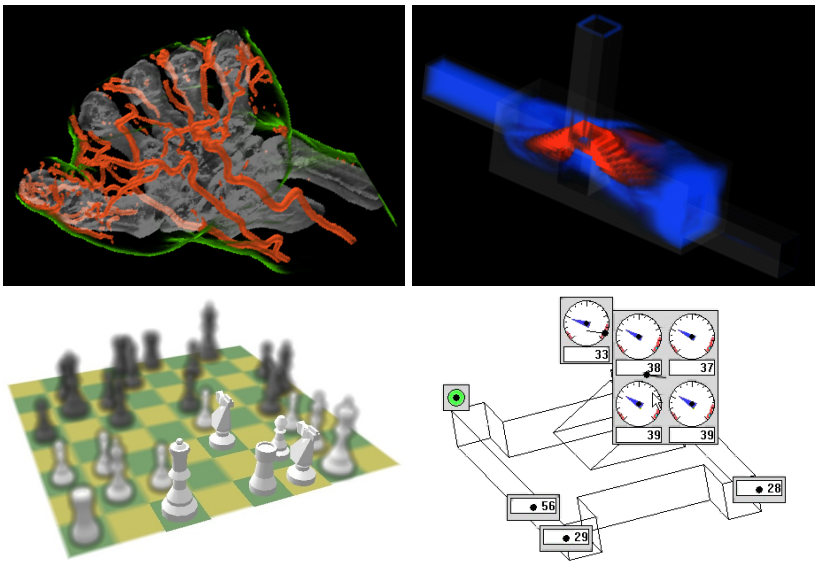
Helwig Hauser

Generalizing Focus+Context Visualization

(Habilitationsschrift)



ausgeführt in den Jahren 1999–2003
zum Zwecke der Erlangung der *venia docendi* (Lehrbefugnis)
im Habilitationsfach “Praktische Informatik”

und eingereicht im März 2004
an der Technischen Universität Wien, Österreich, Fakultät für Informatik,
von Dipl.-Ing. Dr.techn. Helwig Hauser (geb. Helwig Löffelmann),
Lambrechtgasse 16/9, A-1040 Wien, Österreich,
geboren am 16. April 1971 in Wien,
Hauser@IEEE.org.

Preface

Before the actual contents of this thesis are presented, a few words are used to comment on the context of this work, the papers which are contained in this thesis, the thesis itself, and its author.

Most generally, this thesis provides a survey of a subset of Helwig Hauser's research work in the years 1999–2003. After finishing his PhD project on the visulization of dynamical systems (field of flow visualization) in 1998 [80], Helwig Hauser pursued the idea of generalizing focus+context visualization (abbreviated "F+C visualization") in the scope of several projects.

Previously, F+C visualization only has been know in the field of information visualization, and there only in the form of space distortion techniques: comparably large portions of the visualization space are used to visualize certain detailed data parts "in focus" whereas the rest of the data is visually represented "as context" in reduced style (see section 1.2 for more details). Not at the least through the work of Helwig Hauser, F+C visualization now is a more general approach, also including the focus–context discrimination by other visual means, e.g., color, opacity, or style (chapter 1 gives an overview of this generalization). Additionally, F+C visualization now also is used in scientific visualization, e.g., volume visualization and flow visualization, which also can be backtracked to the some work which is presented in the following. This thesis now contains the most important papers with respect to this work.

Chapter 2 describes *two-level volume rendering* (2lVR), which is a new technique for the visualization of segmented volume data, also enabling focus+context visualization through the selective use of color, opacity, and style. Chapter 2 equals a paper which has been published in the journal "IEEE Transactions on Visualization and Computer Graphics" in 2001 [41]. Before, in 2000, Helwig Hauser presented 2lVR at the IEEE Visualization Conference in Salt Lake City, Utah [40]. After the conference this work was selected as an especially interesting contribution which then resulted in an extended version of the paper, published in the journal [41]. Since 2000, several other works by Helwig Hauser as well as by others were based on this paper, e.g., a GPU-based high-quality version of 2lVR with an enriched spectrum of available rendering styles [35] (not in this thesis).

Chapter 3 describes *contour rendering for volume data* as an alternative rendering style for volume visualization which proofed to be especially useful for context rendering in 3D F+C visualization. Chapter 3 equals a paper which has been published in the Proceedings of EUROGRAPHICS 2001 [17]. Contour rendering for volume data has been successfully integrated within two-level volume rendering and is since then a standard option for rendering of volumetric data.

Chapter 4 describes *focus+context visualization of data from 3D flow simulation*, demonstrating how interactive feature specification can be combined with 3D F+C visualization on the basis of volume rendering (2lVR). Chapter 4 equals

a paper which has been published in the Proceedings of the 8th Fall Workshop on Vision, Modeling, and Visualization (VMV) 2003 [39]. This paper is also based on previous work on the F+C visualization of simulation data by Helwig Hauser and others [19, 20], not included as full text within this thesis.

Chapter 5 describes *angular brushing* as a new technique for focus specification with respect to relational data properties when visualized with parallel coordinates. Chapter 5 equals a paper which has been accepted for publication in the Proceedings of the IEEE Symposium on Information Visualization (InfoVis) 2002 in Boston, Massachusetts [37]. The paper version "accepted for publication" is included in this thesis (instead of the ultimately published one [38]), because significant shortening was necessary to gain the finally published version due to page limitations in the proceedings. Accordingly, there is more information in the original (reviewed and accepted) version of this paper.

Chapter 6 describes *semantic depth of field* as a new technique for visually discriminating between focus and context by using a selectively sharp vs. blurred display. Chapter 6 equals a paper which as been published in the Proceedings of the IEEE Symposium on Information Visualization (InfoVis) 2001 in San Diego, California [64]. Later an extended version of this paper has been published in the journal IEEE Computer Graphics and Applications in 2002 [65] (not included in this thesis). In a user study, it was also proven that SDOF indeed can serve as a useful means to realize F+C visualization [66] (full text not in this thesis).

Chapter 7 describes *F+C visualization of process data* based on the traditional approach of space distortion. Chapter 7 equals a paper which has been accepted for publication in the Proceedings of the IEEE Symposium on Information Visualization (InfoVis) 2002 in Boston, Massachusetts [87]. Again, the paper version "accepted for publication" is presented (similar case as with the angular brushing paper, see above) instead of the much shorter, ultimately published article [88].

The generalization of focus+context visualization itself, i.e., the embracing description of the general idea behind all the here presented work, is presented in the first chapter (chapter 1). It serves as a survey of all the related work, brings the other chapters of this thesis in a relation to each other, addresses general aspects of generalized focus+context visualization, and thereby very well acts as an introduction to and overview of all this work. Chapter 1 is the paper accompanying an oral contribution of Helwig Hauser to the Dagstuhl Seminar on Scienitific Visualization in 2003 (exactly about the content of this thesis). High-quality images and further information can be acquired through the World Wide Web and the following URLs:

Two-Level Volume Rendering and Contour Rendering:

```
http://www.VRVis.at/vis/research/two-level/
http://www.VRVis.at/vis/research/npvr/
http://www.VRVis.at/vis/research/voxelstube/
http://www.VRVis.at/vis/research/rtvr/
http://www.VRVis.at/vis/research/compression/
http://www.VRVis.at/vis/resources/diss-LM/
```

F+C Visualization of Simulation Data:

`http://www.VRVis.at/vis/research/volflowvis/`
`http://www.VRVis.at/vis/research/fdl-vis/`
`http://www.VRVis.at/vis/research/smooth-brush/`
`http://www.VRVis.at/vis/research/simvis-time/`
`http://www.VRVis.at/vis/research/diesel-case/`
`http://www.VRVis.at/vis/resources/DA-MMlejnek/`

Angular Brushing

`http://www.VRVis.at/vis/research/ang-brush/`

Semantic Depth of Field:

`http://www.VRVis.at/vis/research/sdof/`
`http://www.VRVis.at/vis/resources/diss-RK/`

Process Visualization:

`http://www.VRVis.at/vis/research/proc-lod/`

General Information:

`http://www.VRVis.at/vis/research/`
`http://www.VRVis.at/vis/`
`http://www.VRVis.at/`

Concerning the type-setting of this thesis it is stated that all papers as described above have been included as they were published (or accepted for publication). Only the type-setting (number of columns, letter face and size, page margins, size of images, etc.) has been unified to the style of this thesis. It is especially noted that, accordingly, no changes to the text of the papers have been made with the exception that all bibliographies have been joined and placed altogether at the end of this thesis.

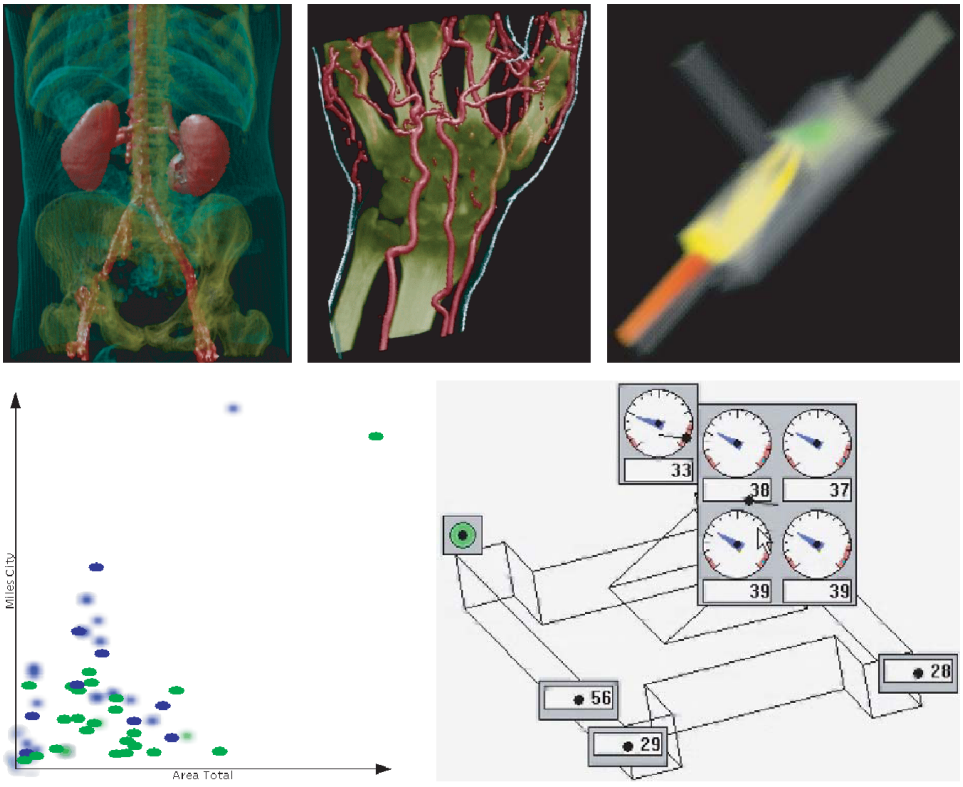
Finally, some concluding remarks are made on the authorship of all the papers included within this thesis. Up to the end of 2003, Helwig Hauser lists more than 50 reviewed or invited papers in his publication list (`http://www.VRVis.at/vis/staff/hauser/publs.html`), none of them is a single-author paper solely by Helwig Hauser (chapter 1 of this thesis will be the first one). The main reason is that Helwig Hauser usually works in collaborative research projects, especially also in conjunction with students (undergraduate as well as graduate, see also Helwig Hauser's Curriculum Vitae at `http://www.VRVis.at/vis/staff/hauser/cv.html`). With respect to the papers included in this thesis, it can be stated that Helwig Hauser either contributed significantly or, moreover, even took the lead in the related projects.

Helwig Hauser, December 22, 2003.

Contents

Chapter 1

Generalizing Focus+Context Visualization



In December 2003, the contents of this chapter have been submitted for publication in the Proceedings of the Dagstuhl 2003 Seminar on Scientific Visualization (paper "**Generalizing Focus+Context Visualization**" by Helwig Hauser).

This paper serves both as an introduction to this thesis as also a survey about all the works that are presented in this thesis. It presents a general idea behind all the papers, which afterwards are included within this thesis. It also is the most

recent work, put into text in December 2003 as a contribution to the Proceedings of the Dagstuhl 2003 Seminar on Scientific Visualization (June 1–6, 2003) where the contents of this chapter also have been presented by Helwig Hauser (on June 3, 2003). Related papers (co-authored by Helwig Hauser) are:

- **Two-Level Volume Rendering** [41], contained in this thesis as chapter 2, and **Two-Level Volume Rendering - Fusing MIP and DVR** [40], an earlier version of this paper
- **Fast Visualization of Object Contours by Non-Photorealistic Volume Rendering** [17], contained in this thesis as chapter 3
- **Interactive Volume Visualization of Complex Flow Semantics** [39], contained in this thesis as chapter 4, **Smooth Brushing for Focus+Context Visualization of Simulation Data in 3D** [20], a tightly related paper, and **Interactive Feature Specification for Focus+Context Visualization of Complex Simulation Data** [19], also a related paper
- **Angular Brushing for Extended Parallel Coordinates** [37], contained in this thesis as chapter 5, later published in a shortened version [38]
- **Semantic Depth of Field** [64], contained in this thesis as chapter 6, and **Focus + Context Taken Literally** [65], an extended version of this paper
- **Process Visualization with Levels of Detail** [87], contained in this thesis as chapter 7, later published in a shortened version [88]

Generalizing Focus+Context Visualization

Helwig Hauser

Abstract

Focus+context visualization is well-known from information visualization: certain data subsets of special interest are shown in more detail (locally enlarged) whereas all the rest of the data is provided as context (in reduced space) to support user orientation and navigation in the visualization. The key point of this work now is a generalized definition of focus+context visualization which extends its applicability also to scientific visualization. We show how different graphics resources such as space, opacity, color, etc., can be used to visually discriminate between data subsets in focus and their respective context. To furthermore demonstrate its general use, we discuss several quite different examples of focus+context visualization with respect to our generalized definition (all of which we have published separately in more detail). Finally, we also discuss the very important interaction aspect of focus+context visualization.

1.1 Introduction

For a long time already, modern society is greatly influenced by computers. Mainly, computers are used to process data of various kind. Additionally, computers are also used to support the acquisition of data, for example, through measurements or computational simulation. Due to a steadily increasing performance of computers (Moore's law), year by year more data is processed. Since users do not extend their capabilities in data-processing at a comparable rate, there is an increasing need for efficient tools to support the processing of large amounts of data.

One very useful opportunity for accessing large amounts of data is visualization. Data is communicated to the user in a visual form to ease processing. Instead of dealing with loads of numbers, the user accesses the data through pictures and a graphical user interface. This approach is especially useful when the data has at least some spatial form inherently associated with it. In many scientific applications, for example, data is tightly related to concrete parts of our real world, e.g., a 3D computer tomography scan of a human body in a medical application or the 3D simulation of air flow around the computer model of a new aircraft.

The main advantage of visualization is that it uses the great bandwidth that is offered by the human visual system for visualization. However, also for visual-

ization the amount of data to be shown simultaneously is limited. For very large data sets, details cannot be shown for all of the data at the same time. In this case, the user usually is offered the opportunity to either get an overview of the data (no details), or zoom into specific parts of the data and get all of the deatils there.

While scientific visualization (SciVis, the visualization of scientific, i.e., inherently spatial data) has been reseinformation visualization) such as bank account data or census data has become popular.arched for dozens of years already, more recently also the visualization of non-scientific, abstract data (InfoVis, information visualization) such as bank account data or census data has become popular. In InfoVis, an additional step is required in the visualization process, i.e., the mapping of non-spatial data to a visual form. As the user has to learn this additional mapping to effectively use the visualization (and to successfully build up a mental map of the data–form relation), more care is required to support the user with orientation in the visualization. Careless zooming across multiple levels of details easily can cause an effect like being lost in too many details. Thus, advanced solutions have been developed in this field to supply users with both overview and details of the data.

1.2 Focus+Context Visualization

In information visualization, an approach called focus+context visualization (F+C visualization) has been developed which realizes the combination of both the general overview and a detailed depiction within one view of the data at the same time. Traditionally, focus+context visualization refers to an uneven distortion of visualization space such that relatively more space is opened for a certain subset of the data (data in focus). At the same time the rest of the visualization is compressed to still show the rest of the data as a context for improved user orientation.

The idea of using different magnification factors for different parts of the visualization (in one image) to display information in a F+C style already dates back to the '70s of the 20^{th} century [25, 53]. Furnas' work on the fisheye view [29] in 1981 often is accepted as the historical start into computer-based F+C visualization. In this work, Furnas describes how information is selected for display in dependence on an a-priori importance and the distance of each data item to the current focus of the visualization. Also in the early 1980s, Spence and Apperley presented the bifocal display as a one-dimensional distortion technique [121] to provide a shrinked context on both the left and the right side of an undistorted focal region in the middle of the visualization.

During the 1980s, both approaches have been generalized and extended [30, 76]. In 1992, Sarkar and Brown presented the graphical fisheye view [113], based on Furnas' work, but more focusing on the graphical appearance of the F+C visualization (comparable to a real fisheye lens). One year later, Sarkar et al. discussed two techniques (orthogonal and polygonal stretching) for F+C visualization based

on the concept of a stretchable rubber sheet [115]. And already in 1994, Leung and Apperley presented a review of distortion-oriented F+C visualization, including additional approaches such as the perspective wall [84] (see also later), and providing a respective taxonomy [77]. They describe techniques of F+C visualization by the characteristics of the magnification function (being the derivative of the tranformation function from the undistorted view to the F+C view). Doing so, two classes of techniques are differentiated: approaches with a continuous magnification function (such as the graphical fisheye [113]) and others (such as the bifocal display [121]). Amongst the latter, the authors differentiate between techniques with piece-wise constant magnification factors (the bifocal display, for example) and others (the perspective wall, for example).

The perspective wall [84], presented by Mackinlay et al. in 1991, is based on the concept of "bending backwards" parts of the display on both the left and the right side of the focus region in the center of the screen (similar to the bifocal display [121]). Perspective projection is used to achieve a variation in magnification factors within this kind of F+C visualization. In 1993, this approach was extended to the so-called document lens – also parts above and below the focal region are used for context visualization.

In the domain of distortion techniques with continuous magnification functions, further extensions have been presented after the first half of the 1990s. In 1995, the three-dimensional pliable surface was presented by Sheelagh et al. [12], also using perspective projection to achieve different magnification factors in different parts of the display. Gaussian profiles are used to generate magnification (thus yielding continuous magnification) and multiple foci are possbile in one view. In 1996, Keahey and Robertson presented non-linear magnification fields as a technique independent from perspective projection and with direct control over the magnification function on every point of a grid over the display [57, 58]. The transformation function is computed in an iterative process, locally optimizing on the difference between the discrete derivative of the transformation field and the input (magnification field).

In 1995, the mapping of hyperbolic space to the plane was used by Lamping et al. [73, 72, 71] to achieve F+C visualization, enabling the visualization of infinite space on a limited screen space (at least in principle). In a similar fashion, Kreuseler et al. used the mapping from spherical space to the plane for F+C visualization [68], allowing for movements of the focal center around the sphere.

The large amount of work on distortion techniques for F+C visualization documents the relevance of this approach, especially in the domain of information visualization. But instead of discussing more details about distortion techniques for F+C visualization or other approaches in this field, we restrict this overview to the above mentioned examples and proceed towards our generalization of F+C visualization. First, we briefly discuss how focus is separated from context, an inherently necessary part of every focus+context visualization.

1.3 Separating Focus from Context

When dealing with focus+context visualization, it is inherently necessary to have a notion of which parts of the data are (at least currently) considered to be "in focus" and what others are not (context part of the data). In the course of this work, we use a so-called *degree-of-interest function* (DOI function), $doi()$, which describes for every item of the data whether (or not) it belongs to the focus (similar to Furnas' definition [29], but normalized to the unit interval):

$$doi_{\text{bin}}(\texttt{data}[i]) = \begin{cases} 1 & \text{if } \texttt{data}[i] \text{ is part of the focus} \\ 0 & \text{if } \texttt{data}[i] \text{ is part of the context} \end{cases}$$

In many application scenarios a binary discrimination between focus and context (as formulated above) is most appropriate, i.e., to assume a sharp boundary between the data items in focus and all the others. In many other cases, however, it is more appropriate to allow for a smooth change of $doi()$-values between the data items in focus and their context (resulting in a *smooth degree of interest* [20]). In other words, the question of whether a data item belongs to the focus (or not) also can be answered by the use of a fuzzy attribute $doi()$:

$$doi(\texttt{data}[i]) = \begin{cases} 1 & \text{if } \texttt{data}[i] \text{ is part of the focus} \\ doi \in]0,1[& \text{if } \texttt{data}[i] \text{ is part of the smooth boundary between focus and context} \\ 0 & \text{if } \texttt{data}[i] \text{ is part of the context} \end{cases} \tag{1.1}$$

Accordingly, a fractional value of *doi* is interpreted as a percentage of being in focus (or interest). A fractional $doi()$-value can be the result of a multi-valued definition with multiple (still discrete) levels of interest, a non-sharp definition of what is interesting (e.g., through a definition which is based on continuous spatial distances), or a probabilistic definition (e.g., through a definition incorporating a certain amount of uncertainty).

Usually, the specification of the $doi()$-function is tightly coupled with the user interface. Different approaches are used to let the user specify which parts of the data (currently) are of special interest (explicit vs. implicit specification, for example). In section 1.5 we discuss the interaction aspect of F+C visualization in more detail.

In traditional F+C visualization (space-distortion techniques), the degree of interest $doi(\texttt{data}[i])$ directly relates to the local magnification used to depict a data item $\texttt{data}[i]$[1]: the larger the $doi()$-value is, the more screen space is used for visualization, at least locally.

[1] this 1:1–relation only holds to a certain extent of accuracy – in general it is not possible to translate every DOI/magnification function into a corresponding transformation function [77, 57].

1.4 Generalized Focus–Context Discrimination

Although the vast majority of research work on F+C visualization has been devoted to space-distortion techniques, the idea of visually discriminating those parts of the data which currently are in focus from all the rest (context information) is more general. In addition to using more space for showing the focus in more detail, other visual dimensions can be used in a similar way. In volume rendering, for example, typically more opacity is used for parts of the data in focus [78], whereas a greater amount of transparency is used for context visualization. Additionally, also color can be effectively used to visually discriminate between different parts of the data. In a system called WEAVE [34], for example, those parts of the data which positively respond to a certain user query (i.e., the current focus) are shown in color, whereas the rest of the data (the context) is shown in gray-scale.

Similarly, other visual dimensions, such as image frequencies, rendering style, etc., can be used for focus–context discrimination (see below for examples). We therefore propose to generalized the definition of focus+context visualization in the following way: *focus+context visualization is the uneven use of graphics resources* (space, opacity, color, etc.) *for visualization* with the purpose of visually discriminating those parts of the data which are in focus from the respective context, i.e., the rest of the data. In table 1.1, we give several examples of F+C visualization which are quite different from each other but which all match the above definition and thereby demonstrate its general character. The examples differ from each other with respect to which graphics resource is (unevenly) used to achieve F+C visualization. Below we discuss some of these examples in more detail (those marked with an asterix in table 1.1). In table 1.2 we provide a side-by-side comparison of five sample techniques (one sample image each) with pointers to other parts of this document with more detail as well as other pieces of related literature.

1.4.1 More Opacity for Visualization in Focus

One alternative style of F+C visualization (alternative to space-distortion techniques) is identified in a domain where usually other objectives, slightly different from focus–context discrimination, actually do govern the development of new techniques. In volume rendering, all from the beginning on [78], a so-called opacity transfer function (OTF) $\alpha()$ was used to deal with the fact that usually not all of the 3D data can be shown simultaneously at full intensity. OTF $\alpha()$ is used to map the data domain to the unit interval ($1 \leftrightarrow$ opaque, $0 \leftrightarrow$ completely transparent). Using an OTF, different values of opacity/transparency are assigned to different parts of the data. This causes that some parts of the data become more prominently visible in the rendered image while others are not (or only hardly) visible at all.

Originally, the use of an opacity transfer function never was argued with the need to discriminate parts of the data “in focus” from their “context”. However,

graphics resource	**approaches**	**sample technique(s)**
space	more space (magnification) for data in focus	graphical fisheye view [113], ...
		F+C process visualization [88]*
opacity	focus rather opaque, context rather transparent	direct volume rendering [78], ...
		RTVR [95]*
color	colored focus in gray context	WEAVE [34], SimVis [20, 19]*
	saturated/light colors for focus	Geospace [82], RTVR [95]*
frequency	sharp focus, blurred context	semantic depth of field [64, 65]*
style	context in reduced style (NPR)	two-level vol.-rend. [40, 41]*
		NPR-contours [17]*

* ... techniques which are described in more detail in section 1.4

Table 1.1: Realizing (generalized) F+C visualization by the uneven use of graphics resources (space, opacity, color, etc.) to discriminate parts of the data in focus from the rest (context) – more details in section 1.4.

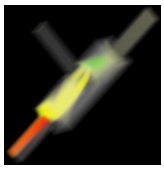
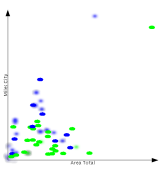
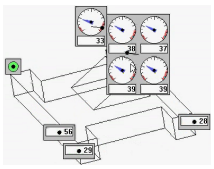
sample image					
graphics resource	opacity	style	color	frequency	space
section	1.4.1	1.4.2	1.4.3	1.4.4	1.4.5
chapter(s)	2, 4	2, 3, 4	4, 5	6	7
paper(s)	[95]	[40, 41, 17]	[20, 19]	[64, 65]	[88]

Table 1.2: Sample images of five different F+C visualization techniques (from left to right): RTVR-based volume rendering, two-level volume rendering, F+C visualization of simulation data, semantic depth of field, F+C process visualization.

the goal to visually bring out certain parts of the data in the visualization while reducing the visual appearance of all the rest very well matches the principal idea of F+C visualization. On the basis of a degree-of-interest function, an OTF can be specified by

$$\alpha(\texttt{data}[i]) = a(doi(\texttt{data}[i]))$$

with $a()$ being the identity map ($a(x) = x$), a simple windowing function (see figure 1.1), or any other (potentially simple) monotonic map from $[0,1]$ to $[0,1]$. When $doi()$, for example, is defined on the basis of a scaled distance from a predefined iso-value – $doi(\texttt{data}[i]) = \max\{1 - s\,|\,\texttt{data}[i] - v_{iso}\,|\,, 0\}$ –, then one of Levoy's OTF (though not yet dependent on the data gradients) is regenerated with $a()$ being the identity map (or a simple window).

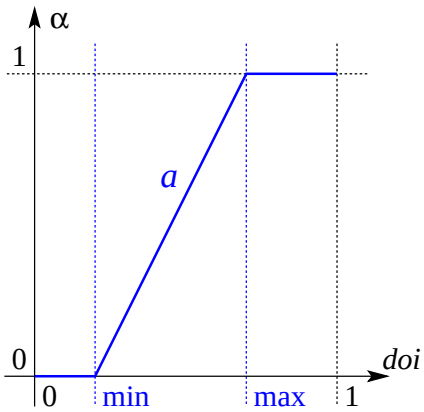


Figure 1.1: A simple "window" often is sufficient to map *doi*-values to α-values: *doi*-values up to a certain minimum are mapped to a minimal value of opacity (often 0), whereas *doi*-values above a certain maximum are mapped to some maximal opacity. In between, a linear map from *doi*-values to α-values is used.

From many years of work on the question of how to specify an optimal opacity transfer function [36] we know that one simple data-dependent function $doi()$ (or $\alpha()$) often is not sufficient to optimally discriminate between focus and context in a visualization of 3D data, e.g., 3D medical data or 3D data from computational simulation. Instead, often sophisticated segmentation algorithms are used to do a proper focus–context discrimination before the actual visualization. The result of a segmentation algorithm usually is an n-valued object map $object()$, telling for each and every data item $\texttt{data}[i]$ which object it belongs to.

In two-level volume rendering (2lVR) [40, 41], such an object map is used to improve the F+C visualization of 3D data: instead of directly deriving $doi()$ from the data, the degree of interest is defined on the basis of $object()$, i.e., for all the objects in the data (and not the singular data items) it is determined how interesting they are. This is done, because in many applications the 3D data anyhow is assumed to be composed of objects (in medical applications, for example, a dataset usually is assumed to be composed of bones, tissue, etc.). Therefore, the user automatically tends to formulate the focus–context discrimination in terms of the data objects (like "I'd like to see the bones and the blood vessels in the context of the skin."). For rendering, two values of opacity are used in two-level volume rendering: in addition to the $object()$-based (global) opacity $\alpha_{global} = a_{global}(doi(object))$, which yields the overall opacity for an object (depending on its degree of interest), a local (object-wise specified) OTF $\alpha_{local}(\texttt{data}[i], object(\texttt{data}[i]))$ is used to individually steer the visual appearance of every object.

For example, assuming $\alpha_{local}(., 1)$ to be a relatively sharp Levoy-OTF (comparably large s) and a_{global} to be the identiy map, object 1 would be rendered like an iso-surface with its importance $doi(1)$ directly relating to its opacity. Through

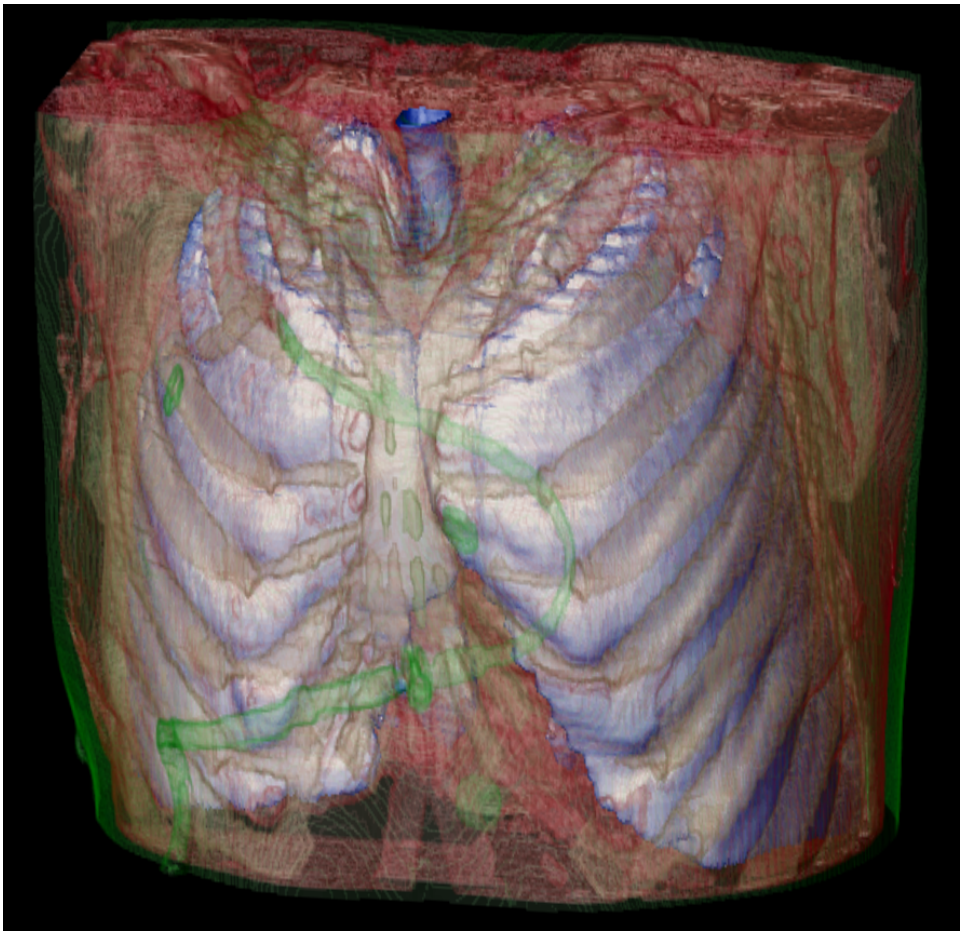


Figure 1.2: A segmented CT-dataset of a chest, visualized using two-level volume rendering. Different values of overall opacity have been used for lung (completely opaque), bones (semi-transparent), and skin (very transparent).

this separation of α_{global} and α_{local} the task of emphasizing certain parts of the data (semantical question) is separated from the question of how to render the different parts of the data (syntactical question). Accordingly, the parameterization of two-level volume rendering (adjustment of opacities) is much more intuitive (when compared to using a standard OTF only) and thus it is possible to achieve better results in shorter time. See figure 1.2 for a sample visualization of segmented 3D chest data with the focus on the two lungs.

In addition to opacity variations, two-level volume rendering also offers alterative ways to achieve a visual focus–context discrimination, for example, by varying the rendering style. But before we futhermore discuss 2lVR, another example for opacity-based F+C visualization is briefly described, which comes from the field of information visualization. Parallel coordinates [47, 49, 48] are a well-established technique for the visualization of high-dimensional data. Every

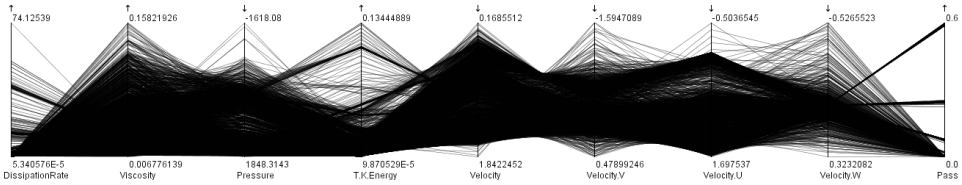


Figure 1.3: 9-dimensional data from computational flow simulation (5400 cells of a T-junction), visualized with parallel coordinates.

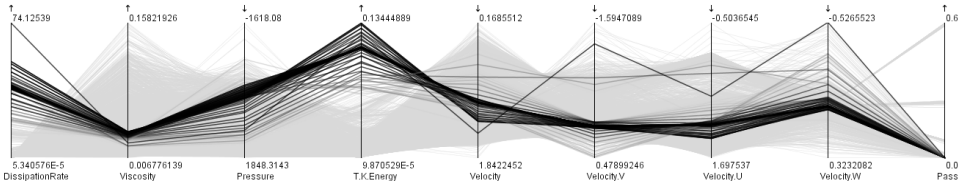


Figure 1.4: DOI-based opacity used to visually separate some parts of the data "in focus" (characterized through rather large values of "T.K.Energy") from all the rest (context).

n-dimensional data item is plotted as a polyline across n parallel axes in screen space such that a data item's polyline intersects the axes exactly at those points which relate to the data item's n attributes (see figure 1.3 for a sample image).

When many data items are to be shown simultaneously (tens of thousands or more), problems with overdraw easily occur: many pixels are covered by several (or even many) polylines. The resulting effect is that the visualization looses effectivness due to visual clutter – a classical scenario where F+C visualization can help. Using a DOI-based opacity to draw semi-transparent polylines over each other [38], an improved display is gained which allows for interactive analysis of the n-dimensional data (see figure 1.4). Note that the ability to interactively focus in such a F+C application is essential here to effectively exploit the visual superiority of this kind of visualization.

1.4.2 Reduced Style for Context Visualization

Another option of visually distinguishing between objects in focus and their context is to use different rendering styles. In two-level volume rendering, for example, it is possible to use different rendering techniques for different objects in the data. On the global level, the different representations of the data objects are combined using standard compositing to achieve the final image. In addition to standard volume rendering (α-compositing), shaded surface rendering, maximum intensity projection (MIP), x-ray rendering, and non-photorealistic contour rendering can be used to depict an object. In table 1.3 some visualization properties are listed for different rendering styles in 3D visualization. A good opacity control, for example, favors the visualization as part of the context, because occlusion is

rendering style	**visualization properties**
α-compositing	conveys appearance of semi-transparent 3D medium (F), opacity difficult to control
shaded surf. displ.	well conveys 3D form (F), good transparency control (C)
max. intensity proj.	good for complex forms (F), limited 3D appearance, good transparency control (C)
x-ray rendering	good for overview (C), complex opacity distribution
contour rendering	reduced appearance (C), little problems with occlusion
	F ... good for focus visualization, C ... good for context visualization

Table 1.3: Visualization properties of different rendering styles for 3D visualization together with a rough assessment of how they can be used for F+C visualization. Depending on whether the focus is inside the context (or outside), or if the context is of complex shape (or a rather coherent object), different combinations of rendering styles yield good results for F+C visualization (details in section 1.4.2).

easier controlled. The ability to visualize 3D form well, as another example, favors the visualization of data parts in focus. In the following we discuss several useful combinations of different rendering styles for F+C visualization.

Shaded surface display very well acts as visualization of objects in focus, especially if the object(s) in focus are inside the context and, consequently, an opaque surface is used for visualization. This way, usually a strong and sharp appearance of the objects in focus is possible with a good communication of 3D shape. For context visualization, in such a case, the use of contour rendering and/or MIP is very interesting. Contour rendering works fine, because of its reduced appearance (lots of object parts are left away whereas only their contours are shown) and the fact that usually the middle parts of the visualization (where the objects in focus are shown) is rarly occluded (see figure 1.5, right image). Additionally, also MIP usually is useful for context visualization because of its easy-to-control opacity – only one data value per viewing ray is chosen for display, all object representatives share the same opacity (see figure 1.5, left image).

In case of context which is inside the objects in focus, like the bones acting as context to blood vessels (as the objects of interest in angiography), for example, shaded surfaces again serve a good job of focus visualization. The surfaces, however, need to be rendered semi-transparently (at least to some extent) to allow the user to peer inside and get visual access to the otherwise occluded context. MIP again is useful for the depiction of the context objects (good transparency control) – see figure 1.6 (left image) for a sample rendering of such a situation. Similarily, an x-ray simulation sometimes is useful for context visualization within objects of interest (see figure 1.6, right image, for an example).

In addition to context rendering, MIP also is useful for depicting objects in focus, especially is they are of complex shape (like an entire system of blood vessels

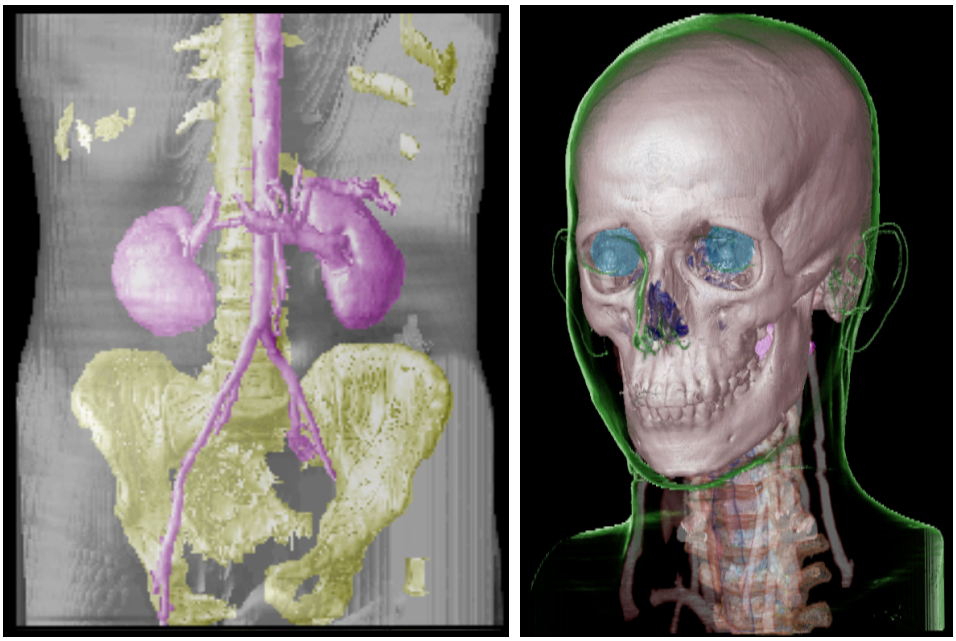


Figure 1.5: MIP is useful for context visualization (skin on left side) because of its easy-to-trim opacity. Contour rendering works very well for context visualization (skin on the right) because of its reduced appearance (little problems with occlusion).

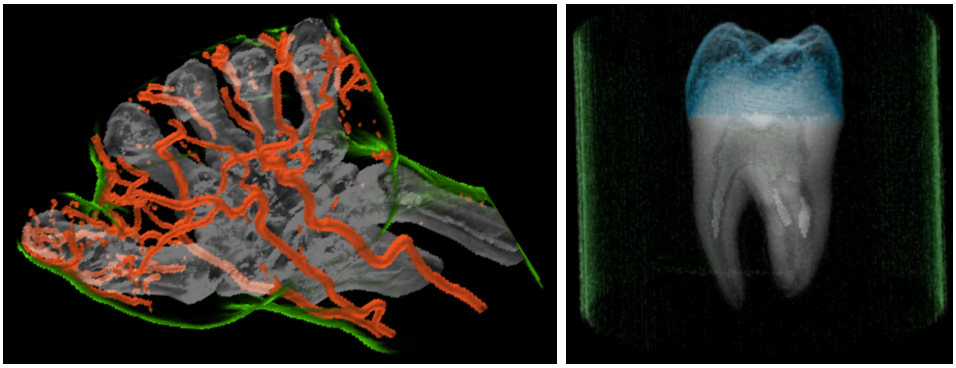


Figure 1.6: F+C visualization of CT data of a human hand (left side): the objects of interest (blood vessels) are drawn as semi-transparent surfaces, whereas the bones are rendered using MIP. Contour rendering has been used to depict the skin. An x-ray simulation has been used to depit the dentine of the tooth on the right side (semi-transparent surface rendering of the adamantine and contour rendering of surrounding material).

