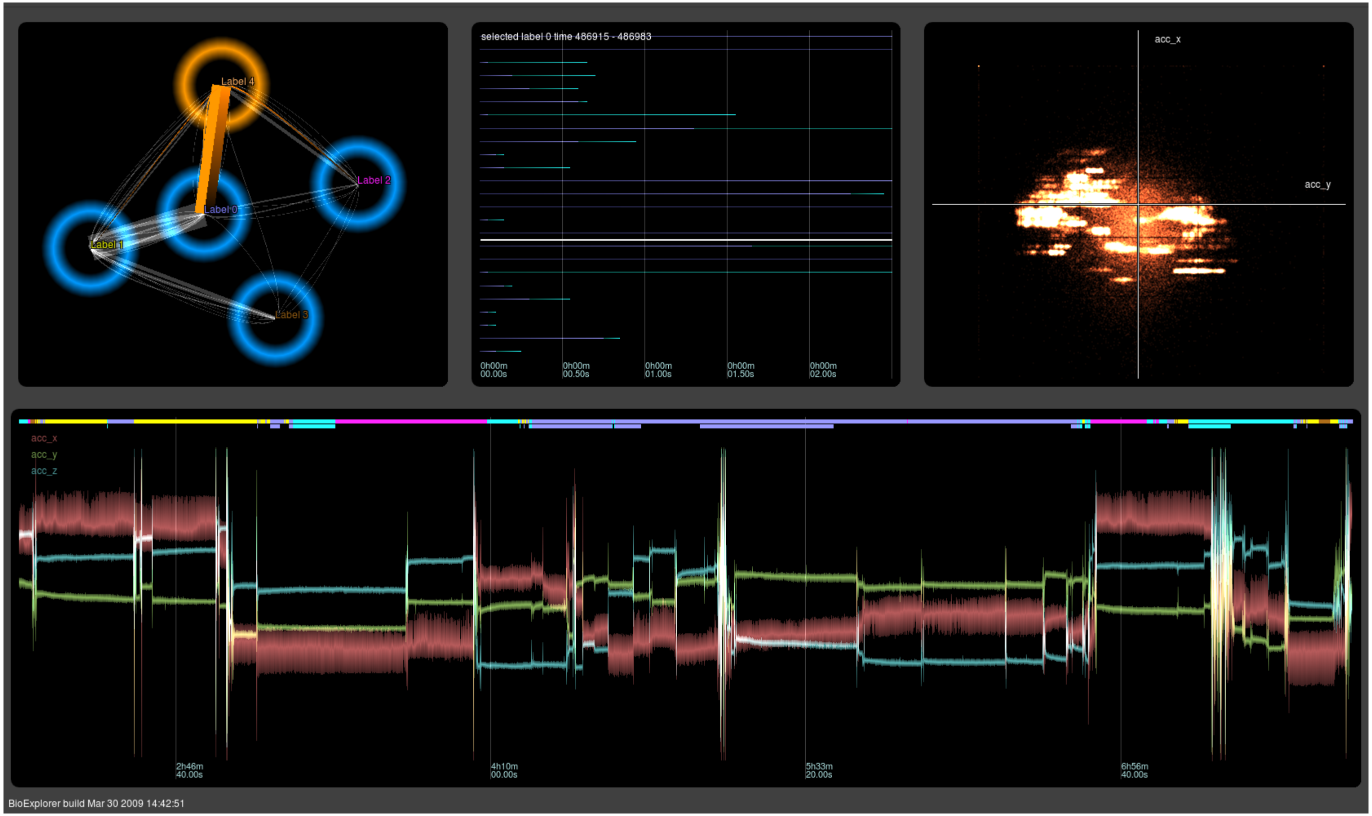


Figure 7.9: A screenshot of the user interface of the exploratory application. The three views on the top of the screen represent (from left to right) the GraphView, the SelectionView and the ScatterPlot. The bottom of the screen shows a ScalarView in which both the accelerometer measurements as well as the assigned labels are shown. For color plate, see Figure A.20 on page 192.

spherical plots, spherical overlays, spherical histograms and a posture state graph are effective at leading to biological understanding. In that work, the biggest aid to simplifying a large dataset is the *Posture State Graph* (PSG). For PSG visualization, each cluster is represented as a node in the graph, and then any transition between nodes is represented as an edge. Edge width can identify the number of transitions between nodes. Nodes correspond to particular orientation and behaviors and therefore present a visual summary of the data. In actual use, the most productive mode was to mix the spherical plot with overlay and PSG which gave good contextual information regarding predominant behaviors, leading to insight about behavioral sequences of the animal. The main problem with such a view was that it became cluttered. It presented a good overview, but still required some effective interaction to understand the relationship between predominant behaviors, unusual behaviors or to appreciate the time spent during particular behaviors.

The new smooth graph and visualization directly addresses the problem of visual clutter by removing the need to have contextual information in the same window. Secondly through the use of curves, higher order transitions can be identified, which provides visual feedback of behavioral sequences, and edge bundling allows a simple appreciation of the frequency of those behavioral sequences, neither of which could be achieved without extensive interaction in previous visualization approaches.

The following sections introduce the multiple linked views of the exploratory framework and the different aspects of the data each presents. Some example explorations of the data are given, with the insight the new approach provides.

7.5.1 Components

The screen is split in four viewports (see Figure 7.9): the GraphView, the SelectionView, the ScatterPlot and the ScalarView. The ScalarView displays the original scalar attributes measured at each time-point in a chart. This allows the user to see the scalar values that were used as a basis for the clustering step. As an overlay above this data, the states are visible as a colored bar. Colors are consistently chosen so that throughout all visualization the same color corresponds to the same cluster label. If a selection is active, for example for a specific sequence of states, then a secondary bar highlights all the ranges in the time-series that match the current criteria. Since the number of data samples can be in the order of millions, the user can freely and quickly zoom in and out to focus on specific ranges in time, similar to [JS98].

The SelectionView gives an instant overview of the currently selected sequences. It depicts the time spent in each of the states within the selected sequences. Through this basic visual representation it provides the ability to make selections based on properties such as total duration of the sequence. This has been a valuable utility to spot and remove 'bad' labels that were introduced due to noise, as these often have a very short duration. The SelectionView also works as an aid to navigating the ScalarView, as when the displayed bars are clicked, the ScalarView smoothly zooms to the corresponding time range.

The ScatterPlot displays a multi-attribute scatter plot of the current selection, assisting in finding correlations between attributes. To cope with the large amount of data samples the points are rendered transparently, to keep the generated image clutter-free.

The GraphView links all the previously presented views by providing the main means of state space exploration. The previously described selection process is fully integrated into the linkage between the views, so that during the construction of a selection set, all views correspond to the data associated with the selection. To cope with the limited amount of screen space available in this split-screen scenario, the graph view supports both zooming and panning, so the area of interest can be changed interactively.

7.5.2 Exploration

To examine the performance of our method on real-world data, we have used the exploratory framework to analyze a data-log containing sensor information from a penguin. Over the entire course of the log, several different activities occur, such as diving and swimming. The data is labelled using a k-means clustering algorithm which assigns labels based on the posture information extracted from the accelerometer data.

Finding repeating patterns

When exploring the penguin data, we are interested in finding the main activities, and then reasoning about any activities that follow a different pattern. The clustering identifies the various states of the animal, and when related to body posture these can be labelled as seen in figure 7.10 (diving, underwater swimming, ascent, steep ascent and surface swimming). The edges show where transitions occur between states, edge thickness shows the frequency of those transitions, and edge continuity shows behavioral sequences within the data.

Over and above the visual information provided by previous techniques, researchers can now see a summary of the behavioral sequences and can begin to reason about the animal from just a static view of the data. For instance we can see a continuous path going from diving, through underwater swimming, ascent, steep ascent, and back to diving. This shows the main cycle of a feeding penguin (we shall call this a dive sequence in the following). By selecting that path (figure 7.11) we are then presented with further statistical summary in the form of the SelectionView which shows us that the dive sequence lasts about 15–20 seconds in the majority of cases. By selecting one of the dive sequences in the SelectionView, the ScalarView shows us the original scalar data where the biologist can verify that this labelling is correct through the pattern known from experience.

In fact the full dive sequence involves another selection of *diving* before the penguin enters the surface swimming state. This is due to the classifier interpreting the body orientation the penguin adopts as it exits the water (to avoid its inherent buoyancy from projecting it out of the water), being similar to that when it prepares to dive.

During development we found it useful to include the ability to sort the SelectionView by the total duration of the sequence. In this case it clearly shows the expected normal distribution for dive length.

From the GraphView we can see another thick path between Surface Swimming and Diving. This is the penguin performing shallow dives in order to swim efficiently just below the surface, before surfacing to breathe, and then diving again. The SelectionView shows that each lasts about 5 seconds. Sometimes it dives a little bit deeper, and goes through the ascent and steep ascent transitions (but this is a rare occurrence in the data as conveyed by the thinner edge transitions).

Noisy data

Another advantage of the higher-order visualization is that the edges that move back-and-forth between two nodes repeatedly are separated from the edges that make the transition as part of a complex higher order transition. For example, the thick straight edge between *Surface Swimming* and *Steep Ascent* indicates that many transitions occur between these two nodes. We can examine this phenomenon closer by building a selection from *Surface Swimming* to *Steep Ascent* and back again. Through examining the SelectionView, we learn that the duration of these two states is often very short, as the colored bars are fairly short. When we select one of the bars, the scalar view zooms in on the time range of the occurrence and we can establish why there are so many back-and-forth transitions. Looking at the scalar values, we see no particular patterns that warrant these erratic state changes, as indicated by the label bar on top of the figure.

The logical explanation for this is that the boundary between these two clusters is particularly sensitive to noise, therefore values that are in between the two clusters are often misclassified. As our sensor data is inherently noisy, this particular group

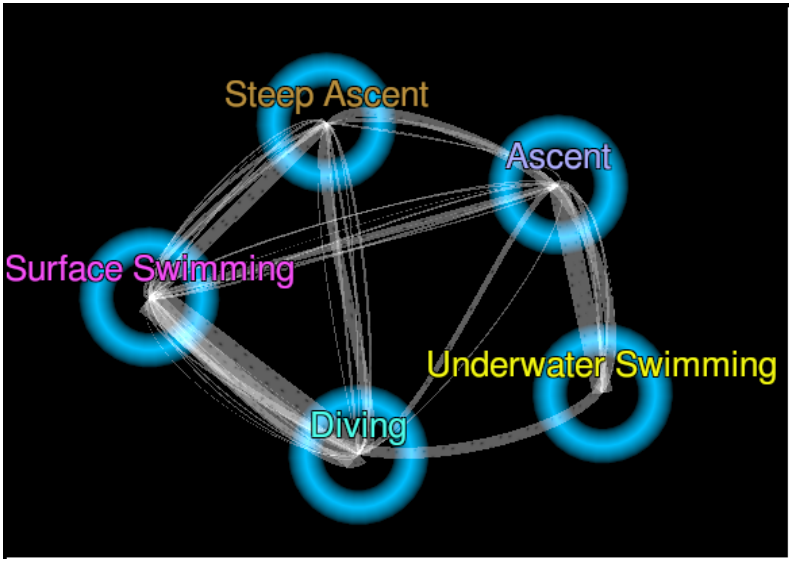


Figure 7.10: Smooth graph of 5 states in penguin dataset. The data has been classified into the states shown using k-means clustering. The label colors correspond to the color bar in the ScalarView and the individual selection bars as seen in Figure 7.11.