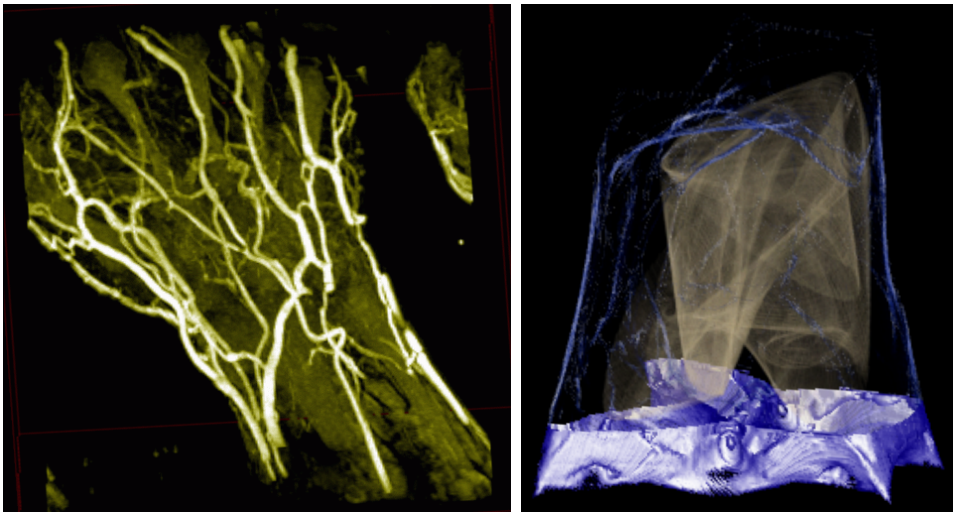


Figure 1.7: Two examples of using MIP for complex objects in focus: the system of blood vessels in the CT hand data (left side) and a chaotic attractor within its basin of attraction on the right side (parts of the basin are shown as shaded surface whereas the rest of the basin is shown using contour rendering to minimize occlusion).

or a chaotic attractor in a dynamical system [2]). In figure 1.7 two examples of such a visualization are given. On the left side MIP is used to show the blood vessels within the CT hand data. On the right side a complex attractor with fractal shape is visualized using MIP. The context (the basin of attraction, in this case) is shown in two ways: whereas the lower parts are shown as a shaded surface, the upper parts are provided using contour rendering only (to reduce problems with occlusion).

1.4.3 Eye-catching Colors for Focus Visualization

In addition to opacity and style as discussed in the previous two sections, also color is effectively used to focus within a visualization. From perseptual research on preattentive vision [128] we know that human observers can very quickly "find" colored parts of a visualization within an otherwise gray-scale depiction. This "search" usually succeeds even before the observer actually starts actively searching, i.e., in a time shorter than 200 ms from stimulus. Accordingly, coloring some parts of a visualization (which are in focus) and showing all the rest in a gray-scale way, also works fine as a F+C visualization technique.

Gresh et al. presented a system called WEAVE [34] which uses this style of F+C visualization for the display of complex simulation data of a beating human heart. Different views of different types of visualization (a scatterplot, a 3D view, etc.) are used to depict and analyse the multi-dimensional simulation data. To assess the large amount of data, the user is able to select certain data subsets of

special interest. These parts of the data are then drawn in color whereas all the rest is displayed in gray-scale style. First of all, the colored parts of the visualization immediately stand out of every view where this kind of focus–context discrimination is used. Secondly, the coloring is done consistently across all the views, so visual linking is established between the views. The same color always indicates the same selection of the data (focus), just visualized differently according to the different views (thereby different characteristics/dimensions of the same data are visualized in the different views). In information visualization this approach is called *linking and brushing* (L&B) – "brushing", because the process of selecting a data subset of interest usually is done directly on one of the linked views, similar as in a drawing program.

In a system called SimVis, we use this approach to visualize data from computational simulation of processes in the automotive industry. An extended brushing technique called *smooth brushing* [20] allows for a gradual transition of the *doi*-function from the subset of interest (focus, $doi = 1$) to the rest (context, $doi = 0$). For visualization, a gradual reduction of color saturation is used to reflect the continuously diminuishing degree of interest. See figure 1.8 for a sample result of this kind of visualization, where a data subset of high pressure and high velocity was selected using smooth brushing in the scatterplot on the right. On the left, a visually linked 3D view shows where those areas of high pressure and high velocity lie in the 3D flow domain (a model of a catalytic converter). In addition to the DOI-based variations of color saturation also the glyph size is varied according to the data item's degree of interest (the more interesting the bigger the glyphs used).

In figure 1.9 volume rendering on the basis of α-compositing [39] was used to depict a subset of a flow through an extended T-junction (characterized through values of high temperature and high turbolent kinetic energy, see scatterplot on the right).

In a system called GeoSpace [82], user queries are answered visually through high-lighting the data parts in focus, i.e., those data items which positively respond to the user query. High-lighting is done, for example, by increasing the color lightness. Thereby the selected data subsets visually stand out from the rest of the depiction. In two-level volume rendering, this approach is used to provide feedback to the user during object selection in the 3D domain. For a short time after the selection of an object in the scene, the selected object is shown with a different transfer function (increased color lightness, increased opacity). Thereby a clear visual linking between an object's name or ID and its visual representation as part of the visualization is established (see figure 1.10).

This is especially useful, when volume visualization is performed in a virtual environment. In this case, especially when 3D objects are to be selected directly through 3D user interaction (for example, by the use of a 3D pointing device), object high-lighting greatly supports the interactive placement of the 3D pointing device. While moving the pointing device, the user immediately gets feedback on

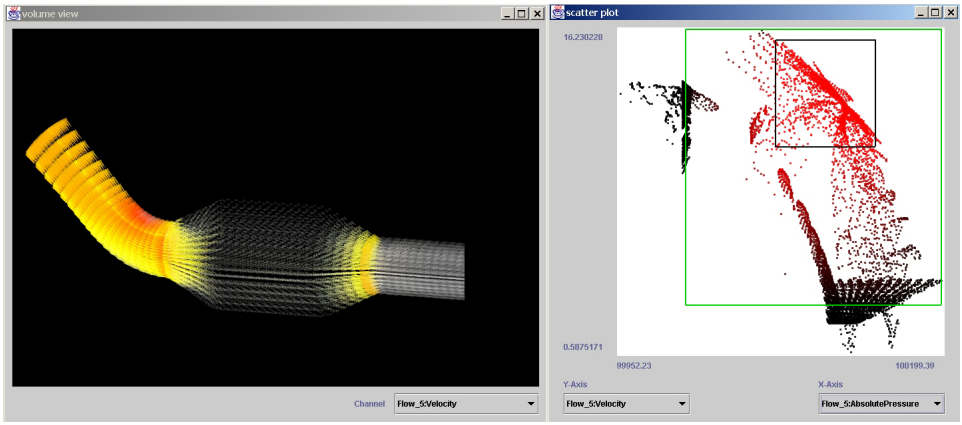


Figure 1.8: F+C visualization of CFD data (flow through a catalytic converter). A data subset, represented by values of high pressure and high velocity, has been selected by smooth brushing on the scatterplot on the right. Gradual changes of color saturation on the left (in the 3D view) represent the smooth degree of interest.

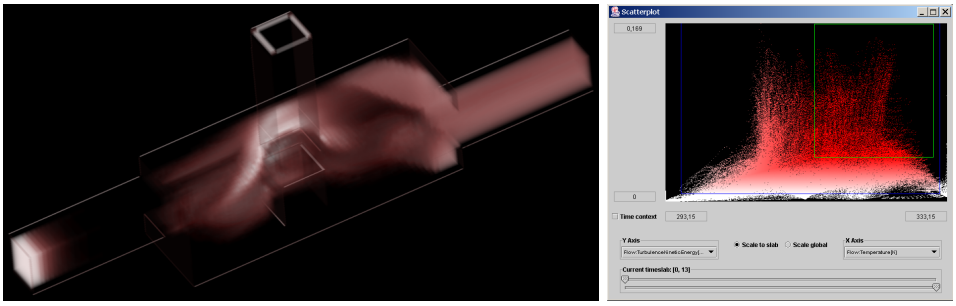


Figure 1.9: Visualization of flow through a T-junction. The visualization focuses on a flow subset which is characterized by high temperature and high turbolent kinetic energy. The junction-geometry is added as context (contour rendering).

which object the pointer currently is pointing towards. Thereby, the user is easily able to efficiently select the one object of special interest without a lot of trial and error (which otherwise is quite normal for 3D direct selection). Figure 1.11 gives a number of snapshots of a video which was taken during a session where the user moved a 3D pointing device around a 3D dataset of a human chest with different segmented parts of the data. Whenever the 3D pointing device enters another object in the scene, the respective object is rendered in a high-lighted fashion according to the above mentioned transfer function alternation.

1.4.4 Band-limited Context

Before we lateron (section 1.4.5) come back to the traditional way of F+C visualization, we furthermore describe one additional way of visually discriminate the vi-

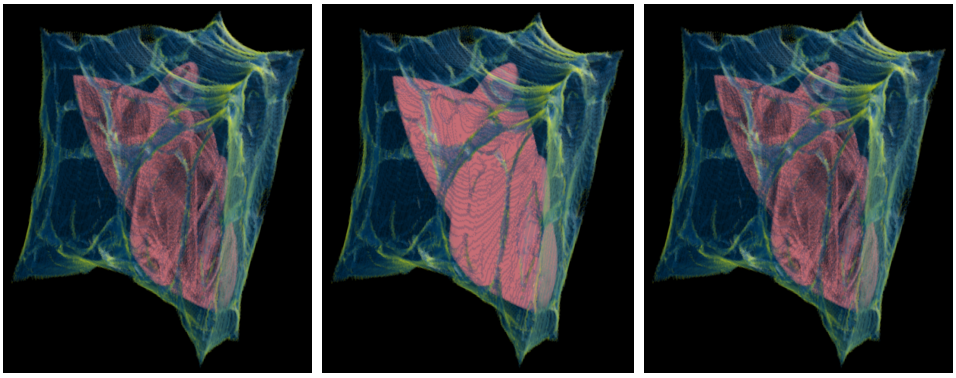


Figure 1.10: Object high-lighting in the course of object selection: before the selection ($t = t_0$, left image), right after the selection of the chaotic attractor ($t = t_0 + \approx \frac{1}{2}$sec., middle image), and a little later after high-lighting ($t = t_0 + \approx 1$sec., right image).

— bones liver kidneys —

Figure 1.11: Several snapshots from a video which was taken through a session where the user moved a 3D pointing device across a 3D dataset of a human chest in a virtual environment. Visual object high-lighting reflects the current 3D position of the 3D pointing device which is very useful to efficiently position the device in 3D space during object selection.

sualization of data parts in focus from all the rest (context). Again (as compared to the use of eye-catching colors for focus visualization, see section 1.4.3) it is an argument from perceptual psychology which motivates this alternative approach: the difference between a sharp and blurred object depiction efficiently can be used for visual focus–context discrimination [64, 65], a technique we call *semantic depth of field* (SDOF). In a user study we could prove that the perceptual identification of sharp objects among blurred others indeed is preattentive [66], i.e., is performed by the human perceptual system within a very short time ($\leq \approx 200$ms).

In 2D, the basic idea of SDOF (semantic depth of field) is (a) to assume a camera model with a depth-of-field effect for rendering and (b) to virtually displace parts of the visualization along the viewing axis to achieve a blurred or sharp depiction of irrelevant and relevant parts of the data, respectively (see figure 1.12). With a lens-based camera, objects are only displayed sharply if they are located at the focal distance from the lens. Object which are displaced along the viewing axis are blurred according to their distance from the lens. Therefore, the displacement in the depth direction is done according to the degree-of-interest values which are

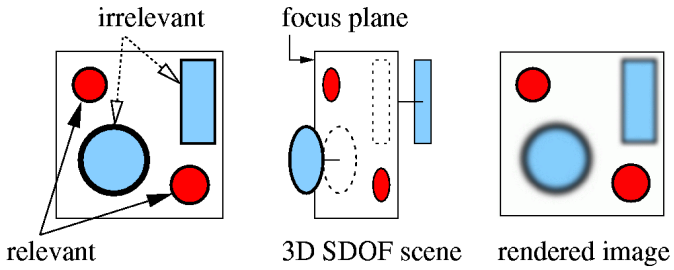


Figure 1.12: The basic idea of SDOF for 2D visualization: assuming a lens-based camera model for rendering, the visualization objects are virtually moved back or forth along the viewing direction to achieve a blurred and sharp depiction for irrelevant and relevant data items, respectively.

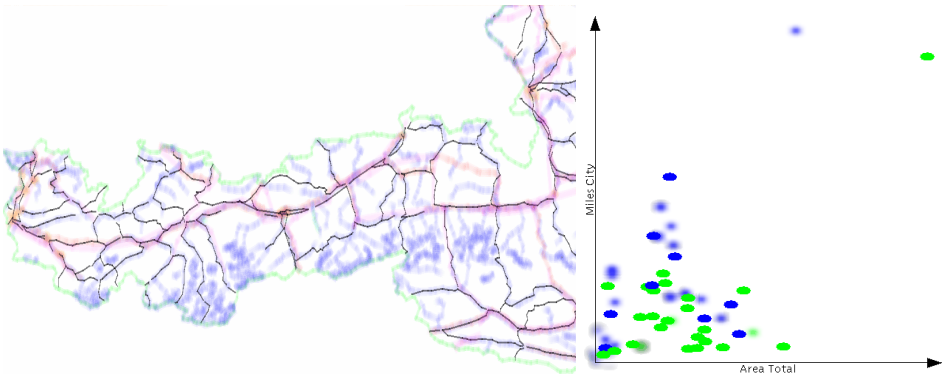


Figure 1.13: Two examples of an SDOF visualization: streets standing out of an SDOF map visualization on the left (other parts of the map blurred) and a scatterplot with SDOF effect on the right.

associated to all the data elements (and not as a spatial function of the data as it is in real-world photography). For 3D visualization, a similar SDOF model exists [64, 65].

Confronted with the result of such an SDOF visualization (see figure 1.13), the user can immediately identify the data subsets in focus (similarily as in photography where sharpness also directly correlates to the fact of being in focus). Therefore, this kind of F+C visualization becomes especially useful when the DOI assignement is done implicitly, e.g., through brushing of invisible dimensions (with a range slider, for example) or through defining the DOI value by how well a data item matches a certain user query [3]. In all these cases the first task a user usually performs is to identify which data items actually have been assigned a high DOI value (and which not). With SDOF this is easily possible as the sharp parts of the visualization, representing the relevant data items, stand out of the depiction automatically.

1.4.5 More Space for Details

After discussing four alternative ways of realizing focus–context discrimination in visualization (based on the variation of opacities, styles, colors, and frequencies), we come back to the traditional way of F+C visualization, i.e., to the variation of magnification factors within a single image. This also thereby completes the picture of our generalization. In section 1.2 we already discussed the extensive block of literature on this kind of F+C visualization. In our case, we have applied this classic principle to process visualization [88] where this has not been done before.

In process visualization, data which is streaming in from a larger number of processes has to be presented to a user such that process surveillance as well as interactive analysis is possible. In analogy to traditional process visualization, where processes are visualized with analog instruments like gauges or other display devices, programs for process visualization (at least partially) mimic this kind of visualization with virtual instruments. One disadvantage of virtual instruments is that they take up quite a lot of screen space. When multiple streams of process data are to be shown simultaneously, not enough screen space is available to show all the data with regularily sized instruments. In such a situation, distortion-based F+C visualization becomes useful.

To achieve F+C visualization of process data, several levels of detail have been designed for the different virtual instruments in use. The different levels of detail use different amounts of screen space, ranging from a small lamp, color-coding the process data, up to a fully fletched virtual instrument, using a hundret times the amount of screen space as compared to the lamp. If not all of the data can be shown at the highest level of detail simultaneously (due to lack of screen space), different levels of detail can be combined according to DOI values of the different data items. See figure 1.14 for an example, where three streams of process data are visualized. DOI values inversely correlate to the distance between the respective virtual instrument and the pointer which is interactively moved by the user (from left to right). Thereby, those virtual instruments which are nearest to the pointer are displayed at the highest level of detail whereas with increasing distance from the pointer lower levels of detail are used.

In process visualization, data usually originates at concrete 3D locations like a sensor at a certain place or a simulation output with a specific 3D position. Accordingly, the visualization of process data can be organized on the screen such that this relation between the virtual instruments and the related 3D model becomes obvious. In a prototype implementation of F+C process visualization, we first draw the underlying 3D model as a wire-frame rendering. Then, the virtual instruments are shown on top of the wire-frame model at those screen coordinates which correlate to the screen-projection of the corresponding 3D locations of the data sources (see figure 1.15).

Figure 1.14: F+C process visualization: depending on where the user points, the virtual instruments are drawn at a smaller or larger level of detail (from left to right: the pointer is moved from left to right).

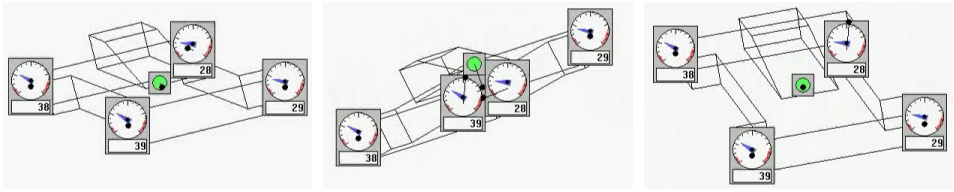


Figure 1.15: 3D anchoring and collision avoidance in F+C process visualization: virtual instruments are placed at the screen-projection of that 3D point which is related to the data origin, for example, a sensor (3D anchoring); to avoid cluttering due to overlapping dials a physically-based spring model is used to relocate instruments such that they do not overlap (collision avoidance).

With such a layout strategy (called *3D anchoring* – the virtual instruments are "anchored" at their respective 3D source locations), it can easily happen that screen projections of sensor locations lie near each other such that a naïve implementation of 3D anchoring would cause overlapping virtual instruments. In our prototype implementation we therefore use a physically-based spring model to resolve for non-overlapping instruments (collision avoidance). See figure 1.15 for three snapshots of this prototype which were taken while the user rotated the 3D model (the black dots, which are connected to the centers of the instruments with black lines, mark the screen-projections of the 3D anchors, i.e., the 2D locations where in the optimal case the virtual instrument should be displayed).

1.5 Interaction

Focus+context visualization requires interaction. Most important, the user needs to have interactive means to focus in a F+C visualization, i.e., he or she needs to steer which parts of the data are to be shown in focus. Accordingly, focussing also includes interactive means to navigate in the visualization, i.e., to change from the visualization of one part of the data (in focus) to another. For applications of F+C visualization, different approaches to focussing are available (see table 1.4 for an overview of some of them), which can be classified with respect to several different aspects. One question is of whether focussing is done directly on the visualization (or not). Another quesion is of whether focussing is done explicitly, i.e., by either directly brushing the data items of interest or naming them explicitly.

focussing	**action**	**selection**	**user**	**sample applications**
brushing	on the view	explicit	active	SimVis [20, 19], ParCoords. [38]
pointing				RTVR [95], ProcVis [88]
selection	off-view			SDOF [64, 65], RTVR [95]
range slider		implicit		SimVis [20, 19], SDOF [64, 65]
querying				SimVis [20, 19]
plot-based		both	passive	SDOF [64, 65]
alerting				ProcVis [88]

Table 1.4: Different approaches to focussing – techniques can be classified according to whether they act directly on the view (or not), their definition is explicit (or implicit), or whether they are triggered by the user (or not). This differentiation is discussed in more detail in section 1.5.

Thirdly, the question of whether the user actively performs the focussing (or the system does it for the user) also classifies the different approaches to focussing.

Most intuitive, *explicit selection* of especially interesting data subsets *directly on the view* results in a (new) specification of the current focus. Prominent examples of this kind of focussing are *brushing* on the one side (as used, for example, in the previously described SimVis system) and *pointing* on the other side (used in F+C process visualization as well as in 3D visualization using RTVR). Similarily, the user can *explicitly focus* by selecting objects through an *off-view list of objects* (as used in volume visualization using RTVR, for example, and the SDOF-visualized map viewer where layers can be selected off-view).

More complex, and a little less intuitive, *implicit selection* also serves for focussing. In the simpler case, selections on invisible axes can be used to describe what currently is most interesting (as also used in SimVis, for example). Alternatively, also complex queries can be used to achieve implicit focussing. As again also used in SimVis – here a separate feature definition language has been developed for the purpose of formally describing what actually is of greatest interest to the user [19].

In addition to methods where the user actively steers which parts of the data are to be visualized in focus, there are other cases, where the system takes over this role. In a tutoring system, for example, a predefined plot describes which parts of the visualization are in focus at which point in time. This kind of focussing was used in a chess tutoring system with the purpose of showing historic competitions to moderately experienced users (see figure 1.16). In F+C process visualization it is possible to let the system assign DOI values according to whether (or not) the values of a certain sensor lie inside (or outside) a certain safety interval. In case of an alert (value out of range) the user immediately is confronted with a F+C display where most visual emphasis is put on the values in question.

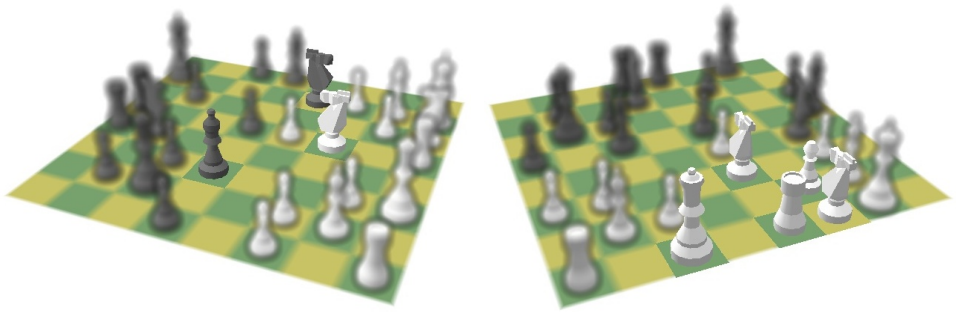


Figure 1.16: SDOF-visualized chess tutoring system: through selective sharpness the system shows which pieces threaten (left image) or cover (right image) the white horse on E3.

1.6 Summary and Conclusions

Taking a step back, we can try to round up the matters discussed up to now and to summarize the most imporant points addressed. In the beginning we started out with a discussion of the well-established approach of *focus+context visualization* (F+C visualization) as known from information visualization. It is usually associated with the process of opening up more space in a visualization for the detailed depiction of some selected parts of the data (those in focus) while still showing the rest of the data in reduced size to provide context information for better orientation and navigation.

This idea of integrating data subsets in focus with their respective context within one visualization also can be found in other fields, especially in scientific visualization. There, however, usually other means than space distortion are used to achieve F+C visualization. In scientific visualization the spatial arrangement of a visualization is tightly coupled with the spatial arrangement of the data origin, e.g., the 3D layout of patients in medical applications or the 3D setup of a flow simulation, and therefore usually resists from uneven distortions. In volume visualization, for example, the use of opacity is varied to achieve F+C visualization of 3D data. In the 3D visualization of segmented data (2lVR), different styles are used to graphically distinguish between objects in focus and their context. Non-photorealistic contour rendering, for example, is very useful for context visualization. In the visualization of data from computational simulation (WEAVE, SimVis), the use of eye-catching colors (within a gray-scale context) also very well serves for F+C visualization. Similarily, the differentiation between a sharp and blurred depiction can yield to F+C visualization (SDOF). All this variety of possible realizations of focus+context visualization yields a more general definition of F+C visualization: focus+context visualization is the uneven use of graphics resources, such as space, opacity, color, frequencies, and style, for visualization (with the purpose to visually discriminate those parts of the data in focus from all the rest).

A discussion of several concrete examples of different types of F+C visualization shows that often several graphics resources are used to do the focus–context discrimination. In F+C volume visualization by the use of RTVR and 2lVR, for example, in some cases all three of opacity, rendering styles, and coloring are varied to achieve F+C visualization (see figure 1.5, left side, for a sample image). In F+C visualization of 3D data from computational flow visualization (SimVis), coloring, opacity, and glyph size are adjusted according to the DOI values of the data to achieve the desired visual discrimination (see figure 1.8, left side, for a sample image). Looking through the glasses of our generalized definition of focus+context visualization at the very broad field of applications shows how useful this approach of graphically integrating data subsets in focus and their respective context within a visualization actually is and how general its applicability is.

In addition to the discussion about different ways to graphically discriminate focus from context, also the interactive aspect of F+C visualization is discussed. Once, focus+context visualization is established, it immediately becomes essential to provide sufficient interactive means for focussing, i.e., to select which parts of the data actually are to be drawn in focus or to navigate through a F+C display. Different options of how to categorize focussing with respect to how it is done (on the view vs. off-view focussing; explicit vs. implicit selection; active/passive user) help to give an overview about available strategies. Another way of looking at focussing, however, is to differentiate user goals: whereas in one case the user wants to see more (details) of certain data subsets ($\rightarrow$ space distortion, style variations), in other cases the user just wants to visually emphasize the graphical depiction of certain parts ($\rightarrow$ opacity, color variations). In again other cases, the visualization goal is to visually attract the user towards a certain subset of the visualization ($\rightarrow$ SDOF, coloring, space distortion). Sometimes, these goals do overlap in an application or are followed upon each other during analysis (first the user needs to be attracted, for example, to a sensor out of range, then the user wants to investigate this sensor data in more detail).

Despite of the principal result that focus+context visualization indeed is generally applicable and useful (almost regardless of the application field), another conclusion of this work is that scientific visualization and information visualization do not lie far apart from each other, but can mutually support each other. There are very good ideas on both sides and visualization systems which integrate approaches from both fields can gain superb advantages over pure SciVis- or InfoVis-solutions.

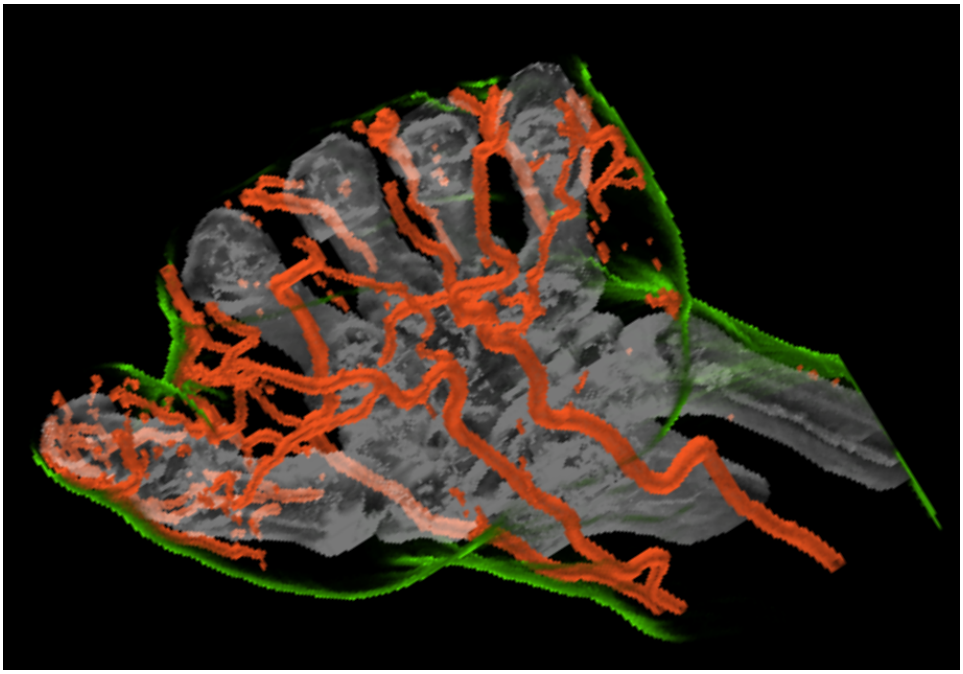
Acknowledgments

This work is based on a lot of related work which would have been impossible without the great contributions of many. To name just a few of them, grateful thanks go to Lukas Mroz, Csébfalvi Balázs, Gian-Italo Bischi, Berk Özer, Anton Fuhrmann, Helmut Doleisch, Martin Gasser, Matej Mlejnek, Markus Hadwiger,

Florian Ledermann, Robert Kosara, Silvia Miksch, Krešimir Matković, Wolfgang Rieger, Wolfgang Meyer, and especially to M. Eduard Gröller and Werner Purgathofer for their patient advice throughout all the years which have been related to this work. For funding, thanks go to K plus, an Austrian governmental funding program, which is supporting VRVis since year 2000 and thus also is responsible that most of the work discussed here actually could be done. For more information, see related papers [17, 19, 20, 38, 39, 40, 41, 64, 65, 66, 88, 95] and `http://www.VRVis.at/vis/`.

Chapter 2

Two-Level Volume Rendering



In 2001, the contents of this chapter have been published in the journal IEEE Transactions of Visualization and Computer Graphics (*IEEE TVCG*) **7**(3), pp. 242–252 (paper "**Two-Level Volume Rendering**" by Helwig Hauser, Lukas Mroz, Gian-Italo Bischi, and Eduard Gröller).

The contents of this chapter (paper) are a result from a collaborative project with Lukas Mroz (one of Helwig Hauser's PhD students), Gian-Italo Bischi (a collaborator from the application side of the project), and Prof. Eduard Gröller (head of the visualization group at the Institute of Computer Graphics and Algorithms, Vienna University of Technology). Related papers (co-authored by Helwig Hauser) are:

- **Two-level volume rendering - fusing MIP and DVR** [40], an earlier version of this paper, published at the IEEE Visualization 2000 Conference –

after the conference this paper was selected as an especially interesting one, which then caused the publication of the paper in an extended version, as presented in this chapter, in IEEE TVCG.

- **Fast Visualization of Object Contours by Non-Photorealistic Volume Rendering** [17], a related paper, also contained in this thesis as chapter 3
- **Interactive Volume Visualization of Complex Flow Semantics** [39], a related paper, also contained in this thesis as chapter 4
- **High-Quality Two-Level Volume Rendering of Segmented Data Sets on Consumer Graphics Hardware** [35], a related paper from 2003 with new extensions and a high-quality and GPU-based implementation
- **RTVR - a flexible java library for interactive volume rendering** [95], a related paper about fast volume rendering
- **Space-Efficient Boundary Representation of Volumetric Objects** [96], a related paper about data compression
- **Interactive High-Quality Maximum Intensity Projection** [97], a related paper about fast MIP
- **Studying Basin Bifurcations in Nonlinear Triopoly Games by Using 3D Visualization** [9], a related paper with more details about the original application

Two-Level Volume Rendering

Helwig Hauser, Lukas Mroz, Gian-Italo Bischi, and Eduard Gröller

Abstract

In this paper we present a two-level approach for volume rendering, i.e., two-level volume rendering, which allows for selectively using different rendering techniques for different subsets of a 3D data-set. Different structures within the data-set are rendered locally on an object-by-object basis by either DVR, MIP, surface rendering, value integration (x-ray-like images), or non-photorealistic rendering. All the results of subsequent object renderings are combined globally in a merging step (usually compositing in our case). This allows to selectively choose the most suitable technique for depicting each object within the data, while keeping the amount of information contained in the image at a reasonable level. This is especially useful when inner structures should be visualized together with semi-transparent outer parts, similar to the focus-plus-context approach known from information visualization. We also present an implementation of our approach, which allows to explore volumetric data using two-level rendering at interactive frame rates.

2.1 Introduction

Irrespective of a certain application, the general goal of visualization is to provide insight into the data of interest. This means that visualization facilitates the process of answering specific user-defined questions about the data under investigation. As a consequence, the question of *what* visualization method is well-suited or even optimal in the context of a specific application is not only dependent on the application data itself, but it also significantly depends on the actual questions of the user.

Especially in cases where the simultaneous visualization of all the data is not possible, e.g., in cases of very large data-sets or data of high dimensionality, the question of *what aspect or what subset* of the data to show becomes a very important decision during investigation. In medical applications, for example, when visualizing three-dimensional data-sets, it is in general not possible to concurrently show all the data. Instead, certain selective representations of the data are used for visualization: iso-surfaces which bound certain subsets of the data, voxels of maximal intensity which are displayed using maximum-intensity projection, as well as several others.

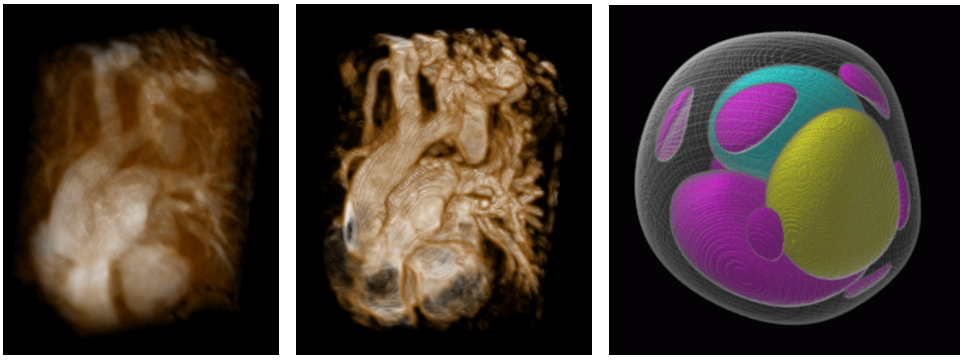
Of course, certain properties of the data under investigation themselves can also imply the use of a specific visualization technique. Given a specific subset of interest, for example, which actually has a sharp boundary that delimits it from

Figure 2.1: Two-level volume rendering of a medical data-set (parts of a human hand): bones are rendered using DVR, surface rendering is used for vessels, and non-photorealistic rendering is used for the skin.

the rest of the data, the use of an iso-surface might be a well-suited choice for visualization. In contrast, another subset of the data, which is characterized by relatively high data-values, might better be visualized by the use of maximum-intensity projection.

In this paper we now present our two-level approach for volume rendering, i.e., *two-level volume rendering* (2lVR), which allows to selectively use individual rendering techniques such as direct volume rendering (DVR), maximum-intensity projection (MIP), iso-surfaces, x-ray-like summation, and non-photorealistic rendering (NPR) for different objects within a common data-set. See Fig. 2.1 for an example, where several different rendering techniques have been used to visualize a medical data-set of a human hand.

An interesting observation about human perception of 3D computer graphics is that viewers seem to inherently separate individual objects from rendered images of 3D scenes. Users tend to see solid objects, which are bounded by opaque or semi-transparent surfaces. Even objects with intrinsic 3D characteristics such as clouds, fire, aso., are perceived as individual entities. Additionally, many visualiza-



(a) (b) (c)

Figure 2.2: Direct volume rendering vs. surface rendering: (a) DVR using a rather smooth TF results in gel-looking objects; (b) DVR using a rather sharp TF simulates surface rendering; (c) surface rendering very well communicates 3D shape.

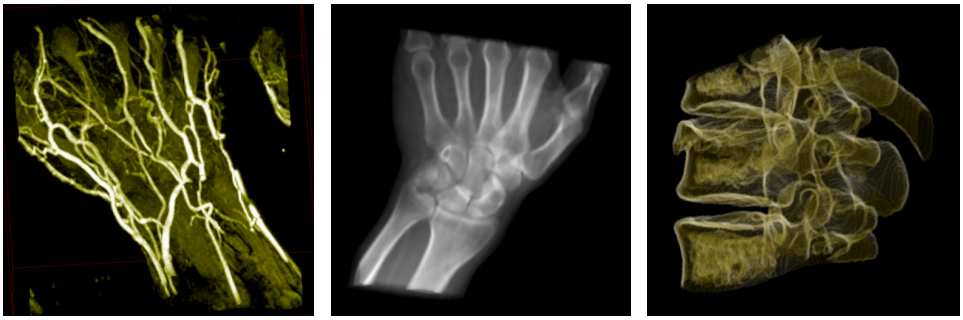
tion goals also require such a notion of data-sets as being composed of individual objects. Medical investigation, for example, often is object-oriented: some specific organ of interest is to be examined whereas all the rest is considered to be context. In the following, we will therefore understand any volumetric data-set as being composed of semantically distinguished subsets of the data, which in turn we will call the *objects* within the data-set.

In this paper, we also demonstrate how two-level volume rendering is used as a focus-plus-context (F+C) technique in volume rendering, comparable to similar solutions which are well-known from information visualization. Especially interesting subsets of the data, which are considered to be "in focus", are rendered, for example, using iso-surfacing or DVR, whereas the rest of the data, i.e., the context in this case, is depicted using semi-transparent MIP or non-photorealistic rendering.

2.2 Experiences with DVR, MIP, etc.

Before we actually describe the new approach of two-level volume rendering, we first give a brief review of some experiences which we gained from previous work with different volume rendering techniques.

One of the standard techniques for displaying volumetric data is direct volume rendering [54, 78], which is based on the sorted composition of visual properties along viewing rays. The most important parameter of DVR probably is the so-called transfer function (TF) which describes the mapping of data-values to optical properties. Depending on the transfer function in use, mainly two typical forms of object appearances can be distinguished: either objects look like semi-transparent gel due to a rather smooth TF, or object boundaries are depicted similar to iso-surfacing as a result of a rather sharp or even binary TF – see Figs. 2.2(a) and



(a) (b) (c)

Figure 2.3: Three alternative methods for volume rendering: (a) MIP displays structures of maximal importance; (b) value integration results in x-ray-like images; (c) non-photorealistic rendering may enhance contours, for example.

(b) for sample results. In general, we experienced that results from DVR usually feature good impression of 3D shape, especially if photorealistic shading is used. Contrarily, occlusion sometimes becomes a significant problem of DVR, which is mainly because pixel-values do not only depend on the local TF mapping, but also on the number of samples to be composited. The latter is not under control of the TF and can drastically vary among all viewing rays, consequently resulting in sometimes large variations with respect to object transparency.

Due to the fact, that it is often very difficult to intuitively set up a proper transfer function, given a specific visualization goal in mind, the design of transfer functions has become a research topic on its own [42, 4, 85, 59, 62]. Producing useful images with DVR is even more difficult if there is no close correspondence between differences in data-values and the discrimination of objects. MRI data-sets, for example, typically contain large differences in data-values for objects of similar type.

Two solutions have been proposed to help in such a case: one is to use an alternative rendering technique for depicting the features of interest – for solutions different to DVR see the paragraphs below. Another approach is to first apply segmentation [130] to the data-set, which is followed by selective rendering of the separated objects, for example, by the use of different transfer functions [126]. Various techniques have been proposed for automatic and semi-automatic segmentation of volumetric data, such as thresholding, water-shed, etc. In this paper we do not focus on segmentation itself, but very well make use of optional segmentation information, if present as a result of a preprocessing step. As two-level volume rendering inherently depends on the notion of a 3D data-set which is composed of distinguishable objects, our implementation of 2lVR also provides a threshold-based segmentation technique for all cases, where no segmentation information is given a priori.

Maximum-intensity projection [142, 112, 99], which is used to display the most important data-values along viewing rays, features a clearly different object ap-

pearance in comparison to DVR. MIP images are usually quite sharp as only one data-value is shown per pixel. Also, assuming that importance is directly correlated with data-values, the more important some data-values are, the more likely it is that they are actually visible in the resulting image – the amount of important information which is hidden is minimal. However, due to the fact that neighboring pixels do not necessarily represent spatially coherent data-values, MIP images tend to look rather flat. This undesirable effect of MIP actually is an inherent property of this rendering method, and can only be diminished to a certain extent by variations of MIP such as local MIP [116], depth-shaded and/or animated MIP, etc. For an example see Fig. 2.3(a).

Surface-based approaches – in contrast to DVR and MIP – are suitable in situations, where sharp object boundaries are present in the data-set and the interior structure of the object is not relevant for visualization. Basically, there are two approaches to extract and render iso-surfaces. One class of techniques deals with the explicit computation of a geometric representation of the iso-surface – the marching cubes algorithm [83] is a well-known example. The second class deals with depicting the surface by means of a transfer function [78]. DVR and surface rendering can also be combined into a hybrid technique depicting both, truly volumetric information, and objects defined by polygonal models [67]. In this paper we integrate the rendering of object surfaces by directly depicting surfaces from the volumetric data [100] – see Fig. 2.2(c) for an example –, not dealing with any polygonal model at all. This is also in contrast to volumetric ray tracing [120], where actual ray tracing is used to compute high-quality images from 3D scenes which are composed of volumetric objects and polygonal models.

We have also implemented value integration along viewing rays through volumetric data for image synthesis. Images, which have been rendered using this technique – see Fig. 2.3(b) for an example –, simulate an x-ray-appearance of the data. This is due to the fact, that value integration is nothing else than the inverse operation compared to data reconstruction within a 3D scanning device such as a CT.

Recently, non-photorealistic rendering (NPR) methods have been proposed for volumetric data [22, 17], which, for example, depict object contours in a view-dependent manner. Here, a great variety of object appearances can be achieved, all useful for different visualization goals. In our case, we found the depiction of object contours by the use of NPR to be especially useful, also as an alternative to traditional rendering methods as described above. Fig. 2.3(c) shows an example of NPR rendering of volumetric data.

Good visualization strongly depends on *what data* is visualized, *what structure* this data consists of, as well as on the *visualization goals* of the user. Depending on these prerequisites several useful approaches exist, and individual decisions (what rendering method to choose) have to be made for specific applications. Two-level volume rendering as described in this paper allows for selectively combining different rendering techniques for different objects within a common data-set.

2.3 Two-level volume rendering

After the preceeding discussion of pros and cons of several volume rendering techniques, we now present two-level volume rendering as a useful way of combining different rendering methods with respect to their respective advantages.

2.3.1 Object discrimination

An important goal for two-level volume rendering not only was to provide the ability for using different rendering techniques for different objects within a 3D scene, but also to come up with visualization results which actually allow to easily discriminate the individual objects from each other. From previous work [51] we know, that displaying multiple semi-transparent surfaces on top of each other, quickly imposes significant problems with the identification and perception of the individual objects. Usually just two or three layers are easily distinguished from rendered images. Therefore, a key feature of two-level volume rendering is that pixel-values are composed of a very limited number of values only.

2.3.2 Two-level rendering

For achieving our goal of generating images which are composed of a few, but meaningful object representatives per pixel, we decompose the rendering calculations into two levels, i.e., local and global rendering. For every pixel, we theoretically investigate a viewing ray through the data and detect what objects are intersected. For every object intersected, a single, representative value is computed (by the use of local rendering). These object representatives are finally composed to yield a pixel value by combining them, usually by the use of DVR compositing (global rendering). Only a small number of data-values are combined for a pixel (one per object), therefore limiting the number of samples which finally make up one pixel-value.

For more specifically explaining two-level volume rendering, it is useful to utilize the model of ray casting. Of course, an implementation of our technique is not bound to this image-order approach. In this paper, for example, we will later present a fast object-order implementation of two-level volume rendering.

For calculating the value of a pixel within the resulting image, we may assume a viewing ray to be cast right into the volumetric data-set (cf. Fig. 2.4). A 3D segmentation mask determines for any voxel within the data which object it belongs to. Therefore, also the viewing ray may be interpreted as being decomposed into several segments, depending on what objects are intersected by the ray. We now utilize this ray segmentation as follows:

While traversing a viewing ray – for example, back-to-front – two tracks of rendering are processed in parallel. As long as one segment of the ray is traversed,

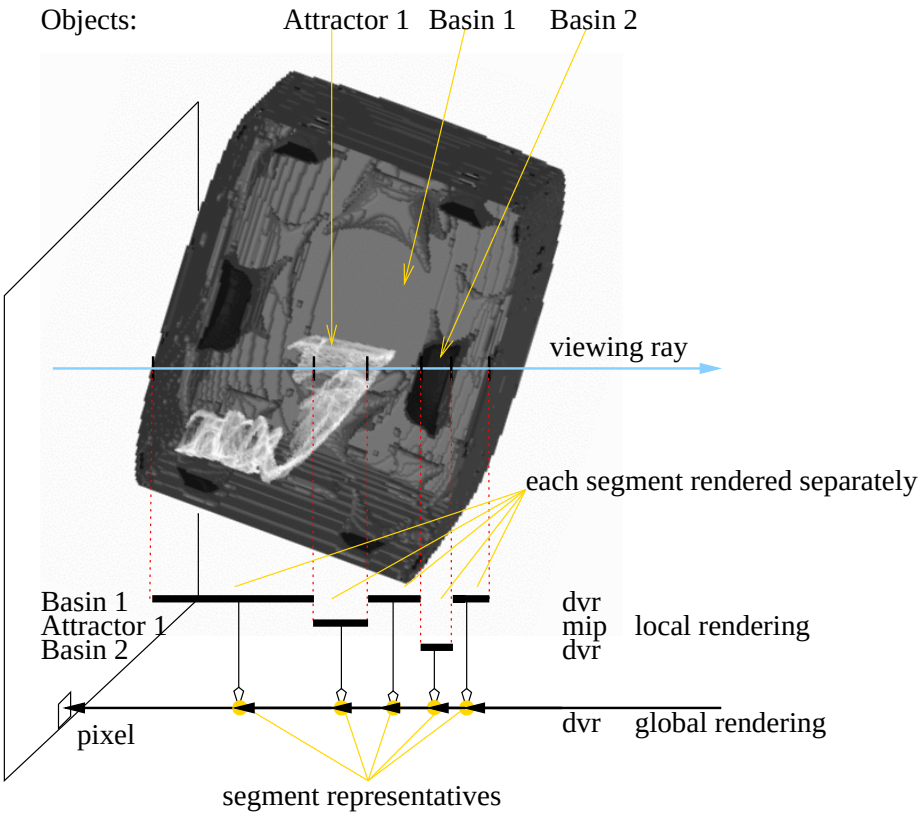


Figure 2.4: Object segmentation implicitly yields viewing rays to be partitioned into segments (one per object intersection).

i.e., as long as the viewing ray traverses one individual object within the data, local rendering is performed to compute an object representative associated to the segment (rendering at the object level). Individual rendering methods can be used for different segments, depending on what objects are traversed. Fig. 2.5, for example, was rendered with DVR using a sharp TF along segments through bones, surface rendering for vessels, and MIP for the skin.

At those points along the ray, where the object classification changes, i.e., at the points where viewing rays cross object boundaries, update steps in the global rendering track are performed. We usually use DVR-compositing on the global level. The only exception, where we could experience that MIP is useful instead, is if all objects in the data-set are rendered by the use of MIP themselves, also. In contrast to standard MIP, this “MIP of MIP” approach allows to easily distinguish between different objects within the scene, as different transfer functions and thus different colors can be assigned to different objects.

Figure 2.5: Combining DVR, surface rendering, and MIP: whereas the bones have been rendered using DVR (sharp TF), and the vessels have been rendered using surface rendering, the skin was depicted using MIP.

2.3.3 Focus-plus-context

Due to the ability to selectively use different rendering techniques for different objects within a 3D data-set, two-level volume rendering strongly supports applications which are of F+C style: depending on whether objects are selected to be "in focus" or not, their visual appearance can be different. Whereas objects "in focus" may feature significant opacity, for example, objects which are considered to be context rather act as semi-transparent reference. See Fig. 2.5 again for an example, where bones and vessels are considered to be "in focus", whereas the skin just acts as the context in this application. Table 2.1 gives an overview about recommended F+C configurations when two-level volume rendering is used. Depending on the internal structure of objects, different rendering techniques seem to be well-suited. Objects "in focus", for example, can very well be rendered using rather opaque surfaces in case of spatially contiguous objects (Fig. 2.5, for example). Objects of interest, which are characterized by a complex or even fractal inner structure, can

	focus	context
DVR	± rather opaque, surface-like	± rather transparent, gel-like
MIP	+ complex focus, high contrast	+ very uniform transparency
surfaces	+ rather opaque	+ semi-transparent
x-ray	−	± inner context
NPR	± only as add-on	+ sparse contours

Table 2.1: F+C configurations for Two-level Volume Rendering

very well be depicted by the use of MIP (Fig. 2.9(c), for example). In case of context visualization, the use of MIP, quite transparent surfaces, and NPR proved to be very useful. MIP is useful, because it features quite homogeneous transparency throughout an entire object (Fig. 2.5, for example). Shape properties of context objects are well-communicated by the use of semi-transparent surfaces. NPR can be used to sparsely apply shape cues of outer context objects (Fig. 2.1, for example).

2.3.4 Technical details

Since the model of two-level volume rendering is based on ray casting, an implementation as image-order technique would be straight-forward. Nevertheless, the idea of two-level rendering has originally been developed to aid exploration of data from the field of complex dynamical systems. Interactivity is crucial for exploring and investigating data, and especially for finding proper settings for optical properties of objects. Thus we present a fast object-order implementation [94], based on shear-warp rendering [70], which allows for interactive visualization of medium-sized data-sets. For performance reasons, no interpolation is done within the volume, each voxel is projected onto exactly one pixel of the base plane. During the warp step, bilinear interpolation is used, also considering the scaling component of the projection matrix.

To allow for efficient skipping of empty space in-between objects, each object within the investigated data is stored separately as an array of all its member voxels. For each voxel, its position and attributes, e.g., data-value and gradient information, are stored. The list is replicated and stored separately for each principal viewing axis (x, y, and z) with voxels sorted by the principal viewing coordinate. For the sake of clarity, the principal viewing direction will be called z from now on.

All voxels of an object which share the same z coordinate are grouped within a so-called `RenderListEntry`, the value of z is stored with the `RenderListEntry` instead with all the voxels (Fig. 2.6). For rendering, the `RenderListEntry`s of all objects are merged into a single `RenderList` sorted by z.

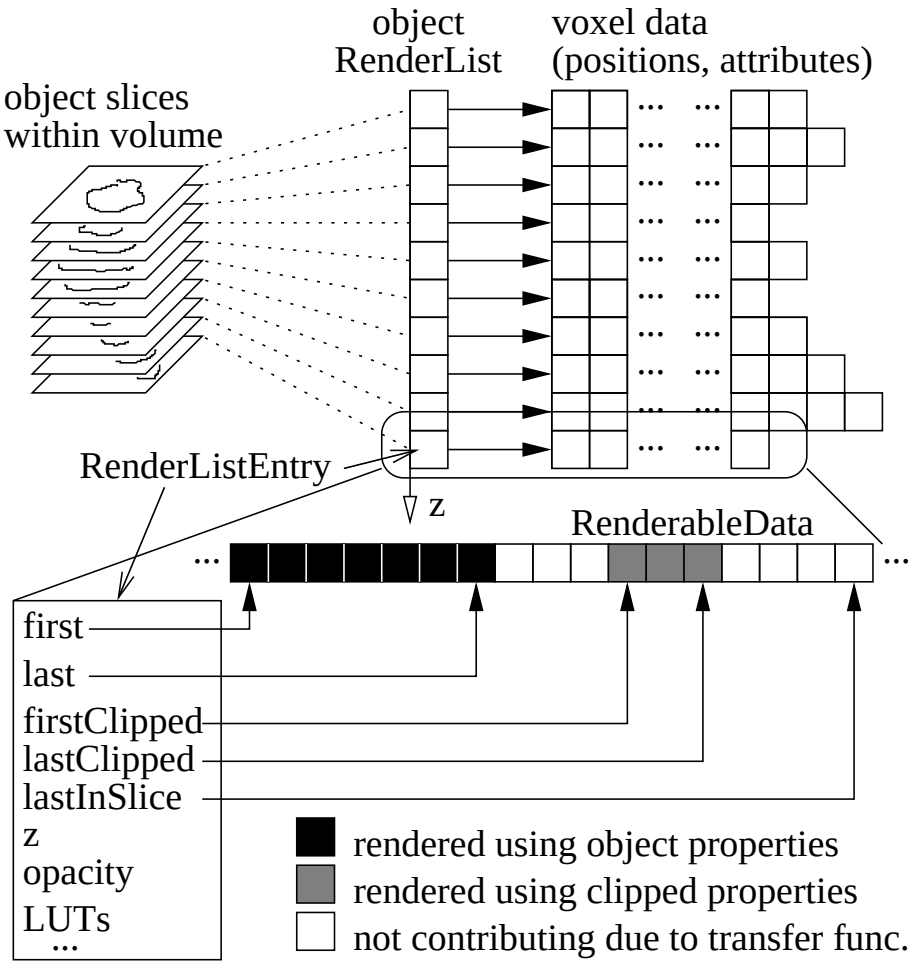


Figure 2.6: Storage scheme for two-level volume rendering.

```
PROCEDURE ComputeRenderList:

FOR pvd (principal viewing dir.) BEING x, y, z:
{ FOREACH obj (object in the scene) DO:
    FOREACH val (depth value regarding pvd) DO:
      LET RenderListEntry[pvd,obj,val] = set of
          all voxels of obj. obj with depth val
  MERGE ALL RenderListEntry[pvd,.,.] into a
    RenderList[pvd] (sorted by val)
}
```

Processing this list sequentially during rendering results, for example, in a back-to-front traversal of the scene, even though the voxels within one `RenderListEntry`, i.e., which are at equal depth from the base plane, are pro-

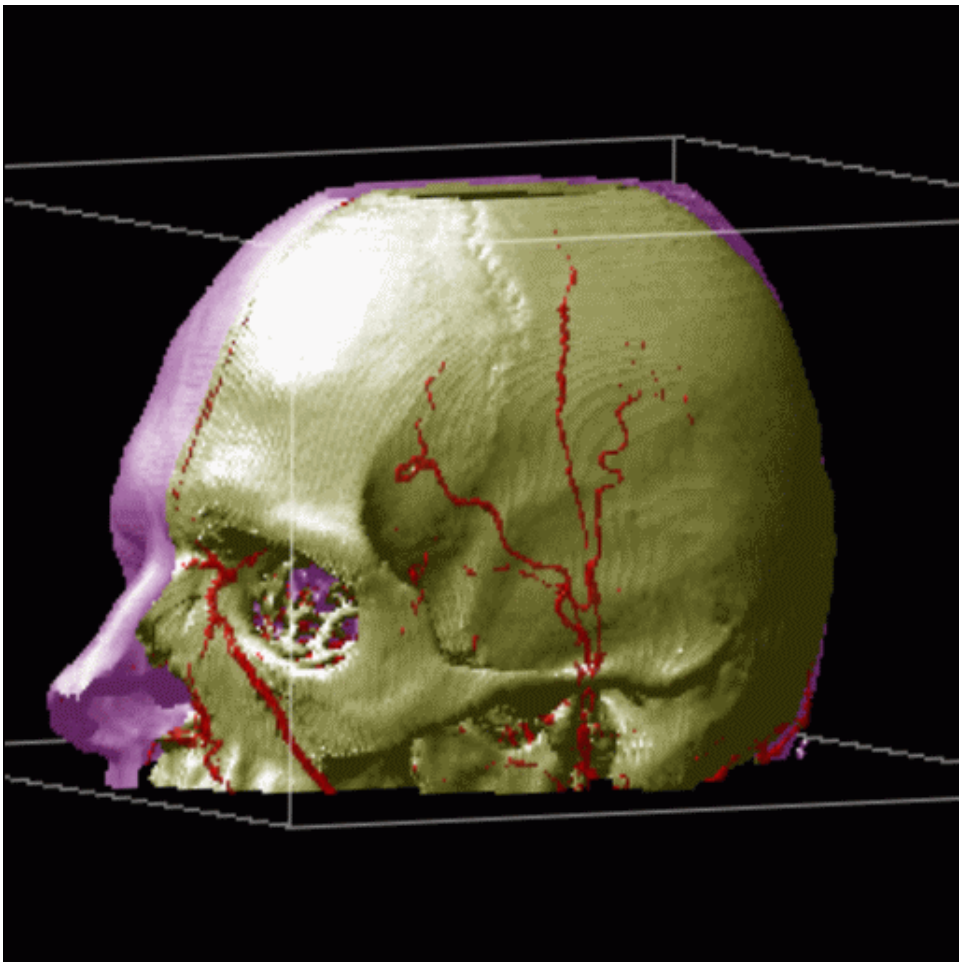


Figure 2.7: Selectively applied clipping planes allow further insight.

jected in an arbitrary order. By storing only voxels which actually belong to objects, empty space in-between is skipped automatically without any effort during rendering.

For implementing two-level volume rendering, two sets of buffers are used for the base plane. One buffer (local object buffer) is used for performing rendering within an object, while a global buffer is used to perform inter-object rendering. In addition to pixel values, each pixel of the object buffer additionally stores the unique ID of the currently front-most object. If a voxel is projected onto the base plane, its ID is compared with the stored ID in the object buffer. If both IDs match, the value in the object buffer is updated using an operation which corresponds to the local rendering mode of the object. If the ID of the voxel differs from the ID of the pixel in the buffer, the viewing ray though this pixel must have entered a new object. The content of the object buffer pixel is combined with the corresponding global buffer pixel using an operation which depends on the global rendering

strategy (usually DVR). Then the object buffer pixel is initialized according to the voxel of the new object and the new local rendering mode.

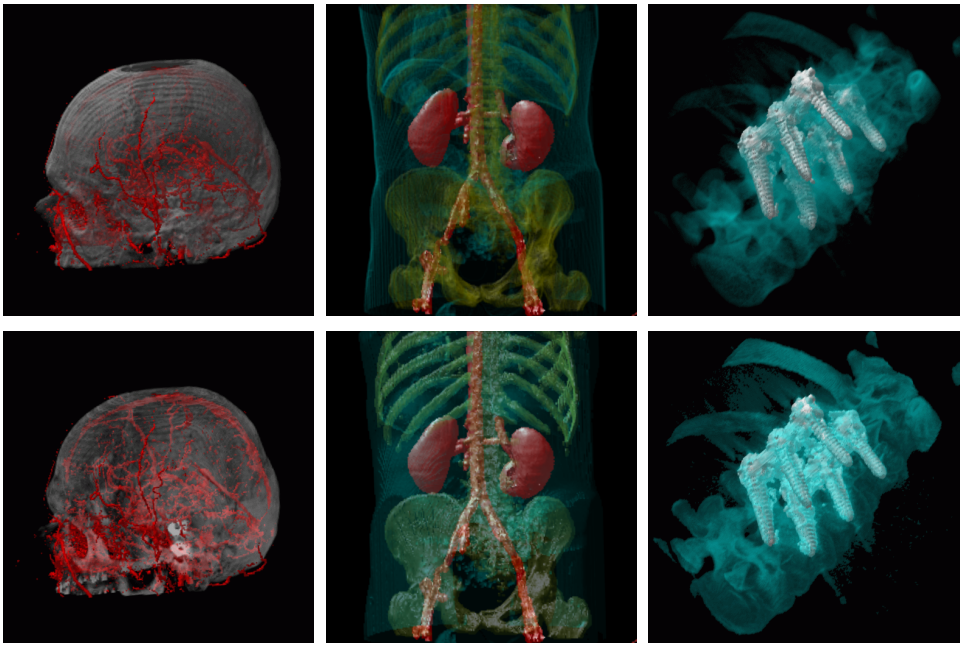
After all voxels have been projected, the contribution of the front-most segment at each pixel has to be included by performing an additional scan of the buffers and merging the segment values left in the local buffer into the global buffer.

As the application of clipping planes is a widely used method for enhancing volume visualization, support for this technique has also been included in our object storage scheme (see Figs. 2.7 and 2.9(c)). The voxel lists used for rendering allow a very efficient implementation of arbitrary cutting planes. As the order in which voxels of the same depth are rendered to the base plane is irrelevant, voxels removed by the application of a cutting plane can be simply moved to the end of the voxel array belonging to a `RenderListEntry`. An end-mark is used to indicate the last visible voxel within a `RenderListEntry` (Fig. 2.6). Removed voxels which are stored after this mark can be ignored during rendering without slowing down the process. Instead, we also enable the clipped parts of objects to be rendered using another rendering techniques (cf. 2.10(c)).

```
PROCEDURE TwoLevelRendering:

Choose pvd (according to current viewing dir.)
Init(localBuf,0); Init(globalBuf,0)
FOREACH RenderListEntry IN RenderList[pvd]:
{ object = RenderListEntry.objectID
  FOREACH voxel IN RenderListEntry.activeSet:
  { pixel = projection(voxel,pvd)
    IF localBuf(pixel).object = object THEN:
      Update(localBuf(pixel),voxel)
    ELSE:
    { Update(globalBuf(pixel),localBuf(pixel))
      Init(localBuf(pixel),voxel)
} } }
FOREACH pixel DO:
  Update(globalBuf(pixel),localBuf(pixel))
image = Warp(globalBuf)
```

For lighting of objects the Phong shading model is employed. The gradient vector at each voxel is pre-calculated during the preprocessing step [102], converted to polar coordinates and quantized to 12 Bit for limiting the storage requirements. During rendering, this representation of the gradient is used to access a look-up table with shading information. The shading table includes the influence of ambient, diffuse, and specular components of the lighting model. It is quickly recalculated for each new viewing or light position and also after a change of material properties.



(a) (b) (c)

Figure 2.8: Comparisons between standard DVR (first row) and two-level volume rendering (second row). (a) Two-level volume rendering allows to show blood vessels within the head, even near the skull. (b) Using MIP for the visualization of context (bones, skin) the object of interest (blood vessels plus stenosis) becomes well visible from all viewing directions. (c) Again MIP proves to work fine for context visualization.

Opacity values for compositing voxels and objects are derived from two sources: first, the ability to change the opacity of a whole object turned out to be useful when adjusting viewing parameters. Secondly, the object-wide opacity can be modulated by a per-voxel opacity which depends on some property of the voxel, like for example data-value, gradient magnitude, or the distance to some other object (see Sect. 2.4.2).

2.4 Applications and results

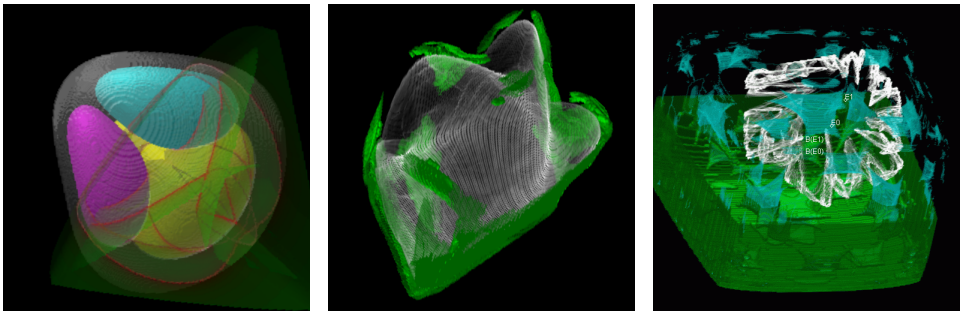
Two-level volume rendering, as described above, can be used in various fields of applications. In this paper we describe two applications and show how two-level volume rendering improves the results.

2.4.1 Medical data visualization

Visualization of medical data is one of the most important application fields of volume rendering. Several rendering techniques are used to depict different features of the data. Pure surface rendering, for example, is used when actual boundaries of objects within the medical data-set are to be shown, e.g., the surface of bones or the skin (cf. Fig. 2.7). Similar to surface rendering, DVR is often used to render tissue transitions within medical data-sets (see Fig. 2.2(b)). Through photorealistic shading, DVR-images as well as surface depictions very well communicate 3D shape. Furthermore, DVR results feature a usually good impression of object inter-relations and depth information. MIP, on the other hand, is very useful for visualizing rather complex structures, like blood vessels, organs, etc. Images usually are less over-loaded, compared to DVR, and focus on the important parts of the data. In Fig. 2.3(a) we show blood vessels together with lower-valued bones by the use of MIP. In addition to DVR and MIP, simple value integration (x-ray-like results) becomes useful when object thickness should be displayed. Also, x-ray-like object appearance might be familiar to user (see Fig. 2.10(a) for an example). Finally, non-photorealistic rendering of medical data-sets has been proven to allow for showing shape cues without the consumption of lots of screen space (cf. Figs. 2.1, 2.10(b), and 2.10(c)).

However, also some disadvantages can be experienced when working with the above mentioned methods: DVR images sometimes tend to be over-loaded with content (blurred images). Due to the convolution character of the compositing step, and the high sensitivity of DVR on the transfer function in use, it happens easily that too many data-values are merged into the final image. The use of gradient information as another input to the transfer function [78] partially improves this situation. For objects with a complex interior structure, nevertheless, it is difficult to avoid this accumulation problem. MIP images, on the other hand, usually lack depth information. This is due to the weak inter-pixel correlation of neighboring viewing rays. Data-maxima often are detected at significantly different depth locations along neighboring rays. Consequently, MIP images usually look rather flat, and view-point variations (via animation or user-interaction) are necessary to re-generate the 3D impression. Also, no occlusion is considered, showing objects, which are actually beyond each other, in an arbitrary order. With standard MIP usually just one structure is visualized, i.e., the object of interest lacks context information.

Two-level volume rendering allows to overcome these difficulties to a certain extent. Context information, for example, can be displayed by the use of MIP, and combined with DVR-rendered inner structures. The MIP-representation of the outer hull avoids the over-loading problem of DVR as MIP hulls feature rather uniform transparency. Through DVR-rendering the 3D shape of inner structures becomes more visible. In Fig. 2.8(a), lower image, the skull is rendered by the use of MIP, producing rather continuous transparency throughout the image. This



(a) (b) (c)

Figure 2.9: Dynamical systems visualized by the use of two-level volume rendering.

enables insight to the blood vessels, which are represented by the use of DVR. Two further comparisons are also depicted in Fig. 2.8. In general, we found it useful to depict the context of DVR-rendered inner structures by the use of MIP [40]. We also experienced that in the case of rather complex structures, MIP is working fine, whereas for objects with limited spatial frequencies usually DVR works fine.

2.4.2 Dynamical system visualization

We also successfully applied two-level volume rendering to discrete dynamical systems in 3D. We are interested in the long-term evolution of a discrete dynamical system, which is given as a set of three difference equations $\mathbf{x}_{t+1} = \mathbf{f}_{\mathbf{p}}(\mathbf{x}_t);\ \mathbf{x}_t,\ \mathbf{x}_{t+1} \in \Omega \subseteq \mathbf{R}^3$, where t denotes the (discrete) time of evolution and $\mathbf{p} \in \Pi \subseteq \mathbf{R}^m$ is a vector of parameters of the dynamical system.

A trajectory (or evolution over time) of the system is defined as the sequence of states $\{\mathbf{x}_t\}_{t\geq 0}$, starting from a given initial condition $\mathbf{x}_0$, by the repeated application (iteration) of the map $\mathbf{f}_{\mathbf{p}}$, i.e., $\mathbf{x}_t = \mathbf{f}^t_{\mathbf{p}}(\mathbf{x}_0)$. Investigating the long-term behavior of such a dynamical system, we are first interested in attracting sets which are present for a specific parameter setting $\mathbf{p}$. Attracting sets might be quite simple, e.g., single points, periodic cycles of limited period, or quasi-periodic trajectories which fill a closed curve, or more complex objects such as chaotic sets, also called strange attractors [33].

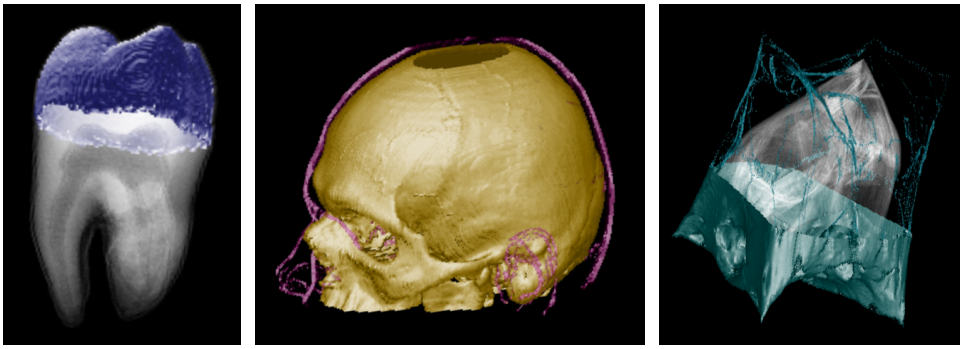
Not only the attractor as the limit of evolution, but also its basin of attraction should be shown. A basin is equal to the set of all starting configurations $\mathbf{x}$, which (in the long-term) finally converge to the corresponding attractor. Similar to the attractors themselves, also the basins may have either simple boundaries, just a box, for example, or complex boundaries which may be formed by many disconnected portions and sometimes may even have a fractal structure. A basin is a set which spatially includes the corresponding attractor. When several attractors

coexist, it becomes very important to visualize the boundaries which separate the corresponding basins together with the attractors which are nested inside them.

In our case, two-level volume rendering is successfully used in the scientific investigation of two discrete systems of international interest. The first one is a three-dimensional system which models a triopoly, i.e., an economic system where three producers share the market of a given product – for example, a competition among three companies in an international context. The question is which company will finally dominate the market, given a particular initial condition (the initial productions of the three companies) and given a certain set of so called reaction-functions $\mathbf{f_p}$ which represent the optimal choices [2]. In Fig. 2.9(a) two-level volume rendering was used to visualize four basins of attraction, three enclosed basins using DVR, and one surrounding basin using MIP. Additionally a so-called critical surface is depicted using MIP. The color of the surface corresponds to the distance to the closest basin boundary. Critical surfaces, which are sometimes responsible for global bifurcations, are basically defined to be the 3D locus where the determinant of the Jacobian matrix of $\mathbf{f_p}$ vanishes.

Another system we are investigating is a three-dimensional non-invertible quadratic, i.e., a second degree map, which can be considered as an extension of a broadly used two-dimensional quadratic map [92]. Such two-dimensional maps have often been used to explain global bifurcations. These are qualitative changes occurring in the structure of the attractors and/or of their basins when the parameters are changed, as a consequence of contacts between particular curves (the so-called contact bifurcations [92]). Two-level volume rendering for the visualization of basins and their associated attractors eases the investigation of bifurcation events in three-dimensional space. A fast visual inspection allows to relate bifurcations to contacts between particular surfaces whose analytic equation is generally unknown. Interactive 3D visualization is crucial for the detection of global bifurcation events in 3D. Fig. 2.9(b) shows, for example, an attractor which is close to a contact with the boundary of its basin. The attractor has been rendered using MIP, the boundary is depicted using DVR. The opacity of the boundary voxels is modulated according to the distance to the closest point of the attractor, clearly showing areas of potential contact. In fact, the use of cutting planes or simple projections does not work in such cases.

In Fig. 2.9(c) MIP was used to render a chaotic attractor. The inner structure of the fractal object becomes visible quite well. Another attractor, also apparent within this parameter setting, is a two-cycle (two points), represented by label "E1". Both basins of attraction are rendered by the use of DVR to better show their 3D shape. They subdivide phase space in a complex way: the basin of attraction corresponding to the simple attractor is a non-connected set, whose distribution in space has a fractal structure.



(a) (b) (c)

Figure 2.10: Further results computed with two-level volume rendering: (a) value integration (x-ray-like depiction) and surface rendering, (b) surface rendering and NPR (c) surface rendering, NPR, and MIP.

2.4.3 User interaction

Two-level volume rendering, as presented in this paper, has been implemented in an interactive Java-based [124] tool. The applet provides the user with a set of interactive tools for exploring and visualizing the data. After loading a data-set, it is decomposed into distinct objects. Segmentation information is always present for data originating from dynamical system computations. Medical data-sets however may not necessarily contain segmentation information. Non-segmented medical data is automatically decomposed using a simple, threshold-based segmentation.

All operations which affect viewing parameters and optical properties of objects are performed interactively. Each object can be either shown or hidden for selective visualization of specific features within the scene. In addition to the global rendering mode, the object-level rendering mode as well as the opacity can be selected separately for each object. Also, the color of objects can be set independently of the range of values of the object's voxels. In our implementation we use color mainly to distinguish between objects within the scene, whereas DVR and MIP mainly work on opacities, color saturation and lightness, as well as shading information. Additional information (like gradient magnitude or distance to another object) can be used to modulate either the color or the opacity of an object's voxels. Cutting planes at certain values of x, y, and z can be applied to subsets of objects and manipulated in real-time.

Using the mouse to directly interact with the scene, further visualization parameters can be influenced: viewing position, zoom factor, position of the light source, material properties, and opacity transfer function. Additionally the windowing metaphor, which is widely employed in medical imaging applications, is used to adjust contrast and emphasize specific data ranges within the color transfer function.

figure	*volume size*	*rendering time*
2.3(a)	$256 * 256 * 100$	0.09 s
2.8(b)	$202 * 152 * 255$	0.18 s
2.8(c)	$256 * 256 * 241$	0.17 s
2.9(c)	$256 * 256 * 256$	0.17 s

Table 2.2: Rendering times.

As empty regions within the volume are not stored in our implementation, we are able to handle 4D data (or even data of higher dimensionality) in a memory efficient way. 4D data is obtained from dynamical system research as parameter variation produces several (10–100) distinct volume data-sets.

2.4.4 Results

Table 2.2 shows the rendering times of some of the figures from this paper measured on an AMD Athlon 600 PC. All images are 512x512 in size. Most of the images are single frames from animation sequences which can be found at the web page of this work (http://www.VRVis.at/vis/research/two-level/), where also further results are shown. Fig. 2.10 shows three of them: surface rendering and value integration (x-ray-like depiction) were used for the tooth data-set, surface rendering together with NPR resulted in the image of the human head, whereas surface rendering, MIP, and NPR were used to depict the data from a dynamical system simulation.

2.5 Summary and conclusions

In this paper we presented the idea of merging several different volume rendering techniques into a two-level volume rendering method. Segmentation information is utilized to apply individual rendering techniques to different objects, which additionally are merged into a resulting image through a global rendering step.

We applied our new technique in two areas, namely medical data visualization and the visualization of dynamical systems. DVR-rendered objects of interest, for example, which are displayed with MIP-rendered context objects proved to be useful in both cases.

The decision what rendering method to choose strongly depends on what data is to be visualized, what structure this data consists of (complex, rather simple, etc.), and what the user actually wants to see within the data. Often, even one data-set contains objects, which differ significantly with regard to their inner structure or boundary surface. Our approach allows to take the most appropriate volume

rendering method on a object-by-object basis, and globally merge the subsequent result to the final image.

We also experienced that certain visualization setups are quite common to various application fields. Focus-plus-context, for example, which is well-known from information visualization, also shows up in visualization goals for medical data, or data of dynamical systems. Users want to peer inside 3D data to investigate some inner structures, but they also would like to keep surrounding objects integrated within the visualization, for spatial reference.

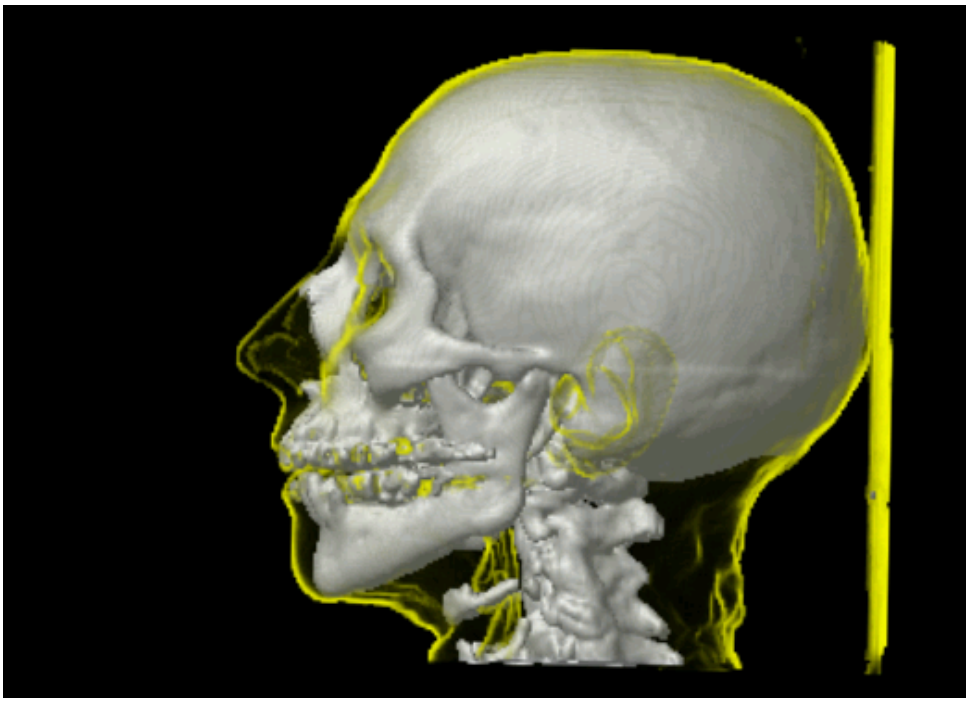
Also, being able to implement this new approach of two-level volume rendering as an interactive tool on PC-hardware, allowed to experience the importance of interactive investigation of volumetric data. Being able to vary viewing parameters as well as visualization attributes interactively significantly helps to generate quite useful results.

Acknowledgments

Acknowledgments go to VisMed, an FFF-funded project (`http://www.vismed.at/`), for the collaboration in the field of medical visualization. We want to thank the radiologists at the Innsbruck University Clinic for providing the medical data-sets shown in this paper. Parts of the work presented in this paper have been carried out as part of the BandViz project (`http://bandviz.cg.tuwien.ac.at/`), which is supported by FWF under project number P 12811. Further parts of this work have been carried out as part of the basic research on visualization at the VRVis Research Center in Vienna, Austria (`http://www.VRVis.at/vis/`), which is funded by an Austrian governmental research program called Kplus.

Chapter 3

Fast Visualization of Object Contours by Non-Photorealistic Volume Rendering



In 2001, the contents of this chapter have been published in the journal Computer Graphics Forum (*Blackwell CGF*) **20**(3), i.e., in the Proceedings of the EUROGRAPHICS 2001 Conference (*EG 2001*), pp. C 452 – C 460 (paper "**Fast Visualization of Object Contours by Non-Photorealistic Volume Rendering**" by Csébfalvi Balázs, Lukas Mroz, Helwig Hauser, Andreas König, and Eduard Gröller).

The contents of this chapter (paper) are a result from a collaborative project with Lukas Mroz (one of Helwig Hauser's PhD students), Csébfalvi Balázs (another PhD student), Andreas König (manager of the project which funded this piece of research), and Prof. Eduard Gröller (head of the visualization group at the Institute of Computer Graphics and Algorithms, Vienna University of Technology). Related papers (co-authored by Helwig Hauser) are:

- **Two-Level Volume Rendering** [41], a strongly related paper, also contained in this thesis as chapter 2, and **Two-level volume rendering - fusing MIP and DVR** [40], an earlier version of this paper
- **Interactive High-Quality Maximum Intensity Projection** [97], a related paper about fast MIP
- **RTVR - a flexible java library for interactive volume rendering** [95], a related paper about fast volume rendering
- **Space-Efficient Boundary Representation of Volumetric Objects** [96], a related paper about data compression
- **High-Quality Two-Level Volume Rendering of Segmented Data Sets on Consumer Graphics Hardware** [35], a related paper from 2003 with an GPU-based implementation

Fast Visualization of Object Contours by Non-Photorealistic Volume Rendering

Csébfalvi Balázs, Lukas Mroz, Helwig Hauser, Andreas König, and Eduard Gröller

Abstract

In this paper we present a fast visualization technique for volumetric data, which is based on a recent non-photorealistic rendering technique. Our new approach enables alternative insights into 3D data-sets (compared to traditional approaches such as direct volume rendering or iso-surface rendering). Object contours, which usually are characterized by locally high gradient values, are visualized regardless of their density values. Cumbersome tuning of transfer functions, as usually needed for setting up DVR views, for example, is avoided. Instead, a small number of parameters is available to adjust the non-photorealistic display. Based on the magnitude of local gradient information as well as on the angle between viewing direction and gradient vector, data-values are mapped to visual properties (color, opacity), which then are combined to form the rendered image (MIP is proposed as the default compositing stragtegy here). Due to the fast implementation of this alternative rendering approach, it is possible to interactively investigate the 3D data, and quickly learn about internal structures. Several further extensions of our new approach, such as level lines are also presented in this paper.

3.1 Introduction

One important reason for visualization to actually be a useful tool for investigating large and complex data-sets is that it allows to view the data of interest by means of different display methods. Especially in the field of volume visualization, where the user aims to peer inside 3D objects, this ability to vary the ways of visualizing the interior of a 3D data-set, is very useful.

Traditionally, so called photorealistic approaches dominate the field of volume visualization. The representation of objects within a 3D data-set by means of iso-surfaces, for example, which themselves are approximated by a large collection of polygons each (cf. marching cubes [83]), as well as the use of transfer functions to map density values to visual properties, which in turn are composed to the final image by the use of alpha-blending along viewing rays (cf. direct volume rendering [21, 78]), are just two prominent examples.

One special challenge of most volume rendering approaches is the specification of parameters such that the resulting images are in fact providing useful insights

into the objects of interest. With iso-surface rendering, for example, the specification of proper iso-values is crucial for the quality of the visualization. When direct volume rendering (DVR) is used, the specification of useful transfer functions has been understood as the most demanding and difficult part [59, 24, 28, 62]. Therefore, techniques, which do not require a lot of parameter tuning, while still conveying useful information about the data-set of interest, proofed to be useful, especially when applied during the exploration phase of data investigation, i.e., when the data should be understood in general.

An interesting experience from working with applications in the field of volume rendering, which is especially important for the work presented in this paper, is that (during investigation) volumetric data often is interpreted as being composed of distinct objects, for example, organs within medical data-sets. For a useful depiction of 3D objects, often boundary surfaces are used as a visual representation of the objects. This is mostly due to the need to avoid visual clutter as much as possible. Even in direct volume rendering, when transfer functions are used which also depend on gradient magnitudes [78], visible structures are significantly related to object boundaries. This is a direct consequence of the fact that voxels, which belong to object boundaries, typically exhibit relatively high values of gradient magnitude, i.e., values of first-order derivative information.

In contrast to photorealistic approaches, which stick to (more or less accurate) models of the illumination of translucent media [90], non-photorealistic techniques allow to depict user-specified features, like regions of significant surface curvature, for example, regardless of any physically-based rendering. Recently, also non-photorealistic rendering techniques, which originally have been proposed for computer graphics in general [111, 74, 23], have been proposed for volume rendering [22, 127], definitely extending the abilities for the investigation of 3D data. Ebert and Rheingans [22], for example, give a broad spectrum of various non-photorealistic visual cues (boundary enhancement, sketch lines, silhouettes, feature halos, etc.) to be easily integrated within the volume visualization pipeline. Also very interesting, other specialized techniques for the representation of object surfaces have been proposed recently like 3D LIC which is based on an eigenvalue/-vector analysis of surface curvature as proposed by Interrante [50], or the representation of surface points by the use of small geometric objects as proposed by Saito [110].

In this paper, we present a non-photorealistic rendering technique for volumetric data, which does not depend on data-values, but on the magnitude of gradient information, instead. Thereby, 3D structures, which are characterized by regions exhibiting significant changes of data-values, become visible without being obstructed by visual representations of rather continuous regions within the 3D data-set. We discuss the importance of high-quality gradient reconstruction, which in our case is performed by a sophisticated method based on linear regression [102].

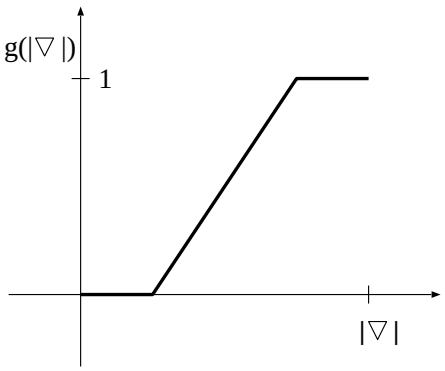


Figure 3.1: The $g(|\nabla|)$ windowing function.

3.2 Contour Rendering

In this section we present the non-photorealistic rendering (NPR) model which we use to display the contours of the objects within the volumetric data-set. The following characteristics of the rendering model in use are crucial:

- no a priori knowledge about the data values
- enhancement of internal details
- just a few rendering parameters, fast fine tuning
- support of optimization for fast previewing

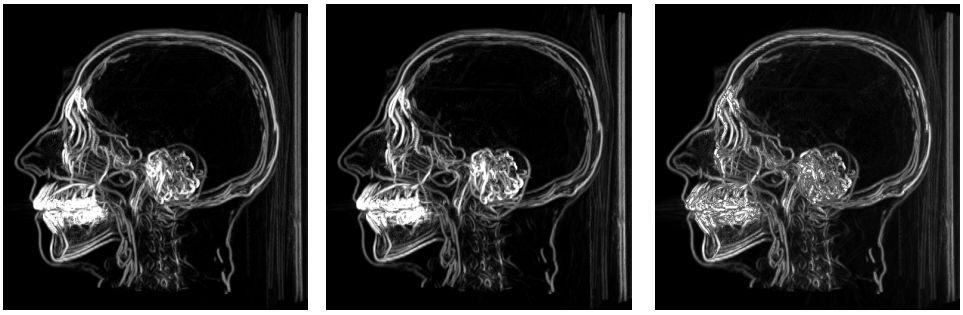
Since we need a model, which is independent of data-values, as a general tool for volumes of various origins, the data values themselves should not directly contribute to the final result. Instead of the original data-values, a function $g(|\nabla|)$ of gradient magnitudes is used (note, that for the sake of simplicity '∇' is used here as a shortcut for '$\nabla \mathrm{data}_i$') in order to characterize the "surfaceness" of a voxel.

This function can be defined as a windowing function by default determining the range of interest in the domain of gradient magnitudes (Figure 3.1). Our model enhances the voxels with higher value of function g emphasizing the surfaces of different objects.