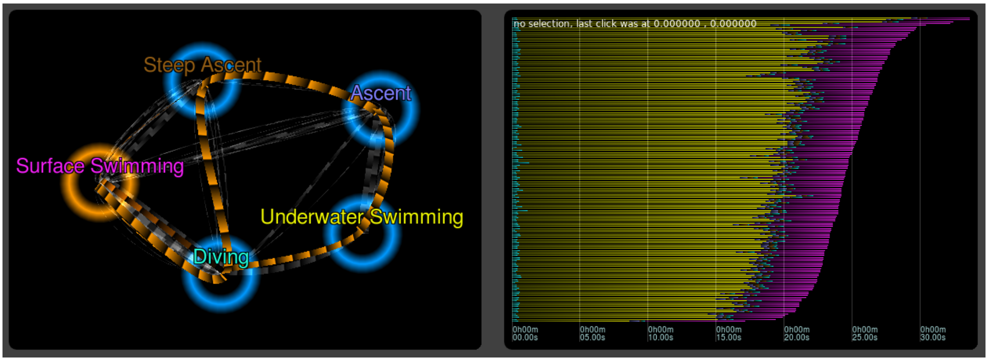


Figure 7.11: Selection of the penguin's feeding cycle by querying the selecting the sequence of nodes corresponding to diving, underwater swimming, ascent, steep ascent, diving and surface swimming. For color plate, see Figure A.19 on page 191.

of transitions should be refined before further analysis. Another course of action can be to decide that the two states probably refer to a very similar posture, and that they should therefore be merged into a single state. We can visually perform this action by positioning the two nodes on top of each other, or we can reprocess the label data and substitute one state for the other to get rid of the state entirely.

7.6 Conclusions and Future Work

We have presented a novel visual representation for state transition graphs that focuses on the temporal coherence between sequences of transitions. Our representation uniquely visualizes higher-order state transitions using a smooth spline representation for the edges. The method we designed adheres closely to the principles of continuity, connectedness and closure.

The higher order transitions are represented as curved lines in such a way that each sequence of states forms a smooth continuous curve through their respective nodes. The visual continuity of the curve is easily tracked by the human eye, making correlations between in-going edges and out-going edges visible in an intuitive way. The continuity holds for state sequences of arbitrary length.

The fact that the presented method uniquely represents each Nth-order state transition visually has proven useful in finding frequently occurring state transitions in noisy real world data. We have demonstrated this and other issues by exploring a real-world dataset.

The limitations of our method manifest when there are either a large number of states to visualize, or when even higher order state transitions need to be visualized. In both cases the visual complexity increases greatly, complicating the use of our method. In the former case, the high number of nodes clutters the display and leaves little room for routing the edges. In the latter case, the higher order of transitions simply leads to more permutations of the spline interpolation, or more types of edges. This can also be seen in table 7.1, where increasing the order of the transitions leads to an exponential increase in the number of edges. However, for the datasets we have experimented with up to now, neither of these limitations has posed any problems.

In future work, we are planning to investigate whether the idea of representing higher order state transitions using curved edges can be successfully applied to other state transition graph visualization methods in addition to two dimensional node-link diagrams.

We are also planning to continue our investigation into the visualization of oriented edges by using animated texture, taking into account the recent work by Holten and van Wijk [Hv09].

Although tested through investigation with a biological sensor dataset, it is also important to evaluate the performance of curved edges when employed for typical tasks. We are planning a user study of the type carried out by Ware and Bobrow [WB04], and Holten and van Wijk [Hv09]. In particular we flag that the choice of an appropriate basis for comparison will require a careful experimental design.

Conclusions and Future Work | 8

8.1 Conclusions

In this thesis, we presented a number of new methods representing different aspects of a linked view based visual analysis approach for multi-field data. We strongly believe that human-in-the-loop visual data analysis on large and complex datasets is best aided by multiple linked and conceptually different representations, as demonstrated in Chapter 4 to 7. By abstracting the data hierarchically using a set of linked domains, meaningful high level representations can be built that still are linked to low-level detailed representations. Through grouping, clustering and aggregation the raw data is linked to summarized entities at the higher level, either by use of pattern recognition techniques or by manual interaction.

We applied these techniques to a number of different fields, ranging from neurological imaging to the study of marine animal behavior.

The following paragraphs provide a short overview of the conclusions of the main chapters of this thesis.

8.1.1 Interactive selection

In Chapter 2, we introduced two contributions to the field of DTI visualization. Firstly, we presented a technique to enable real-time selection of fiber bundles by manipulating multiple convex selection objects. We showed that our technique performs an order of magnitude faster than existing methods. By using an efficient spatial query structure, selection queries with three objects in a full brain dataset with 2.7 million fiber points could be performed at interactive rates.

Secondly, we tested the reproducibility of our selection method. For 10 subjects, two users independently selected the fiber bundles corresponding to the left and the right cingulum. Our analysis of the test data showed both a significant variation of the fractional anisotropy in the cingula over the different subjects and a significantly high correlation between the measurements by the two observers. This inter-observer reproducibility indicated that this type of approach can be used as a measurement tool in pre-clinical studies.

8.1.2 Combined visualization of multiple modalities

In Chapter 3 we presented an ordered toolset of visualization techniques for the interactive and simultaneous exploration of anatomical MRI, fMRI and DTI datasets with the goal of providing information about the resection area around a brain tumor. This visualization is embedded in a conceptual framework for multi-modal visual exploration.

Unique to our approach was the combined visualization across any of the three domains (tractography, functional and structural) and, very importantly, the rich interaction between these three domains. Our approach fused the full brain tractography with other domains and allows the user to filter information interactively by changing the necessary constraints to control the area of interest, both in location as well as in subject.

8.1.3 Linking physical and feature space views

In Chapter 4 we presented a dynamic coordinated view approach for the interactive visualization of (medical) multi-field data. We showed how scatter plots can be created with arbitrary projections of high-dimensional feature space. Subsequently, we demonstrated how techniques from pattern analysis can be interactively applied to facilitate a visualization-based multi-field exploration process. Using two case studies, we demonstrated the application of our approach to real medical data.

8.1.4 Visual data exploration

In Chapter 5, we presented a conceptual framework that provides structures and methods to ease the interpretation and analysis of longitudinal multi-field data.

We showed that the MULTI-2 framework is able to provide interactive exploration on large multi-modal datasets. We demonstrated a direct two-way coupling between statistical aggregate values and the underlying data, making it easy to investigate the data behind the values.

8.1.5 Interactive exploration of large temporal datasets

In Chapter 6, we demonstrated the use of our interactive tool for exploring large volumetric datasets using parallel coordinate plots (PCPs), linked views, and interactive brushing selection with three large time-varying datasets. The results in terms of clarity of visualization and interactive response times were quite encouraging.

We showed that PCPs can be fruitfully used with large full-sized time-varying volumetric datasets, and that it is possible to integrate PCPs into an interactive multiple-linked-views type of environment, even when handling datasets with thousands of time-steps, totalling 20+ gigabytes in size.

8.1.6 Smooth graphs

In Chapter 7, we presented a novel visual representation for state transition graphs that focuses on the temporal coherence between sequences of transitions. Our representation uniquely visualizes higher-order state transitions using a smooth spline representation for the edges. The method we designed adheres closely to the principles of continuity, connectedness and closure as discussed by Ware [War04].

The higher order transitions are represented as curved lines in such a way that each sequence of states forms a smooth continuous curve through its respective nodes. The visual continuity of the curve is easily tracked by the human eye, making correlations between in-going and out-going edges visible in an intuitive way. The continuity holds for state sequences of arbitrary length. The unique visual encoding proved useful in finding frequently occurring state transitions in noisy real-world data.

8.2 Discussion

Throughout the chapters of this thesis, we have always striven to find the most fitting evaluation methodology to validate our methods. Where our work resulted in an implementation prototype we made use of a combination of expert review and heuristic evaluation to study its utility. At a lower level, we have measured application performance, as interactivity is an important factor.

As opposed to the other chapters, Chapter 5 is a position paper. An experimental prototype is currently being developed to further the validation. This prototype will play a key role in the future to provide a high-level validation of the proposed methods.

The proposed systems are complex, with a high level of communication between the components, and they often have a human in the loop. As such, there is no single best evaluation methodology. The most suitable approach is the use of case studies at the top level, combined with user studies of the most used components, taking careful note of positioning in the context of Munzner [Mun09a].

The techniques we presented scale well with the size of the data, even up to several gigabytes in size, as is particularly prominent in Chapter 6. However, this only applies to the technical side of the issue: scalability in the sense of human interaction and perception is a different problem. It is clear that some of the representations we have chosen only work up to a certain number of elements. Scatter plots and parallel coordinates scale fairly well when clutter suppression techniques are applied, but the state-transition graphs quickly become cluttered as the number of nodes in the graph rises.

Scalability towards more diverse and complex data is an issue that requires further research, as more and more of the representations become unusable for large datasets. One of the most effective ways to handle large datasets is to keep the number of items represented in the visual representation limited. There are two key methods to reduce the data: Filtering and aggregation, as described by Munzner et al. [Mun09b].

Filtering implies taking a subset of the data, for example one that corresponds to the current selection criteria, as demonstrated by the fiber selection in Chapter 2 and by the volume selection in Chapter 6.

Aggregation is an orthogonal approach, where items are joined together in groups, which in turn can be represented by a single item each. In Chapter 5, we briefly touched on the subject of aggregation by taking average voxel values over larger volumes, and using that value as a feature over time.

The combination of these two techniques, combined with pattern recognition and feature selection, is very promising, and leads to many interesting research opportunities.

8.3 Future Work

This work forms a basis for the human-in-the-loop exploratory analysis of data at different abstraction levels simultaneously. Because of the huge variations across abstraction levels in both the amount as well as the nature of the data visualized, a mix of visual representations is required. The representations used at the lowest levels of abstraction traditionally belong to scientific visualization, where data is spatial in nature and often represented as dense fields. At higher abstraction levels, such as feature spaces, the spatial nature of the data may be discarded, and techniques from information visualization, such as scatter plots and parallel coordinate plots, may be more applicable. At even higher abstraction levels, the number of entities to visualize becomes even lower due to aggregation, but the nature of the entities becomes more complex, so that iconic representations gain importance.