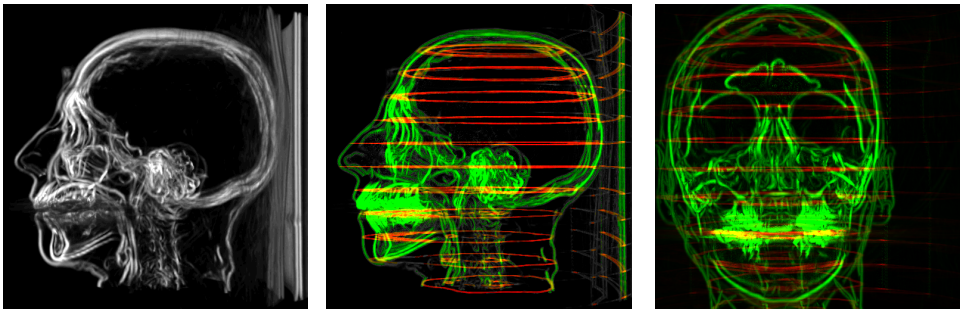
The traditional way of iso-surface enhancement is to use a transfer function, where the opacities are weighted by the gradient magnitudes [78]. This solution has the following drawbacks:

- time consuming transfer function specification
- limited number of iso-surfaces rendered at the same time
- internal details like cavities can be easily missed
- ray-casting-based rendering is computationally expensive

Considering these disadvantages we do not follow the idea of rendering several iso-surfaces. Instead of this we propose a non-photorealistic visualization model.



a) projection: MIP b) proj.: MIP + depth cueing c) projection: LMIP



d) opacity modulated e) plus level lines f) front view

Figure 3.2: Different options of using non-photorealistic rendering for the visualization of object contours within volumes.

Our goal is to give a first 3D impression about the content of the volume preferably without missing important features. Therefore, only the silhouette lines of iso-surfaces are rendered in order to avoid the hiding of important details. Previously this idea has been used only for single iso-surfaces defined by a density threshold and reconstructed by the "marching cubes" algorithm. The silhouette enhancement can be adapted to volumes, where the contours of all the objects can be rendered at the same time providing images which are rich in details. In our model we use a view-dependent function $s(P, V)$ which assigns higher weights to voxels belonging to an object contour:

$$s(P, V) = (1 - |\nabla(P) \cdot V|)^n, \tag{3.1}$$

where P is the position of a given voxel and vector V is the viewing direction and n is an exponent controlling the sharpness of the contour.

There are several opportunities of using weighting functions g and s in calculating the pixel colors corresponding to the viewing rays. For example, an intensity value for a sample position P can be calculated using the following formula:

$$\begin{aligned} I(P, V) &= g(|\nabla(P)|) \cdot s(P, V) \\ &= g(|\nabla(P)|) \cdot (1 - |\nabla(P) \cdot V|)^n \end{aligned} \tag{3.2}$$

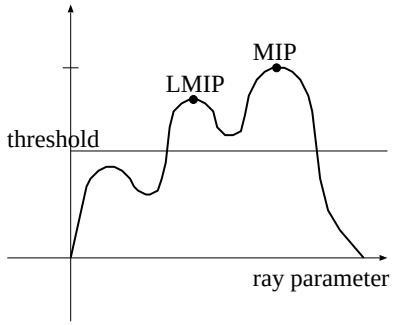


Figure 3.3: Local maximum intensity projection (LMIP).

After having these view-dependent intensities calculated at the sample points along a viewing ray we can use maximum intensity projection (MIP), thus the highest sample intensity is assigned to the given ray. Note that, the classification according to function $I(P, V)$ results a sparse volume significantly reducing the number of voxels which contribute to the final image. This approach has two advantages. There is no visual overload (unlike in direct volume rendering, when several iso-surfaces are rendered at the same time) and this property can also be exploited in the optimization of the rendering process as we will show it in the further discussion. Figure 3.2(a) shows the CT scan of a human head rendered using the contour enhancement function $I(P, V)$, where you can see the contours of the skull, the skin, and the brain at the same time.

If simple maximum intensity projection is used, depth information is not appearent in the image. The spatial impression can be improved by using simple depth cueing where the intensities of sample points are weighted according to their distance from the viewpoint. Figure 3.2(b) shows an image generated with this depth-cueing extension. Here for instance, the contours of the ears are better recognizable than in Figure 3.2(a).

Even if depth cueing is used it can happen that some higher intensity contours hide the weaker contours which are closer to the viewer. This might be confusing in correct interpretation of the image. In medical imaging practice usually local maximum intensity projection (LMIP) is applied instead of traditional MIP in order to avoid such problems [116]. The idea is to project the first local maximum intensity onto the image plane, which is higher than a predefined threshold (Figure 3.3).

This technique can be adapted also to the view-dependent intensities defined in our method. Figure 3.2(c) shows an image rendered using the LMIP of our contour enhancing intensity function $I(P, V)$.

Another alternative of keeping the depth information is to assign the values of function g as colors and opacities modulated by the view-dependent intensity function $I(P, V)$ (Equation 3.2) to the sample points and to perform an alpha-blending rendering.

This approach demonstrated in Figure 3.2(d) emphasizes rather the contours of larger objects and the smaller details are less visible. This version is appropriate for data sets which contain many high frequency regions. In this case only the main characteristic contours are enhanced.

The basic contour projection approach can be extended by also rendering some additional characteristic lines like level lines. For instance, every twentieth slice is selected in the volume and independently from the viewing direction the values of function g are assigned as intensities to the voxels of the selected slices. These assigned intensities are rendered by maximum intensity projection as well but using a different color channel in order to distinguish the level lines from the object contours (Figure 3.2(e)). Of course these level lines can be perpendicular to any arbitrary direction as well.

Figure 3.2(e) clearly illustrate that the additional characteristic lines improve the spatial impression without hiding the important features. After having the image generated the user can interactively and separately scale the intensity of different color channels. It is also possible to simply switch off the display of additional characteristic lines.

Figure 3.4 shows internal organs of a human body rendered by our combined technique. Without taking care about the different density ranges the contours of the skin, the lungs, the spine, and the pelvis are visualized at the same time. The branches of the bronchia inside the lungs are also visible. These are typical low density cavities which can be hardly rendered using the conventional transfer function based ray casting.

Figure 3.2(f) shows also the rendering of such internal cavities, where generating the front view of the human head the frontal sinuses are clearly visible.

For rendering the CT scan of human vertebrae containing screws, which is depicted in Figure 3.5 it would be confusing to use any additional characteristic lines. Note that, the visualization of the screws would require quite some time to find an appropriate transfer function if traditional direct volume rendering was applied.

3.3 Gradient Estimation

Usually in volume rendering the conventional method of central differences is used for gradient estimation. Since it takes only a narrow voxel neighborhood into account it causes typical staircase artifacts. Contour enhancing as it is described in this paper is very sensitive to accuracy in gradient directions, therefore a more sophisticated gradient estimation is required.

We use the results of our previous work published in [102]. The gradient estimation method presented there is based on linear regression, where in a local

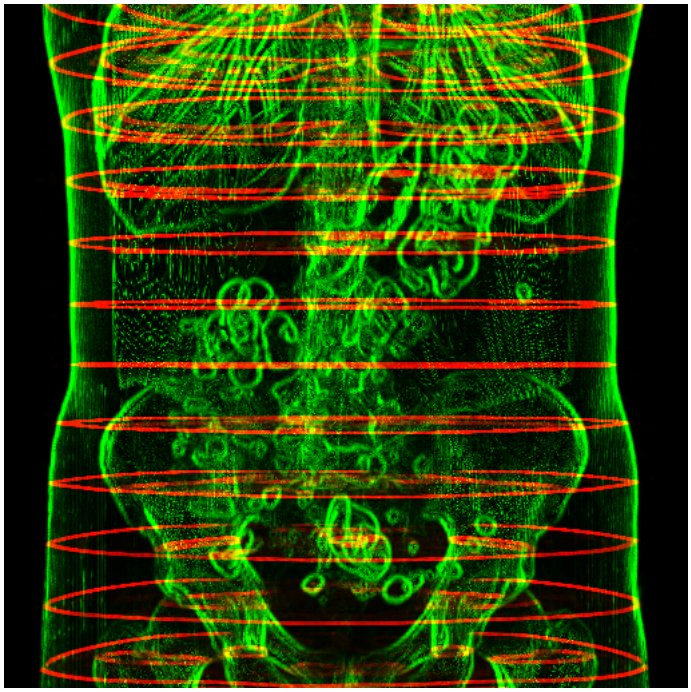


Figure 3.4: Non-photorealistic visualization of internal organs.

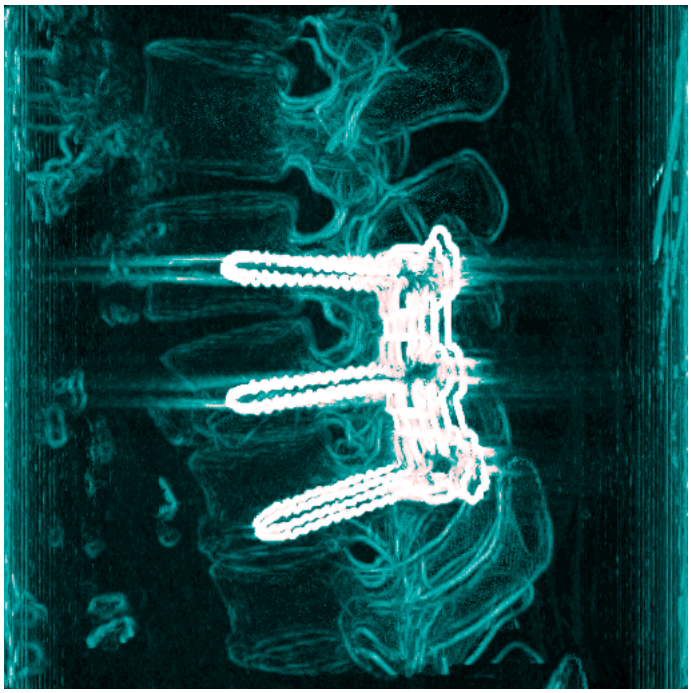


Figure 3.5: Non-photorealistic rendering of human vertebrae with screws inside.

neighborhood the density function is approximated by a 3D hyperplane. The error of the approximation is defined as a mean square penalty function which is evaluated at each neighboring voxel position using a ring off weighting of error contributions. The minimization of the equation system leads to a computationally efficient convolution. Since the surface inclination is estimated from a larger voxel neighborhood, therefore we obtain a smooth gradient function.

3.4 Rendering

Two different implementations of the non-photorealistic method have been used for generating the images shown in this paper. A high-quality, perspective ray caster (Fig. 3.2, 3.4, and 3.5) and an interactive shear-warp based implementation (Fig. 3.8) which is discussed below in detail.

When the lighting model from equation 3.2 is used, the usual spatial queues which are offered by conventional rendering with a physics-based lighting model are not available. If in addition MIP is used to obtain pixel colors, depth information is also lost due to the inherent lack of occlusion within MIP images. The ability to interactively modify the viewing parameters and to change the view-point becomes crucial for understanding the result of the visualization process.

A precondition to achieve interactive rendering performance for volume visualization without special volume rendering hardware like a VolumePro board [103] is the ability to exclude from rendering parts of the volume which do not contribute to the resulting image with utmost efficiency. Classical volume rendering is based on compositing the contributions of samples along rays according to their opacity. In this case either “empty”, entirely transparent regions, or parts of the volume located within high-opacity objects which are occluded by outer parts of the object can be omitted. Skipping of empty regions is usually performed using either a hierarchical approach (for example based on octrees [70]) or by encoding the distance to the closest non-transparent structure at each sample within the volume [13]. The skipping of non-contributing data within high-opacity regions can be done using front-to-back rendering and early ray termination.

If equation 3.2 is used for determining the color of a voxel, it’s potential influence on the resulting image depends on two factors: First, the windowed magnitude of the gradient at the voxel, and second on the angle between the gradient and the current viewing direction. Similarly to the definition of classical opacity transfer functions, the influence of gradient magnitude on a voxel changes only if the windowing function is modified. Voxels with $g(|\nabla(P)|) = 0$ can easily be skipped using one of the established techniques for empty space encoding. Most of the remaining voxels with a higher gradient magnitude also do not contribute any significant information to a contour image, due to the influence of the $(1-|\nabla(P)\cdot V|)^n$ term. Unfortunately, this term is view dependent, making any of the conventional volume region skipping techniques infeasible.

In the following sections we will discuss two approaches for skipping non-contributing voxels within the volume. Their applicability is closely coupled to the used rendering technique. The method used to achieve interactive rendering is therefore outlined first.

3.4.1 Preprocessing

Effective rendering of volumetric data sets where voxel contributions depend on gradient magnitude, requires methods which are able to efficiently deal with "sparse" volumes. For example, potentially contributing voxels can be extracted from the volume and to stored in a list. For each of those voxels, it's position and relevant attributes, like data value or gradient direction and magnitude are stored. The voxels within the list are ordered to meet requirements of a specific rendering technique. For MIP for example, sorting and projecting the voxels by data value eliminates the need for maximum search and allows efficient skipping of voxels mapped to black [98]. For DVR, sorting the potentially contributing voxels by depth eliminates the need to process empty voxels while still maintaining a correct order of compositing [100]. Voxels from common sized (256^3) data sets stored in this way can be rendered at interactive frame rates using for example shear-warp based projection [40]. To achieve interactive frame rates, this method performs parallel projection only, using nearest neighbor interpolation for projecting voxels onto the base plane.

A quantization of gradient vectors to 12-16 bit allows to perform shading by a single table lookup using the gradient vector as an index. The effort for computing the lookup table is negligible. For scanned data sets even the quantization to only 12 bit per gradient still provides visually satisfactory results. Despite of the reduction in accuracy by quantization, performing gradient reconstruction using a high-quality method is still recommended, to avoid staircase artefacts as introduced by using the central differences method.

3.4.2 Optimizations for MIP

In our case, voxel intensities are not constant data values from the volume as commonly used for MIP projection. Intensities $I(P, V)$ result from a function which depends on the viewing direction and gradient magnitude. This property makes a global pre-sorting of voxels by intensity impossible. However, proper ordering of the voxels can be used to group and efficiently skip groups of voxels mapped to black either by windowing of the gradient magnitude or by the influence of the current viewing direction.

For maximum intensity projection, the order of projecting voxels is not relevant as $max(a, b) = max(b, a)$. Thus, voxels do not have to be ordered and projected in spatial order. If we instead group voxels with the same or a similar gradient

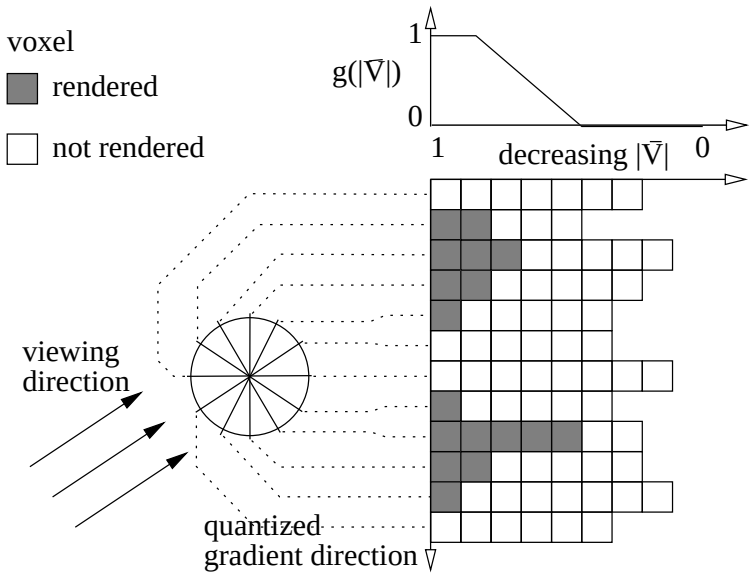


Figure 3.6: Voxel ordering for MIP (2D). Voxels are grouped by (quantized) gradient direction. Within the groups, voxels are sorted by gradient magnitude. Only groups with $S(P,V) > \epsilon$ are rendered, within a group rendering is stopped after the first voxel with $I(P,V) < \epsilon$

direction, we can exploit the fact, that voxels which are not part of a contour for the current viewing direction, are mapped do low intensity values. Entire groups of voxels with a similar gradient direction can be skipped, if the intensity $S(P,V)$ of a representative of this group is below some ϵ (see Fig. 3.6). The quantization of gradient vectors for rendering leads to the required clustering of voxels into groups with the same gradient representation. For typical data sets, over 75% of all voxel scan be skipped by exploiting just this scheme. Furthermore, within a group of voxels with the same gradient representation, voxels can be sorted by gradient magnitude. If projection of voxels within a group starts with voxels with the highest gradient magnitude, processing of the group can be stopped as soon as the first voxel with an intensity $I(P,V)$ below ϵ has been projected.

This arrangement of voxels allows to skip non-contributing parts of the data with utmost efficiency. The disadvantage of this optimization is the restriction of the compositing process to maximum intensity selection. Due to the arbitrary spatial ordering of the voxels, blending of voxel contributions is not feasible.

3.4.3 Optimizations for Back-to-front Compositing

To maintain full flexibility in the choice of compositing operations, like local MIP or alpha-blending, a spatially consistent ordering of the projected voxels has to be maintained. If shear-warp based projection is used, only the order in which

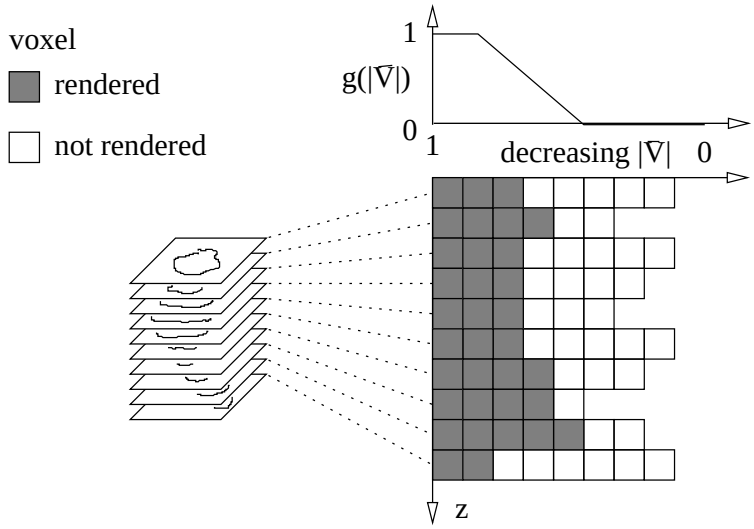
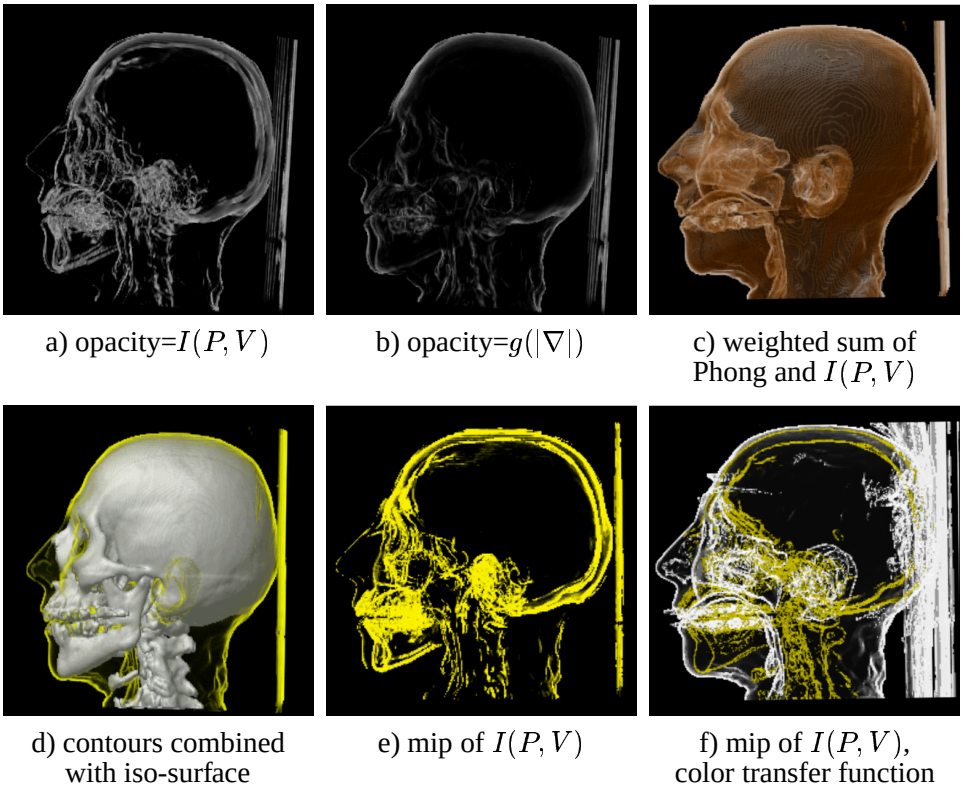


Figure 3.7: Voxel ordering for back-to-front rendering: Voxels within each slice are sorted by gradient magnitude. Voxels which are mapped to 0 by $g(|\nabla|)$ can be skipped efficiently.

slices of voxels are projected is relevant (consistently back-to-front or front-to-back). Within a slice of voxels, the ordering is not relevant as long as projections of neighboring voxels do not influence each other. Nearest neighbor interpolation for the projection to the base-plane meets this requirement. The grouping into slices has to be performed for each principal viewing axis -x, y, and z separately. Depending on the viewing direction, one of the three copies is used for rendering (the usual approach to shear-warp rendering).

Voxels with a gradient magnitude below a specific threshold do not provide a useful contribution to an image rendered using our model. Only about 25% of all voxels have a sufficiently high gradient magnitude, and are thus included into the extracted data structure, thus keeping the memory requirements at a reasonable level.

Within a slice, voxels are sorted according to gradient magnitude. During rendering, only voxels which are not mapped to black due to their gradient magnitude (see Fig. 3.7) have to be considered. Voxels mapped to black due to the currently used $g(|\nabla|)$ are located at the end of the voxel list of the slice and can be efficiently skipped. Compared to the MIP-only ordering of voxels described in the previous section, significantly more voxels have to be rendered. Voxel skipping is only based on the "surfaceness" (i.e. gradient magnitude) of a voxel, but not on the view-dependent property of being part of a contour.



a) opacity=$I(P,V)$ b) opacity=$g(|\nabla|)$ c) weighted sum of Phong and $I(P,V)$

d) contours combined with iso-surface e) mip of $I(P,V)$ f) mip of $I(P,V)$, color transfer function

Figure 3.8: Interactive non-photorealistic rendering results

3.4.4 Interactive rendering – discussion

As usual for interactive rendering, a high frame rate is achieved by trading off rendering quality and accuracy against speed. The most important factor for the quality of contour images is the accuracy of gradient vectors (Sec. 3.3). By quantizing gradients to a few bit for interactive rendering, limits are set to the exponent n in Equ. 3.1. High values of n result in very sharp and thin contours. The quantization error of gradients close to the contour is therefore amplified and results in too bright or too dark voxel contributions. For a quantization to 12 bit as used by our implementation, an exponent around 4 provides a sufficiently narrow contour without producing disturbing artefacts (Fig. 3.8).

Due to more efficient skipping of black voxels and a simpler compositing operation for projecting a voxel, rendering using MIP is faster (see Tab. 3.1) than when blending of voxel contributions is used. Although MIP allows to depict the most significant features of a volume (see Fig. 3.8e, f), the lack of occlusion and depth information in MIP images may be a disadvantage. The high interactivity of non-photorealistic rendering using MIP compensates for this disadvantage by adding time as an additional degree of freedom for the visualization (i.e. interactive viewpoint changes).

	volume size	*voxels rendered*	*time*
head/mip	$256^2 * 225$	102k	85ms
head/blend		366k	150ms
screws/mip	$256^2 * 241$	337k	130ms
screws/blend		942k	270ms

Table 3.1: Rendering Times

More flexibility is gained by using opacity-modulated blending of voxel contributions for rendering. The rendering times are acceptable (Tab. 3.1), although slower than for MIP. Depending on the source of voxel opacity, different effects can be achieved. By setting the opacity equal to voxel intensity (Fig. 3.8a) an effect similar to MIP is achieved, with the difference, that occlusion and spatial ordering of the voxels is taken into account. Contours in areas with a higher gradient magnitude are depicted brighter than in areas with lower gradient magnitude. If opacity is derived from $g(|\nabla|)$ only, the resulting image displays a blended set of surfaces with lighted contours (Fig. 3.8b). This approach can be also well used to enhance contours [22] in addition to Phong shading for surface rendering (Fig. 3.8c). For segmented data which allows to distinguish between different objects, non-photorealistic methods can be easily combined with other rendering methods, for example with conventional surface rendering (Fig. 3.8d).

The rendering times provided in table 3.1 have been measured using a Java implementation of the algorithms on a PII/400MHz processor with Sun JDK 1.3 for Windows.

3.5 Conclusion

In this paper an interactive non-photorealistic volume rendering technique has been presented. It is proposed for fast viewing of different types of data sets without assuming any a priori knowledge about the density values. Our simplified visualization model does not require a time-consuming transfer function specification which is necessary in traditional direct volume rendering. It has been shown that using a simple contour enhancement approach different important details can be visualized simultaneously. This is hardly possible with rendering of semi-transparent iso-surfaces.

Our view-dependent voxel classification results in a sparse volume, therefore the number of voxels which contribute to the final image is significantly reduced. On one hand using this approach the visual overload can be avoided without missing the internal features, like low density cavities. On the other hand the data reduction can be exploited in the optimization of the rendering process. Since real-time rotation is very important in order to improve the spatial impression an interactive

rendering technique has been presented as well. The main characteristic of the interactive approach is a reordering of voxels during a preprocessing step in a way, that voxels mapped to black during rendering due to their gradient magnitude or due to the direction of their gradient vectors can be skipped with high efficiency. This is a pure software-based acceleration method which provides high frame rates even on low-end machines. It also supports the combination of traditional direct volume visualization and non-photorealistic volume rendering.

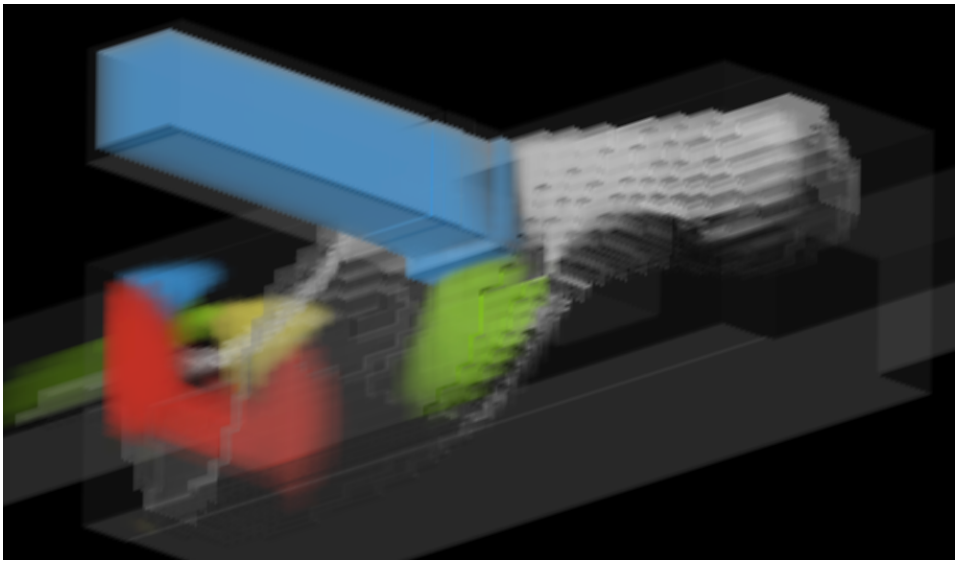
Acknowledgements

The work presented in this publication has been funded by the $V^{is}M^{ed}$ project (`http://www.vismed.at`). $V^{is}M^{ed}$ is supported by *Tiani Medgraph*, Vienna (`http://www.tiani.com`) and by the *Forschungsförderungsfond für die gewerbliche Wirtschaft*. Parts of this works have been done in the VRVis research center, Austria (`http://www.vrvis.at/vis/`), which is funded by the Austrian governmental research program Kplus.

The CT scans of the human head and the engine block were obtained from the Chapel Hill Volume Rendering Test Dataset. The data was taken on the General Electric CT scanner and provided courtesy of North Carolina Memorial Hospital. The CT scan of human vertebrae was provided by *Tiani Medgraph*.

Chapter 4

Interactive Volume Visualization of Complex Flow Semantics



In 2003, the contents of this chapter have been published in in the Proceedings of the 8th International Fall Workshop "Vision, Modeling, and Visualization 2003" (*VMV 2003*), pp. 191–198 (paper "**Interactive Volume Visualization of Complex Flow Semantics**" by Helwig Hauser and Matej Mlejnek).

The contents of this chapter (paper) are a result from a collaborative project with Matej Mlejnek (one of Helwig Hauser's diploma students). Related papers (co-authored by Helwig Hauser) are:

- **Two-Level Volume Rendering** [41], a strongly related paper, also contained in this thesis as chapter 2, and **Two-level volume rendering - fusing MIP and DVR** [40], an earlier version of this paper
- **RTVR - a flexible java library for interactive volume rendering** [95], a related paper about fast volume rendering

- **Smooth Brushing for Focus+Context Visualization of Simulation Data in 3D** [20], a tightly related paper, and **Interactive Feature Specification for Focus+Context Visualization of Complex Simulation Data** [19], also a tightly related paper

Interactive Volume Visualization of Complex Flow Semantics

Helwig Hauser and Matej Mlejnek

Abstract

Comprehending results from 3D CFD simulation is a difficult task. In this paper, we present a semantics-based approach to feature-based volume rendering of 3D flow data. We make use of interactive feature specification to acquire so-called degree-of-interest (DOI) values, which describe what is most interesting to the user (at a current point in time). We adapt three visualization techniques to achieve better visualization answers to specific user questions: (a) isosurfacing the degree of interest – a triplet of isosurfaces represents certain levels of interest; (b) feature volumes – volume rendering is used to depict 3D distributions of DOI values; and (c) interest-based seeding of streamlines, resulting in streamlines which are automatically placed in regions of special interest. We utilize HW-accelerated resampling and fast shear-warp volume rendering (RTVR) for real-time viewing, also providing two-level volume rendering, which allows to integrate all of the above-mentioned approaches.

4.1 Introduction

The visualization of flow datasets which result from computational simulation gains increasing importance due to the increased use of flow simulation in several fields of modern industry – a well-known example is the simulation of air flow around models of new flight vehicles. In our case, the visualization users simulate fluid and/or gas flows within complex 3D geometries, such as combustion processes in the automotive environment. Due to the size and the complexity of their data, user-oriented visualization is needed to help with answering specific questions about the data during data analysis and exploration. Also, visualization proves to be very useful when the simulation process itself is investigated – questions such as of whether the grid design was appropriate with respect to the simulated flow features or of whether the boundary conditions actually provided a useful interface between the “world setting” of the simulation and its inner processes are frequently checked after simulation.

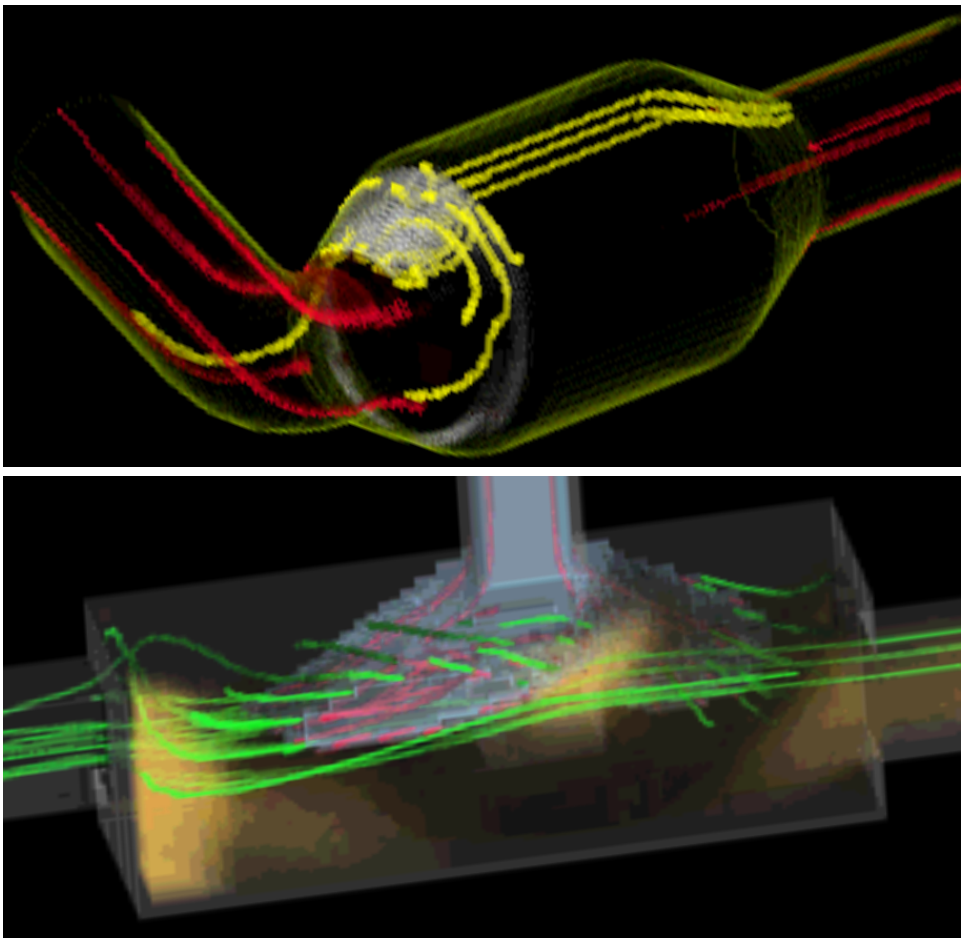


Figure 4.1: Visualization of a backflow region in a catalytic converter (upper image) and the mixing of hot and cold water (lower image).

4.1.1 Flow data and visualization

The visualization of flow data still is a challenging field of research and applications. This is because usually huge lots of data-items need to be visualized, often complex grid structures are involved, and real flows usually are three-dimensional and time-dependent (posing major challenges such as tremendous resource requirements, dealing with unsteady data, and handling of occlusion). Results from computational flow simulation also often provide several additional data attributes per data-item such as pressure, temperature, etc., requiring visualization techniques for multi-dimensional data.

To cope with the above mentioned challenges, visualization techniques are required, which focus on especially interesting subsets of the data. If not all of the data is shown simultaneously, occlusion problems are reduced and resource re-

quirements are moderated. Feature-based visualization of flow data is one very useful approach as well as focus+context visualization of data from flow simulation. While literature [105] already provides very good solutions in feature-based visualization – examples are iconic visualization of flow features [108, 131] or the use of flow topology for visualization [43, 117] –, focus+context visualization of flow data only recently became an active field of research [19]. The work presented here builds upon this latter approach.

With respect to the multi-dimensional character of data from computational flow simulation, visualization techniques which extend the usual domain of scientific visualization are needed. In information visualization we find many useful approaches to the visualization of multi-dimensional data [11, 63]. The integration of techniques from information visualization such as scatterplots, histograms, and parallel coordinates [48] already proved to be useful, also in scientific visualization [34].

4.1.2 Flow data and volume visualization

Volume rendering is one of the most prominent fields of visualization research. Very useful results have been achieved in medical applications. Volume rendering also has been applied to data from flow simulation [16, 122]. However, the specification of proper transfer functions – which already is a challenging problem in 3D medical visualization – is an even greater challenge in 3D flow visualization. This probably is due to the fact that medical data and flow data inherently have different data characteristics. Whereas in medical data we usually have tissue boundaries (which usually are to be shown in medical visualization), in flow data we often miss such rather sharp boundaries between subsets of the data (of course there are counter-examples such as shock waves, also).

Flow features often are smoothly delimited. Thus tools, which already proved their excellence in medical applications such as isosurfaces (computed with marching cubes [83]) and alpha-compositing of gradient-weighted voxel data [78, 61], do not produce as meaningful results as in medical visualization. The respective visualization metaphor of representing data by some kind of thresholding needs to be adapted to data from flow simulation. In this paper, we adapt – as one part of our contribution (section 4.4) – isosurfacing and alpha-compositing for volume rendering to better match the special situation of 3D flow visualization.

4.1.3 User questions and visualization

The other part of our contribution in this paper is that we more directly involve the user within the visualization process. We do so to more specifically respond to the actual user questions. A typical user question would enquire all those places in the flow domain where the temperature is high and the flow is slow, for example.

Therefore, a tool for interactive analysis should (a) allow the users to easily formulate their questions and (b) also provide useful visualization answers, which are tailored to the user questions. In our software, we use an interactive assessment step in the first place [19, 20], which is based on multiple, linked views as well as on interactive brushing to determine what the user currently is most interested in (see section 4.2 for more details).

So the two main contributions of this paper are (a) the embedding of voxel-based volume rendering in our framework for interactive analysis and exploration of 3D flow data (through resampling to a Cartesian grid, see section 4.3) and (b) the adaptation of voxel-based iso-surfacing, volume rendering, and dense streamlines (section 4.4) to better reflect the specific user interests (utilizing degree-of-interest values for visualization). All of the here described extensions have been integrated within a real-time volume rendering software (RTVR [95], see section 4.5 before results, conclusions, etc.).

4.2 Interactive Data Assessment

In this section, we describe how a system of multiple, linked views is used (together with interactive brushing) to determine what the user currently is most interested in. We also describe degree-of-interest (DOI) values which are used to represent the results of the data assessment.

4.2.1 Linking and brushing

In the interactive assessment step, the user investigates the multi-dimensional simulation data by using multiple, linked views, which all show the same data, but usually different data attributes thereof. We provide scatterplots, histograms, and also parallel coordinates for flow data assessment. On demand, the user brings up views as required and adjusts them to show exactly those data dimensions which currently are most related to the user's current interest (see figure 4.2 for an example).

After interactive exploration in these views, the user starts to formulate what he or she is most interested in: to do so, the user works out interesting data-items by iteratively restricting the subset of interest with respect to certain data dimensions. This is achieved by interactively brushing what actually is shown in the views. Step by step, the user concretizes the description of his or her current interest. 3D visualization, as described below (section 4.4), supports this step by interactive spatial feedback.

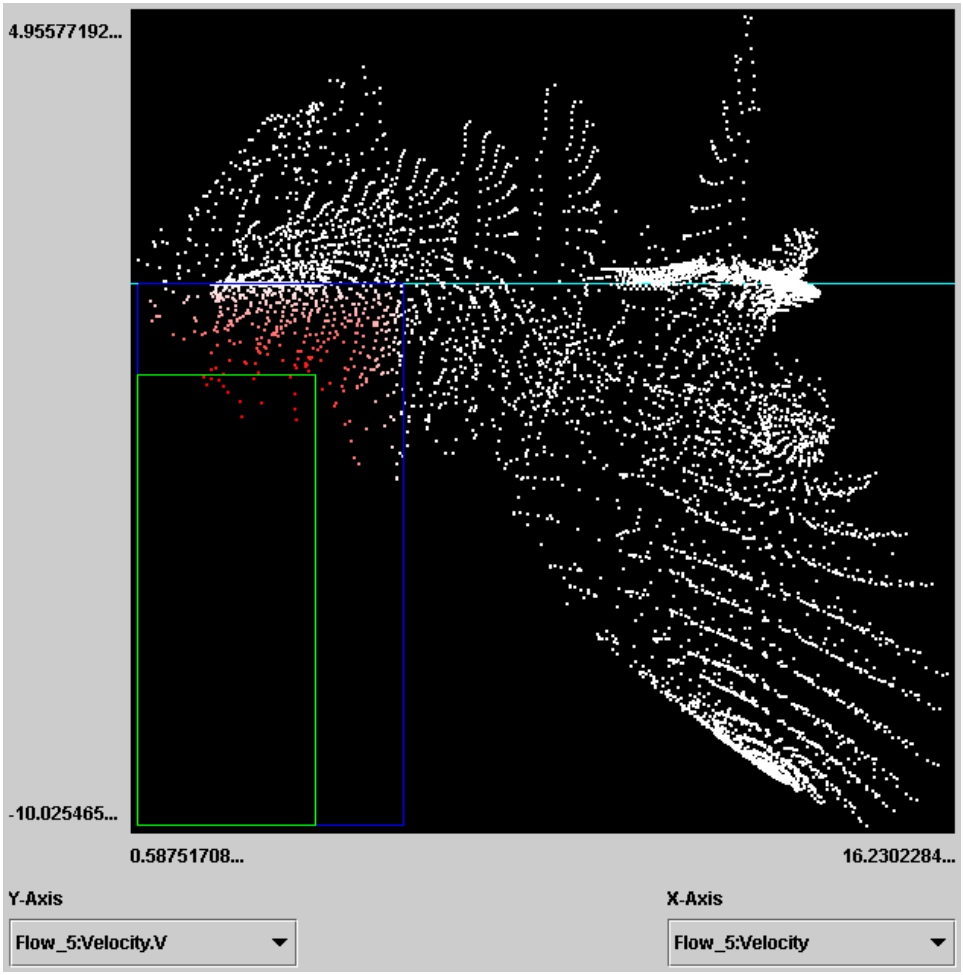


Figure 4.2: A scatterplot of the catalytic converter data (x: over-all velocities, y: vertical flow components). Interactive brushing was used to select data with a significant "downwards" flow component.

4.2.2 Degree-of-interest values

As already mentioned, we use degree-of-interest (DOI) values to represent what the user (currently) is most interested in. In the case of several concurrent user questions, multiple DOI values are attached to each data-item. Smooth brushing [20] is used to gain DOI values which non-binarily are distributed in the interval between 0 (no interest at all) and 1 (100% interest). Thereby, the user can work around the case that the smooth distribution of data properties across the flow domain does not allow for a sharp and still meaningful segmentation into interesting versus not interesting parts of the flow. This also can be interpreted as assigning a fuzzy set membershipwith respect to the data subset of interest.

To enable users to specifically formulate their questions it is necessary to allow complex combinations of simple restrictions. In the interface of our application, logical AND- and OR-combinations are provided through the use of modifier keys while brushing with the mouse. The application explicitly represents a composite brush in terms of a so-called feature definition language [19]; a separate view, similar to a regular tree viewer of a file system, allows to directly manipulate the settings of each and any brush via sliders and numeric input. DOI values are computed from the logical composition tree by hierarchically using a min- or a max-combination where a t-norm (AND) or a t-conorm (OR) is required, respectively.

Although there are interesting alternatives for logical combinations of fuzzy sets [60], we chose min/max for our system, allowing for fast computations and also the drop-off to zero, when combining two fractional values, is minimal with respect to other alternatives.

4.3 HW-Accelerated Resampling

Flow data-sets from computational simulation usually are laid out on non-Cartesian grids such as unstructured grids. To enable fast volume visualization, we use a variant of a hardware-accelerated resampling approach [136] to provide fast access to the flow data with respect to Cartesian coordinates.

The simulation results we are using are given in the form of cell-based data, i.e., data attributes such as the flow vector, pressure, etc., are given for each grid cell, as opposed to a per-vertex specification.

For fast volume visualization it is crucial to efficiently interrogate any data attribute with respect to (almost) arbitrary spatial location. Since we usually need access to multiple attributes during visualization, we decided to not directly resample all the data themselves, attribute by attribute, but to compute an intermediate 3D redirection table that maps spatial locations to the corresponding original grid cells.

Such a redirection table uses much less memory than resampling all the original attributes into individual Cartesian grids. The redirection table is a 3D Cartesian grid of a given resampling resolution (usually we use 512 voxels along the dimension of greatest extent), where each voxel contains the ID of the original grid cell corresponding to the spatial coordinates of that voxel.

In order to compute this table, we first decompose the original grid cells into tetrahedra. Each tetrahedron is assigned the ID of the corresponding original cell. Since our resampler employs graphics hardware for the actual resampling process, we split up the integer ID into four 8-bit parts that fit into the R, G, B, and A channels of a hardware color specification. During resampling, each cell is assigned the constant RGBA color corresponding to its ID and rasterized into the output grid, yielding a redirection table of 32-bit cell ID voxels.

Then, since the original data attributes are available for each cell, mapping a Cartesian location to any of its corresponding attributes becomes a simple two-step process. First, the location is mapped to its corresponding cell ID via the redirection table. Then, the cell ID is mapped to the actual desired attribute corresponding to that cell.

Through this kind of resampling we trade memory-efficiency and flexibility for limited visual quality of the results – the redirection table basically equals a zero-order reconstruction of the data. However, high quality output anyhow is not essential for exploration/analysis of simulation data.

4.4 Visualization Mapping

In this section we describe adaptations of three well-know visualization algorithms (isosurfacing, volume rendering, and streamline rendering) to better match the situation of visualizing 3D flow semantics.

4.4.1 Isosurfacing the degree of interest

Isosurfaces are a very useful instrument to visualize scalar data in 3D. This is mostly due to their visually sharp appearance. When shaded, very good depth perception is achieved. The worth of an isosurface, however, is dependent on how meaningful the respective isovalue is (see also related work about how to find meaningful isovalues [4, 134]).

An isosurface is a sharp visual distiction between data-items with a reference-attribute lower than the isovalue from those with larger attribute values. If the distribution of the investigated data attribute is rather smooth with respect to its spatial locations, then an isosurface can provide reduced information in places where there is not much change in the data attribute of interest – a small variation of the isovalue can then cause drastic changes in the geometric layout of the isosurface.

Below, we now present two variants of standard isosurfaces which better reflect their sharpness with respect to the underlying data, thus being better suited for rather smooth data such as flow data from computational simulation. Since all of the here described volume visualization techniques are integrated within a volume rendering setup, the below described isosurfaces are represented as voxel-surfaces [95].

Gradient-dependent opacity of isosurfaces

As a first extension, we present isosurfaces which exhibit a surface opacity which is dependent on the gradient magnitude with respect to the data attribute under

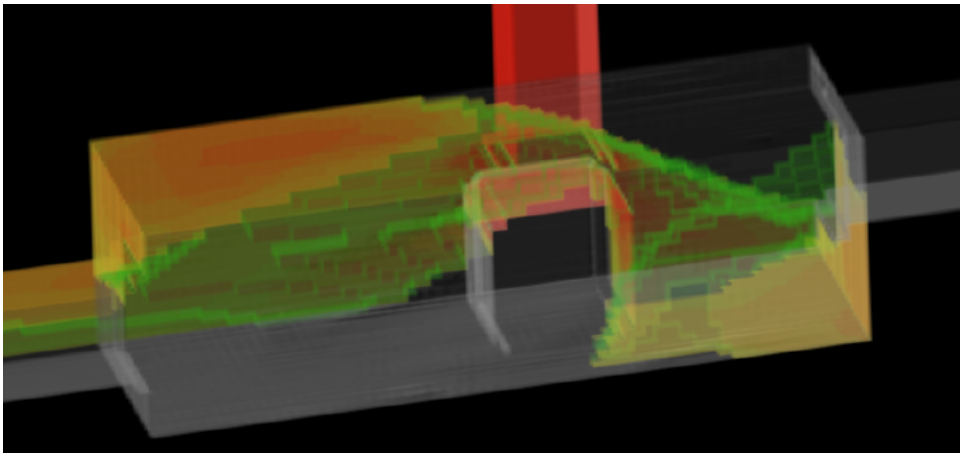


Figure 4.3: Isosurface with respect to 316 K (hot water floating in from the top inlet), whose voxels are more opaque the larger the temperature gradient is at the same voxel. Similarly, the color of the isosurface varies from red (large gradients) via yellow to greed. In this case the resampling resolution was much higher than the original grid resolution, yielding staircase-effects at the isosurface.

investigation. Similar to windowing of intensity values in medical visualization, the opacity $\alpha_{\mathbf{v}}$ of an isosurface-voxel $\mathbf{v}$ is defined as

$$\alpha_{\mathbf{v}} = \begin{cases} \alpha_{\mathrm{hi}} & \text{if } |\mathbf{g}_{\mathbf{v}}| > g_{\mathrm{hi}} \\ \alpha_{\mathrm{lo}} & \text{if } |\mathbf{g}_{\mathbf{v}}| < g_{\mathrm{lo}} \\ \alpha_{\mathrm{lo}} + \frac{|\mathbf{g}_{\mathbf{v}}| - g_{\mathrm{lo}}}{g_{\mathrm{hi}} - g_{\mathrm{lo}}} \cdot (\alpha_{\mathrm{hi}} - \alpha_{\mathrm{lo}}) & \text{else} \end{cases}$$

where $\mathbf{g}_{\mathbf{v}}$ denotes the gradient of the investigated data attribute, g_{lo} and g_{hi} bound a range of gradient magnitudes in which the surface opacity linearly increases from α_{lo} to α_{hi}. While g_{lo} often is set to 0, setting an upper limit of $g_{\mathrm{hi}} \ll \max_{\mathbf{v}} |\mathbf{g}_{\mathbf{v}}|$ especially is useful when parts of the isosurface coincide with the boundary geometry of the flow where usually large gradients occur.

A sample image demonstrating this kind of a isosurface is shown in figure 4.3. The more transparent the isosurface is, the less it actually tells due to small gradients at the respective places. Also, a color transfer function has been used which assigns three different colors with respect to small (green), medium, and large gradient magnitudes (red). This color scheme is an additional help with the correct interpretation of the isosurface.

Isosurface triplets

Another extension of standard isosurfaces are isosurface triplets, which also help to correctly read the sharpness of an isosurface. Instead of one isosurface, a set of three semi-transparent isosurfaces is rendered (also compare to related work [31]).

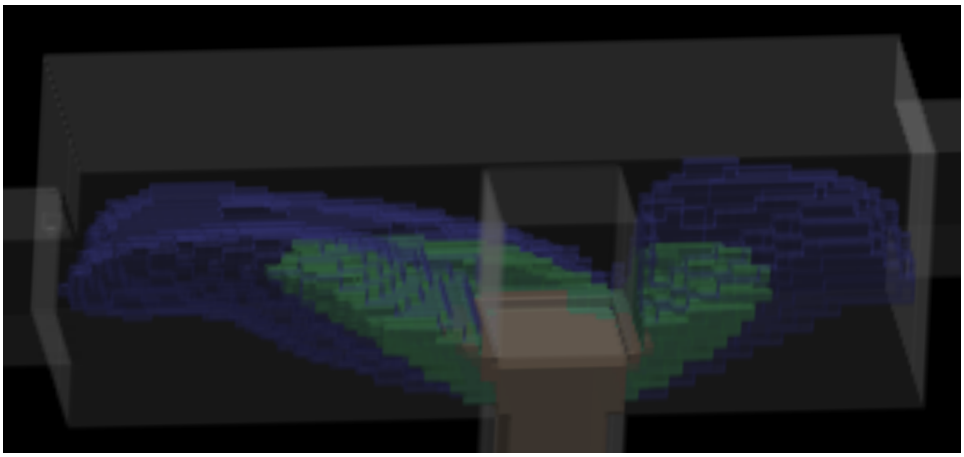


Figure 4.4: Isosurface triplet used to represent a user's interest in hot regions of the flow. Red denotes maximal interest (hottest parts), green medium interest, and blue least interest, all according to an interactive DOI assessment. Again, the blocky appearance of the surface structures result from the relatively low grid resolution with respect to resampling and rendering.

With respect to one so-called reference isosurface (isovalue f_{ref} and surface opacity α_{ref}), two additional isosurfaces (isovalues f_{a} and f_{b}, opacities α_{a} and α_{b}) are specified. We propose two different specification modes for isosurface triplets: *reference-inside* and *intermediate-reference*.

In the *reference-inside* mode of an isosurface triplet, surface isovalues are sorted $f_{\mathrm{a}} < f_{\mathrm{b}} < f_{\mathrm{ref}}$, i.e., the reference isosurface is the inner-most, assuming that the data attribute under investigation exhibits increasing values outside-in. Usually an equidistant spacing between the three isovalues is used such that $(f_{\mathrm{ref}} - f_{\mathrm{b}}) = (f_{\mathrm{b}} - f_{\mathrm{a}}) = f_{\mathrm{sep}}$, where the user can adjust f_{sep} in the user interface. For the *reference-inside* type of isosurface triplets, increasing values of surface opacities proved to be useful: $\alpha_{\mathrm{a}} < \alpha_{\mathrm{b}} < \alpha_{\mathrm{ref}}$ with $\alpha_{\mathrm{ref}} \approx 1$. Again, an equidistant spacing of the three opacities is useful. The *reference-inside* isosurface triplet is useful for DOI visualization, using $f_{\mathrm{ref}} = 1 - \varepsilon$ to represent the 100%-interest subsets with the reference isosurface, and $f_{\mathrm{sep}} = 25\%$, for example, to also denote where interest drops to 75% and 50%, respectively. In figure 4.4 we demonstrate this kind of DOI visualization (red: 100%-interest, blue: 50%-interest).

In the *intermediate-reference* mode of an isosurface triplet, isovalues are sorted $f_{\mathrm{a}} < f_{\mathrm{ref}} < f_{\mathrm{b}}$, i.e., the reference isosurface is in-between the two others. Two different variants of setting the isovalue spacing proved to be useful: equidistant spacing with $(f_{\mathrm{b}} - f_{\mathrm{ref}}) = (f_{\mathrm{ref}} - f_{\mathrm{a}}) = f_{\mathrm{sep}}$, where the user sets f_{sep}, and proportional spacing with

$$\frac{f_{\mathrm{ref}} - f_{\mathrm{a}}}{f_{\mathrm{ref}} - f_{\mathrm{lo}}} = \frac{f_{\mathrm{b}} - f_{\mathrm{ref}}}{f_{\mathrm{hi}} - f_{\mathrm{ref}}}$$

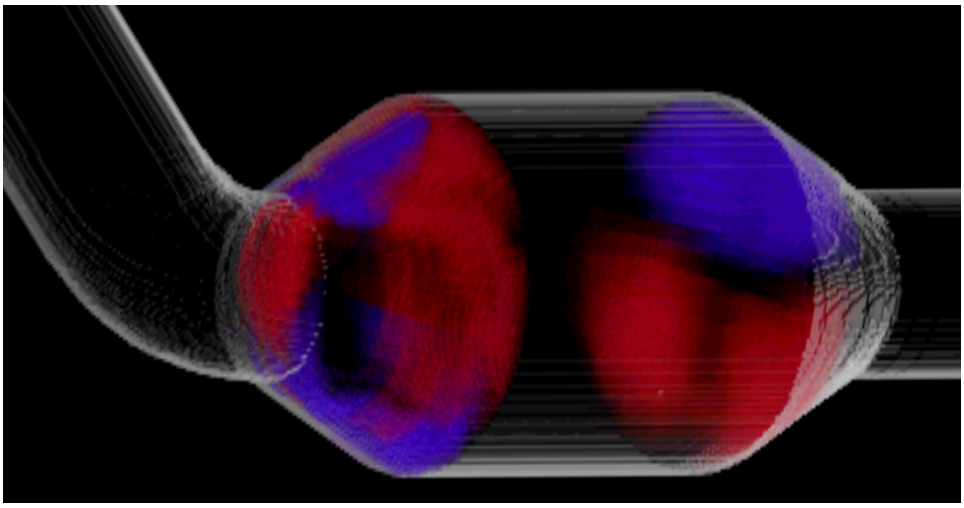


Figure 4.5: Volume rendering of all those places in a catalytic converter where flow vectors exhibit a relatively strong vertical flow component (red: upwards, blue: downwards) – a recirculation bubble becomes apparent in the upper part, before conversion block.

with f_{lo} and f_{hi} denoting the value range of the data attribute under investigation. In this case, the isovalues of the additional two isosurfaces partition the value range below and above the reference isovalue proportionally. Thereby, for example, it is possible to easily adjust the isosurface triplet in a way such that, regardless which isovalue is used for the reference isosurface, the outer isosurface always represents the property boundary halfway to minimum (with respect to the data attribute under investigation), and the inner isosurface halfway to maximum. For the surface opacity setting, usually $\alpha_{\mathrm{a}} = \alpha_{\mathrm{b}} = (\alpha_{\mathrm{ref}} - \alpha_{\mathrm{sep}})$ is used with the user setting $\alpha_{\mathrm{ref}} \gg \alpha_{\mathrm{sep}} \gg 0$.

In addition to the specific settings of the isosurface opacities as described above, coloring the isosurfaces in red (reference isosurface), green (inner), and blue (outer) is useful for a better visual distinction of the stacked and semitransparent isosurfaces.

Our implementation also supports the use of isosurface-quintuplets. However, using more than three semi-transparent isosurfaces stacked over each other, really poses difficulties for visual discrimination. Future work could clarify whether a more sophisticated modelling of the isosurface opacity, like the use of curvature-directed strokes [51] can furthermore improve the 3D perception of isosurface triplets/quintuplets.

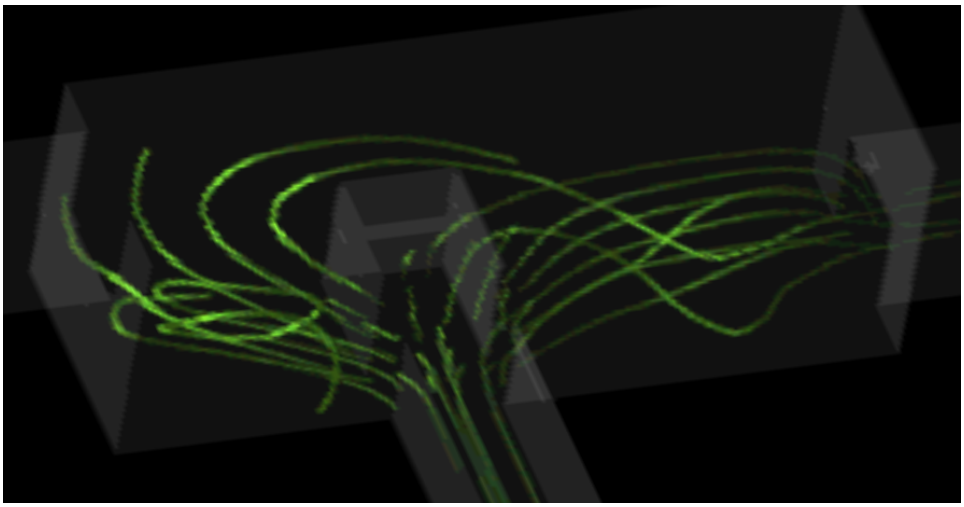


Figure 4.6: Illuminated streamlines representing a user's interest in streamlines whose origins coincide with the hot inflow.

4.4.2 Volume rendering of DOI values

Another useful volume visualization technique for scalar data laid out in three-space is volume rendering by means of alpha-compositing [78]. In our case we use this technique to visualize DOI values instead of original data values. We map DOI values to voxel opacities in a one-to-one manner such that DOI values of 1 appear opaque (or at least with maximal opacity) and DOI values of 0 are not visible at all.

The application of this kind of DOI visualization usually is based on the definition of multiple DOI attributes (in the sense of segmenting the flow data into several interesting subsets). We use the hue of voxel colors to visually differentiate between different DOI objects. Usually we vary the color intensity according to Phong illumination [104]. When multiple volumetric DOI objects overlap in one voxel, one voxel overrides the other based on a random choice – since such cases usually are rare (due to the segmentation approach used), this solution has been sufficiently accurate in our case. In figure 4.5 we see two DOI objects (one red and one blue) with varying opacities according to their DOI values.

4.4.3 Interest-dependent streamlines

Streamlines are a very useful instrument to visualize instantaneous vector fields. Due to their character of visually connecting flow locations along virtual paths of massless particles, streamlines give an intuitive image of a flow dataset. Evenly-spaced streamlines, which fill up the flow domain in a well-distributed way, are well-accepted for two-dimensional flows [52]. However, in 3D flow visualization, selective streamline placement is needed to avoid problems with occlusion [6].

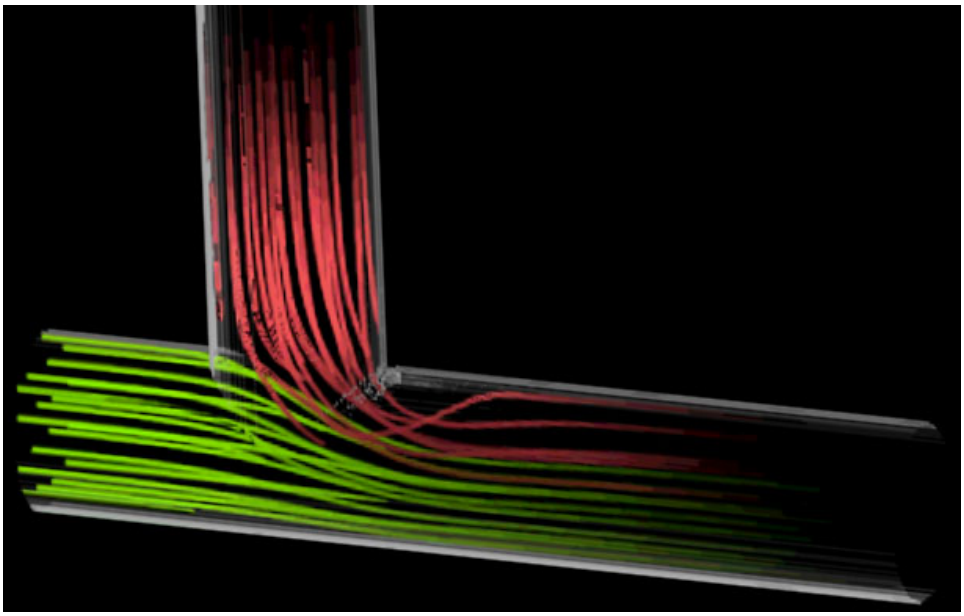


Figure 4.7: Illuminated streamlines representing a user's interest in how two flows join in a simple T-junction. Two regions of interest were used to gain two (differently colored) objects of streamlines.

In previous work [81] we use the vicinity to topological structures such as characteristic trajectories in the flow data to selectively seed streamlines. In this work, we now use the spatial distribution of DOI values to selectively seed streamlines. To do so, evenly-spaced streamline seeding [52] is extended to 3D. Streamlines are only seeded in regions where the DOI value is 1. From there, streamlines are integrated forwards and backwards until they either exit the flow domain, violate the minimal distance to other streamlines, or just get too long with respect to a user-defined maximal length of streamlines.

To integrate DOI-based streamlines within our volume visualization setup, we assume that streamlines have a certain width and rasterize them into our voxel grid such that the voxel opacities are set according to the proportional overlap of the streamline and the respective voxel. We use a $5 \times 5 \times 5$ sub-voxels accuracy when computing the overlap of (thick) streamlines and voxels, thus gaining anti-aliased streamlines, which is of great importance for useful results. Furthermore, voxel opacities are also dimmed down according to the DOI values exhibited at the respective streamline points. Thereby, the streamlines fade out according to the degree of interest. For lighting, we use the illuminated streamlines model [141] to furthermore support the 3D perception of streamlines in our visualization. Figure 4.6 demonstrates the use of DOI-based streamlines on the example of a T-junction.

4.5 Real-time Volume Rendering

Gradient-dependent isosurfaces and isosurface triplets, as well as volume rendering of DOI values and interest-weighted streamlines have been integrated on the basis of RTVR [95], which is a fast volume rendering software providing several different modi of volume rendering such as standard alpha-compositing, maximum-intensity projection (MIP), and non-photorealistic contour rendering [17]. The use of RTVR for flow volume rendering provided flexibility with respect to modeling as well as fast rendering. These advantages came at the cost of limited visual quality, which however is not so important during analysis and exploration.

Two-level volume rendering [41] is used for datasets which are composed of several, explicitly separated objects so that different rendering modi can be used for different objects.

In this work, we build up RTVR-objects for every representation of a user's particular interest. Thereby, the user interactively composes a set of volumetric objects (extended isosurfaces, truly volumetric objects, and streamlines, recall section 4.4) in dependence on as many data attributes as necessary, in simple or in complex manner (recall section 4.2). For the case that two or more objects make use of one voxel, different strategies apply. When multiple iso-surface voxels coincide within on voxel, preintegration of object values is performed to mix the different surface contributions accordingly. Streamline voxels and isosurface voxels override volume voxels, whereas between streamline voxels and isosurface voxels, the object with the more recent specification overrides the other. For every object the user can decide in which style the respective object should be rendered.

Isosurfaces usually are of one color and shaded while their opacity may vary with respect to the data gradient. Isosurfaces also can be rendered by using the non-photorealistic contour rendering, thereby giving very good context for the entire visualization (see the visualization of the boundary geometry in figure 4.7 for an example). For volume rendering, either shaded voxels are used or voxels whose color intensities depend on the DOI attribute. Voxel opacities usually represent the DOI attribute which the voxel object was made dependent on during specification. Streamlines are either illuminated [141] or colored such that the intensity represents the DOI attribute for which the streamlines have been requested. Opacity of streamlines is used for anti-aliasing.

In general, we use hue to discriminate objects from each other. This approach already proved to be very useful in medical applications [95] and also in the context of flow data, color coding the different 3D objects different from each other significantly eases the perceptual distinction of them.

RTVR provides several useful tools for volume visualization which also in this application helped a lot with comprehending the results of computational flow simulation: interactive change of lighting conditions, the interactive change of an

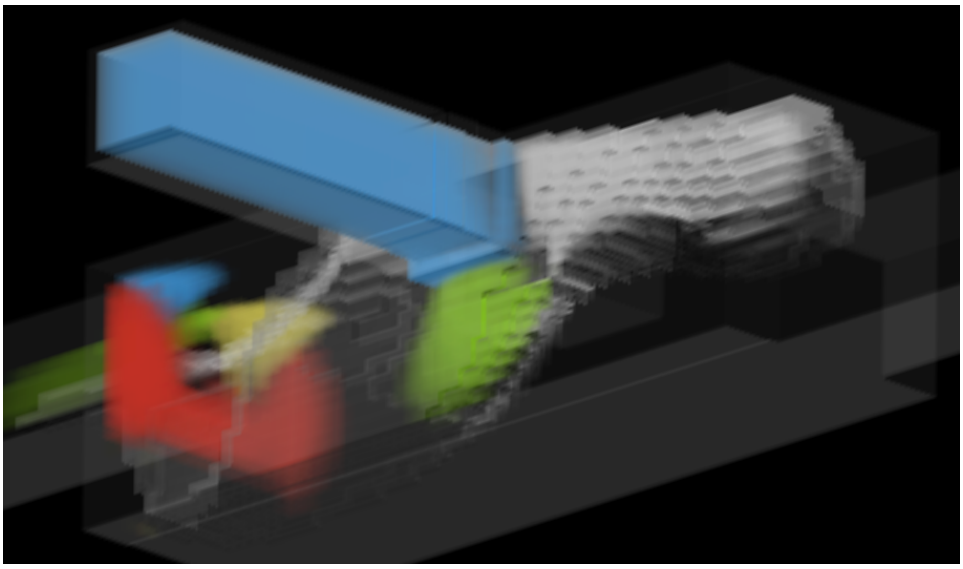


Figure 4.8: Flow through the extended T-junction (main flow from right to left). Volume rendering was used to encode flow regions with flow which is mostly aligned with one of major axes (blue: $-y$-flow, like from the secondary inlet; yellow: $+y$-flow; red: upwards; green: downwards). A gradient-dependent isosurface was used to show the extent of medium hot temperature (hot coming in from the upper inlet).

object-wide overall opacity (in addition to the per-voxel opacity transfer function), and, of course, the interactive change of viewing conditions.

4.6 Results

Following, we describe two application cases in which the previously described visualization technique helped to answer specific user questions.

4.6.1 Flow through a catalytic converter

One investigated dataset comprises the results of a simulation of gas flow through a catalytic converter. With respect to this case, special interest is put on a recirculation zone, right before the reaction chamber. For visualization, the user first describes his/her interest by (a) a brush on $-x$-velocities (as the positive x-axis coincides with the main flow direction through the catalytic converter) and (b) a restriction to the region before the reaction chamber, combined with a logical AND. The thereby defined DOI attribute is then used to selectively place streamlines as shown in figure 4.1, upper image (the DOI values are also volume rendered in grey). In addition to the streamlines, volume rendering in red is used to visualize

regions of relatively high turbolent-kinetic energy. The whole catalytic converter is shown as context using a contour rendering.

Another approach to this case is to investigate upwards and downwards flow which at the same time does not exhibit a significant flow component into the main flow direction. Figure 4.5 shows a volume rendering with upwards flow in red and downwards flow in blue – note the different patterns of upwards/downwards flow before (with the recirculation) and after the reaction chamber.

4.6.2 Extended T-junction

The other case presented here is an extended T-junction (with an extended chamber around the junction and an obstacle in front of the secondary inlet). Cold water is floating in from the primary inlet, and hot water is entering the T-junction through the secondary inlet) after the first third of the time-span of the simulation.

In this example, the user primarily is interesting in the mixing behavior of this flow-junction. The obstacle in the middle of the T-junction causes the hot inflow to split – one part turns directly to the exit whereas the other part traverses an extra-loop before it also develops towards the exit of this structure (see figure 4.6 for several streamlines demonstrating this behavior). Figures 4.3 and 4.4 use the two here described isosurface extensions to show certain degrees of temperature in response to the hot inflow from the one side.

In figure 4.8, several volumetric objects are used to partition parts of the flow in regions which exhibit flow that is mainly in the direction of one of the three main axes. A gradient-dependent isosurface was used to depict the boundary of the incoming hot flow – this isosurface is more opaque in regions with larger temperature-gradients. The separation of the incoming flow, due to the obstacle, is very well visible in this example, also.

Figure 4.9 visualizes vortical structures in the data as well as the boundary between hot and cold flow (in blue). Additionally, a gradient-dependent isosurface is shown which visualizes flow regions with a major y-component, i.e., towards the inlet in the front. From this isosurface, information can be derived about where in the flow vortical structures could be, e.g., before and after the obstacle.

4.7 Summary and conclusions

In this paper we present semantics-based visualization of flow data from computational simulation. We describe how the user utilizes interactive feature specification to tell the system what he or she currently is interested in. We propose (a) gradient-dependent isosurfaces to also visualize the sharpness of an isosurface, (b) isosurface triplets with specific relations for the three isovalues involved, (c) volume rendering of DOI values, and (d) interest-dependent streamlines, which are based

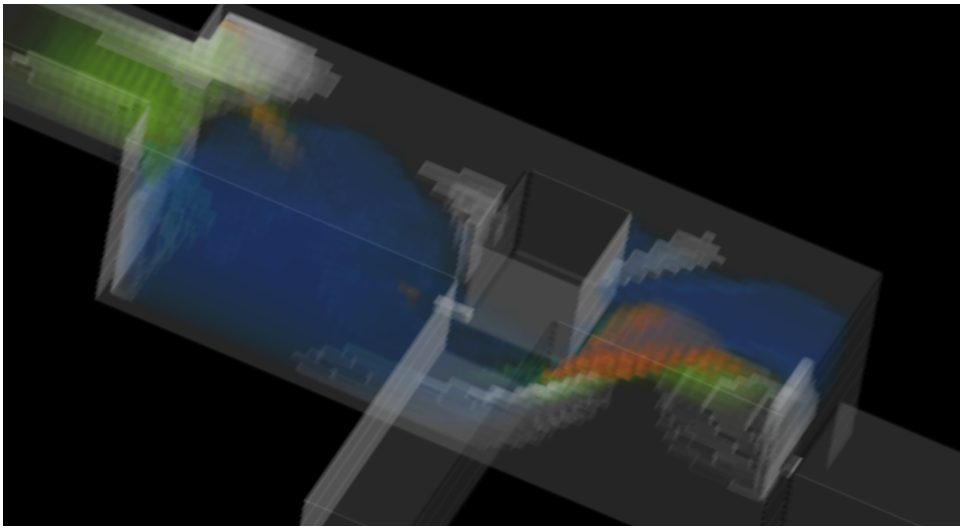


Figure 4.9: Flow through the extended T-junction. Volume rendering was used to show flow subsets which exhibit relatively high values of turbulent-kinetic energy while also having a strong flow component into(green)/against(red) the main direction of flow. Another volume-rendered subsets of the data denotes regions of medium-hot fluid, whereas a gradient-dependent isosurface was used to work out those parts of the flow where a larger y-component of the flow is given, i.e., towards the viewer (more details: section 4.6.2).

on the evenly-spaced streamline seeding algorithm but only are placed in regions of high user-interest.

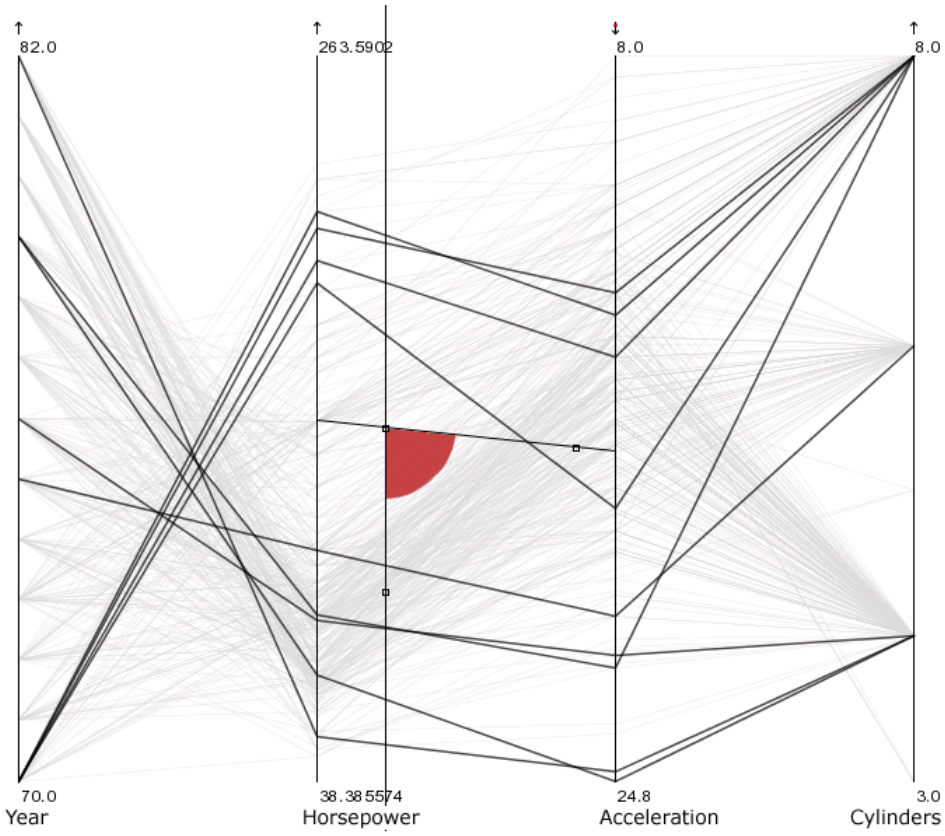
Conclusions of this work are (a) that multiple data attributes usually are necessary to really comprehend results from computational flow simulation and that for this reason a flow visualization system should offer easy-to-handle access to all the information which is provided by the simulation, (b) that selective visualization, in our case, focus+context visualization of 3D flow data is a really good approach to cope with the large amount of information to be communicated – for deciding *what* to show, the user and his/her questions about the data should be considered, and (c) that due to the different requirements of different cases it is very useful to have an integrated solution which provides different means/techniques for the visualization.

Acknowledgements

This work has been at the VRVis Research Center in Vienna, Austria (`http://www.VRVis.at/`). VRVis is funded by an Austrian governmental program called K plus. All data visualized in this paper are courtesy of AVL List GmbH. Special thanks go to Helmut Doleisch, Martin Gasser, Markus Hadwiger, Robert Kosara, and Lukas Mroz for their help in the scope of this project.

Chapter 5

Angular Brushing of Extended Parallel Coordinates



In 2002, the contents of this chapter have been accepted for publication in the Proceedings of the IEEE Symposium on Information Visualization 2002 (*IEEE InfoVis 2002*) as short paper (paper "**Angular Brushing of Extended Parallel Coordinates**" by Helwig Hauser, Florian Ledermann, and Helmut Doleisch).

The contents of this chapter (paper) are a result from a collaborative project with Florian Ledermann (one of Helwig Hauser's undergraduate students) and Helmut Doleisch (a collaborating colleague). Related papers (co-authored by Helwig Hauser) are:

- **Angular Brushing for Extended Parallel Coordinates** [38], the shortened version of this paper, published in the Proceedings of IEEE InfoVis 2002
- **Smooth Brushing for Focus+Context Visualization of Simulation Data in 3D** [20], a tightly related paper, and **Interactive Feature Specification for Focus+Context Visualization of Complex Simulation Data** [19], also a tightly related paper

Angular Brushing of Extended Parallel Coordinates

Helwig Hauser, Florian Ledermann, and Helmut Doleisch

Abstract

In this paper we present several extensions to the well-known InfoViz technique of parallel coordinates, mainly concentrating on brushing and focus+context visualization. First, we propose *angular brushing* as a new approach to high-light rational data-properties, i.e., features which depend on two data dimensions (instead of one). We also demonstrate *smooth brushing* of parallel coordinates as an intuitive tool for specifying non-binary degree-of-interest functions (then used for F+C visualization). Thirdly, we show how *composite brushes* provide lots of flexibility during data exploration. Additionally, we also present several further and more general extensions to parallel coordinates which there are (a) *histograms* to be used as axis overlays, (b) *interaction features* such as axis re-ordering, flipping, scaling and panning, and (c) *detail on demand*, implemented as a mouse-over effect.

Keywords: information visualization, parallel coordinates, brushing, linear correlations, focus+context visualization

5.1 Introduction

For many years already, parallel coordinates [47, 49, 48] are well-known and an often used technique in information visualization (InfoViz). Multi-dimensional datasets are visualized by drawing a poly-line – for every n-dimensional data-point – across coordinate axes which are laid out in parallel, side-by-side (see figure 5.1 for an example, four out of a dozen dimensions shown). The poly-lines intersect the parallel coordinate axes at points which correspond to the respective values (dimensions) of the particular data-points. Thereby, up to a dozen of data dimensions (or even more) are well visualized, especially when the visualization is combined with proper interaction features such as real-time re-ordering of coordinate axes or interactive brushing in single dimensions [86]. If used as an interactive tool, parallel coordinates are especially effective in visualizing correlations between neighboring axes, outliers, etc.

Brushing – as already mentioned above, brushing is a very effective technique for specifying an explicit focus during information visualization [7, 133, 138]. The

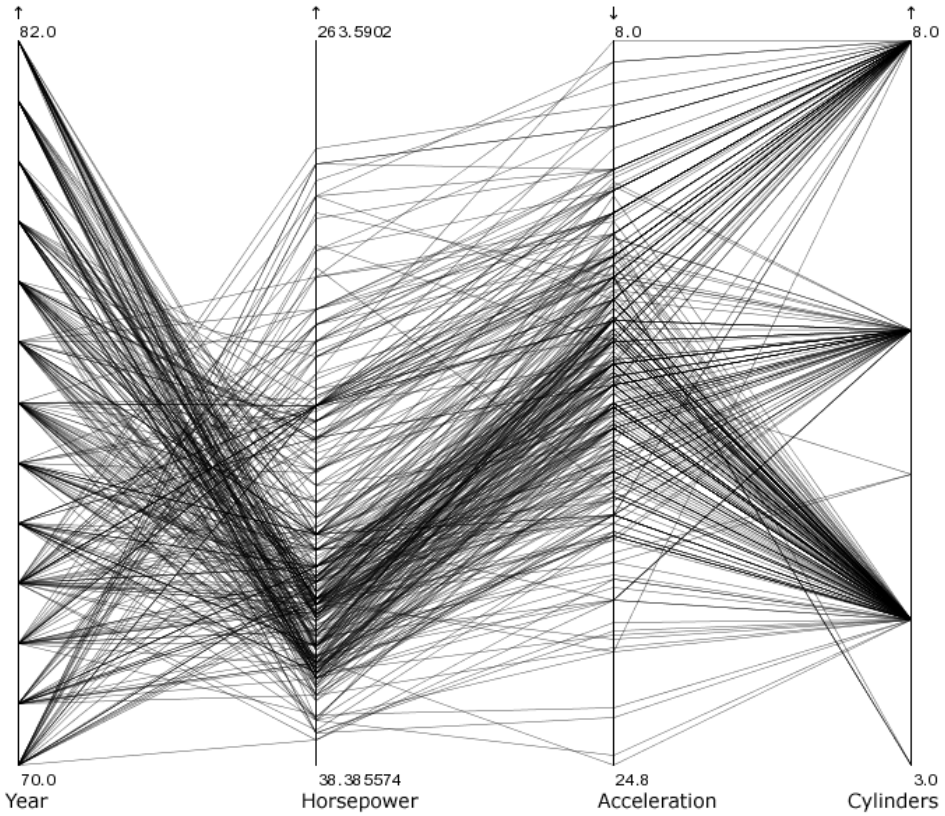


Figure 5.1: Parallel coordinates, a sample view: the *cars* data-set is shown, four (out of twelve) data dimensions are visualized.

user actively marks sub-sets of the data-set as being especially interesting, for example, by using a brush-like interface element. If used in conjunction with multiple linked views, brushing can enable users to understand correlations across multiple dimensions [7, 10, 44, 34].

Focus+context visualization – brushing also is very useful to steer a drill-down into visualizations of really big data-sets [119] – by specifying a (limited and limiting) focus, more details can be shown for the selected data-points. This relates to another very important InfoViz concept which is focus-plus-context (F+C) visualization [30, 11]. The basic idea of F+C visualization is to jointly visualize some parts of the data in detail – usually those parts which are of prime interest at a particular point in time – and also show the rest of the data-set, but in reduced style (smaller, accumulated, etc.) – this is done to maintain the user's orientation and to ease navigation in very large data-sets.

Degree-of-interest functions – for specifying which parts of the data are to be drawn in detail, i.e., for specifying the focus, a degree-of-interest (DOI) function

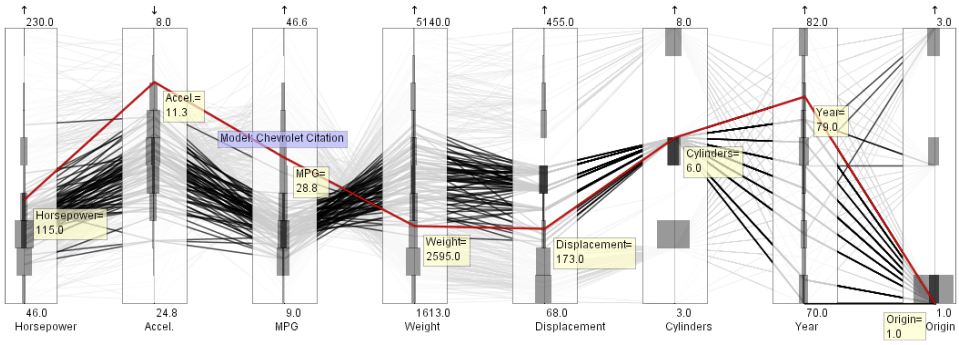


Figure 5.2: Extended parallel coordinates, a sample view of the *cars* data-set: cars with six cylinders were emphasized through brushing, histograms are laid over axes, and one data-point is shown with all details.

is used [30], usually assigning a value of 1 or 0 to each data-point, depending on whether the data-point is of current interest or not. Brushing is an effective method to actively (and explicitly) assign values of 1 to data-points of interest [7, 138].

Brushing extensions – in addition to standard brushing, several useful extensions have been proposed previously. The introduction of composite brushes [86], for example, enabled users to more specifically define their focus. However, it was also shown that interaction elements of the user interface must be chosen carefully due to the very high number of possible ways of compositing a complex brush [138]. The combination of brushing and non-binary degree-of-interst functions [86] proved to be useful for data-sets with gradual changes in at least one of their data attributes, for example, data from scientific simulation [20]. Also, rather complex extensions to brushing have been proposed as, for example, structure-based brushes, dealing with hierarchical data-sets [27].

In this paper, we propose several small but very effective extensions to brushing as an interaction technique for parallel coordinates (like angular brushing or smooth brushing, see section 5.2). Thereby, parallel coordinates can be furthermore improved as an effective tool for information visualization and for visual data mining.

General extensions to parallel coordinates – during the long time that they have been an integral part of InfoViz research, several extensions to the original parallel coordinates have been presented. A hierarchical extension was proposed [26], for example, integrating automatic clustering to deal with really big data-sets. Furthermore, a number of 3D extensions to parallel coordinates were presented [135], including extruded parallel coordinates or linking with wings, for example. Also, higher-order parallel coordinates were proposed [125], using higher-order splines to interpolate values on more than two axes.