Finding the right mix between these techniques, and adapting techniques from one field to work on the other, is one of the greatest challenges that lie ahead. Possible extension areas of this research are discussed in the following sections.

Exploration through linked domains

When visualizing data using different representations at the same time, each particular view of the data has its own merits. A lower-dimensional projection of feature space data can be seen as such a parameterized view. The parameterization in this case corresponds to the chosen projection, and the effectiveness of the visualization heavily relies on choosing the right parameters.

Currently, we often select these parameterizations by a pre-defined process (using principal component analysis for example), or just leave the selection completely to the user (presenting a scatter plot matrix, or using scatter plot dice). We strongly feel that these parameters should be an important part of the exploration process.

In Chapter 4, we showed that statistical methods can be used to generate projections from selections. Supplementing this with brushing methods, to define selections within a given projection, these provide the user with powerful tools for exploring multi-field data.

The goal is not only to give the user insight into the data, but also to give the computer insight into what the user wants. Often, the user performs a repetitive task during exploration, such as looking for a specific pattern in a scatter plot. If the computer can learn, from a few examples, which pattern we are looking for, it can assist in the exploration by pre-selecting the data shown. In recent work, Ma [Ma07] has applied machine learning techniques to volume visualization. His work sets out how techniques from pattern recognition form a key component of intelligent interfaces for data visualization. While Ma's work focusses on single-volume datasets, we foresee that these techniques can be extended to fulfill an important role in the exploration of multi-field high-dimensional data within a multiple coupled view framework.

Image processing

The use of image processing filters is a powerful tool to introduce derived attributes. We have demonstrated this using LH-spaces (see 4.5.1), and also in a more structural way using generic image processing operations (see 5.3.2). Looking further in this direction of extending the input data, there are a number of continuous-domain extensions such as scale-space image processing, for which few visualization and pattern discovery techniques are known. It would be quite beneficial to further study how these higher-dimensional data can be explored through multiple linked views.

Distance functions

One domain we would also like to explore further are distance functions. Most of the techniques that we are currently using aim at reducing high dimensional data to a lower dimensionality, at which common visualization techniques can be more easily applied. However, this type of projection from a high to a low dimensional space imposes limitations on the distance metric used.

It makes sense to explore the distance functions defined on high dimensional spaces for two reasons. Firstly, several projection techniques attempt to find a projection that conserves the distances locally between neighbors, yet the distance metric that is used to determine these neighbors is often difficult to visualize. Secondly, clustering techniques can often function directly on the higher dimensional space once a good high-dimensional distance metric has been defined, providing a labelling of samples without the need for a lower dimensional representation.

In these spaces, many different distance metrics have been suggested, and often the one that should be used is highly task-specific. Keeping this in mind, the exploration and visualization of distance functions could probably become a major component of a multiple linked view for images containing high dimensional data. Specifically, a method such as that presented by Paulovich et al. could help project the data to a sufficiently low dimension to make interactive exploration effective [VPNM06].

Large medical cohort studies

Future work would be to further test the techniques presented in this thesis on a larger scale. We aim to analyze the data from a large medical study, where both image processing as well as pattern discovery will likely form a major part. By using aggregations and hierarchical linked domains, it will be possible to explore the data in a much more coherent fashion.

As an example, consider the analysis of the growth of tumor over time. After registration of source MRI images, we could brush a selection inside and outside the tumor to train a classifier. If the classification is not satisfactory, we can either change the selections interactively, or add more derived attributes. When a good classifier has been found, we can directly apply it to all other scans and show the results in a comparative fashion. At this point, we can assess the necessity of the used MRI imaging sequences by looking at the feature space distribution of voxels classified as tumorous.

Once we have the classified voxels, surfaces can be extracted that delineate the tumor. By assigning attributes to the surface, such as shape descriptors, we can quickly examine the spread and distribution of the detected tumors by opening a feature space view of the shape attributes. The Linkage between all these views makes is possible to pick an outlier in the shape space and go directly to the corresponding dataset and voxel selection, which makes understanding the processing pipeline and its results far more intuitive.

Bibliography

[AdOL04] A.O. Artero, M.C.F. de Oliveira, and H. Levkowitz, *Uncovering clusters in crowded parallel coordinates visualizations*, INFOVIS '04: Proceedings of the IEEE Symposium on Information Visualization (2004), 81–88.

[AM07] Hiroshi Akiba and Kwan-Liu Ma, *Eurovis07: Joint eurographics - ieee vgtc symposium on visualization, norrköping, sweden, 23-25 may 2007*, EuroVis (Ken Museth, Torsten Möller, and Anders Ynnerman, eds.), Eurographics Association, 2007, pp. 115–122.

[Asi85] Daniel Asimov, *The grand tour: a tool for viewing multidimensional data*, SIAM J. Sci. Stat. Comput. **6** (1985), no. 1, 128–143.

[ASM$^+$04] David Akers, Anthony Sherbondy, Rachel Mackenzie, Robert Dougherty, and Brian Wandell, *Exploration of the Brain's White Matter Pathways with Dynamic Queries*, Proceedings of IEEE Visualization 2004 (Holly Rushmeier, Greg Turk, and Jarke J. van Wijk, eds.), 2004, pp. 377–384.

[BA86] Andreas Buja and Daniel Asimov, *Grand tour methods: an outline*, Proc. 17th Symp. on the Interface of Computer Sciences and Statistics, Elsevier North-Holland, Inc., 1986, pp. 63–67.

[BBP$^+$05] Jorik Blaas, Charl P. Botha, Bart Peters, Frans M. Vos, and Frits H. Post, *Fast and reproducible fiber bundle selection in DTI visualization*, Proceedings of IEEE Visualization 2005 (Claudio Silva, Eduard Gröller, and Holly Rushmeier, eds.), October 2005, pp. 59–64.

[BBP07] Jorik Blaas, Charl P. Botha, and Frits H. Post, *Interactive visualization of multi-field medical data using linked physical and feature-space views*, Proc.

Eurographics / IEEE-VGTC EuroVis (Ken Museth, Anders Ynnerman, and Torsten Möller, eds.), 2007, pp. 123–130.

[BH07] Raphael Bürger and Helwig Hauser, *Visualization of multi-variate scientific data*, EuroGraphics 2007 State of the Art Reports (STARs), 2007, pp. 117–134.

[BKP^{+}04] Anders Brun, Hans Knutsson, Hae-Jeong Park, Martha E. Shenton, and Carl-Fredrik Westin, *Clustering Fiber Traces Using Normalized Cuts*, MICCAI 2004 (C. Barillot, D.R. Haynor, and P. Hellier, eds.), vol. LNCS 3216, Springer-Verlag, 2004, pp. 368–375.

[BPP^{+}00] P.J. Basser, S. Pajevic, C. Pierpaoli, J. Duda, and A. Aldroubi, *In vivo fiber tractography using DT-MRI data*, Magnetic Resonance Imaging **44** (2000), no. 4, 625–632.

[CBT06] *Primary brain tumors in the united states - statistical report 1998-2002*, Central Brain Tumor Registry of the United States (2005-2006), 10.

[Cle93] W. S. Cleveland, *Visualizing data*, Hobart Press, Summit, New Jersey, U.S.A., 1993.

[CMR07] M. Caat, N.M. Maurits, and J.B.T.M. Roerdink, *Design and evaluation of tiled parallel coordinate visualization of multichannel eeg data*, Visualization and Computer Graphics, IEEE Transactions on **13** (2007), no. 1, 70–79.

[CZQ^{+}08] Weiwei Cui, Hong Zhou, Huamin Qu, Pak Chung Wong, and Xiaoming Li, *Geometry-based edge clustering for graph visualization*, IEEE Transactions on Visualization and Computer Graphics **14** (2008), no. 6, 1277–1284.

[DGH03] Helmut Doleisch, Martin Gasser, and Helwig Hauser, *Interactive feature specification for focus+context visualization of complex simulation data*, VISSYM '03: Proceedings of the symposium on Data visualisation 2003, Eurographics Association, 2003, pp. 239–248.

[DKS05] Roman Dementiev, Lutz Kettner, and Peter Sanders, *STXXL: Standard template library for XXL data sets*, Algorithms - ESA 2005 : 13th Annual European Symposium (ESA 2005) (Palma de Mallorca, Spain) (Gerth StÃÿlting Brodal and Stefano Leonardi, eds.), Lecture Notes in Computer Science, vol. 3669, European Association for Theoretical Computer Science (EATCS), Springer, October 2005, pp. 640–651.

[EGM07] David Eppstein, Michael T. Goodrich, and Jeremy Yu Meng, *Confluent layered drawings*, Algorithmica **47** (2007), no. 4, 439–452.

[FHH93] David J. Field, Anthony Hayes, and Robert F. Hess, *Contour integration by the human visual system: Evidence for a local "association field"*, Vision Research **33** (1993), no. 2, 173–193.

[FMHC07] Zhe Fang, Torsten Möller, Ghassan Hamarneh, and Anna Celler, *Visualization and exploration of time-varying medical image data sets*, GI '07: Proceedings of Graphics Interface 2007 (New York, NY, USA), ACM, 2007, pp. 281–288.

[FR91] Thomas M. J. Fruchterman and Edward M. Reingold, *Graph drawing by force-directed placement*, Softw. Pract. Exper. **21** (1991), no. 11, 1129–1164.

[FWR99] Ying-Huey Fua, Matthew O. Ward, and Elke A. Rundensteiner, *Hierarchical parallel coordinates for exploration of large datasets*, Proc. IEEE Visualization (Los Alamitos, CA, USA), IEEE Computer Society Press, 1999, pp. 43–50.

[GJL$^+$09] Ed Grundy, Mark W. Jones, Robert S. Laramee, Rory P. Wilson, and Emily F. Shepard, *Visualization of sensor data from animal movement*, Eurographics/ IEEE-VGTC Symposium on Visualization (Eurovis) 2009, Computer Graphics Forum **28** (2009), no. 2, 815–822.

[GK03] M. Graham and J. Kennedy, *Using curves to enhance parallel coordinate visualisations*, Information Visualization, 2003. IV 2003. Proceedings. Seventh International Conference on, July 2003, pp. 10–16.

[GPS$^+$03] M. Guye, G.J. Parker, M. Symms, P. Boulby, C. A. Wheeler-Kingshott, A. Salek-Haddadi, G. J. Barker, and J. S. Duncan., *Combined functional MRI and tractography to demonstrate the connectivity of the human primary motor cortex in vivo.*, NeuroImage, 2003, pp. 1349–1360.

[GRW$^+$00] D.L. Gresh, B.E. Rogowitz, R.L. Winslow, D.F. Scollan, and C.K. Yung, *Weave: a system for visually linking 3-d and statistical visualizations applied to cardiac simulation and measurement data*, Proceedings of IEEE Visualization 2000, 2000, pp. 489–492, 597.