In this paper, we also present a number of further general extensions to parallel coordinates, which we have found useful during data investigation (such as adding histograms on single axes or a detail-on-demand option, see figure 5.2 and section 5.3).

The remainder of this paper is structured as follows: first we present several extensions which special focus on brushing (section 5.2). Then, we continue with further, more general extensions to parallel coordinates (section 5.3). Later in this paper, we also shortly present some implementation issues, talking about fast rendering, for example (section 5.4). The application itself and some results from visual data analysis are demonstrated in section 5.5.

5.2 Extended Brushing for Parallel Coordinates

As already discussed above shortly, brushing effectively enables the user to (explicitely) specify his or her focus. In parallel coordinates, this usually is accomplished by marking a certain data sub-set of interest according to values within a single data dimension. In the *cars* data-set [132], for example, a user could focus on cars with four cylinders by brushing the respective portion of the *cylinders*-axis. See figure 5.3 for a sample view – note, that the third axis ("Acceleration") was flipped, since values are given in seconds 'til 60 MPH, and therefore a smaller number represents a higher acceleration.

In our application of visually investigating multi-dimensional and large data-sets originating in a physically-based simulation of dynamic processes (computational fluid dynamics, CFD, for example, see section 5.5 for more details), we found standard brushing of parallel coordinates very useful, however, we needed to go further. The requirement was to specify a focus not only in dependence of one data attribute (one dimension), but to do so with respect to at least two of them – like brushing all those data-points which feature a z-velocity larger than the respective x-velocity. The consequence was to introduce a new brushing technique which we called *angular brushing.*

5.2.1 Angular Brushing of Parallel Coordinates

One strength of parallel coordinates is its effectiveness of visualizing relations between coordinate axes. Bringing axes next to each other in an interactive way, the user can investigate how values are related to each other with special respect to two of the data dimensions. This way, it is not only possible to see whether data-points exhibit high or low values in single dimensions, but also their relation can be understood. More specifically, the slope of the line-segments in-between two axes tells the user, whether there is a positive or negative correlation in-between values (of course assuming that the user set up a proper mapping). Outliers also

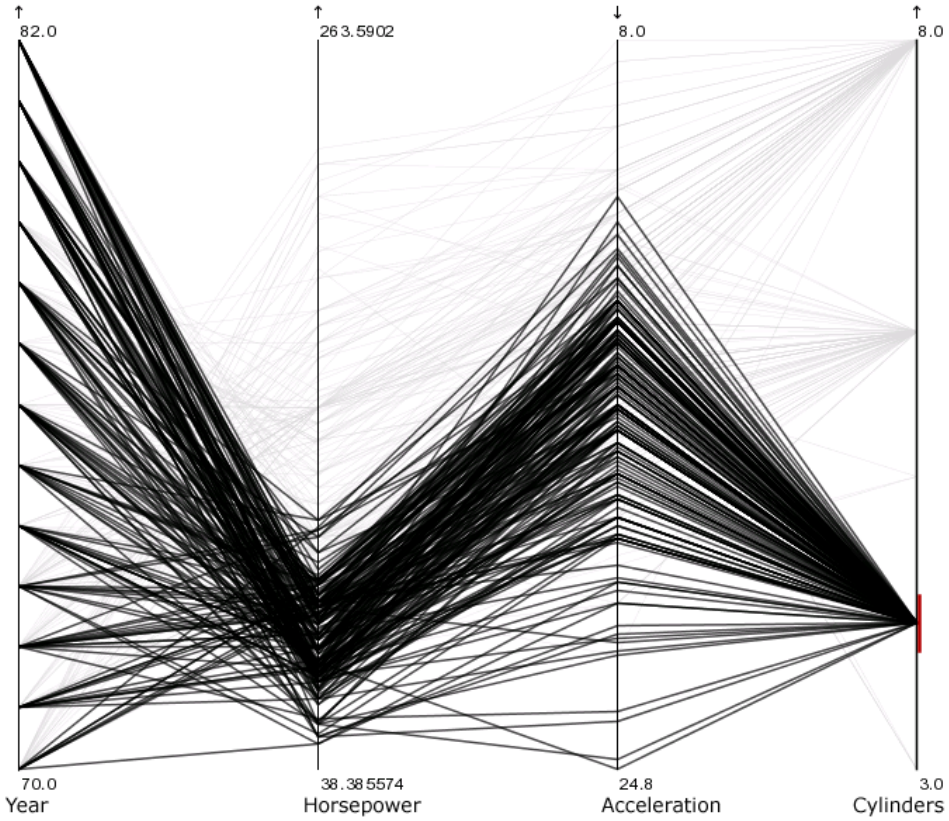


Figure 5.3: Brushing parallel coordinates, an example: all cars with four cylinders are marked and emphasized (only four data dims. shown).

can be easily found with respect to two data dimensions – when all but a few line-segments have a positive slope, then the few others clearly stand out (see figure 5.4 for an illustration).

In this paper, we demonstrate how this feature of parallel coordinates easily can be exploited for an extended brushing technique, i.e., angular brushing. In addition to standard brushing, which primarily acts along the (parallel) axes, we enable the space in-between axes for brushing interaction, as well. The user can interactively specify a sub-set of slopes which then yields all those data-points to be marked as part of the current focus, which exhibit the matching correlation in between the brushed axes. See figure 5.4 for an example, where all negative slopes have been marked in-between the second and third axis.

Angular brushing as described above is a useful extension to parallel coordinates, also because it very well corresponds to the effective visual cues provided by parallel coordinates. Inherently, angular brushing opens the extended field of brushing multi-dimensional properties of high-dimensional data-sets. In addition to composite brushes which are composed multiple single-axis brushes by the use

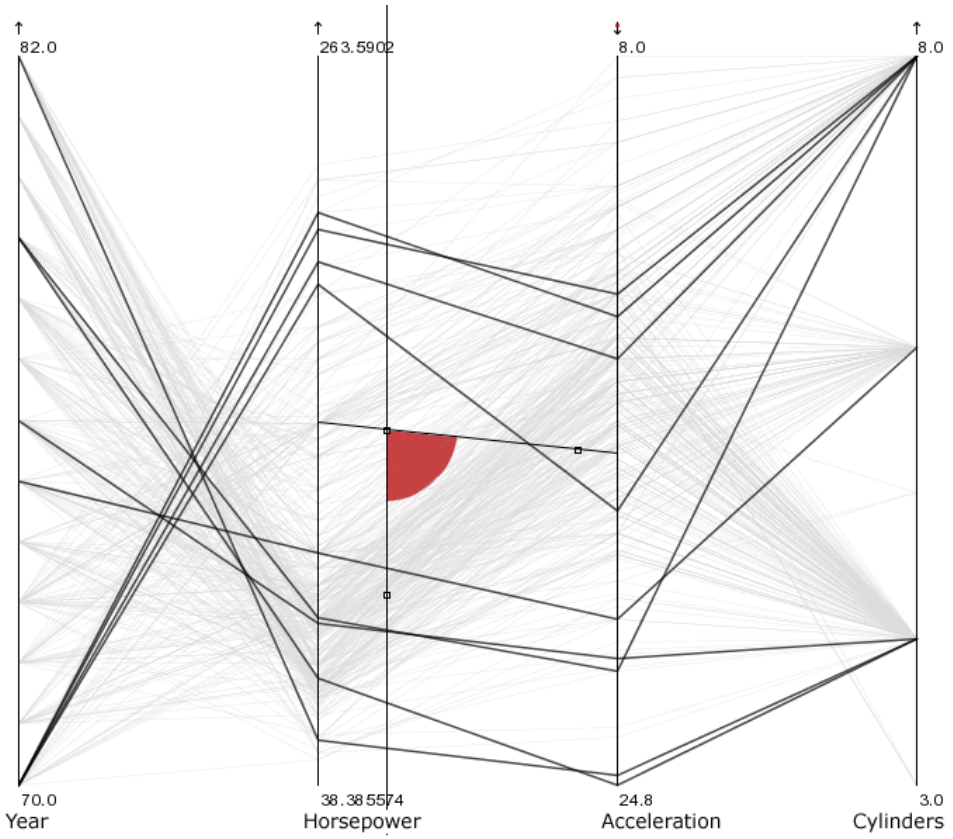


Figure 5.4: Reading between the lines: whereas most line-segments go up in-between the 2nd and the 3rd axis (visualizing a positive correlation of values there), just a few go down – those have been emphasized through angular brushing.

of logical operators (see also below in section 5.2.3), angular brushes allow the selection of rational properties with respect to two of the data dimensions. When compared to composite brushes by the use of a scatterplot visualization, we clearly see that brushes based on logical operators select sub-sets which are aligned with the display axes, whereas angular brushes select sub-sets which are aligned with the diagonals when visualized in a scatter-plot (see also figure 5.5 for comparison).

5.2.2 Smooth Brushing and Non-Binary DOIs

Many well-known F+C techniques in InfoViz such as fisheye views [29, 30], for example, do not use a discrete distinction between focus and context, but allow a multi-valued or even continuous transition, which inherently supports the mental connection between data-points in focus and their context. This corresponds to a degree-of-interest function, which non-binarily maps into the $[0, 1]$-range.

Often, such a non-binary DOI function is defined by means of spatial distances, i.e., the DOI-value reflects the distance of a data-point from a so-called center-of-

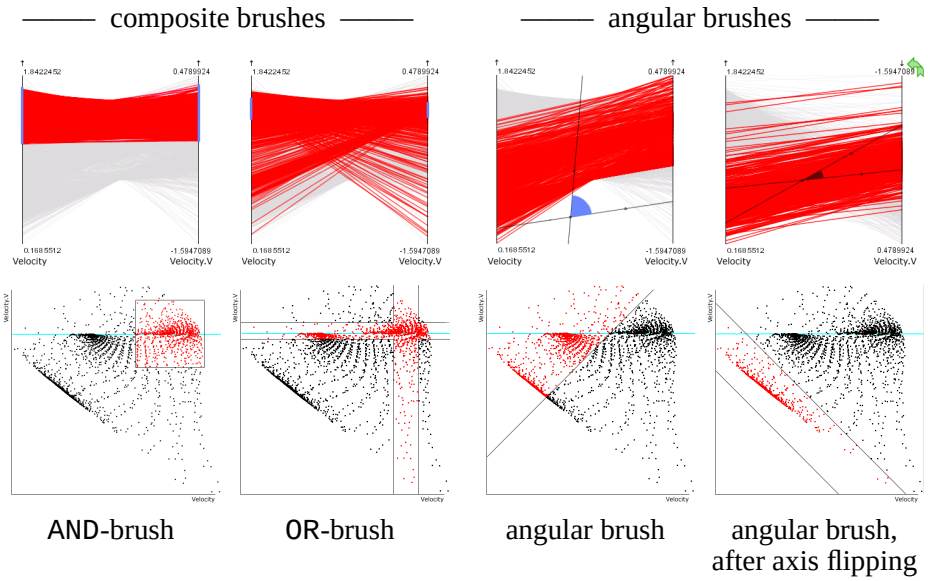


Figure 5.5: Comparing composite brushes (on the left) and angular brushes (on the right), using parallel coordinates (top row) and scatterplots (bottom row): composite brushes address "horizontal/vertical" features, whereas angular brushes emphasize "diagonal" feature.

interest (COI), which in these cases often is specified explicitly, for example, by pointing to it. In the implicit case, i.e., when querying is used for the specification of the DOI function, fuzzy logic can be used in the style of probabilistic responses.

Brushing also easily combines with non-binary DOI functions [86], i.e., when smoothly delimited brushes are used. Thereby, at the brush boundaries, fractional DOI-values between 0 and 1 are assigned. When composite brushes are composed through the use of logical operators, fuzzy logic is employed to aggregate final DOI-values for every single data-point. In previous work, we already demonstrated the useful application of smooth brushing in the F+C visualization of multi-dimensional data, originating in 3D flow simulation [20].

In conjunction with parallel coordinates we extend the interactively specified brush by a certain, user-defined percentage to accommodate a smooth gradient of DOI-values at its borders. When combining smooth brushes through logical operators, we apply the Łukasiewicz norm [118], known from fuzzy logic as a comparably large norm [60], i.e., a norm which (when applied repeatedly) only slowly converges to 0 and therefore better conserves the transitional DOI-gradients, for computing AND- as well as OR-combinations. See figure 5.6 for an example of smoothly brushed parallel coordinates.

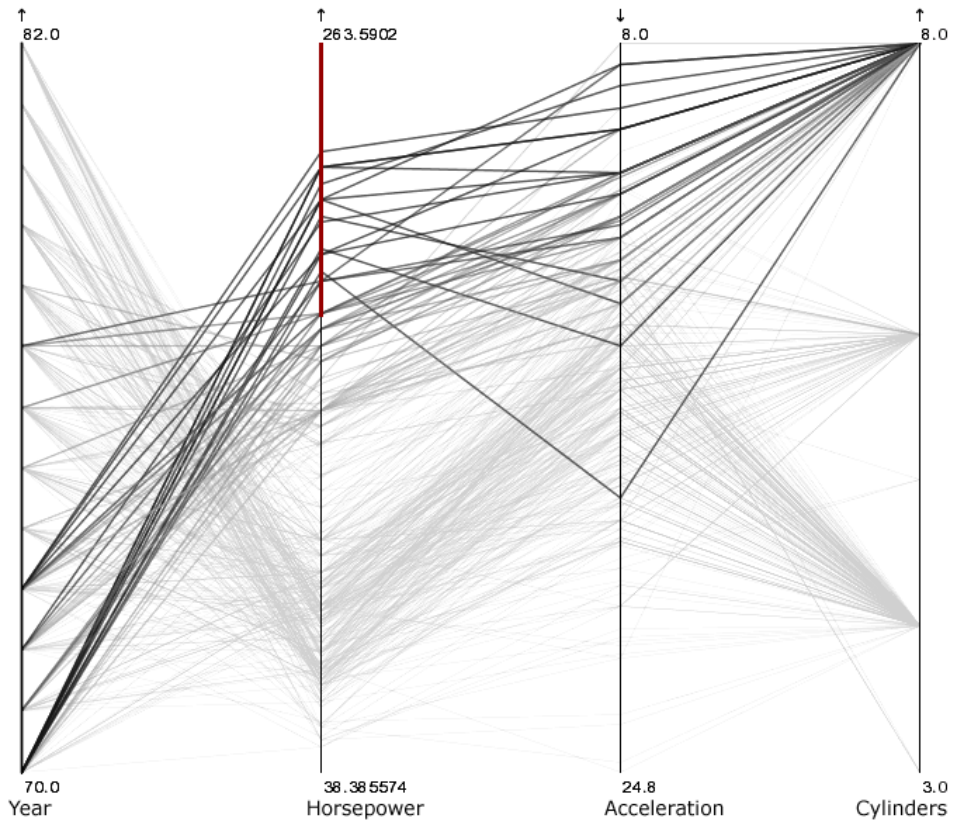


Figure 5.6: Smooth brushing, an example: note the gradual changes of drawing intensity which reflect the respective degree of interest, after smooth brushing of the 2nd axis.

5.2.3 Composite Brushes for Parallel Coordinates

Shortly after the original idea of brushing, also the logical combination of brushes was proposed as a useful extension [86]. As soon as the user's imagination about the features he or she wants to see within a data-set gets more and more concrete, a need for more flexible and multi-dimensional descriptions of features arises. User-centered specifications like *all those cars which are fast but do not cost a lot*, for example, need to be supported by visualization. Composite brushes, i.e., brushes which are composed of simple ones by the use of logical operators, are an intuitive solution to the above mentioned requirement.

In our application (see also section 5.5), composite brushes were to be enabled on the basis of single-axis brushes, angular brushes (see above, section 5.2.1), and smooth brushes (see above, also). We therefore implemented the following features to end up with a consistent solution:

Multiple brushes – whenever a new brushing interaction is triggered, a new brush is instantiated, represented by a new entry in a separate brush list (sep-

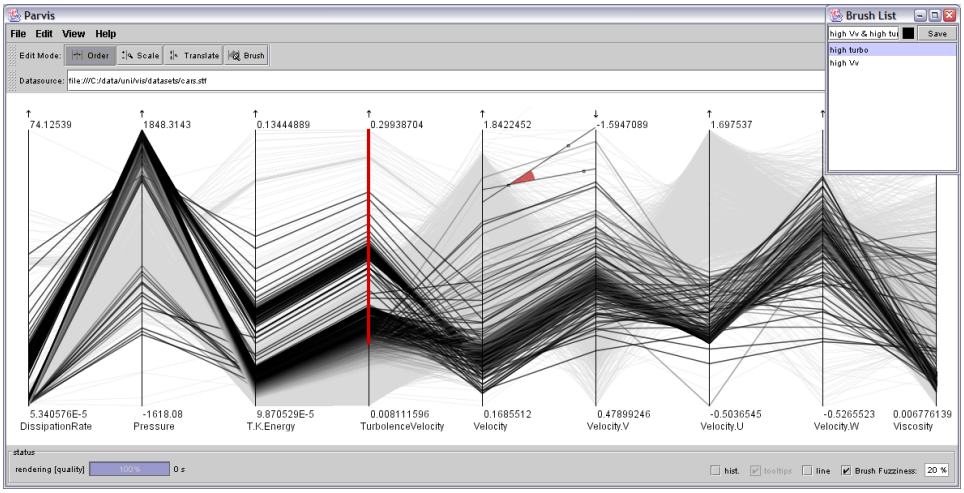


Figure 5.7: An example of a composite brush (AND): a CFD data-set is shown with all data-points high-lighted, which exhibit high values of turbolent velocity (single-axis brush, smoothly along "TurbolenceVelocity"-axis) and relatively low y-velocity ("Velocity.V") when compared to 3D-velocity (angular brush after axis flipping).

arated part of the user interface). New brushes can be named by simply editing the new entry in the brush list. Hereby, a useful overview is provided while still enabling the user to switch between previously defined brushes (to compare them, for example), or to combine them to form composite brushes.

Composite brushes – single brushes easily can be combined to form composite brushes through the application of logical operators. A nice little feature in our application is that new (composite) brushes automatically derive their names from the brushes involved in the composition. Composite brushes also can be constructed by adding new constraints interactively, using modifier keys through interaction. As already described above, composite brushes are computed based on the Łukasiewicz norm, when smooth brushes are involved (see also figure 5.7).

Support of linking – in our application, DOI-values are considered to be first-rank data-derivatives, i.e., brushes can be loaded from and saved to disk, or exchanged with other views, for example. Thereby, the here described parallel coordinates also can be used as one of several linked views, supporting a *linking-and-brushing* style of application. See figure 5.8 for an example where parallel coordinates were combined with a 3D scientific view. In this example, a scatter-plot was used for brushing and the parallel coordinates as well as the 3D view are linked for F+C visualization.

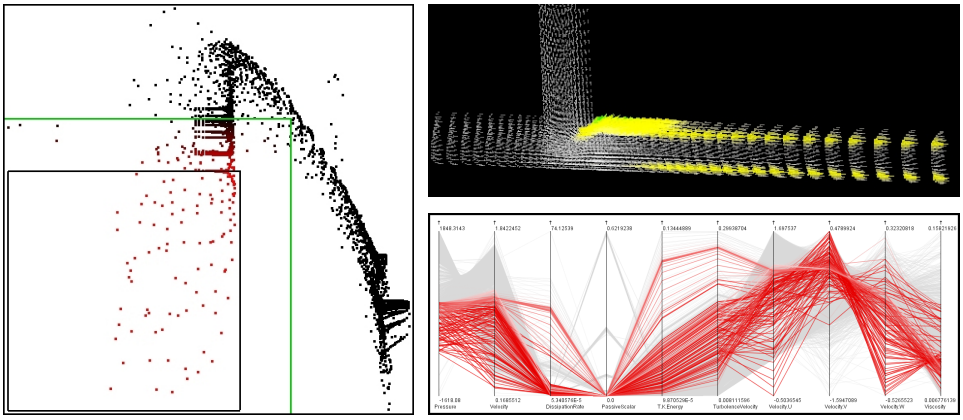


Figure 5.8: Linking and brushing, an example: in a scatter-plot (shown on the left side), smooth brushing was used to mark data-points of low pressure and low velocity; a linked 3D view (on the top right) shows the same data with the brushed data-points high-lighted; thirdly, the parallel coordinates view (on the lower right) also shows the same data, also high-lighting the brushed sub-set.

5.3 Further Extensions for Parallel Coordinates

In addition to the brushing extensions as described above, we also developed several further extensions to parallel coordinates, which we found useful during data investigation. Next to simple but yet very useful interaction features such as coordinate flipping, the addition of histograms along all of the parallel axes was found to be especially useful.

5.3.1 Histograms over Parallel Coordinates

Similar to a previously published solution [138], we also compute histograms for all the axes shown in the parallel coordinates view. This is especially useful, when many data-points are to be shown. While users might be troubled with resolving clutter problems with an over-full display, histograms give a very intuitive clue on where data-points accumulate along coordinate axes.

Depending on a user-defined sub-division, multiple bins of equal extension are generated, and the data-points are sorted into these bins during the build-up of the parallel coordinates view. Afterwards, a semi-transparent representation of the bins' cardinality is plotted on top of the parallel coordinates view, comparable to a histogram along this data dimension. See figure 5.2 for an example of parallel coordinates plus histogram visualization along axes.

5.3.2 Flexible Layout with Parallel Coordinates

From the early beginning of parallel coordinates on, interaction was a crucial component of parallel coordinates views to enable the user to work with the view and to interactively explore the data under investigation.

In our application, we implemented a number of partly obligatory and partly new features in conjunction with the goal of providing a flexible layout the user can work with:

Axes re-ordering – probably most important is that the user can interactively re-order the layout of the axes in use. In our application, single axes can be dragged around, and automatically position themselves right in the gap (between two other axes, or at the boundary), where the user drops them.

Flipping of axes – also very useful for data investigation is an option which allows users to flip certain axes, i.e., to require a top-down ordering of values instead of the standard bottom-up style. By doing so, correlations between axes can be better investigated. Also, this option is especially important in conjunction with angular brushing as it enables the user to brush sub-sets, which – in a scatter-plot – would be aligned along the sub-dominant diagonal (see figure 5.5).

Changing the mapping – the default in our application is to use a min-max mapping of values to coordinate axes, i.e., minimal and maximal values are extracted before mapping, then values are linearly mapped such that minimal values come lowest whereas maximal values are mapped to the top of the coordinate axes. Additionally, two other modes can be used: (a) The user can enforce that zero is mapped to the same height along all the coordinate axes. (b) The min-max mapping can be correlated along all the axes, resulting in a consistent mapping for of them. Of course, if not all of the scales are comparable, this mode can result in a roughly distorted view.

Axis scaling and panning – the user can interactively scale and pan the mapping for single axes. On the first sight, this only adds more flexibly with adjusting the view layout. In our application, this option also provides important means when working in-between dimensions. Especially in conjunction with angular brushing, the ability to scale and pan one of the two axes (later to be brushed), proved to be very useful.

Deletion and addition of axes – in addition to interactive re-ordering of axes as shortly described above, we also enable the user to interactively delete single axes from the view. This is used to clarify the view, when not all of the information needs to be shown concurrently. We also allow the addition of axes to the view, even if they are already in use: often, single data dimensions are of special interest to the user and then it is very useful to have such an axis repeatedly in the view, especially as correlations in-between axes are best visualized when axes are next to each other.

5.3.3 Detail on Demand

We also implemented a detail-on-demand feature, which we realized with a mouse-over effect. Thereby the user can interactively browse through the display and whenever the mouse moves over a poly-line, all the details are shown for the respective data-point. We also make use of this feature to display textual data attributes, which by default are not drawn as axes at all (problem of mapping).

5.4 Implementation

To deal with real-world data-sets – in our case consisting of 10^4–10^6 data-points each – we quickly experienced the limitations of a brute-force implementation of parallel coordinates. Simply drawing multiple line-segments for each and every data-point already poses quite a load on the graphics hardware – for data-sets as in our application, quickly millions of line-segments need to be drawn. We also needed semi-transparency, coloring, and anti-aliased line-segments to nicely integrate all of the features described in this paper.

In the following, we shortly describe some of the approaches we utilized to still provide real-time interactivity:

Exploiting coherence – during interaction, the application seeks to re-use as much of the current visual output as possible. When single axes are flipped, for example, only the inter-axis space on the left- and right-hand side of the flipped axis are re-rendered, all the rest is reused.

Progressive rendering – for immediate feed-back the application uses a preview-style of rendering, i.e., without anti-aliasing or semi-transparency. As soon as there is time for better rendering, the application computes a high-quality result and replaces the preview with the improved image.

Bi-threaded implementation – to de-couple rendering and interaction, we use a bi-threaded implementation. One thread serves the user interface and guarantees interactive response, while the other thread deals with rendering, which sometimes lasts for a number of seconds, especially for bigger data-sets.

5.5 Application and Results

In our application we investigate multi-dimensional data-sets which originate in CFD simulations (run by one of our partners), which usually exhibit a dozen of data dimensions, or more. During flow simulation, values like temperature, pressure, velocity, dissipation rate, and many others, are computed for all cells within a

detailed CFD grid. Data like this needs careful analysis which usually is non-trivial, not only because the data is of high dimensionality, often laid out in 3D space or even time-dependent.

Parallel coordinates, with all the extensions presented in this paper, proved to be very useful for data investigation. The interactive character of the here described implementation allows for fast and flexible data exploration, even when simultaneously investigating multiple dimensions. Linking parallel coordinates and other types of views (like a 3D SciViz view, shown in figure 5.8 on the top right) and using our brushing features eased data exploration. Whereas parallel coordinates are used to find interesting data-points, a linked 3D view can simultaneously show where those data-points reside in 3D space. Also the combination with scatter-plots was found to be useful, primarily because composite brushes are specified more easily in a scatter-plot than in parallel coordinates.

Figures 5.5, 5.7, and especially 5.8 show results of data exploration by the use of parallel coordinates. The data shown is the result of a simulation of flow through pipes shaped as a T-junction. Flow is coming from the left and the top (when referring to figure 5.8, top left) and leaving the T-junction to the right. In figure 5.7, a composite AND-brush was used to mark all those data-points which, on the one hand, exhibit rather high values of turbolence (smoothly brushed) and which are part of flow regions that are well aligned with the negative y-axis ("Velocity.V"), on the other hand.

5.6 Summary and Conclusions

In this paper, we have presented several useful extensions to parallel coordinates, especially focussing on brushing interactions. We proposed *angular brushing* as a new technique to effectively select data sub-sets which exhibit a data correlation along (at least) two axes. We also showed how *smooth brushing* can be used, even in combination with *composite brushes.* We also showed how histgrams can be used as axis overlays to further support data exploration.

Concluding, we may say that we re-experienced the advantages of an interactive visualization tool when doing data exploration. We also conclude that the worth of a parallel coordinate visualization increases drastically when interaction features like axis re-ordering, etc., are added.

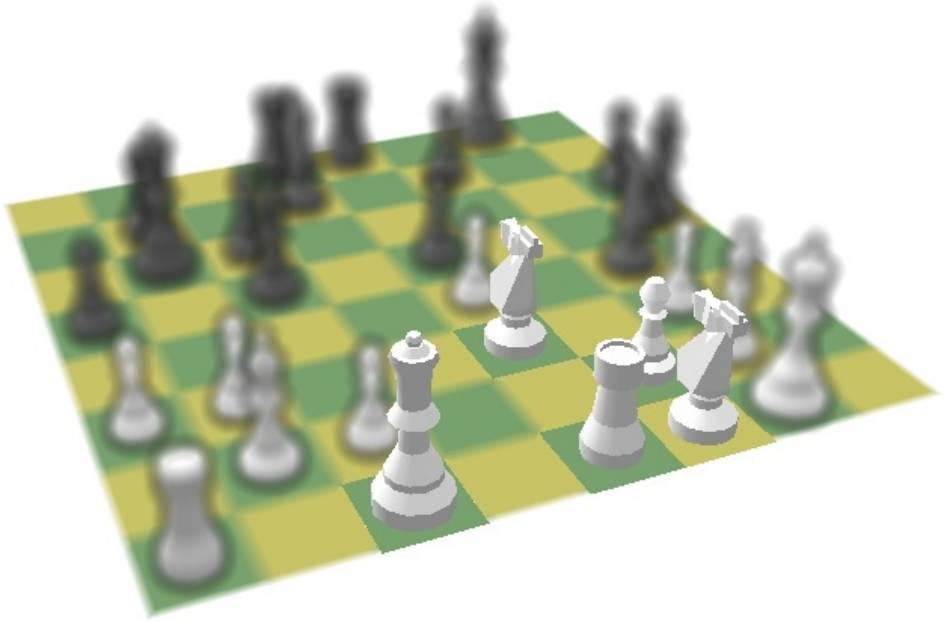
Acknowledgements

Parts of this work have been done in the scope of the basic research on visualization (http://www.VRVis.at/vis/) at the VRVis Research Center (http://www.VRVis.at/) in Vienna, Austria, which is funded by an Austrian governmental research program called K plus. The authors also thank AVL

(`http://www.AVL.com/`), i.e., one of VRVis' partner companies, for providing real-world data to evaluate our software prototype. Last, but not least, the authors thank Robert Kosara for his valuable hints during the preparation of this paper.

Chapter 6

Semantic Depth of Field



In 2001, the contents of this chapter have been published in the Proceedings of the IEEE Symposium on Information Visualization 2001 (*IEEE InfoVis 2001*), pp. 97–104 (paper "**Semantic Depth of Field**" by Robert Kosara, Silvia Miksch, and Helwig Hauser).

The contents of this chapter (paper) are a result from a collaborative project with Robert Kosara (one of Helwig Hauser's PhD students) and Silvia Miksch (Robert Kosara's university-side PhD supervisor). Related papers (co-authored by Helwig Hauser) are:

- **Focus + Context Taken Literally** [65], an extended version of this paper
- **Useful Properties of Semantic Depth of Field for Better F+C Visualization** [66], a related paper about results from a user study about semantic depth of field

Semantic Depth of Field

Robert Kosara, Silvia Miksch, and Helwig Hauser

Abstract

We present a new technique called *Semantic Depth of Field* (SDOF) as an alternative approach to focus-and-context displays of information. We utilize a well-known method from photography and cinematography (depth-of-field effect) for information visualization, which is to blur different parts of the depicted scene in dependence of their relevance. Independent of their spatial locations, objects of interest are depicted sharply in SDOF, whereas the context of the visualization is blurred. In this paper, we present a flexible model of SDOF which can be easily adopted to various types of applications. We discuss pros and cons of the new technique, give examples of application, and describe a fast prototype implementation of SDOF.

6.1 Introduction

Whenever large amounts of data are to be investigated, visualization potentially becomes a useful solution to provide insight into user data. Especially for exploration and analysis of very large data-sets, visualization not only needs to provide an easy-to-read visual metaphor, but also should enable the user to efficiently navigate the display, allowing for flexible investigation of arbitrary details.

Focus and Context (F+C) techniques enable the user to investigate specific details of the data while at the same time also providing an overview over the embedding of the data under investigation within the entire dataset. But F+C encompasses a number of very different techniques that achieve similar goals in very different ways.

6.1.1 Different Kinds of Focus and Context

The most prominent group of F+C methods are *distortion-oriented* [77] or *spatial methods.* The geometry of the display is distorted to allow a magnification of interesting information without losing the (less magnified) context. It is thus possible to navigate information spaces that are far too large to be displayed on a screen. Examples are fisheye views [30, 114], hyperbolic trees [68, 73, 101], stretchable rubber sheets [115], etc. Distortion-oriented techniques are usually used in an explicit way, by actively bringing the interesting objects into focus, e.g. by clicking on objects or dragging them around.

For smaller numbers of objects that have a lot of data associated with them, a visualization method is useful that shows just a limited number of data dimensions, and allows the user to select which of the objects are to be shown in more detail – we call these *dimensional methods.* The context in this case are not only the other objects, but also the remaining data dimensions. This type of method also shows more detail, but in terms of data dimensions, not screen size. Examples are magic lenses [123] and tool glasses [8], where the user moves a window over the display, the objects inside which are displayed in more detail.

The third type of focus and context allows the user to select objects in terms of their features, not their spatial relations; usually by assigning a certain visual cue to them – we therefore call these methods *cue methods.* They make it possible to query the data for information which is not immediately visible in the initial visualization, while keeping the original layout, and thus not destroying the user's mental map [93]. An example for such a system is a Geographic Information System (GIS) that makes it possible to display crime data, certain cities, or hospitals [82]. This data is displayed in the same context as before, but the relevant parts of the display have a higher color saturation and opacity than the rest. This leads the viewer's attention to the relevant objects easily without removing context information. In contrast to distortion-oriented techniques and magic lenses, with this type of method, the user first selects the criteria, and then is shown all the objects fulfilling them.

The technique presented in this paper is of the third type, but we use a different visual cue for discriminating focus and context.

6.1.2 The Uses of Blur and Depth of Field

Blur is usually considered to be an imperfection: it makes features harder to recognize and can hide information completely. But the difference between sharp and blurred parts of an image is a very effective means of guiding the viewer's attention. In photography, the depth-of-field (DOF) effect leads to some parts of the image being depicted sharply, while others are blurred [1]. The viewer automatically looks at the sharp parts, while the blurred parts provide non-disturbing context for the objects of interest (see Fig. 6.1 for an example). The same effect is also used in cinematography [55], where focus changes can guide the audience's attention from one character to another, from a character to an object he or she just noticed, etc.

Because the human eye (like every lens system) also has limited DOF [32], an important characteristic of human vision is that whenever we get interested in a specific part of our environment, we 1) bring the the object of interest into the center of our eye (where the area of most acute vision, the *fovea centralis*, is located), and 2) focus on that object. From the above applications of DOF (photography and cinematography), we know that this process is easily inverted: If we display sharp

Figure 6.1: A lantern with a bridge as context.

objects in a blurred context, the viewer's attention is automatically guided to the sharp objects. This also gives us reason to believe that DOF is perceived preattentively, i.e. within 50ms of exposure to the stimulus, and without serial search [128]. This means, it very efficiently makes use of the bandwidth of the human visual system to convey a lot of information in very little time.

We have developed an F+C technique we call *Semantic Depth of Field* (SDOF) for information visualization, which renders objects sharp of blurred, depending on their current relevance. It thus makes use of the phenomena described above to guide the viewer's attention.

6.2 Related Work

There have been surprisingly few attempts to use DOF or blur in visualization at all; the ones relevant to this work are shortly summarized here.

In a system for the display of time-dependent cardio-vascular data [139], a stereoscopic 3D display is included that is controlled by the viewer's eyes. Like a microscope, only one thin slice through the data appears sharp, all others are blurred and therefore almost invisible. Eye tracking equipment determines what the user is looking at, and that point is brought into focus. This makes it possible to concentrate on one detail without the surrounding structures confusing the viewer. Later work [140] describes "non-linear depth cues", which means displaying structures that currently are of interest (like single organs) in focus, and other objects out of focus, not based on their distance from the camera, but on their importance.

The Macroscope [79] is a system for displaying several zoom levels of information in the same display space. For this purpose, the images on all levels are drawn over each other, with the more detailed ones drawn "in front", i.e., drawn over the less magnified layers. The layers' transparency can be changed so that the background (context) can be more or less visible. The less detailed layers are blurred so as to not distract the viewer, but serve as context.

The most interesting existing approach for this work is a display of geographical information [14]. In this system, up to 26 layers of information can be displayed at the same time. Each layer has an interest level associated with it that the user can change. The interest level is a combination of blur and transparency, making less interesting layers more blurred and more transparent at the same time. This work does not seem to have been followed up on recently.

In this paper, we describe a general model of SDOF, i.e., of selectively using sharpness vs. blur to emphasize/deemphasize certain parts of the data. We clearly embed SDOF within the scope of information visualization and computer graphics. In addition to the above examples, we provide a flexible solution which easily is adopted to various kinds of applications, as demonstrated later on.

6.3 Semantic Depth of Field (SDOF)

SDOF allows the user to select relevant parts of a visualization that are then pointed out by deemphasizing all the rest through blur. The discrimination between relevant and irrelevant objects can be binary (an object is either relevant or irrelevant) or continuous (an object can have a relevance value between the two extremes).

Different relevance metrics for objects have to be offered by the application, that have to deal with the specific information and tasks the application is made for. Examples for binary relevance measures are the set of chessmen that threaten a specific piece in a chess tutoring system (see Fig. 6.5c and the accompanying

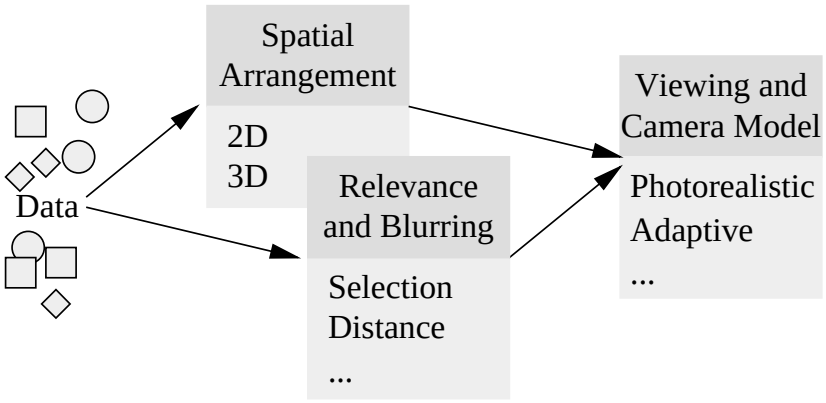


Figure 6.2: SDOF Building Blocks.

video), the layer containing roads in a GIS application (Fig. 6.5d), or all incidents related to high blood glucose in a graphical patient record. Continuous functions could express the age of files in a file system viewer (Fig. 6.5a), the recent performance of stocks in a stock market browser, or the distance of cities from a specified city in terms of flight hours.

The building blocks of SDOF are discussed in the following subsections, and are summarized in Fig. 6.2 and Tab. 6.1.

6.3.1 Spatial Arrangement

In information visualization, usually some kind of layout algorithm is used to arrange objects in the visualization space (typically 2D or 3D). The special challenge of information visualization is the fact that data often does not have any inherent structure that naturally translates to a layout. Mapping functions are a very important part of visualization because they determine how well the user can build a mental map that he or she can use to understand and navigate the visualization. Changing the layout often means having to learn a new layout, and thus losing one's ability to navigate easily.

In our model, the spatial mapping function is called **place**; it translates from the original data space to an intermediate visualization space (2D or 3D).

6.3.2 Relevance and Blurring

Independently of the spatial arrangement, the blur level of each object is determined. This is done in two steps: First, each object is assigned a relevance value r by the relevance function rel. The value of r is in the interval $[0; 1]$, where 1 means

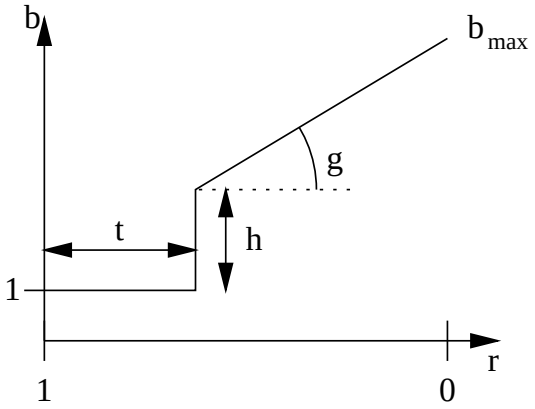


Figure 6.3: The Blur function.

the object is maximally relevant, and 0 means the object is completely irrelevant. This relevance value is translated into a blur value b through the blur function blur.

The relevance function is application-specific and thus can be very different between applications (see Sect. 6.5.2 for examples). The blur function can theoretically also take on any shape, but we have found the function depicted in Fig. 6.3 to be sufficient for our current uses. The user can specify the threshold value t, the step height h, and the maximum blur diameter $b_{\max}$. The gradient g is then calculated by the application.

6.3.3 Viewing and Camera Models

In order to provide a consistent model, and to embed the idea of SDOF in existing work in computer graphics, we discuss camera models for generating images with SDOF. Depending on whether the visualization space is two- or three-dimensional, different camera models can be used to finally achieve the SDOF effect. The camera provides two functions: **camera** projects data values from an intermediate space (where the information was laid out by the **place** function) to screen space; and dof, which calculates the blur level of each data item depending on its z coordinate and the z_{focus} value the camera is currently focused at.

In the following, we describe two camera models: a regular photo-realistic camera ($\mathbf{camera}_{\text{P}}$) can be used in the 2D case; for 3D, we present the *adaptive camera* ($\mathbf{camera}_{\text{A}}$).

2D SDOF and the Photo-realistic Camera

In the 2D case, objects get a third coordinate in addition to their x and y values. This additional z value depends on the intended blur diameter b of the object: If

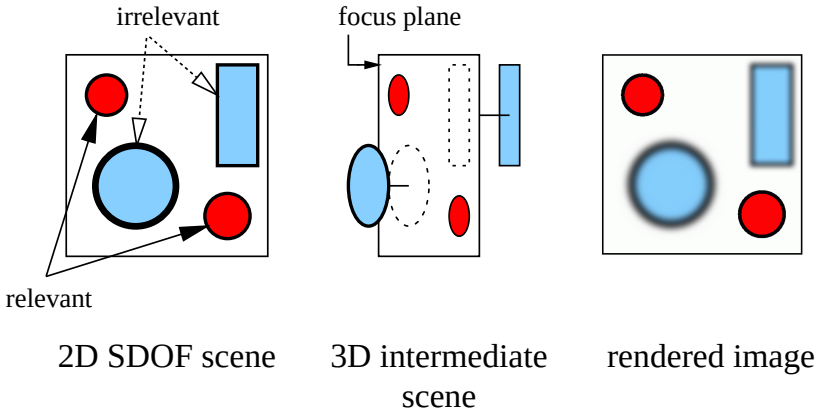


Figure 6.4: The photo-realistic camera and 2D SDOF.

the camera is focused at z_{focus}, an object with intended blur b has to be moved to a distance of z from the lens of the camera (see Fig. 6.4):

$$b = \mathsf{dof}_{\text{P}}(z, z_{\text{focus}}) = \left| D \frac{f(z_{\text{focus}} - z)}{z_{\text{focus}}(z - f)} \right| \tag{6.1}$$

$$z = \mathsf{dof}_{\text{P}}^{-1}(b, z_{\text{focus}}) = \frac{D + b}{\frac{D}{z_{\text{focus}}} + \frac{b}{f}} \tag{6.2}$$

where D is the effective lens diameter as defined in the thin lens model [75], and f is the focal length of the lens in use.