[HMM00] I. Herman, G. Melancon, and M.S. Marshall, *Graph visualization and navigation in information visualization: A survey*, IEEE Transactions on Visualization and Computer Graphics **6** (2000), no. 1, 24–43.

[Hol06] D. Holten, *Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data*, IEEE Transactions on Visualization and Computer Graphics **12** (2006), no. 5, 741–748.

[HPS+03] T. Hendler, P. Pianka, M. Sigal, M. Kafri, D. Ben-Bashat, S. Constantinu, M. Grait, I. Fried, and Y. Assat, *"two are better than one": Combining fMRI and DTI based fiber tracking for effective pre-surgical mapping*, Proc. Intl. Soc. Mag. Reson. Med, 2003, p. 394.

[Hv09] Danny Holten and Jarke J. van Wijk, *A user study on visualizing directed edges in graphs*, CHI '09: Proceedings of the 27th international conference on Human factors in computing systems (New York, NY, USA), ACM, 2009, pp. 2299–2308.

[Ins85] Alfred Inselberg, *The plane with parallel coordinates*, The Visual Computer **1** (1985), no. 4, 69–91.

[ISNC05] L. Ibanez, W. Schroeder, L. Ng, and J. Cates, *The ITK Software Guide*, Kitware, Inc. ISBN 1-930934-15-7, second ed., 2005.

[JJ95] Thierry Jéron and Claude Jard, *3d layout of reachability graphs of communicating processes*, GD '94: Proceedings of the DIMACS International Workshop on Graph Drawing (London, UK), Springer-Verlag, 1995, pp. 25–32.

[JLC07] Jimmy Johansson, Patric Ljung, and Matthew Cooper, *Eurovis07: Joint eurographics - ieee vgtc symposium on visualization, norrköping, sweden, 23-25 may 2007*, EuroVis (Ken Museth, Torsten Möller, and Anders Ynnerman, eds.), Eurographics Association, 2007, pp. 35–42.

[JLJC05] Jimmy Johansson, Patric Ljung, Mikael Jern, and Matthew Cooper, *Revealing structure within clustered parallel coordinates displays*, INFOVIS '05: Proceedings of the IEEE Symposium on Information Visualization, 2005, pp. 125–132.

[JMM+05] C. Johnson, R. Moorhead, T. Munzner, H. Pfister, P. Rheingans, and T. Yoo, *Nih-nsf visualization research challenges report*, IEEE Press, 2005.

[JS98] D.F. Jerding and J.T. Stasko, *The information mural: a technique for displaying and navigating large information spaces*, Visualization and Computer Graphics, IEEE Transactions on **4** (1998), no. 3, 257–271.

[JTJ04] J. Johansson, R. Treloar, and M. Jern, *Integration of unsupervised clustering, interaction and parallel coordinates for the exploration of large multivariate data*, INFOVIS '04: Proceedings of the IEEE Symposium on Information Visualization (2004), 52–57.

[Kin04] Gordon Kindlmann, *Superquadric Tensor Glyphs*, Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization, 2004, pp. 147–154.

[Kin10] Gordon Kindlmann, *The TEEM toolkit*, May 2010, http://teem.sf.net/.

[KWH00] Gordon Kindlmann, David Weinstein, and David Hart, *Strategies for Direct Volume Rendering of Diffusion Tensor Fields*, IEEE Transactions on Visualisation and Computer Graphics (2000), no. 6(2), 124–138.

[LCG+02] John Peter Lee, Daniel Carr, Georges Grinstein, John Kinney, and Jeffrey Saffer, *The next frontier for bio- and cheminformatics visualization*, IEEE Computer Graphics and Applications **22** (2002), 6–11.

[LHG+06] David Luebke, Mark Harris, Naga Govindaraju, Aaron Lefohn, Mike Houston, John Owens, Mark Segal, Matthew Papakipos, and Ian Buck, *Gpgpu: general-purpose computation on graphics hardware*, SC '06: Proceedings of the 2006 ACM/IEEE conference on Supercomputing (New York, NY, USA), ACM, 2006, pp. 208+.

[LT05] M. Leuschel and E. Turner, *Visualizing larger state spaces in ProB*, Proceedings of the International Conference of B and Z Users, 2005.

[Ma07] Kwan-Liu Ma, *Machine learning to boost the next generation of visualization technology*, IEEE Computer Graphics and Applications **27** (2007), 6–9.

[MCCvZ99] Susumu Mori, B. J. Crain, V. P. Chacko, and Peter C. M. van Zijl, *Three-dimensional tracking of axonal projections in the brain by magnetic resonance imaging*, Ann Neurol., 1999, pp. 265–269.

[MKO+08] Philipp Muigg, Johannes Kehrer, Steffen Oeltze, Harald Piringer, Helmut Doleisch, Bernhard Preim, and Helwig Hauser, *A four-level focus+context approach to interactive visual analysis of temporal features in large scientific data*, vol. 27, 2008, pp. 775–782.

[Mun09a] Tamara Munzner, *A nested process model for visualization design and validation*, IEEE Trans. Vis. Comput. Graph. **15** (2009), no. 6, 921–928.

[Mun09b] Tamara Munzner, *Visualization*, third ed., ch. 27, pp. 675–707, AK Peters, 2009.

[MW91] John J. Miller and Edward J. Wegman, *Construction of line densities for parallel coordinate plots*, pp. 107–123, Springer-Verlag New York, Inc., New York, NY, USA, 1991.

[NH06] M. Novotný and H. Hauser, *Outlier-preserving focus+context visualization in parallel coordinates*, IEEE Trans. on Visualization and Computer Graphics **12** (2006), no. 5, 893–900.

[Obe05] M.F.X.J. Oberhumer, *Lzo real-time data compression library*, 2005, http://www.oberhumer.com/opensource/lzo/.

[ODH$^+$07] Steffen Oeltze, Helmut Doleisch, Helwig Hauser, Philipp Muigg, and Bernhard Preim, *Interactive visual analysis of perfusion data*, IEEE Trans. Vis. Comput. Graph. **13** (2007), no. 6, 1392–1399.

[Ooi87] Beng Chin Ooi, *Spatial kd-Tree: A Data Structure for Geographic Database.*, Büro, Technik und Wissenschaft, 1987, pp. 247–258.

[Pre08] A.J. Pretorius, *Visualization of state transition graphs*, Ph.D. thesis, Eindhoven University of Technology, 2008.

[Pv06] A. Johannes Pretorius and Jarke J. van Wijk, *Visual analysis of multivariate state transition graphs*, IEEE Transactions on Visualization and Computer Graphics **12** (2006), no. 5, 685–692.

[Pv08] ______, *Visual inspection of multivariate graphs*, Computer Graphics Forum **27** (2008), no. 3, 967–974.

[Rob07] Jonathan C. Roberts, *State of the art: Coordinated & multiple views in exploratory visualization*, CMV '07: Proceedings of the Fifth International Conference on Coordinated and Multiple Views in Exploratory Visualization (Washington, DC, USA), IEEE Computer Society, 2007, pp. 61–71.

[SCB98] Deborah F. Swayne, Dianne Cook, and Andreas Buja, *XGobi: Interactive dynamic data visualization in the x window system*, J. Comp. Graph. Stat. **7** (1998), no. 1, 113–130.

[SJW$^+$04] S.M. Smith, M. Jenkinson, M.W. Woolrich, C.F. Beckmann, T.E.J. Behrens, H. Johansen-Berg, P.R. Bannister, M. De Luca, I. Drobnjak, D.E. Flitney, R. Niazy, J. Saunders, J. Vickers, Y. Zhang, N. De Stefano, J.M. Brady, and P.M. Matthews, *Advances in functional and structural mr image analysis and implementation as fsl.*, 208–219.

[SLBC03] Deborah F. Swayne, Duncan Temple Lang, Andreas Buja, and Dianne Cook, *GGobi: evolving from XGobi into an extensible framework for interactive data visualization*, Comput. Stat. Data Anal. **43** (2003), no. 4, 423–444.

[SRL$^+$00] Peter Sutherland, Anthony Rossini, Thomas Lumley, Nicholas Lewin-Koh, Julie Dickerson, Zach Cox, and Dianne Cook, *Orca: A visualization toolkit for high-dimensional data*, J. Comp. Graph. Stat. **9** (2000), no. 3, 509–529.

[SVSG06] Petr Sereda, Anna Vilanova, Iwo W. O. Serlie, and Frans A. Gerritsen, *Visualization of boundaries in volumetric data sets using lh histograms*, IEEE Trans. on Visualization and Computer Graphics **12** (2006), no. 2, 208–218.

[The00] H. Theisel, *Higher order parallel coordinates*, Proc. Vision, Modeling and Visualization 2000 (Saarbrücken) (B. Girod, G. Greiner, H. Niemann, and H.P. Seidel, eds.), 2000, pp. 119–125.

[TMA01] Melanie Tory, Torsten Möller, and M. Stella Atkins, *Visualization of time-varying MRI data for MS lesion analysis*, Proc. SPIE Medical Imaging 2001: Visualization, Display, and Image-Guided Procedures, vol. 4319, 2001, pp. 590–598.

[TOW$^+$03] I. F. Talos, L. O'Donnell, C.-F. Westin, S. K. Warfield, W. M. Wells, S. S. Yoo, L. Panych, A. Golby, H. Mamata, S. E. Maier, P. Ratiu, C. G. Guttmann, P. Mc. L. Black, F. A. Jolesz, and R. Kikinis, *Diffusion tensor and functional MRI fusion with anatomical MRI for image guided neurosurgery*, Sixth International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI'03) (Montreal, Canada), November 2003, pp. 407–415.

[vHR08] F. van Ham and B. Rogowitz, *Perceptual organization in user-generated graph layouts*, IEEE Transactions on Visualization and Computer Graphics **14** (2008), no. 6, 1333–1339.

[VPNM06] Fernando Vieira Paulovich, Luis Gustavo Nonato, and Rosane Minghim, *Visual mapping of text collections through a fast high precision projection technique*, Proceedings of the conference on Information Visualization (Washington, DC, USA), IEEE Computer Society, 2006, pp. 282–290.

[vvv01] Frank van Ham, Huub van de Wetering, and Jarke J. van Wijk, *Visualization of state transition graphs*, Information Visualization, 2001. INFOVIS 2001. IEEE Symposium on, 2001, pp. 59–66.

[vvv02] ———, *Interactive visualization of state transition systems*, IEEE Transactions on Visualization and Computer Graphics **8** (2002), no. 4, 319–329.

[WAPW06] C. Ware, R. Arsenault, M. Plumlee, and D. Wiley, *Visualizing the underwater behavior of humpback whales*, IEEE Computer Graphics and Applications **26** (2006), no. 4, 14–18.

[War94] Matthew O. Ward, *Xmdvtool: integrating multiple methods for visualizing multivariate data*, VIS '94: Proceedings of the conference on Visualization '94, IEEE Computer Society Press, 1994, pp. 326–333.