The above equations apply to camera models such as distribution ray tracing [15], linear post-filtering [106], etc.

If the camera uses perspective projection, objects also have to be scaled (and possibly moved) to compensate for depth effects that are not desired in this case.

3D SDOF and the Adaptive Camera

In the 3D case, of course, it is not possible to directly map blur factors to depth values, because the spatial arrangement of data items already contains a third dimension. However, using a simple extension of the photo-realistic camera, it is possible to also handle the 3D case.

The adaptive camera is a modification of a photo-realistic camera that can change its focus for every object point to be rendered. This is easily possible with object-order rendering, but can also be achieved when rendering in image order. In contrast to the photo-realistic camera, the adaptive camera can render SDOF in 2D and 3D scenes. The photo-realistic camera is, in fact, a special case of the adaptive camera (which simply stays focused at the same distance for the whole image).

$$\mathbf{data}[i] \xrightarrow{\mathbf{place}_{2D}} \begin{pmatrix} \hat{x} \\ \hat{y} \\ b = \\ \mathsf{blur}(r) \end{pmatrix} \xrightarrow{\mathbf{view}_{2D}} \begin{pmatrix} x \\ y \\ z = \\ \mathsf{dof}_{\mathrm{P}}^{-1}(b) \end{pmatrix} \xrightarrow{\mathbf{camera}_{\mathrm{P}}} \begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix}$$

with $\xrightarrow{\mathsf{rel}} r \xrightarrow{\mathsf{blur}}$ and $\xrightarrow{\mathsf{dof}_{\mathrm{P}}^{-1}}$

$$\mathbf{data}[i] \xrightarrow{\mathbf{place}_{3D}} \begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ b = \\ \mathsf{blur}(r) \end{pmatrix} \xrightarrow{\mathbf{view}_{3D}} \begin{pmatrix} x \\ y \\ z \\ z_{\mathrm{focus}} = \\ \mathsf{dof}_{\mathrm{A}}^{-1}(b) \end{pmatrix} \xrightarrow{\mathbf{camera}_{\mathrm{A}}} \begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix}$$

with $\xrightarrow{\mathsf{rel}} r \xrightarrow{\mathsf{blur}}$ and $\xrightarrow{\mathsf{dof}_{\mathrm{A}}^{-1}}$

Table 6.1: All steps necessary for visualizing data values $\mathbf{data}[i]$ with 2D (top) and 3D SDOF (bottom).

Function $\mathsf{dof}_{\mathrm{A}}$ is defined like $\mathsf{dof}_{\mathrm{P}}$ in Eq. 6.1. Different to the 2D case, now the inversion of $\mathsf{dof}_{\mathrm{A}}$ must be resolved for z_{focus}-values:

$$b = \mathsf{dof}_{\mathrm{A}}(z, z_{\mathrm{focus}}) \quad = \quad \mathsf{dof}_{\mathrm{P}}(z, z_{\mathrm{focus}}) \tag{6.3}$$

$$z_{\mathrm{focus}} = \mathsf{dof}_{\mathrm{A}}^{-1}(b, z) \quad = \quad \frac{D}{\frac{D+b}{z} - \frac{b}{f}} \tag{6.4}$$

An example for an adaptive camera is splatting [45, 137], which is a volume rendering technique, but which also can be used for information visualization. By changing the size of the splat kernel depending on the b value of a data point, SDOF can be implemented easily.

Another possibility is to use pre-blurred billboards (Sect. 6.6 and [91]). Objects are rendered into memory, the images are then blurred and mapped onto polygons in the scene.

6.4 Properties and Applicability

This section discusses some high-level properties of SDOF, how it can be principally applied, and what challenges it brings with it.

6.4.1 Properties

SDOF, being yet another F+C highlighting technique, has the following properties that make it an addition to the current toolbox:

- SDOF does not distort geometry. It is therefore usable when sizes (of objects or parts of objects (glyphs)) and positions are used as visual parameters. We also believe that it is easier to recognize blurred icons than distorted ones.

- SDOF does not alter colors. If color is used to convey meaning, or the visualization is to be used by color-blind people, SDOF can be used instead of color highlighting. This also means that SDOF is independent of color, and can therefore be used when only gray-scale is available (e.g., printed images).
- SDOF changes the irrelevant objects, rather than the relevant ones. It is therefore useful whenever the relevant objects contain a lot of information whose perception might be impeded by changes.

6.4.2 Applicability

SDOF requires concrete queries to the data (which can be simple, but have to be formulated nonetheless), and is therefore useful for analyzing and presenting data.

SDOF can serve as an additional aid to guide the viewer's attention, together with brighter colors, etc., or as a completely separate dimension of data display. Because blur is very naturally associated with importance (even more than color), we do not believe that it is suitable for true multi-variate data visualization. It can, however, add another dimension for a short time, when the displayed data is to be queried.

Blurring needs space, so when a lot of very small objects are depicted, it is only of little use. The application can deal with this problem by drawing the objects in such an order that sharp objects are drawn on top of blurred ones. But this can introduce artifacts, where parts of the display appear sharp only because of the contrast between sharp objects and the background.

6.4.3 Challenges

SDOF images depend on the output device (similar to tone mapping [89], for example). The reason for this is that blur is not an absolute measure, but depends on the viewing angle that the image covers – this is also the reason why small images look sharper than larger ones: the circles of confusion are not visible in the smaller version, or at least to a smaller extent. We use a calibration step at program startup to account for this problem (see Sect. 6.5.1).

Images that contain SDOF effects are also problematic when lossy compression is used (like MPEG, JPEG, etc.). In this case, artifacts can be introduced that create a high contrast in a blurred area, and thus distracting the user. But SDOF is most useful in interactive applications, so this problem should play no big role in practice.

6.5 Parameterization

Parameterization of SDOF consists of two parts: Adaptation to current viewing parameters and user interaction to change the relevance mapping.

6.5.1 Output Adaptation

We ask the user to select two blur levels on program startup: a) the minimal blur level that can be easily distinguished from a sharp depiction – this value translates to the step height h in Fig. 6.3; b) the minimal difference in blur that can be distinguished – this value can be used to calculate g, if the smallest difference between any two r values is given. Because this is generally not the case, the blur function is adapted for every image after examining the r values of all objects. These values can vary with the use of the generated image (printing out, projecting onto a wall, etc.), the use of different screens, etc.

6.5.2 User Interaction

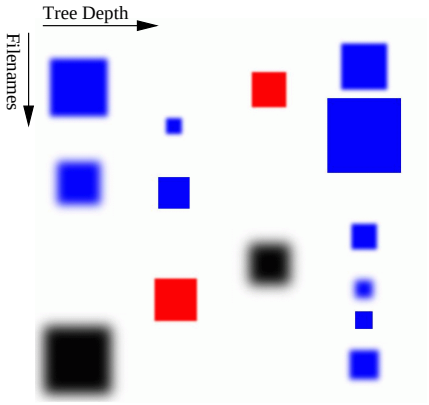
Interaction is a key part of SDOF. Blurred objects are unnatural, and it is therefore important for the user to be able to change the relevance mapping and blur function quickly, and to return to a depiction that shows all objects in focus.

Depending on the application, there are different usage patterns. In many applications, it is useful to be able to point at an object and say “Show me all objects that are older than this”, “Show me all chessmen that cover this piece” (Fig. 6.5e), or “Show me the cities weighed by their railway distance from this city”.

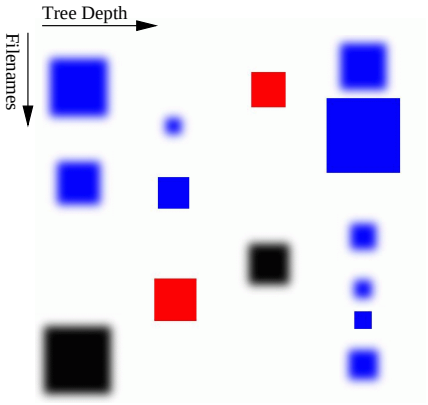
Another way is to select values independently of objects: “Show me all threatened chess pieces of my color”, “Show me all files that were changed today” (Fig. 6.5b), or “Show me all current patients weighed by their need for drug xyz”.

An additional feature we believe is useful is the *auto focus*. After a prespecified time, it makes all objects appear sharp again, thus making examination of all objects easier (this function can be turned off).

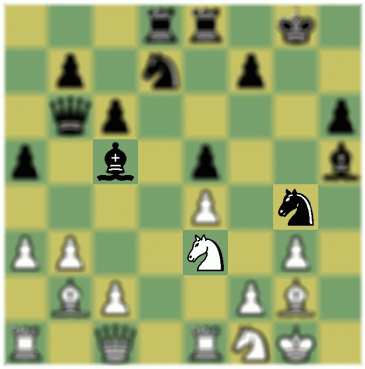
Transitions between different displays are always animated to enable the user to follow the change and immediately see which objects are relevant in the new display. This is another reason for separating r and b (see section 6.3.2): The animation is done between the old and the new b values, rather than the r values. This is because the blur function can contain discontinuities that can lead to jumps between blur levels of objects, and are therefore undesirable.



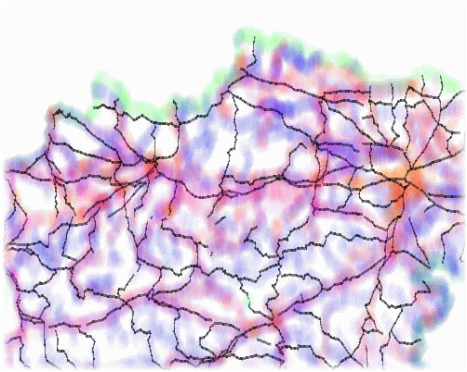
a) A file browser showing the age of files through blur. Continuous relevance function.



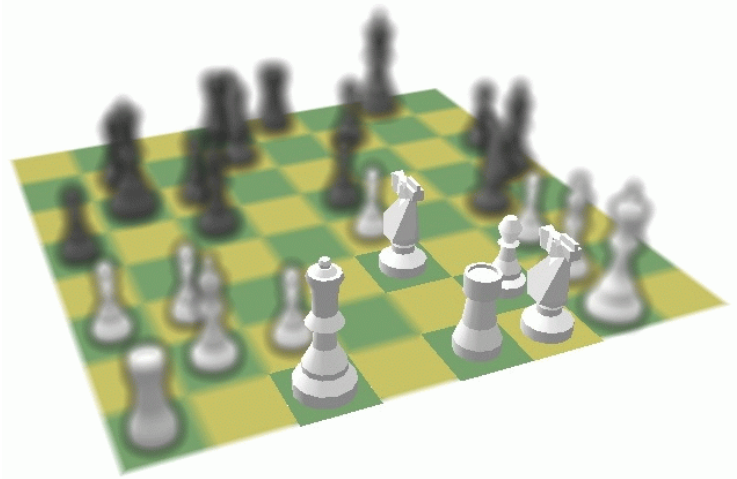
b) A file browser showing today's files sharply, older ones blurred. Binary relevance function.



c) A chess tutoring system showing the chessmen that threaten the knight on e3.



d) A Geographic Information System (GIS) showing the roads layer in focus.



e) A chess tutoring system showing the chessmen that cover the knight on e3.

Figure 6.5: SDOF in action. See Sect. 6.5.2 and 6.3 for details.

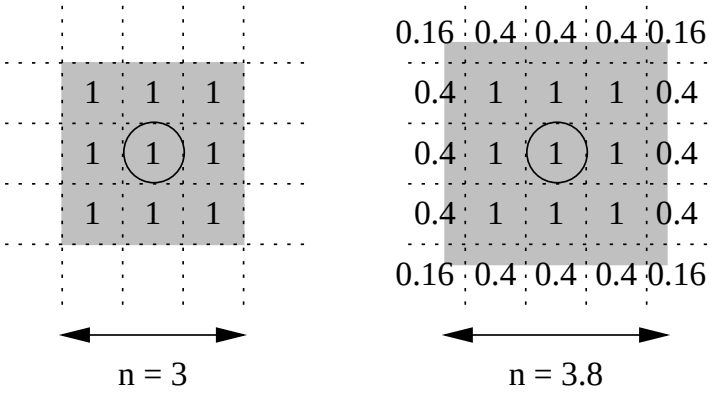


Figure 6.6: The Box Filter (left), and the generalized box filter for arbitrary sizes (right).

6.6 Implementation

A method in information visualization should not only be visually effective, but also fast, so that it can be used interactively. Blurring used to be a very slow operation because it involves a sum of three color components of many neighboring pixels for every single pixel in the image, and is still not supported by hardware except in high-end graphics workstations. We have implemented SDOF using texture mapping hardware, which makes it fast on currently available consumer PCs. The described method is an implementation of the adaptive camera model (see Sect. 6.3.3).

Blur can be understood as a convolution operation of the image with a blur kernel. In photography, this blur kernel ideally is round, but usually is a regular polygon with six to eight vertices, due to the shape of the aperture.

The more common type of blur kernel in computer science is the box filter (Fig. 6.6, left). It has the big advantage of being separable [91], which reduces its computational cost from $O(n^2)$ to $O(2n)$, where n is the filter size. It can also be generalized quite easily to arbitrary sizes (Fig. 6.6, right) other than just odd integers. This implementation directly uses b as its filter size n.

Using graphics hardware is different from a software implementation of a filter in that it does not sum up the color values of surrounding pixels for every single pixel. Rather, it adds the whole image to the frame buffer in one step by drawing it onto a textured polygon (this is done by blending with a special blend function). When the image is drawn in different positions (with one pixel distance between the images), several image pixels are added to the same frame buffer pixel. Because of the limited accuracy of the frame buffer (typically eight bits per color component), this can only be done for small values of n (we have found $n \leq 4$ to yield acceptable images).

For larger blur diameters, we use a two-step approach. First, we sum up four images into the frame buffer, with their color values scaled so that the sum uses the entire eight bits. We then transfer this image to texture memory (this is a fast operation) and use this auxiliary sum as the operand for further calculations. The auxiliary sum already contains the information from four addition steps, so when summing them up further, only one quarter of the addition steps is needed. Because all the values in the box filter (except for the border, which is treated separately) are equal, all auxiliary sums are equal – they are only displaced. This means, that the auxiliary sum only needs to be computed once (as well as another auxiliary for the borders). Summing up auxiliary sums is therefore not only more accurate, it is also faster.

For blur diameters larger than 20 pixels, we first scale the image to one quarter of its size, then blur with half the diameter, and then scale it back ("quarter method").

Using the described method, it is possible to run applications – like the ones shown in the images and the accompanying video – at interactive frame rates (at leat 5 frames per second) on cheap consumer graphics hardware. This number is likely to increase with some further optimizations as well as the use of multitexturing (which is supported by more and more consumer graphics cards).

6.7 Evaluation

To show that SDOF is actually perceived preattentively, and to demonstrate its usefulness in applications, we are currently performing a user study with 16 participants. We want to find out a) if SDOF is, indeed, perceived preattentively, which includes the detection and localisation of targets, as well as the estimation of the number of targets on screen (as a number relative to all objects in the image) in the presence of distractors; b) how many blur levels people can distinguish, and how blur is perceived (e.g., linear, exponential, etc.); c) how blur compares to other visual cues which are known to be perceived preattentively (such as color and orientation); and d) how well SDOF can be used to solve simple problems with simple applications (where the emphasis is on the use of SDOF). This study is still in progress at the time of this writing, but we will publish the results as soon as they are available.

6.8 Conclusions and Future Work

We have presented an extension to the well-known depth-of-field effect that allows objects to be blurred depending on their relevance rather than on their distance from the camera. This technique makes it possible to point the user to relevant objects, without distorting the geometry and other features of the visualization.

Because of the similarity to the familiar depth-of-field effect, and the fact that DOF is an intrinsic part of the human eye, we believe that it is a quite natural metaphor for visualization and can be used quite effortlessly by most users.

SDOF can be used when analyzing and presenting data, and also seems to be effective as a tool for pointing information out in tutoring systems.

We expect to learn a lot about SDOF's properties during our user study, and will use this information to define criteria when and how SDOF can be best used.

As one of the next steps, we want to investigate the applicability of SDOF to other areas of scientific visualization, like volume and flow visualization.

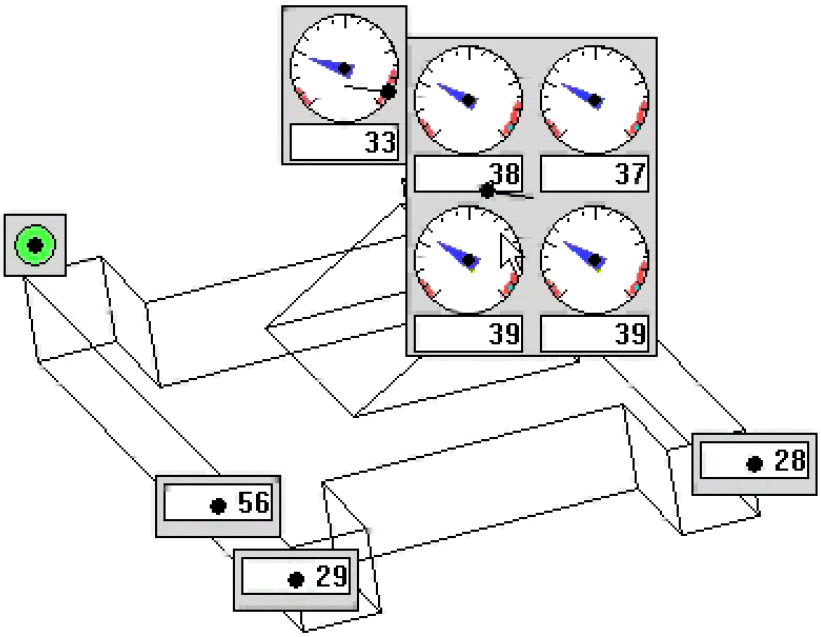
We also want to find out how SDOF can be applied to human computer interaction, to enable the user to grasp important information faster, and to be alerted to important changes without being distracted too much.

Acknowledgments

We would like to thank Markus Hadwiger for his help in coming up with a fast method for rendering SDOF images. This work is part of the Asgaard Project, which is supported by *Fonds zur Förderung der wissenschaftlichen Forschung* (Austrian Science Fund), grant P12797-INF. Parts of this work have been carried out in the scope of the basic research on visualization at the VRVis Research Center (`http://www.VRVis.at/vis/`) in Vienna, Austria, which is funded by an Austrian governmental research program called Kplus.

Chapter 7

Process Visualization with Levels of Detail



In 2002, the contents of this chapter have been accepted for publication in the Proceedings of the IEEE Symposium on Information Visualization 2002 (*IEEE InfoVis 2002*) as short paper (paper "**Process Visualization with Levels of Detail**" by Krešimir Matković, Helwig Hauser, Reinhard Sainitzer, and Eduard Gröller).

The contents of this chapter (paper) are a result from a collaborative project with Wolfgang Rieger (one of Helwig Hauser's diploma students), Krešimir Matković and Reinhard Sainitzer (both from the application side of this project, and Prof. Eduard Gröller (head of the visualization group at the Institute of Computer Graphics and Algorithms, Vienna University of Technology). There were no further directly related papers (co-authored by Helwig Hauser) from this project.

Process Visualization with Levels of Detail

Krešimir Matković, Helwig Hauser, Reinhard Sainitzer, and Eduard Gröller

Abstract

In this paper we demonstrate how we applied information visualization techniques to process monitoring. Virtual instruments are enhanced using *history encoding* – instruments are capable of displaying current value and the value from the near past. *Multi-instruments* are capable of displaying several data sources simultaneously. *Levels of detail* for virtual instruments are introduced where the screen area is inversely proportional to the information amount displayed. Furthermore the monitoring system is enhanced by using *3D anchoring* – attachment of instruments to positions on a 3D model –, *collision avoidance* – a physically based spring model prevents instruments from overlapping –, and *focus+context rendering* – giving the user a possibility to examine particular instruments in detail without loosing the context information. Two applications were developed, a prototype application and a commercial process monitoring tool.

Keywords: process monitoring, process visualization, information visualization, levels of detail, focus+context visualization, virtual instruments.

7.1 Introduction

Information visualization (InfoViz) has emerged as an important field of research in the past few years. Process visualization as one of its sub-fields is directly related, but nevertheless much older. Processes have been visualized and monitored using traditional, analog instruments since the early days of industrialization. Analog instruments were followed by the use of digital instruments, and finally, computers are increasingly used to visualize and monitor processes.

There are a lot of software tools available, offering a comfortable and intuitive way for process visualization. Examples of widely used instrument libraries and tools are products by Global Majic Software [46] and Quinn-Curtis Inc. [107]. A wide-spread monitoring tool also is LabVIEW System from National Instruments [69]. There are hundreds of instruments available within this software. Another offerer in this field is GE Fanuc Automation with the DataViews system [18] and a large base of virtual instruments as well.

Most of these tools and libraries try to mimic conventional instruments, without taking advantage of well-known InfoViz methodology. Virtual instruments are

much more flexible than conventional settings with real instruments. The virtual instruments are easier customized, but essentially they are still just an electronic version of the real instruments.

In this paper, we describe how process visualization can benefit from information visualization. We have improved features of singular instruments, as well as global monitoring features, and finally we developed two applications which illustrate the methods described in this paper.

Features added to virtual instruments are: history encoding, multi-instruments, and levels of detail (LOD). Including history encoding allows to display the current value and the values from the near past simultaneously. We achieve this by visually augmenting the common gauge and bar instrument. The historic values are displayed using less saturated colors and smaller needles or bars. In this way history values can be distinguished from the current value on a first sight, but still give an additional information about sensor behavior in the near past.

Multi-instruments display more than one data source simultaneously, making it easier to compare them. A multi-gauge, for example, has several needles, each of them in a different color and length. Values corresponding to the needles can be easily compared. In fault-tolerant systems often several sensors measure the same phenomenon. In this case multi-instruments are well-suited in emphasizing a malfunctioning sensor.

Levels of detail is an interesting choice in structuring the concurrent display of many instruments. Various types of instruments of different sizes are suitable for a LOD approach. If, e.g., four redundant sensors should be visualized, an instrument which covers a small area and which displays only the collective state of all sensors (ok/not-ok) can be considered as the first (and roughest) LOD. The next level could be an instrument which displays the average value of all the four sensors. Such an instrument covers more area on the screen, but gives more information as well. The following level could be a multi-instrument, displaying all four values, and finally the highest LOD would be represented by four single instruments. The highest level occupies the largest area, giving the most information at the same time.

Besides augmenting the instruments themselves, the global monitoring system was improved as well. Global features added are focus+context rendering [56] and collision avoidance.

Focus+context (F+C) approaches make a huge information space manageable for a user by coarsely representing the entire information in the available screen space. The user is enabled to display selected parts (focus) enlarged but embedded into the surrounding environment (context). In the example described above, it is possible to choose the level of detail of an instrument depending on the degree of interest. The degree of interest can be determined either explicitly by the user – the user points to the focus –, or it can be data-driven. Here the degree of interest might increase whenever a specific event occurs, e.g., a data value leaves the admissible range.

Increasing the level of detail, increases the required display area of the instrument, and it potentially occurs that some of the instruments overlap after a level change. In order to avoid instrument overlap, a collision avoidance algorithm based on a physically based spring model was implemented.

The above described features will be explained in more detail using gauge, bar, LED, and numeric instruments. The bar instrument is an instrument where the length of a solid bar corresponds to the measured value. It's analog counterpart is a mercury thermometer, or a tube manometer. The LED instrument mimics a single LED and can represent two states in our case. The numeric instrument is an instrument that displays measured data on a numeric display. Data can be displayed as a decimal, hexadecimal or binary number.

All of the new features are implemented in two applications. A placement prototype application [109] and a commercial TTPView 3.0 application [129].

7.2 New Features for Virtual Instruments

The idea of using virtual instead of classical instruments is not new. However, most of the virtual instruments try to mimic real ones, and offer a faster, more comfortable and more flexible way of arranging a new set of measuring instruments. We improve virtual instruments by adding some new features and techniques which are well-known from information visualization. We not only improve the instrument management but also augment the virtual instruments as compared to their real counterparts. We made virtual instruments capable of displaying the history of data (history encoding). Our virtual instruments are also able to simultaneously display several data values within one instrument, so that divergences in redundant information are easily recognizable (multi-instruments). Finally, we introduce a LOD representation for virtual instruments.

7.2.1 History Encoding in Virtual Instruments

It is often important not only to read the current value of a sensor, but also to get a clue what was the value in the near past. Of course it is possible to use some kind of oscilloscope instrument or time chart, but their size is often a limiting factor. We added history encoding to the bar and gauge instrument. Such an augmented instrument takes no more place than a common bar/gauge instrument, but gives far more information to the user. In the case of the bar instrument, history is encoded through horizontal lines on the left of the numerical scale. Length and saturation of a line encode the "recentness" of a value, i.e., longer and more saturated lines correspond to newer data values. Figure 7.1(a) illustrates a bar instrument without history encoding. Just the current value is depicted, redundantelly through a bullet on the numerical scale as well as a solid bar to the left of the scale. In figures 7.1(b)–7.1(e) data history is depicted. Figure 7.1(b) and (c) show an over time

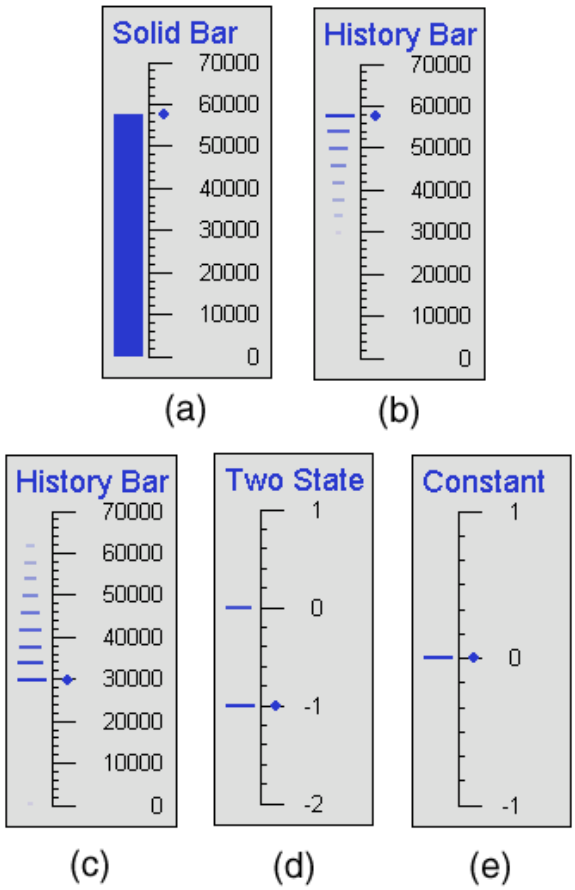


Figure 7.1: Solid bar (a), history bar with increasing (b) and decreasing (c) value, two state (d) and constant value (e) history bar

monotonically increasing, respectively decreasing data value. Figure 7.1(d) shows a quantity that is switching between two values and figure 7.1(e) shows a quantity that is constant over time. A simple solid bar instrument would not be able to capture these vastly different temporal evolutions.

The same idea has been applied to the gauge instrument. The history needles in the gauge instrument are reduced to the needle tips in order to make the whole instrument more clear. Figure 7.2 illustrates the history gauge instrument for a uniformly increasing data value.

History encoding also relates to the visualization of gradient information in the data, similar as, for example, often used in scientific visualization (compare to gradients in volume visualization, for example). By showing not only the immediate

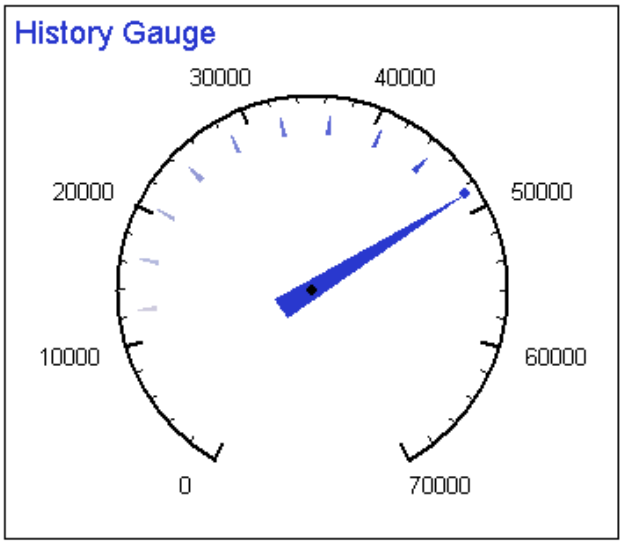


Figure 7.2: History gauge depicting a uniformly increasing value

values, but also a notion of temporal derivatives, the user gains more information about the sensor data.

7.2.2 Multi-Instruments

Common practice in fault tolerant systems, e.g., planes, locomotives, modern cars, etc., is to install several sensors for the same data. Important system decisions must not be based on a single measurement alone. In a faultless situation all sensors deliver the same data. If a sensor is out of order, its measurements should be neglected, and the maintainer of the system should get the information that the sensor is not working properly. If a separate instrument is used for each sensor, it is very hard to see if there are any deviations in the measurements, unless one value is significantly different from the others.

In order to make it easier to visualize such cases, we introduce multi-instruments. The multi-bar and the multi-gauge instrument, display several distinct data sources simultaneously, but using the same scale. The multi-bar instrument uses more bars beside each other, and the multi-gauge instrument uses more needles. The needles have different length, and shorter needles are placed on top of longer needles, so that the user immediately sees all the values. If there is a deviation in the values, one (or more) not matching needle(s) will be easily spotted. Figure 7.3 illustrates a multi-bar and a multi-gauge instrument. The user easily notices that a sensor represented by the yellow bar/needle obviously malfunctions. The multi-instrument approach has its drawbacks as well. Although the needles in

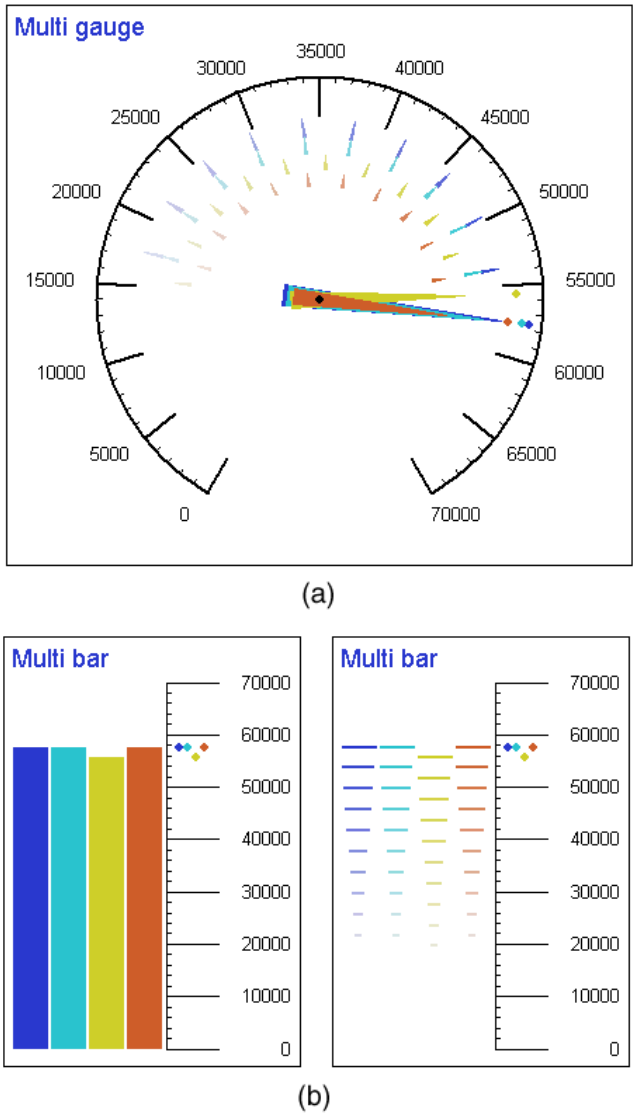


Figure 7.3: Multi-gauge and multi-bar

the multi-gauge instrument are color-coded and have different lengths, it is not so easy to perceive separate needles when they overlap. Several, single instruments are certainly easier to read but require much more space.

Multi-instruments can also be used whenever it is necessary to visualize related data which does not have to be identical. An example is the visualization of a distribution of a data value using three needles. One needle displays the average, and other two needles the standard deviation of the distribution.

In order to make it easier to read the instruments, two more features are added. First a bullet on the scale indicates the current value. This is especially useful for

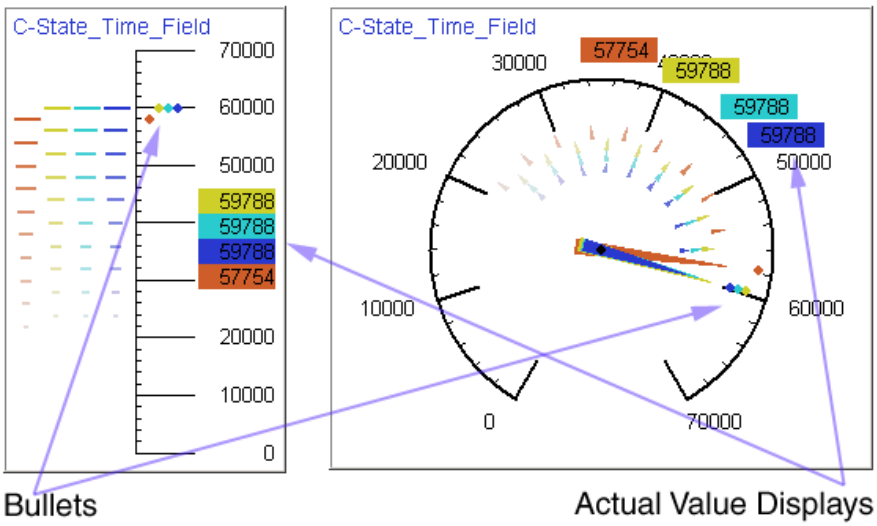


Figure 7.4: Bullets and actual value displays in the multi-bar and multi-gauge instruments

multi-instruments where the numerical scale is not next to all bars/needles. Furthermore, an actual value display is added as well. An actual value display shows the current value as decimal number in a numeric display. The position of the actual value display changes and closely follows the current data value. The display is positioned using a running average making it easier to read the numerical value in case of fast changes or oscillating values. Furthermore, actual value displays never overlap, which makes them easier to read as well. Figure 7.4 shows these two features.

7.2.3 Levels of Detail

Multi-instruments have not only the advantage of making it easier to compare related values, but they save screen space as well. A multi-bar instruments with four bars occupies a far smaller area than four single bar instruments. This led us to the next new feature in process visualization, i.e., levels of detail. An example is the following. Four single bar instruments are the finest level of detail, the multi-bar instrument representing the same four values is the next lower level, a numeric instrument which displays the average value and a LED instrument are further levels of detail. Figure 7.5 illustrates the example, and gives the area usage for each level (highest level = 100%). In this way, it is possible to use the LED instrument if everything is running OK, and than automatically switch to a higher level if some data does not fulfill predefined conditions. There are other possibilities how the levels can be switched, and some of them will be described in the following section.

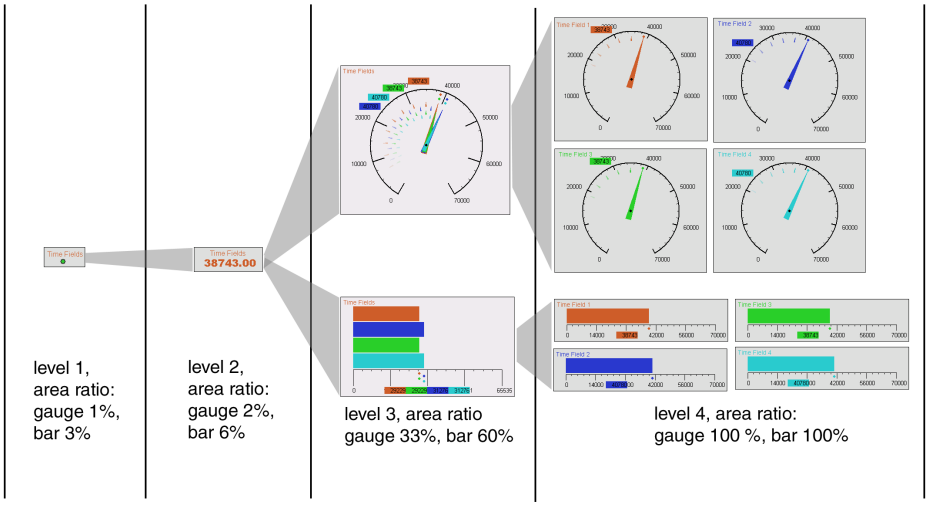


Figure 7.5: Various LODs, building up a tree of instruments, with corresponding area sizes

7.3 F+C Process Visualization

The LOD idea itself leads to a F+C process visualization. The focus+context approach is a well-known technique in information visualization. The surrounding environment, i.e., the context, makes this approach different from simple zooming-in, where an area is displayed in another scale but the connection to the environment is lost.

The F+C principle is partially based on the fact that humans can best perceive a circular area with a limited diameter in the center of their view [11]. The important information should be placed in this focus, but the information outside the ”most interesting” area, is not unimportant. This peripheral area (context) helps us to orient, and brings surrounding information for the instruments in focus.

According to Kosara et al. [64] the F+C methods can be divided into three groups. The first, and widest spread group of F+C methods are distortion-oriented or spatial methods. The idea is to distort the display to allow a magnification of interesting areas without losing the context which will be displayed on a coarser scale. A lot of metaphors relating this kind of F+C techniques to real world objects have been introduced like fisheye views [114, 30], stretched rubber sheets [115], etc.

The second group of methods are dimensional methods, which are suitable for objects with a lot of data associated with them. Objects in focus will be displayed using more dimensions of the data. An example of such a method is the magic lens [123].

Finally, the last group of F+C methods is called cue methods. These approaches allow the user to select objects in terms of their features, and not their spatial relations. An example is the semantic depth of field approach [64] which blurs different parts of the scene in dependence of their relevance. In this way the context information is still present, but the important information is in depicted sharp, and can be easily recognized by the user.

Our idea is to use the LOD principle introduced in the previous section for our F+C approach. The degree of interest corresponds to the level of detail of the instrument. The larger the degree of interest is, the higher level of detail is used. This can be considered, according to the above introduced categorization as a dimensional F+C method. Increasing the level of detail, however, increases the size of an instrument, and instrument overlapping possibly would occur. Some kind of rearrangement of the instruments should be done in this case (assuming there is enough free space on the screen) which makes our method a spatial method too. Therefore, our approach represents a combination of a dimensional and a spatial F+C method.

7.3.1 3D Anchoring and Interaction

While process visualization usually is done in 2D, the data often comes from a 3D environment. Of course it would be possible to use 3D visualization with its advantages like similarity to real-world situations and flexibility which is offered by the third dimension. On the other hand 3D visualization has some drawbacks as well. The computing power needed for 3D visualization is significantly higher than the computing power needed for 2D, the objects are often occluded by other objects, and there should be some navigation mechanism in order to visually access all objects. The fact that common output devices (monitors) and input devices (mouse) are actually 2D devices, makes manipulating and using 2D visualization more intuitive for a typical user.

Virtual instruments represent real measuring sensors, that are placed in a 3D environment, and send data to the system. The user positions instruments on the screen, specifying the instruments current and desired position (which can but must not be the same). Since the real sensors exist in a 3D environment, e.g., a racing car, virtual instruments could be attached to the 3D model of the real environment and then mapped to the screen. This attachment of an instrument to a 3D position of a 3D model is called anchoring. If data that is going to be visualized comes from the wheel sensors of a car, the corresponding virtual instruments can be anchored to the wheels on the 3D model. In this way, the user can immediately recognize which data is represented by which instrument. The user can for example rotate the 3D model, and the assigned instruments will follow the movement.

The instruments are not simply placed in the 3D world and then mapped, because they would be easily unreadable after applying projection. Instead, after

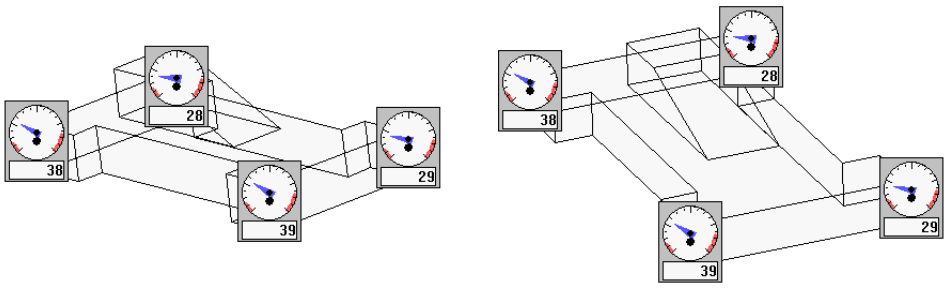


Figure 7.6: 3D anchoring: virtual instruments follow the 3D model

projection the instruments are placed at the positions of anchoring points (projected now). In this way the instruments are always visible, and easily readable. Figure 7.6 shows an example of four instruments assigned to the four wheels of a schematic car model. The anchored instruments follow changes of the underlying 3D model.

7.3.2 Collision Avoidance

In principle, anchored instruments, which follow a 3D model, often would overlap. It is possible to let the user solve the overlapping problem by arranging the instruments himself. It would however be far more comfortable if the system accomplishes this task.

In this case, it is necessary to develop a collision avoidance system which will prevent the instruments from overlapping. The algorithm implemented [109] uses a physically based spring model [5].

A system of objects and springs is defined, such that there is a spring between each instrument's current and its desired position. The desired position is the projected anchor point. The tension of a spring is proportional to the distance between the instrument's current and desired position. Furthermore, a minimal allowed distance between two instruments is defined. If two instruments come closer than the minimal allowed distance, repelling forces are added to the system. This keeps instruments apart from each other, and avoids overlap. As soon as the instruments are far enough from each other the repelling forces are not acting any more.