[War04] Colin Ware, *Information visualization: Perception for design (morgan kaufmann interactive technologies series)*, 1st ed., Morgan Kaufmann Publishers, 2004.

[WB97] Pak Chung Wong and R. Daniel Bergeron, *30 years of multidimensional multivariate visualization*, Scientific Visualization, Overviews, Methodologies, and Techniques (Washington, DC, USA), IEEE Computer Society, 1997, pp. 3–33.

[WB04] Colin Ware and Robert Bobrow, *Motion to support rapid interactive queries on node–link diagrams*, ACM Transactions on Applied Perception **1** (2004), no. 1, 3–18.

[WGP96] Rainer Wegenkittl, Meister Eduard Gröller, and Werner Purgathofer, *A guided tour to wonderland: Visualizing the slow-fast dynamics of an analytical dynamical system*, Tech. Report TR-186-2-96-11, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Favoritenstrasse 9-11/186, A-1040 Vienna, Austria, April 1996, human contact: technical-report@cg.tuwien.ac.at.

[WKL99] David Weinstein, Gordon Kindlmann, and Eric Lundberg, *Tensorlines: Advection-Diffusion based Propagation through Diffusion Tensor Fields*, Visualization '99, proceedings, 1999, pp. 249–530.

[WMB$^+$99] C.-F. Westin, S.E. Maier, B.Khidhir, P. Everett, F. A. Jolesz, and R. Kikinis, *Image processing for diffusion tensor magnetic resonance imaging*, Proceedings of Second Int. Conf. on Medical Image Computing and Computer-assisted Interventions (MICCAI'99), 1999, pp. 441–452.

[WMM$^+$02] C.-F. Westin, S. E. Maier, H. Mamata, A. Nabavi, F. A. Jolesz, and R. Kikinis, *Processing and visualisation for diffusion tensor MRI*, Medical Image Analysis **6** (2002), 93–108.

[WN08] D. Whalen and M.L. Norman, *Competition data set and description*, 2008 IEEE Visualization Design Contest, 2008, http://vis.computer.org/VisWeek2008/vis/contests.html.

[ZL05] Song Zhang and David H. Laidlaw, *DTI Fiber Clustering and Cross-subject Cluster Analysis*, Proceedings of ISMRM (Miami, FL), May 2005.

List of Figures

List of Tables

Summary

This thesis investigates methods for the visualization of multi-field medical data. In the medical field, data complexity has been growing consistently over the past years. Not only the size of the data grows, but also the need to visualize beyond traditional boundaries.

We present a number of novel facets that encompass a general approach to the exploration of multi-field data. We strongly believe that human-in-the-loop visual data analysis on large and complex datasets is best aided by multiple linked different representations.

The presented techniques demonstrate how complex data from multiple modalities can be visualized and interactively explored. We explore the use of linked selections to aid in reducing the complexity of the visualizations. Using multiple-linked views, we can integrate multiple orthogonal representations of the data simultaneously.

We have applied aforementioned techniques in the design and implementation of a number of prototype frameworks, with applications ranging from brain imaging for neurosurgical planning to the study of the behavior of marine animals through the use of sensor data.

We also present a conceptual framework for studying complex longitudinal data, by means of aggregation and multi-level visualization. We successfully adapted techniques from information visualization in order to use them on datasets that are orders of magnitude larger than they are originally used for.

Samenvatting

Dit proefschrift beschrijft het onderzoek naar methoden voor het visualiseren van multi-field medische datasets. Medische datasets zijn steeds groter en complexer geworden over de afgelopen jaren. Omdat complexiteit van de data steeds groter wordt, loopt men veelal tegen de grenzen aan van de huidige visualisatietechnieken.

Wij beschrijven een aantal nieuwe componenten die een generieke aanpak vormen voor de exploratie van multi-field datasets. We zijn van mening dat voor de visuele analyse van grote en complexe datasets het best gebruik gemaakt kan worden van verschillende gekoppelde visuele representaties.

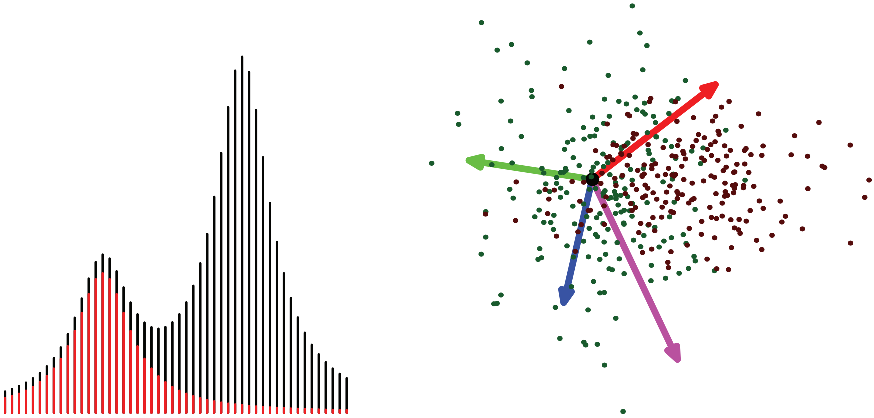
De hierin beschreven technieken laten zien hoe complexe data van verschillende moditeiten gevisualiseerd en interactief exploratief bekeken kan worden. Wij hebben gekozen voor gekoppelde selecties tussen meerdere representaties om de visuele complexiteit laag te houden. Door gebruik te maken van deze koppeling kunnen verscheidene orthogonale representaties geïntegreerd worden tot een effectief werkend systeem.

De bovengenoemde technieken hebben wij toegepast op het ontwerpen en implementeren van een aantal prototype systemen. De toepassingen van deze systemen lopen uiteen van neurochirurgische planning, tot aan het bestuderen van het leefgedrag van penguins.

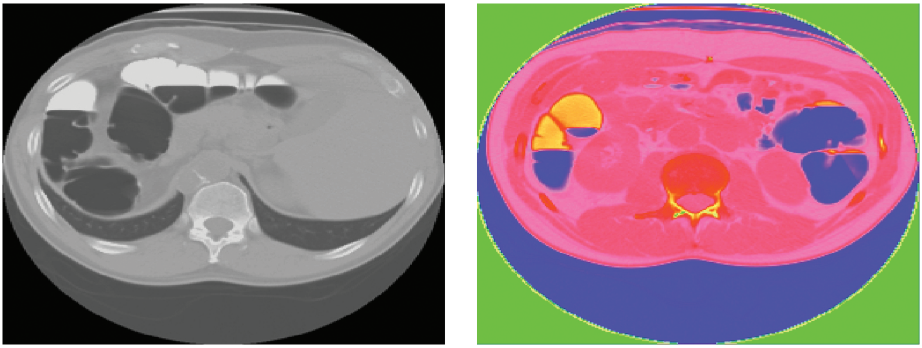
We hebben ook een conceptueel framework ontworpen, met het oog op de analyse van longitudinale medische studies, waarbinnen aggregatie en multi-level visualisatie kernelementen zijn. In het kader hiervan hebben wij een aantal technieken uit de informatie visualisatie aangepast zodat ze bruikbaar zijn op datasets die orde groottes meer elementen bevatten dan waarvoor zij van oudsher gebruikt worden.

Color Plates

A



Histogram Scatter plot

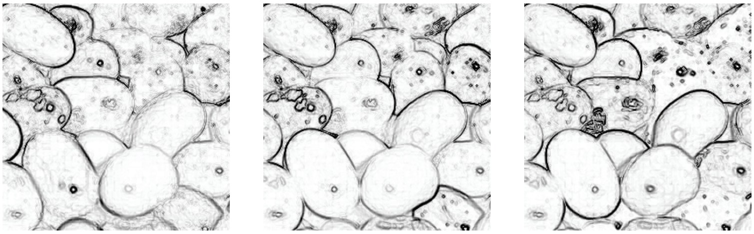


Slice-Viewer Color Slice-Viewer

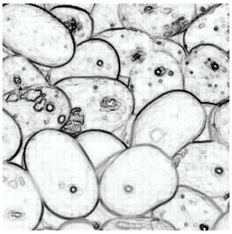
Figure A.1: Analysis of CT colonoscopy data using a multiple linked view system (see Chapter 4). The four components used are (from top-left to bottom-right): histogram, scatter plot, slice-viewer and pseudo-color slice-viewer.

(a) Color image

(b) Red channel (c) Green channel (d) Blue channel



(e) Red edges (f) Green edges (g) Blue edges



(h) Combined edges

Figure A.2: Edge detection on a color photograph (a) of jellybeans. When visualizing each channel (red, green and blue) independently (b,c,d), some of the beans have the exact same intensity. Looking at edges detected in the three images (e,f,g), it is clear that some of the edges are only present in a single color channel. When all color channels are used to compose an edge image (h), all beans are clearly delineated, even though we are visualizing the edge strength as a single scalar channel.

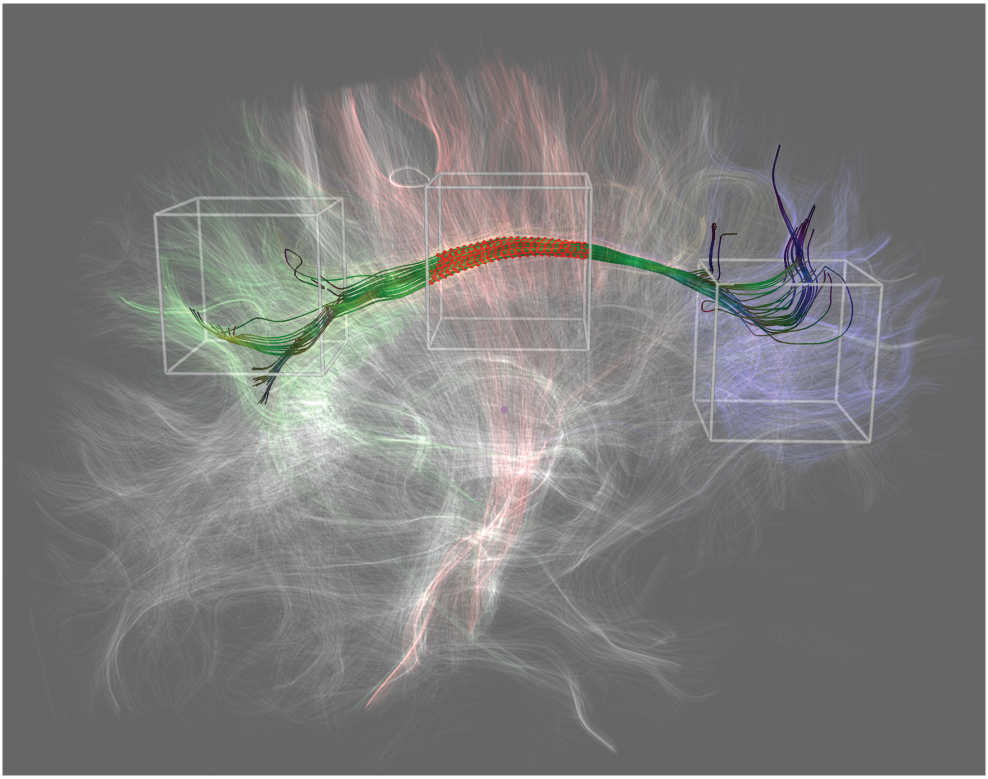


Figure A.3: The three-box selection method in progress. A pre-selection is made of all fibers that intersect all three boxes. Out of this pre-selection, only fiber sections that are contained within the center box are selected for further processing.