In our prototype application the user determines the initial position by placing instruments and specifying their ideal positions (anchor points on the 3D model). When all instruments are placed, the spring model tries to assume a minimum-tension state by incrementally moving the instruments in the spring force directions. The process is repeated until a stable state is found, or until a user-defined time elapses. During the rearrangement, the stiffness of springs successively in-

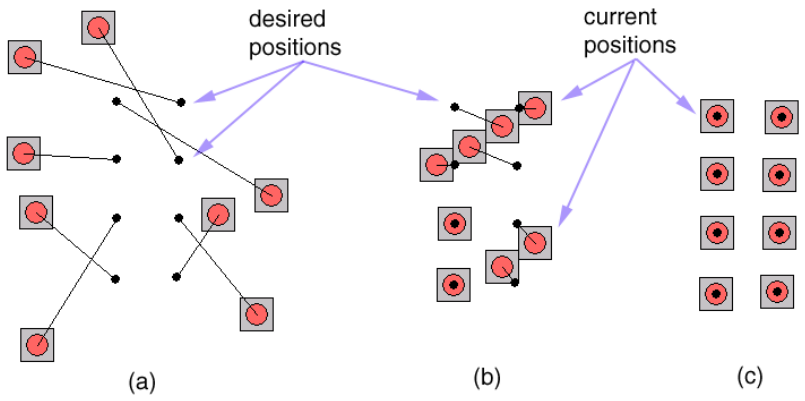


Figure 7.7: Collision avoidance of virtual instruments: (a) start configuration, (b) rearrangement is going on, and (c) final solution

creases. This simulates annealing to avoid oscillations between two stable solutions (instruments jumping between different positions). The process of automatic placement is illustrated in figure 7.7.

7.3.3 Focus+Context Rendering

The LOD principle and automatic instrument placement are used for focus+context rendering. The idea is to use different levels of detail for different degrees of interest. The instruments with the highest degree of interest will be represented using the highest LOD, and the level of detail will decrease with decreasing degree of interest. After a level changes it is sometimes necessary to rearrange the instruments. Automatic collision avoidance described in section 7.3.2 is used in such a case. Figure 7.8 illustrates the change of levels, and rearrangement of the instruments.

An instrument's degree of interest can be increased either explicitly or implicitly. By explicitly changing the degree of interest the user selects the instruments that are more interesting. This can be done in two ways.

One way is to equip the user with a sort of magic lens [123] which can be coupled to the cursor. In this way the area around the cursor is considered to have a high degree of interest, and the degree decreases with the distance from the cursor.

Another way, could be to define a fixed area on the screen which represents the region with the highest degree of interest. This can be the area around the center of the screen, or area around the golden cut point, or any other user selected area. The closer the instrument is to the center of the area, the higher level of detail is used to

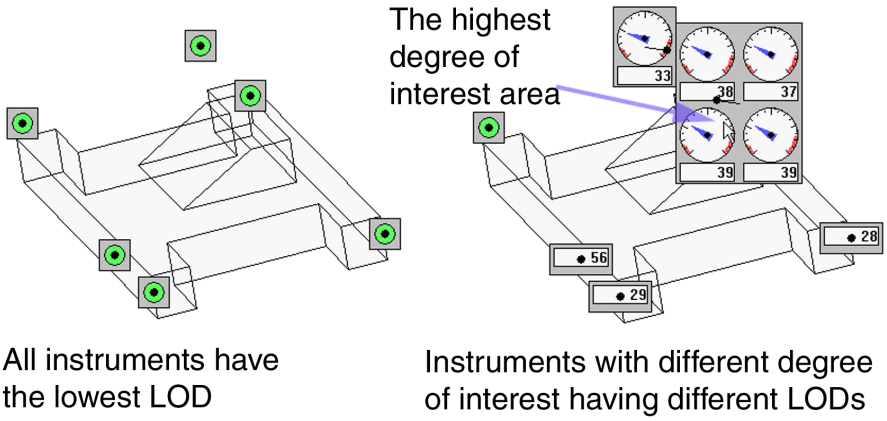


Figure 7.8: Changing the degree of interest

represent the instrument. In this way the user can examine a particular instrument by bringing it into a predefined screen area.

Implicitly changing the degree of interest is coupled with predefined, undesired system states. If an unwanted state occurs, e.g., a value exceeds a predefined range, the instrument's degree of interest is increased automatically, to notify the user and get his attention.

7.4 Application and Evaluation

The principles described in this paper are implemented in a placement prototype [109] and in a commercial application TTPView 3.0 [129]. The placement prototype realizes focus+context visualization as well as collision avoidance as described in section 7.3. The TTPView 3.0 application is far more complex. It is a commercial product which is intended to be used on a daily basis.

TTPView is a high-speed bus monitoring tool for the TTP system. TTP stands for Time Triggered Protocol, which represents one of two available technologies in the field of real-time control systems. One is event-triggered, and the second one is time-triggered. The TTPView 3.0 is easily connected to the system, and gets all information about existing data sources from the system. The data sources are logically organized in trees, and the user can easily drag-and-drop wanted items into a view. Each tree item has a default instrument assigned to it, but the user can switch to another virtual instrument. All available instruments can be configured by: adding a caption string (description of data source) and unit string (physical unit of the data like RPM, MPH,..), adding an instrument background image, switching the scales and actual value displays on or off, changing size and color

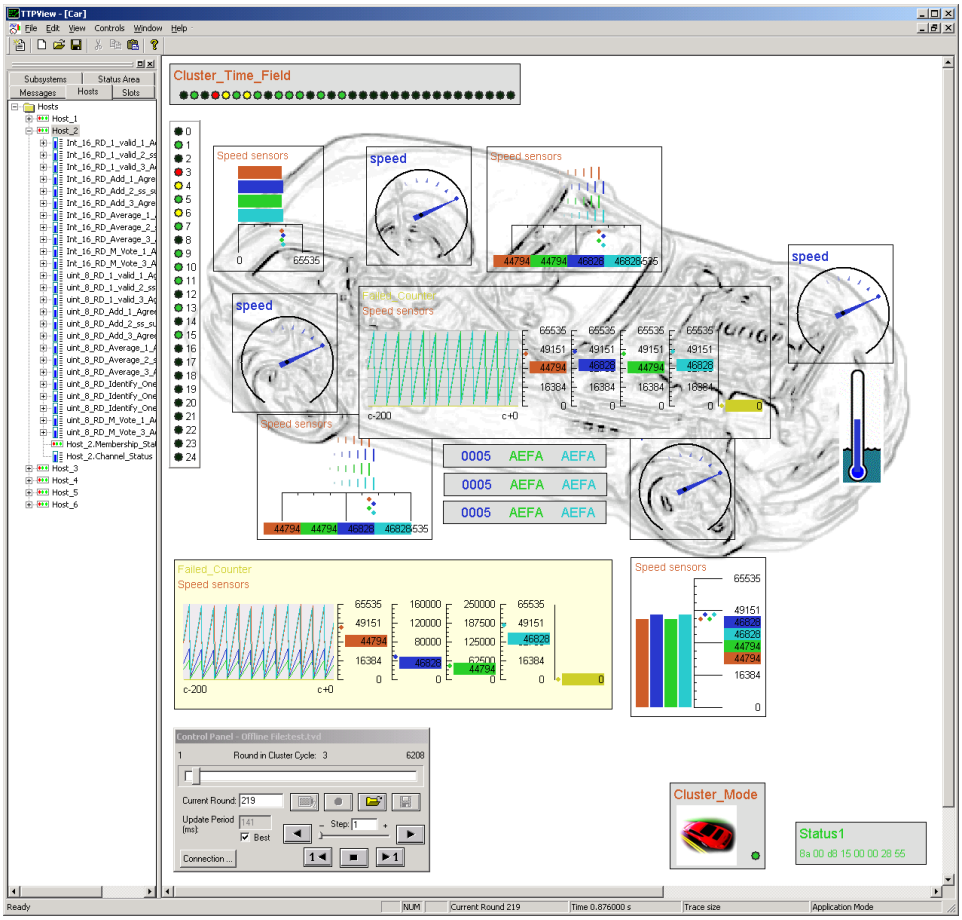


Figure 7.9: A screen shot of TTPView 3.0

etc. It is easy for the user to configure the views according to his or her needs. The user can create several views as well. The views themselves may have a background image and a user-selected background color. Figure 7.9 shows some of these features.

The feature to add an instrument background has turned out to be very useful. Using cleverly designed background images, it is often immediately obvious what type of data the instruments display. The bar instrument in figure 7.10 shows temperature and the gauge instrument in the same figure shows revolutions per minute (RPM). Both data types are obvious from the design of the background image of the instrument. Note that the instruments in figure 7.10 are the same bar and gauge instruments as shown throughout the paper, but with scales switched off, and instrument background images added.

TTPView supports the following instruments: bar, gauge, trace, led, numerical, icon, and text instrument. The bar, gauge, LED, and numerical instruments were described in previous sections. Here we just briefly describe the other instruments.

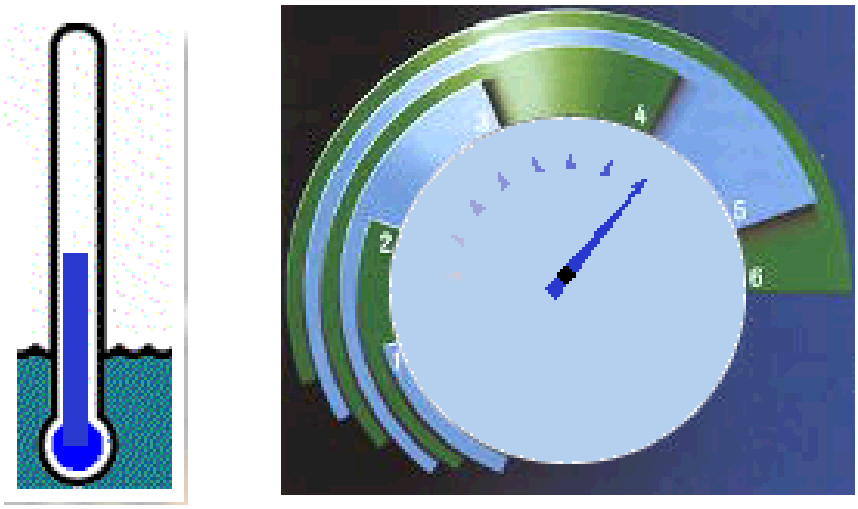


Figure 7.10: The bar instrument with a thermometer background and the gauge instrument showing RPMs

The trace instrument displays the temporal evolution of a data value as a poly-line in a time vs. data-value plot. The trace instrument supports more data sources, which makes it a multi-trace instrument. It has a separate scale for each data source (note that multi-bar and multi-gauge instruments have only one scale). This can be useful to study correlations between different sources, e.g., correlation of speed, RPM, and engine temperature.

The icon and text instruments are designed to visualize discrete sets of values. An icon or a string is assigned to individual intervals of the input data. An example could be to assign traffic-light icons with green, yellow and red light switched on according to a three state input.

The user can define a range of valid values for all instruments. If a value exceeds the admissible range a warning sign is added to the instrument. A message describing when the error occurred, and what was wrong is printed on the screen.

7.5 Conclusion

This paper presents several novel approaches to process visualization. We did not only try to mimic real instruments, but added features to the virtual instruments that can not be realized using their real counterparts. Furthermore we have enhanced the whole process visualization system, and we have implemented two applications.

The new features added to the virtual instruments can be summarized as:

History encoding, which makes it possible to display the values from the near past together with the current value. This allows to see the evolution of the value without increasing the display size of the instrument.

Multi-instruments are capable of displaying several values simultaneously. A multi-instrument makes it easier to compare redundant values, and saves screen space compared to single instruments which display the same data.

The levels-of-detail approach makes it possible to use instruments of different sizes to represent the same data. Increasing the level of detail will increase the amount of information shown, but will take up more screen space.

The advanced features in instruments made it possible to improve the global monitoring system by applying:

The focus+context principle, which allows the user to explore particular instruments in more detail (using higher levels of detail) without loosing the overall context. The method implemented is a combination of a dimensional and a spatial F+C method. The degree of interest can be explicitly set by the user, or can be data driven, increasing the degree if some special, predefined data state occurs.

3D Anchoring was used since our data originates from a 3D environment. Instruments are anchored on the 3D model, but special attention is taken when instruments are projected to 2D screen space in order to make them always visible.

Collision Avoidance was implemented using a physically based spring model. This was necessary to avoid instrument overlapping. Overlapping can be avoided only if the total instrument area is not larger than the total screen area.

The methods described in this paper were implemented in two applications: a placement prototype with F+C visualization and collision avoidance, and TTPView 3.0, as a commercial application, developed with the TTTech company [129], which is used on a daily basis.

This paper shows how InfoViz techniques can be applied to process monitoring and visualization. We did not try only to mimic the real instruments, but also to enhance them, and the whole monitoring system, as well. As a result, a system offering more information, using not much more screen space, and giving the user more flexibility is realized.

Acknowledgements

The authors thank Wolfgang Rieger for his inputs and his help in implementing the software, Herbert Buchegger for his help in implementation and Wolfgang Meyer for valuable discussions and ideas. Furthermore we would like to thank the TTTech company from Vienna (`http://www.tttech.com/`). TTTech helped us in understanding the needs in process visualization, and the TTPView 3.0 was made for TTTech. Parts of this work were carried out in the scope of applied and basic research at the VRVis Research Center in Vienna (`http://www.VRVis.at/`), Austria, which is funded by an Austrian governmental research program called K plus.

Bibliography

[1] Ansel Adams. *The Camera*. Little Brown & Company, 1991.

[2] Hamdy Agiza, Gian-Italo Bischi, and Michael Kopel. Multistability in a dynamic cournot game with three oligopolists. *Mathematics and Computers in Simulation*, 51:63–90, 1999.

[3] Christopher Ahlberg and Ben Shneiderman. Visual information seeking: tight coupling of dynamic query filters with starfield displays. In *Proceedings of ACM CHI 1994 Conference on Human Factors in Computing Systems*, pages 313–317, 1994.

[4] Chandrajit Bajaj, Valerio Pascucci, and Daniel Schikore. The contour spectrum. In *Proceedings of IEEE Visualization 1997*, pages 167–174, 1997.

[5] David Baraff and Andrew Witkin. Phisically based modeling: principles and practice. In *SIGGRAPH 1999 Course Notes*, 1999.

[6] Dirk Bauer, Ronald Peikert, Mie Sato, and Mirjam Sick. A case study in selective visualization of unsteady 3D flow. In *Proceedings of IEEE Visualization 2002*, pages 525–528, 2002.

[7] Richard Becker and William Cleveland. Brushing scatterplots. *Technometrics*, 29(2):127–142, 1987.

[8] Eric Bier, Maureen Stone, Ken Pier, William Buxton, and Tony DeRose. Toolglass and magic lenses: the see-through interface. In *Proceedings of ACM SIGGRAPH 1993*, pages 73–80, 1993.

[9] Gian-Italo Bischi, Lukas Mroz, and Helwig Hauser. Studying basin bifurcations in nonlinear triopoly games by using 3d visualization. *Nonlinear Analysis*, 47(8):5325–5341, 2001.

[10] Andreas Buja, John McDonald, John Michalak, and Werner Stuetzle. Interactive data visualization using focusing and linking. In *Proceedings of IEEE Visualization 1991*, pages 156–163, 1991.

[11] Stuart Card, Jock MacKinlay, and Ben Shneiderman. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers, 1998.

[12] Sheelagh Carpendale, David Cowperthwaite, and David Fracchia. 3-dimensional pliable surfaces: for the effective presentation of visual information. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, Information Navigation, pages 217–226, 1995.

[13] Daniel Cohen and Zvi Sheffer. Proximity clouds: an acceleration technique for 3D grid traversal. *The Visual Computer*, 11(1):27–38, 1994.

[14] Grace Colby and Laura Scholl. Transparency and blur as selective cues for complex visual information. In *SPIE Vol. 1460, Image Handling and Reproduction Systems Integration*, pages 114–125, 1991.

[15] Robert Cook, Thomas Porter, and Loren Carpenter. Distributed ray tracing. In *Proceedings of ACM SIGGRAPH 1984*, pages 137–145, July 1984.

[16] Roger Crawfis, Nelson Max, Barry Becker, and Brian Cabral. Volume rendering of 3D scalar and vector fields at LLNL. In *Proceedings Supercomputing 1993*, pages 570–576, 1993.

[17] Balázs Csébfalvi, Lukas Mroz, Helwig Hauser, Andreas König, and Eduard Gröller. Fast visualization of object contours by non-photorealistic volume rendering. In *Proceedings of EUROGRAPHICS 2001*, pages C 452–C 460, 2001.

[18] Data Views by GE Fanuc Automation, http://www.dvcorp. com.

[19] Helmut Doleisch, Martin Gasser, and Helwig Hauser. Interactive feature specification for focus+context visualization of complex simulation data. In *Proceedings of the Joint IEEE TCVG – EUROGRAPHICS Symposium on Visualization 2003*, pages 239–248, 2003.

[20] Helmut Doleisch and Helwig Hauser. Smooth brushing for focus+context visualization of simulation data in 3D. In *Proceedings of WSCG 2002*, pages 147–154, 2002.

[21] Robert Drebin, Loren Carpenter, and Pat Hanrahan. Volume rendering. In *Proceedings of ACM SIGGRAPH 1988*, pages 65–74, August 1988.

[22] David Ebert and Penny Rheingans. Volume illustration: non-photographic rendering of volume models. In *Proceedings of IEEE Visualization 2000*, pages 195–202, 2000.

[23] Gershon Elber. Interactive line art rendering of freeform surfaces. In *Proceedings of EUROGRAPHICS 1999*, volume 18(3), pages 1–12, 1999.

[24] Shiaofen Fang, Tom Biddlecome, and Mihran Tuceryan. Image-based transfer function design for data exploration in volume visualization. In *Proceedings of IEEE Visualization 1998*, pages 319–326. IEEE, 1998.

[25] William Farrand. *Information Display in Interactive Design.* PhD thesis, Department of Engineering, University of California, Los Angeles, CA, 1973.

[26] Ying-Huey Fua, Matthew Ward, and Elke Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. In *Proceedings of IEEE Visualization 1999*, pages 43–50, 1999.

[27] Ying-Huey Fua, Matthew Ward, and Elke Rundensteiner. Structure-based brushes: a mechanism for navigating hierarchically organized data and information spaces. *IEEE Transactions on Visualization and Computer Graphics*, 6(2):150–159, 2000.

[28] Issei Fujishiro, Taeko Azuma, and Yuriko Takeshima. Automating transfer function design for comprehensible volume rendering based on 3D field topology analysis. In *Proceedings of IEEE Visualization 1999*, pages 467–470. IEEE, 1999.

[29] George Furnas. The FISHEYE view: a new look at structured files. Technical Memorandum #81-11221-9, Bell Laboratories, Murray Hill, New Jersey 07974, U.S.A., 1981. Reprinted in Card et al., Readings in Information Visualization: Using Vision to Think.

[30] George Furnas. Generalized fisheye views. In *Proceedings of the ACM Conference on Human Factors in Computer Systems*, SIGCHI Bulletin, pages 16–23, New York, U.S.A., 1986. Association for Computer Machinery.

[31] Thomas Gerstner. Multiresolution extraction and rendering of transparent isosurfaces. *Computers & Graphics*, 26(2):219–228, 2002.

[32] Bruce Goldstein. *Sensation and Perception.* Brooks/Cole Publishing Company, 5^{th} edition, June 1998.

[33] Celso Grebogi, Edward Ott, and James Jorke. Chaos, strange attractors, and fractal basin boundaries in nonlinear dynamics. *Science*, 238:256–261, 1987.

[34] Donna Gresh, Bernice Rogowitz, Raimond Winslow, David Scollan, and Christina Yung. WEAVE: a system for visually linking 3-D and statistical visualizations, applied to cardiac simulation and measurement data. In *Proceedings of IEEE Visualization 2000*, pages 489–492, 2000.

[35] Markus Hadwiger, Christoph Berger, and Helwig Hauser. High-quality two-level volume rendering of segmented data sets on consumer graphics hardware. In *Proceedings of IEEE Visualization 2003*, pages 301–308, 2003.

[36] Hans-Peter Pfister and William Lorensen and Chandrajit Bajaj and Gordon Kindlmann and William Schroeder and Lisa Sobierajski-Avila and Ken Martin and Raghu Machiraju and Jinho Lee. Visualization viewpoints: the transfer function bake-off. *IEEE Computer Graphics and Applications*, 21(3):16–23, 2001.

[37] Helwig Hauser, Florian Ledermann, and Helmut Doleisch. Angular brushing for extended parallel coordinates. Technical Report TR-VRVis-2002-015, 2002. http://www.VRVis.at/vis/.

[38] Helwig Hauser, Florian Ledermann, and Helmut Doleisch. Angular brushing for extended parallel coordinates. In *Proceedings of the IEEE Symposium on Information Visualization 2002*, pages 127–130. IEEE, October 2002.

[39] Helwig Hauser and Matej Mlejnek. Interactive volume visualization of complex flow semantics. In *Proceedings of the 8th Fall Workshop on Vision, Modeling, and Visualization*, pages 191–198, München, Germany, November 2003.

[40] Helwig Hauser, Lukas Mroz, Gian-Italo Bischi, and Eduard Gröller. Two-level volume rendering - fusing MIP and DVR. In *Proceedings of IEEE Visualization 2000*, pages 211–218, 2000.

[41] Helwig Hauser, Lukas Mroz, Gian-Italo Bischi, and Eduard Gröller. Two-level volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):242–252, 2001.

[42] Taosong He, Lichan Hong, Arie Kaufman, and Hans-Peter Pfister. Generation of transfer functions with stochastic search techniques. In *Proceedings of IEEE Visualization 1996*, pages 227–234, 1996.

[43] James Helman and Lambertus Hesselink. Visualization of vector field topology in fluid flows. *IEEE Computer Graphics and Applications*, 11(3):36–46, 1991.

[44] Chris Henze. Feature detection in linked derived spaces. In *Proceedings of IEEE Visualization 1998*, pages 87–94, 1998.

[45] Jian Huang, Klaus Mueller, Naeem Shareef, and Roger Crawfis. Fastsplats: optimized splatting on rectilinear grids. In *Proceedings of IEEE Visualization 2000*, pages 219–226, Salt Lake City, USA, October 2000. IEEE.

[46] IAL by Global Majic Software, http://www.globalmajic. com.

[47] Alfred Inselberg. The plane with parallel coordinates. *The Visual Computer*, 1(2):69–92, 1985.

[48] Alfred Inselberg. A survey of parallel coordinates. In Hans-Christian Hege and Konrad Polthier, editors, *Mathematical Visualization*, pages 167–179. Springer Verlag, Heidelberg, 1998.

[49] Alfred Inselberg and Bernard Dimsdale. Parallel coordinates: a tool for visualizing multidimensional geometry. In *Proceedings of IEEE Visualization 1990*, pages 361–378, 1990.

[50] Victoria Interrante. Illustrating surface shape in volume data via principal direction-driven 3D line integral convolution. In *Proceedings of ACM SIGGRAPH 1997*, pages 109–116, August 1997.

[51] Victoria Interrante, Henry Fuchs, and Stephen Pizer. Illustrating transparent surfaces with curvature-directed strokes. In *Proceedings of IEEE Visualization 1996*, pages 211–218, 1996.

[52] Bruno Jobard and Wilfrid Lefer. Creating evenly-spaced streamlines of arbitrary density. In *Proceedings of the EUROGRAPHICS Workshop on Visualization in Scientific Computing 1997*, pages 43–56. Springer, 1997.

[53] Naftali Kadmon and Eli Shlomi. A polyfocal projection for statistical surfaces. *The Cartography Journal*, 15(1):36–41, 1978.

[54] James Kajiya. Ray tracing volume densities. In *Proceedings of ACM SIGGRAPH 1984*, pages 165–174, 1984.

[55] Steven Katz. *Film directing shot by shot: visualizing from concept to screen.* Focal Press, 1991.

[56] Alan Keahey. The generalized detail-in-context problem. In *Proceedings of the IEEE Symposium on Information Visualization 1998*, pages 44–52. IEEE Computer Society Press, 1998.

[57] Alan Keahey and Edward Robertson. Techniques for non-linear magnification transformations. In *Proceedings of the IEEE Symposium on Information Visualization 1996*, pages 38–45. IEEE, 1996.

[58] Alan Keahey and Edward Robertson. Nonlinear magnification fields. In *Proceedings of the IEEE Symposium on Information Visualization 1997*, pages 51–58. IEEE, October 1997.

[59] Gordon Kindlmann and James Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In *Proceedings of the IEEE Symposium on Volume Visualization 1998*, pages 79–86, 1998.

[60] Erich Klement, Radko Mesiar, and Endre Pap. *Triangular Norms*, volume 8 of *Trends in Logic*. Kluwer Academic Publishers, Dordrecht, 2000.

[61] Joe Kniss, Gordon Kindlmann, and Charles Hansen. Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In *Proceedings of IEEE Visualization 2001*, pages 255–262, 2001.

[62] Andreas König and Eduard Gröller. Mastering transfer function specification by using volumepro technology. In *Proceedings of SCCG 2001*, pages 279–286, 2001.

[63] Robert Kosara, Helwig Hauser, and Donna Gresh. An interaction view on information visualization. In *EUROGRAPHICS 2003 State-of-the-Art Reports*, pages 123–137, Granada, Sept. 2003.

[64] Robert Kosara, Silvia Miksch, and Helwig Hauser. Semantic depth of field. In *Proceedings of the IEEE Symposium on Information Visualization 2001*, pages 97–104. IEEE Computer Society Press, 2001.

[65] Robert Kosara, Silvia Miksch, and Helwig Hauser. Focus + context taken literally. *IEEE Computer Graphics and Applications*, 22(1):22–29, 2002.

[66] Robert Kosara, Silvia Miksch, Helwig Hauser, Johann Schrammel, Verena Giller, and Manfred Tscheligi. Useful properties of semantic depth of field for better f+c visualization. In *Proceedings of the Joint IEEE TCVG – EUROGRAPHICS Symposium on Visualization 2003*, pages 205–210, 2003.

[67] Kevin Kreeger and Arie Kaufmann. Mixing translucent polygons with volumes. In *Proceedings of IEEE Visualization 1999*, pages 191–198, 1999.

[68] Matthias Kreuseler, Norma López, and Heidrun Schumann. A scalable framework for information visualization. In *Proceedings of the IEEE Symposium on Information Vizualization 2000*, pages 27–38, October 2000.

[69] LabView by National Instruments, http://www.natinst.com/ labview.

[70] Philip Lacroute and Marc Levoy. Fast volume rendering using a shear-warp factorisation of the viewing transform. In *Proceedings of ACM SIGGRAPH 1994*, pages 451–459, 1994.

[71] John Lamping and Ramana Rao. The hyperbolic browser: a focus + context technique for visualizing large hierarchies. *Journal of Visual Languages and Computing*, 7(1):33–35, 1996.

[72] John Lamping and Ramana Rao. Visualizing large trees using the hyperbolic browser. In *Proceedings of the 1996 Conference on Human Factors in Computing Systems, CHI*, pages 388–389, April 1996.

[73] John Lamping, Ramana Rao, and Peter Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proceedings CHI 1995*, pages 401–408. ACM, 1995.

[74] John Lansdown and Simon Schofield. Expressive rendering: a review of nonphotorealistic techniques. *IEEE Computer Graphics and Applications*, 15(3):29–37, May 1995.

[75] Hsien-Che Lee. Review of image-blur models in a photographic system using principles of optics. *Optical Engineering*, 29(5):405–421, May 1990.

[76] Ying Leung. Human-computer interface techniques for map based diagrams. In *Proceedings of the Third International Conference on Human-Computer Interaction*, volume 2 of *Designing and Using Human-Computer Interfaces and Knowledge Based Systems; Graphics*, pages 361–368, 1989.

[77] Ying Leung and Mark Apperley. A review and taxonomy of distortion-oriented presentation techniques. *ACM Transactions on Computer-Human Interaction*, 1(2):126–160, June 1994.

[78] Marc Levoy. Display of surfaces from volume data. *IEEE Computer Graphics & Applications*, 8(5):29–37, 1988.

[79] Henry Lieberman. A multi-scale, multi-layer, translucent virtual space. In *IEEE International Conference on Information Visualization*, pages 126–131, London, September 1997. IEEE.

[80] Helwig Löffelmann. *Visualizing Local Properties and Characteristic Structures in Dynamical Systems*. PhD thesis, Vienna University of Technology, Austria, 1998. http://www.VRVis.at/vis/resources/diss-HL/.

[81] Helwig Löffelmann and Eduard Gröller. Enhancing the visualization of characteristic structures in dynamical systems. In *Proceedings of the EUROGRAPHICS Workshop on Visualization in Scientific Computing 1998*, pages 59–68. Springer, 1998.

[82] Ishantha Lokuge and Suguru Ishizaki. Geospace: an interactive visualization system for exploring complex information spaces. In *CHI 1995 Proceedings*, pages 409–414, 1995.

[83] William Lorensen and Harvey Cline. Marching cubes: a high resolution 3D surface construction algorithm. In *Proceedings of ACM SIGGRAPH 1987*, pages 163–189, 1987.

[84] Jock Mackinlay, George Robertson, and Stuart Card. The perspective wall: detail and context smoothly integrated. In *Proceedings of ACM CHI 1991 Conference on Human Factors in Computing Systems*, Information Visualization, pages 173–179, 1991.

[85] Joe Marks, Brad Andalman, Paul Beardsley, William Freeman, Sarah Gibson, Jessica Hodgins, Thomas Kang, Brian Mirtich, Hans-Peter Pfister,

Wheeler Ruml, Kathy Ryall, Josh Seims, and Stuart Shieber. Design galleries: a general approach to setting parameters for computer graphics and animation. In *Proceedings of ACM SIGGRAPH 1997*, pages 389–400, 1997.

[86] Allen Martin and Matthew Ward. High dimensional brushing for interactive exploration of multivariate data. In *Proceedings of IEEE Visualization 1995*, pages 271–278, 1995.

[87] Krešimir Matković, Helwig Hauser, Reinhard Sainitzer, and Eduard Gröller. Process visualization with levels of detail. Technical Report TR-VRVis-2002-016, 2002. http://www.VRVis.at/vis/.

[88] Krešimir Matković, Helwig Hauser, Reinhard Sainitzer, and Eduard Gröller. Process visualization with levels of detail. In *Proceedings of the IEEE Symposium on Information Visualization 2002*, pages 67–70. IEEE Computer Society, 28–29 October 2002.

[89] Krešimir Matković, Laszlo Neumann, and Werner Purgathofer. A survey of tone mapping techniques. In *Proceedings of SCCG 1997*, pages 163–170, Budimerce, Slovakia, 1997. Comenius University.

[90] Nelson Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, June 1995.

[91] Tom McReynolds and David Blythe. Advanced graphics programming techniques using OpenGL. SIGGRAPH 2000 Course 32, Course Notes, 2000.

[92] Cristian Mira, Laura Gardini, Alexandra Barugola, and Jean-Claude Cathala. *Chaotic Dynamics in Two-Dimensional Noninvertible Maps*. World Scientific, Singapore, 1996.

[93] Kazuo Misue, Peter Eades, Wei Lai, and Kozo Sugiyama. Layout adjustment and the mental map. *Journal of Visual Languages and Computing*, 6(2):183–210, June 1995.

[94] Lukas Mroz. *Real-Time Volume Visualization on Low-End Hardware*. PhD thesis, Vienna University of Technology, Austria, 2001. http://www.cg.tuwien.ac.at/ mroz/diss/.

[95] Lukas Mroz and Helwig Hauser. RTVR - a flexible java library for interactive volume rendering. In *Proceedings of IEEE Visualization 2001*, pages 279–286, October 2001.

[96] Lukas Mroz and Helwig Hauser. Space-efficient boundary representation of volumetric objects. In *Proceedings of the Joint IEEE TCVG – EUROGRAPHICS Symposium on Visualization 2001*, pages 193–202, 2001.

[97] Lukas Mroz, Helwig Hauser, and Eduard Gröller. Interactive high-quality maximum intensity projection. In *Proceedings of EUROGRAPHICS 2000*, pages 341–350.

[98] Lukas Mroz, Andreas König, and Eduard Gröller. Real-time maximum intensity projection. In *Proceedings of the Joint IEEE TCVG – EUROGRAPHICS Symposium on Visualization 1999*, 1999.

[99] Lukas Mroz, Andreas König, and Eduard Gröller. Maximum intensity projection at warp speed. *Computers and Graphics*, 24(3):343–352, 2000.

[100] Lukas Mroz, Rainer Wegenkittl, and Eduard Gröller. Mastering interactive surface rendering for Java-based diagnostic applications. In *Proceedings of IEEE Visualization 2000*, pages 437–440, 2000.

[101] Tamara Munzner. Drawing large graphs with H3Viewer and Site Manager. In *Proceedings of Graph Drawing 1998*, number 1547 in Lecture Notes in Computer Science, pages 384–393. Springer Verlag, August 1998.

[102] Laszlo Neumann, Balázs Csébfalvi, Andreas König, and Eduard Gröller. Gradient estimation in volume data using 4D linear regression. In *Proceedings of EUROGRAPHICS 2000*, pages C–351–C–357, 2000.

[103] Hans-Peter Pfister, Jan Hardenbergh, Jim Knittel, Hugh Lauer, and Larry Seiler. The volume pro real-time ray-casting system. In *Proceedings of ACM SIGGRAPH 1999*, pages 251–260, 1999.

[104] Bui-Tuong Phong. Illumination for computer generated pictures. *Comm. of the ACM*, 18(6):311–317, 1975.

[105] Frits Post, Benjamin Vrolijk, Helwig Hauser, Robert Laramee, and Helmut Doleisch. Feature extraction and visualization of flow fields. In *Eurographics 2002 State-of-the-Art Reports*, pages 69–100, September 2002.

[106] Michael Potmesil and Indranil Chakravarty. A lens and aperture camera model for synthetic image generation. In *Proceedings of ACM SIGGRAPH 1981*, pages 297–305, August 1981.

[107] Real-Time Graphics Tools by Quinn-Curtis, http://www. quinn-curtis.com.

[108] Freek Reinders. *Feature-based Visualization of Time-dependent Data*. PhD thesis, Delft University of Technology, The Netherlands, 2001.

[109] Wolfgang Rieger. Neue Strategien für interaktive Processvisualisierung. Master's thesis, Vienna University of Technology, 2002.

[110] Takafumi Saito. Real-time previewing for volume visualization. In *Proceedings of the IEEE Symposium on Volume Visualization 1994*, pages 99–106. ACM SIGGRAPH, October 1994.

[111] Takafumi Saito and Tokiichiro Takahashi. Comprehensible rendering of 3-D shapes. In *Proceedings of ACM SIGGRAPH 1990*, pages 197–206, August 1990.

[112] Georgios Sakas, Marcus Grimm, and Alexandros Savopoulos. Optimized maximum intensity projection. In *Proceedings of 5th EUROGRAPHICS Workshop on Rendering Techniques*, pages 55–63, Dublin, Ireland, 1995.

[113] Manojit Sarkar and Marc Brown. Graphical fisheye views of graphs. In *Proceedings of ACM CHI 1992 Conference on Human Factors in Computing Systems*, Visualizing Objects, Graphs, and Video, pages 83–91, 1992.

[114] Manojit Sarkar and Marc Brown. Graphical fisheye views. *Communications of the ACM*, 37(12):73–83, December 1994.

[115] Manojit Sarkar, Scott Snibbe, Oren Tversky, and Steven Reiss. Stretching the rubber sheet: a metaphor for visualizing large layouts on small screens. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, Visualizing Information, pages 81–91, 1993.

[116] Yoshinobu Sato, Nobuyuki Shiraga, Shin Nakajima, Shinichi Tamura, and Ron Kikinis. LMIP: local maximum intensity projection - a new rendering method for vascular visualization. *Journal of Computer Assisted Tomography*, 22(6):912–917, 1998.

[117] Gerik Scheuermann, Heinz Krüger, Martin Menzel, and Alyn Rockwood. Visualizing nonlinear vector field topology. *IEEE Transactions on Visualization and Computer Graphics*, 4(2):109–116, April – June 1998.

[118] Berthold Schweizer and Abe Sklar. *Probabilistic Metric Spaces*. Noth-Holland, New York, 1983.

[119] Ben Shneiderman. The eyes have it: a task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Symp. on Visual Languages*, pages 336–343, 1996.

[120] Lisa Sobierajski and Arie Kaufman. Volumetric ray tracing. In *Proceedings of the IEEE Symposium on Volume Visualization 1994*, pages 11–18, 1994.

[121] Robert Spence and Mark Apperley. Data base navigation: an office environment for the professional. *Behaviour and Information Technology*, 1(1):43–54, 1982.

[122] Aleksander Stompel, Eric Lum, and Kwan-Liu Ma. Feature-enhanced visualization of multidimensional, multivariate volume data using non-photorealistic rendering techniques. In *Proceedings of Pacific Graphics 2002*, pages 394–402, 2002.

[123] Maureen Stone, Ken Fishkin, and Eric Bier. The movable filter as a user interface tool. In *Proceedings of ACM CHI 1994 Conference on Human Factors in Computing Systems*, volume 1 of *Information Visualization*, pages 306–312, 1994.

[124] Sun Microsystems. Java. URL: `http://java.sun.com/`.

[125] Holger Theisel. Higher order parallel coordinates. In *Proceedings of the 5th Fall Workshop on Vision, Modeling, and Visualization*, pages 415–420, Saarbrücken, Germany, November 2000.

[126] Ulf Tiede, Thomas Schiemann, and Karl-Heinz Höhne. High quality rendering of attributed volume data. In *Proceedings of IEEE Visualization 1998*, pages 255–262, 1998.

[127] Steve Treavett and Min Chen. Pen-and-ink rendering in volume visualisation. In *Proceedings of IEEE Visualization 2000*, pages 203–210. IEEE Computer Society Technical Committee on Computer Graphics, 2000.

[128] Anne Treisman. Preattentive processing in vision. *Computer Vision, Graphics, and Image Processing*, 31:156–177, 1985.

[129] TTTech Computertechnik AG, http://www.tttech.com.

[130] Jayaram Udupa and Gabor Herman. *3D Imaging in Medicine*. CRC Press, 1999.

[131] Theo van Walsum, Frits Post, Deborah Silver, and Frank Post. Feature Extraction and Iconic Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2(2):111–119, June 1996.

[132] Pantelis Vlachos and Mike Meyer. StaLib – data, software and news from the statistics community. http://lib.stat.cmu.edu/, 2002.

[133] Matthew Ward. XmdvTool: integrating multiple methods for visualizing multivariate data. In *Proceedings of IEEE Visualization 1994*, pages 326–336, 1994.

[134] Gunther Weber, Gerik Scheuermann, Hans Hagen, and Bernd Hamann. Exploring scalar fields using critical isovalues. In *Proceedings of IEEE Visualization 2002*, pages 171–178, 2002.

[135] Rainer Wegenkittl, Helwig Löffelmann, and Eduard Gröller. Visualizing the behavior of higher dimensional dynamical systems. In *Proceedings of IEEE Visualization 1997*, pages 119–125, 1997.

[136] Manfred Weiler and Thomas Ertl. Hardware-software-balanced resampling for the interactive visualization of unstructured grids. In *Proceedings of IEEE Visualization 2001*, pages 199–206, October 2001.

[137] Lee Westover. Footprint evaluation for volume rendering. In *Proceedings of ACM SIGGRAPH 1990*, pages 367–376, August 1990.

[138] Graham Wills. 524,288 ways to say "this is interesting". In *Proceedings of the IEEE Symposium on Information Visualization 1996*, pages 54–61, 1996.

[139] Steven Wixson. Four-dimensional processing tools for cardiovascular data. *IEEE Computer Graphics and Applications*, 3(5):53–59, August 1983.

[140] Steven Wixson. The display of 3D MRI data with non-linear focal depth cues. In *Computers in Cardiology*, pages 379–380. IEEE, September 1990.

[141] Malte Zöckler, Detlev Stalling, and Hans-Christian Hege. Interactive visualization of 3D-vector fields using illuminated streamlines. In *Proceedings of IEEE Visualization 1996*, pages 107–113, 1996.

[142] Karel Zuiderveld, Anton Koning, and Max Viergever. Techniques for speeding up high-quality perspective maximum intensity projection. *Pattern Recognition Letters*, 15:507–517, 1994.

Acknowledgements

The last lines of this text should be words of gratitude to all those who have helped to make this work possible. First of all, I have to say "Thank You!" to all those students of mine who collaborated with me – without them none of the papers presented in this thesis would have been possible. Accordingly, many thanks go to **Lukas Mroz** (PhD student, finished 2001, worked on RTVR, 2lVR, etc.), **Robert Kosara** (PhD student, finished 2001, worked on SDOF), **Csébfalvi Balázs** (PhD student, finished 2001, worked on contour rendering), **Helmut Doleisch** (PhD student since 2000, works on F+C visualization of simulation data), **Markus Hadwiger** (PhD student since 2000, works on GPU-based visualization), **Matej Mlejnek** (diploma student, finished 2003, worked on the combination of interactive feature sprecification and flow visualization based on volume rendering), **Florian Ledermann** (undergraduate student, worked on angular brushing), **Wolfgang Rieger** (diploma student, finished 2002, worked on F+C process visualization), **Martin Gasser** (undergraduate student, worked on the F+C visualization of simulation data).

Furtheron I'd like to say "Thank You!" to several colleagues who I collaborated within the scope of this work. Special thanks go to **Gian-Italo Bischi** (researcher in the field of 3D dynamical systems, an application for two-level volume rendering), **Krešimir Matković** (collaborative researcher in the field of scientific visualization), **Silvia Miksch** (collaborative researcher in the field of information visualization), **Werner Purgathofer** (head of the Institute of Computer Graphics and Algorithms at Vienna University of Technology, Austria, for his general support during all the years), and especially to **Meister Eduard Gröller** (for his father-like support and help during all my research career up to now).

Of course, thanks also go to all institutions and funding agencies/programs which supported parts of this work. Especially to mention are the **Institute of Computer Graphics and Algorithms** at Vienna University of Technology, Austria (there I was employed for several years), the **VRVis Research Center** in Vienna (since 2000 I am working for VRVis), and the funding agencies "**Fonds zur Förderung der wissenschaftlichen Forschung (FWF)**" and "**Forschungsförderungsfond für die gewerbliche Wirtschaft (FFF)**" which supported this work through the projects VisMed, BandViz, and Asgaard.

Finally my deepest thanks go to my wonderful wife **Maria Hauser** and my lovely sons **Jakob** and **Oskar Hauser** – through them true happiness came into my life which significantly contributed such that I had the power to actually accomplish this work – special thanks to you!

Thank you all!