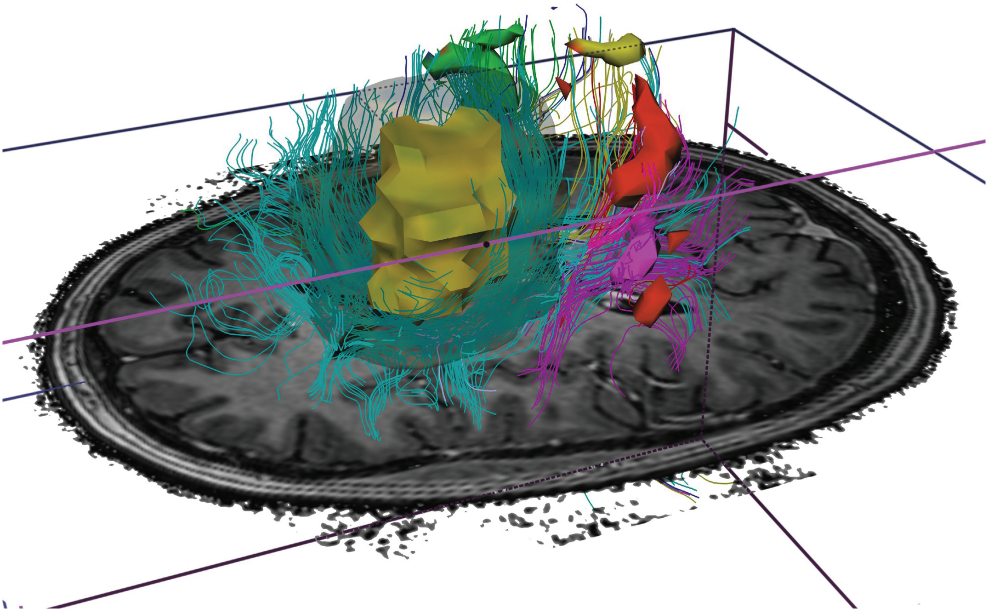


Figure A.4: Combined visualization of a brain tumor, the white matter tractography and the functional areas associated with motor tasks. The tumor is surrounded by a safety area, which is used to filter and color the fibers in the tractography. Both the tumor and the activation areas directly influence the color of the fibers.

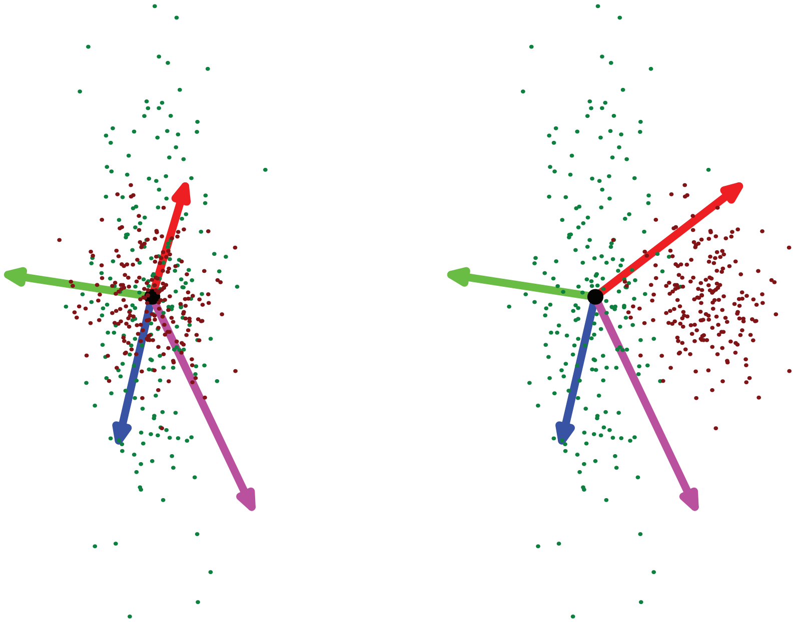


Figure A.5: Two clusters of points separated by a single feature. The separation can easily be found by moving the axis (red) corresponding to the separating feature. After moving the axis (right), the separating clusters can be distinguished.

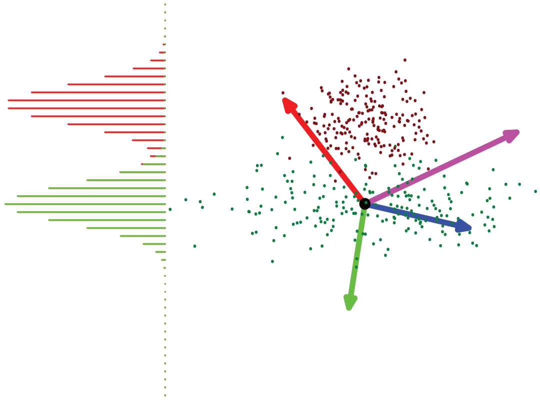


Figure A.6: A histogram linked to a scatter plot. The histogram uses the Y coordinates of the projected points in the scatter plot as a source.

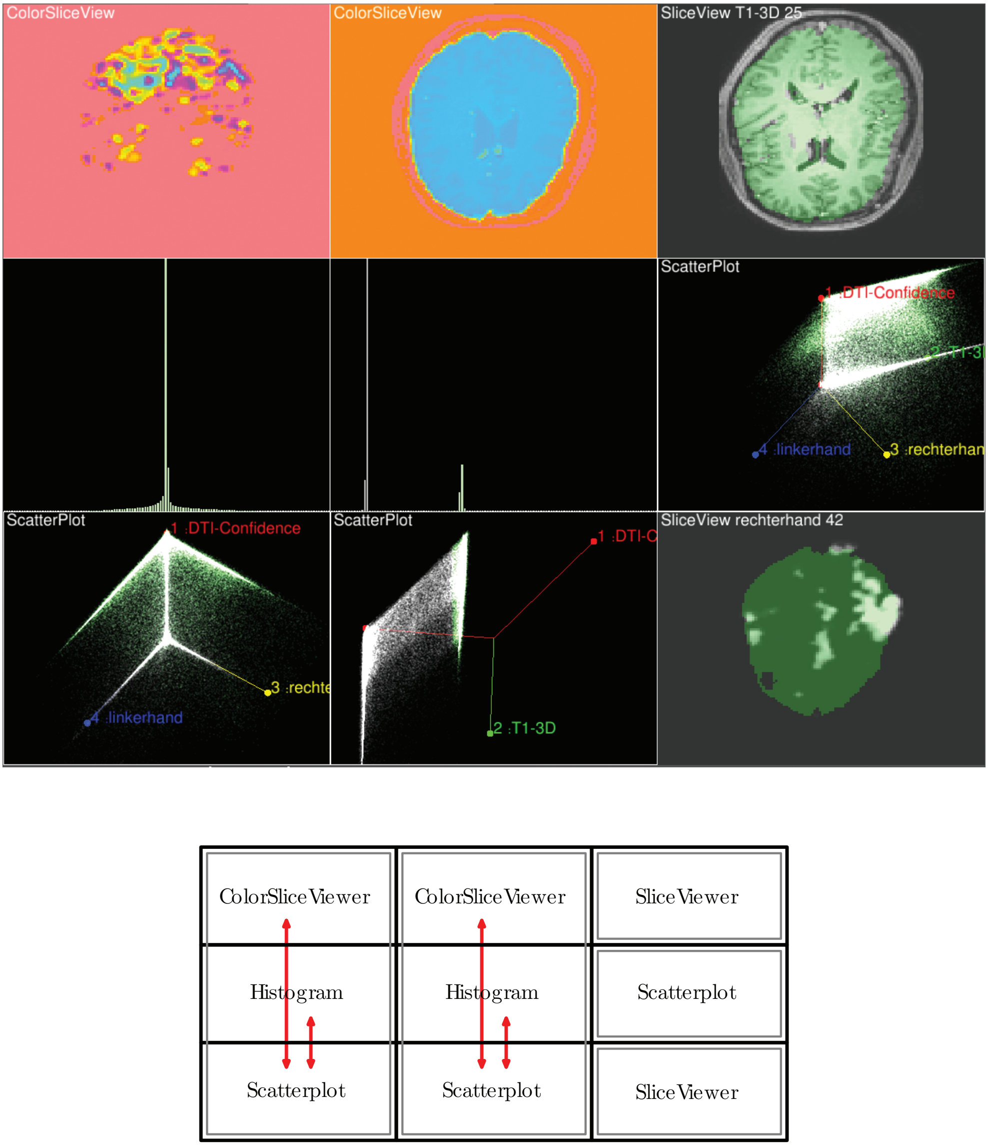
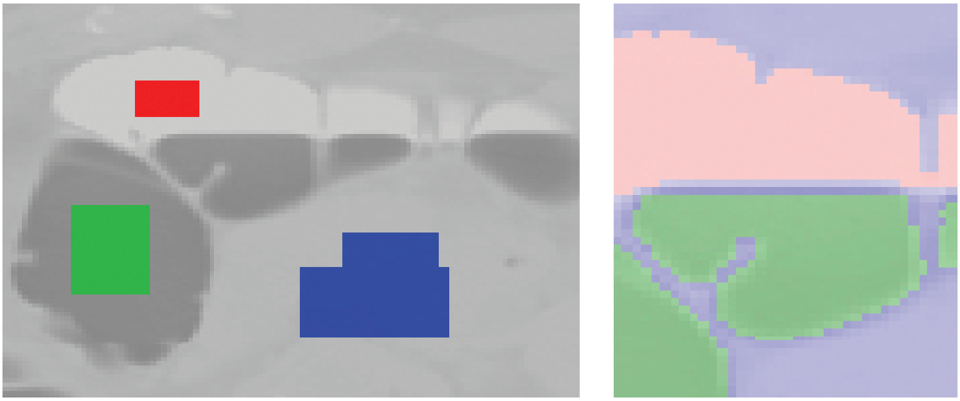


Figure A.7: Top: The interface during a user session. Bottom: The component grid, where red arrows indicate the coupling between components.



(a) Initial Selection (b) Expanded Selection

Figure A.8: (a) shows the initial selection created through brushing in the slice-viewer. Red represents contrast liquid, green represents gas and blue marks tissue. (b) shows a detail of the expanded selection through application of the kNN classification process, note that the boundary is not correctly classified.

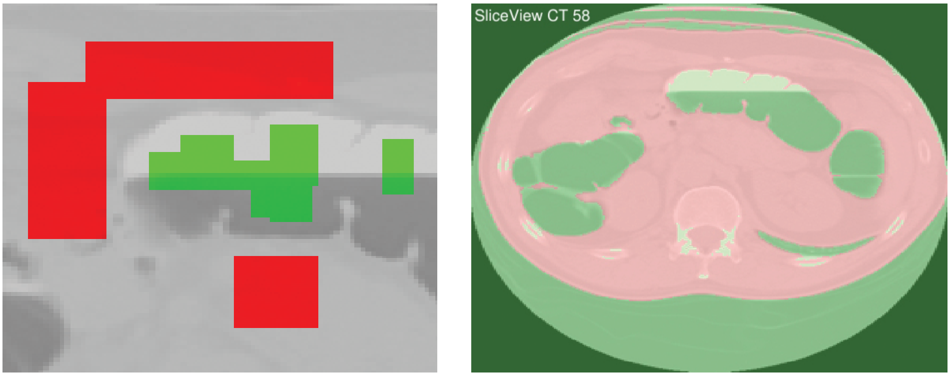


Figure A.9: Expanded selection through application of the kNN classification process on the feature space that was expanded with the derived LH feature.

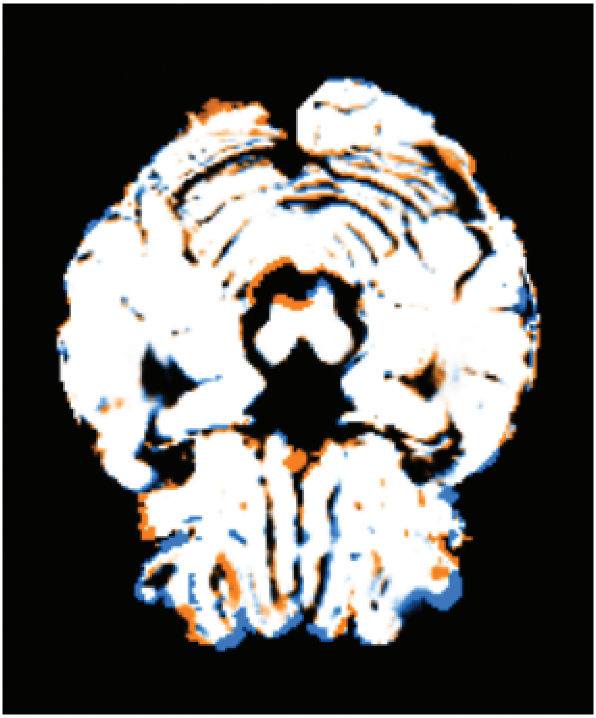


Figure A.10: The slice viewer shows a blue-orange image fusion of two features. One feature represents the base-line image (blue) while the other feature corresponds to an image taken six months later (orange). Areas where only orange or blue is seen correspond to changes, either due to changes in the brain, or for example due to misalignment during registration.

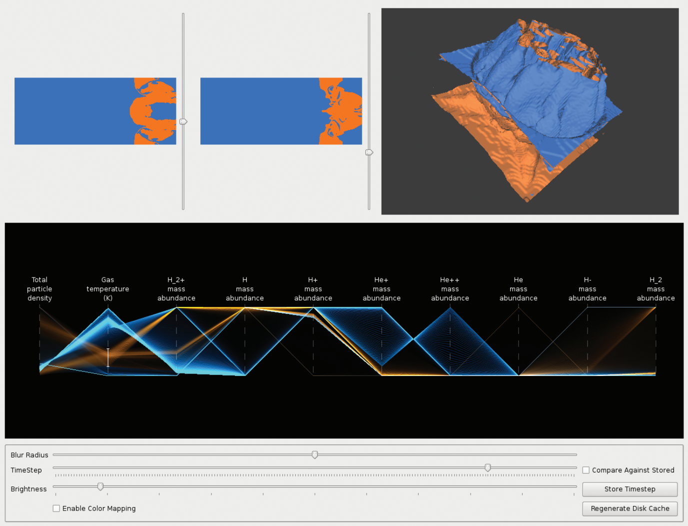


Figure A.11: The main user interface of the demonstrator application. The top part of the screen contains spatial viewing components (two slice viewers and a 3D isosurface view). The parallel coordinate plot is positioned in the middle, and the control interface is positioned at the bottom.

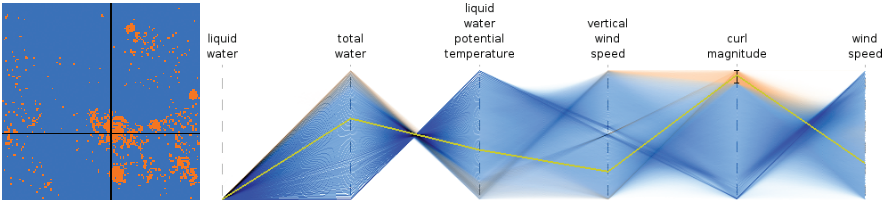
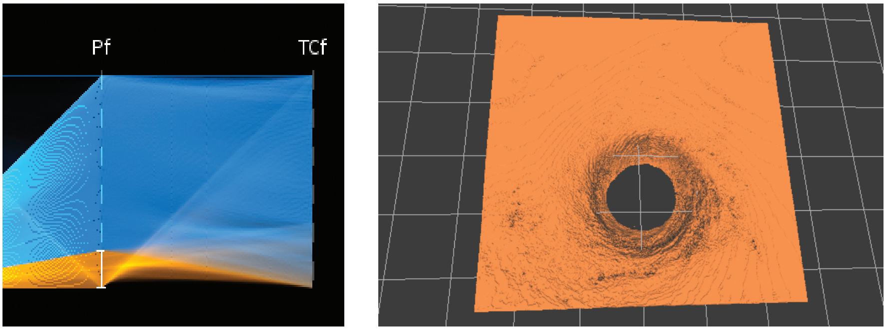


Figure A.12: A slice viewer in which a point is selected (left) linked to a parallel coordinate plot (right) which displays the selected value as a bright yellow line.

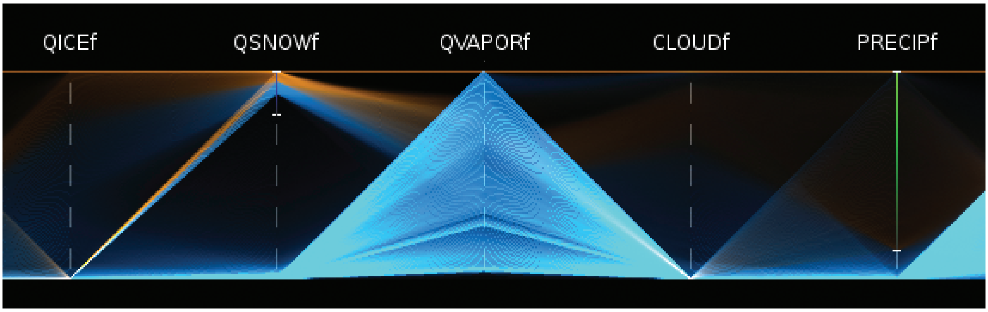
Figure A.13: Defining a colormap using two selections. Two ranges are selected in the PCP on the left, which correspond to two separate color maps in the slice view on the right. Bright blue pixels correspond to a high gas temperature, and bright green pixels to a high H_2^+ mass abundance.



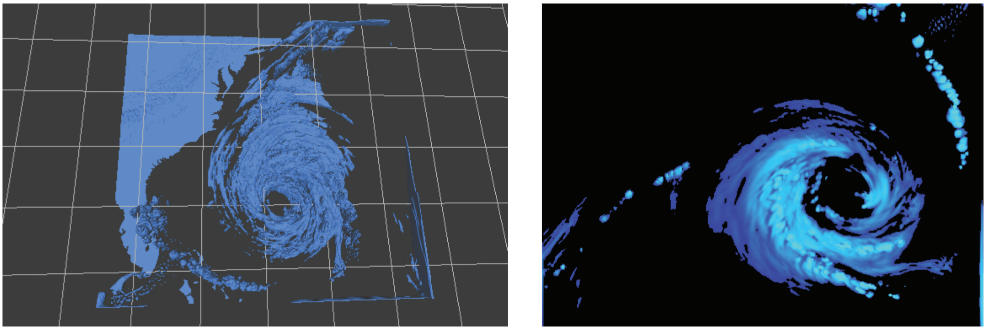
(a) PCP selection

(b) 3D view

Figure A.14: Selection of low pressure areas (shown in orange) reveals the area of low-temperature near the eye of the hurricane. The compact horizontal shape of the orange band in the PCP reveals that low-pressure areas mostly have a low-temperature as well.



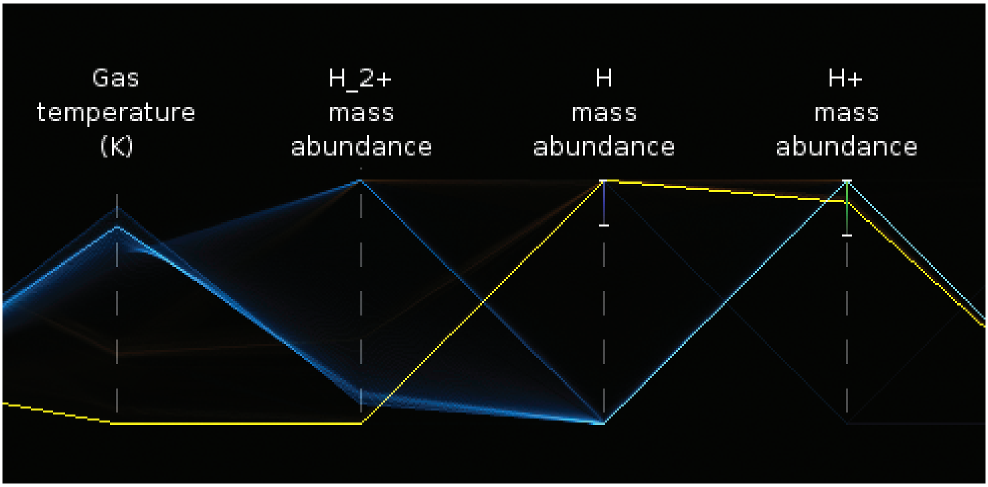
(a) PCP selection



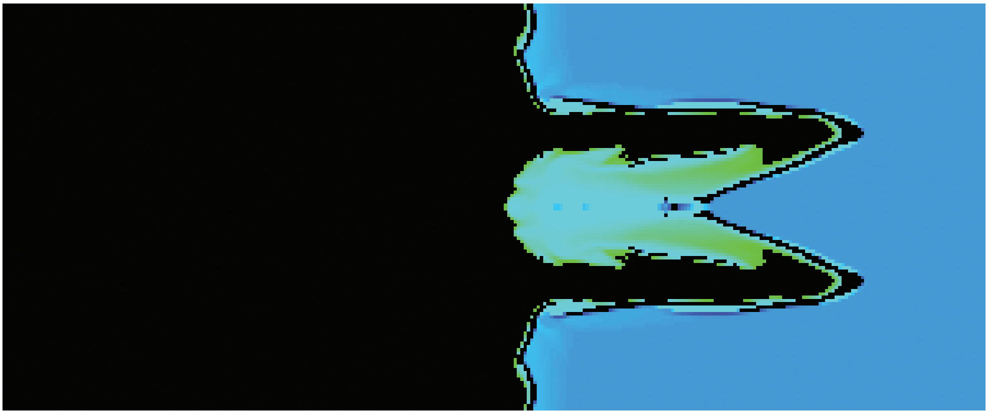
(b) 3D view

(c) Color-mapped slice view

Figure A.15: Exploration of the hurricane Isabel dataset. The combination of high precipitation and snow has been selected so that the blue colors in the slice view correspond with snow while green and cyan correspond to areas with high precipitation.

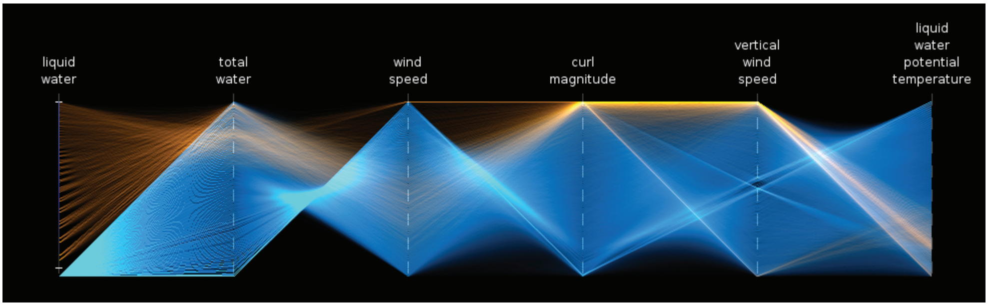


(a) PCP

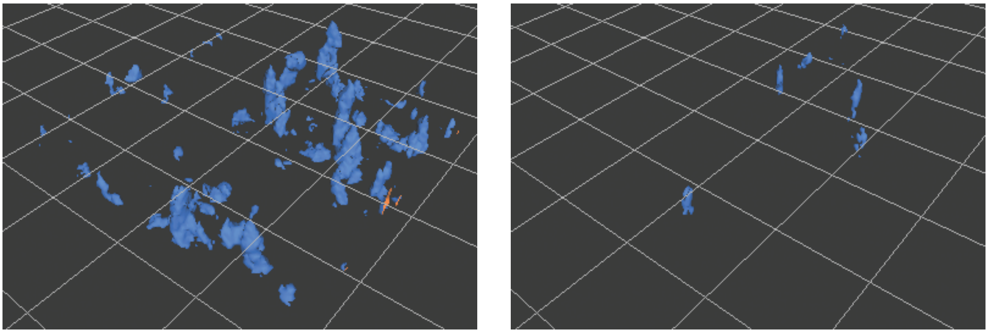


(b) Slice

Figure A.16: A PCP showing the correlation between H and H^+ mass abundance. The PCP uses two color-mapped selection ranges so that the colors in the slice viewer represent the H mass abundance (blue) and the H^+ mass abundance (green).



(a) PCP selection of clouds



(b) All clouds

(c) High-windspeed clouds

Figure A.17: Clouds can be selected based on the liquid water attribute (a,b). The orange lines show the distribution of the other attributes within the clouds. When the high wind-speed outlier is selected, only a select subset of the clouds is visible (c).

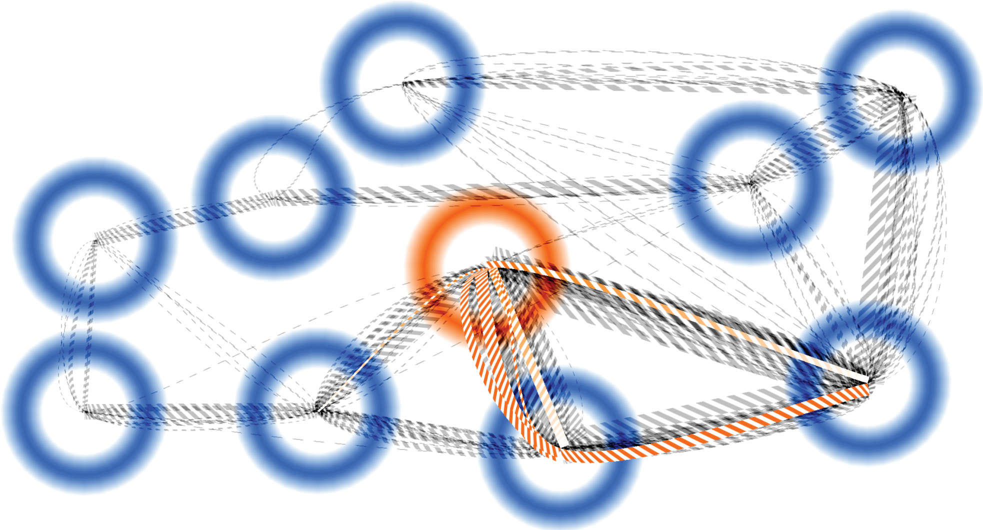
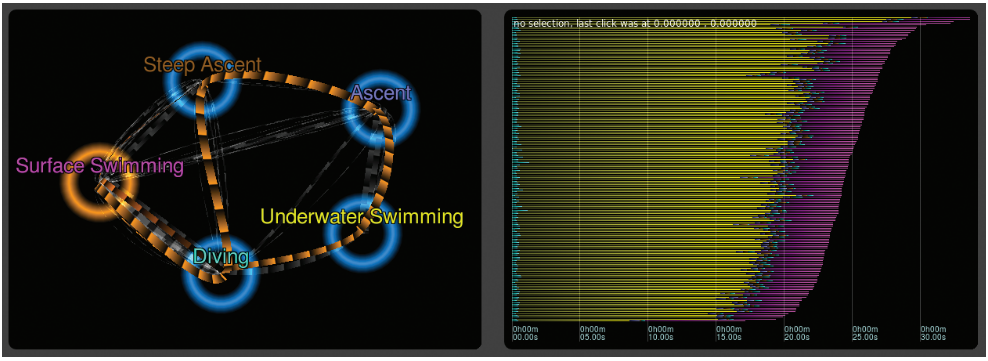


Figure A.18: A smooth graph representation of a labeled biological time-series. Each ring represents a state, and the edges between states visualize the state transitions. This graph uses smooth curves to explicitly visualize third order transitions, so that each curved edge represents a unique sequence of four successive states. The orange node is part of a selection set, and all transitions matching the current selection are highlighted in orange.



no selection, last click was at 0.000000 , 0.000000

Figure A.19: Selection of the penguin's feeding cycle by querying the selecting the sequence of nodes corresponding to diving, underwater swimming, ascent, steep ascent, diving and surface swimming.

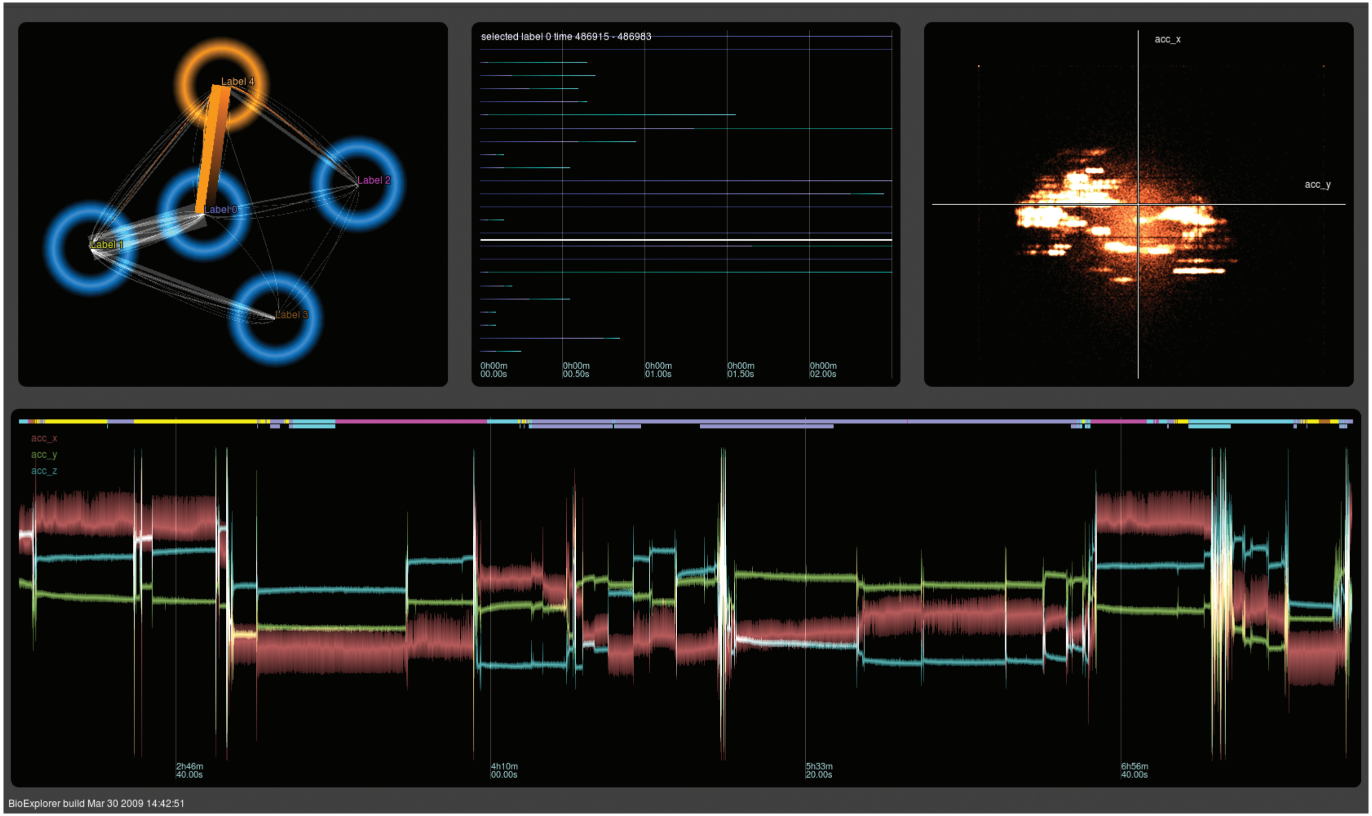


Figure A.20: A screenshot of the user interface of the exploratory application. The three views on the top of the screen represent (from left to right) the GraphView, the SelectionView and the ScatterPlot. The bottom of the screen shows a ScalarView in which both the accelerometer measurements as well as the assigned labels are shown.