DISSERTATION

Visual Analysis of Complex Simulation Data using Multiple Heterogenous Views

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Doktors der technischen Wissenschaften

unter der Leitung von

Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller,
Institut E186 für Computergraphik und Algorithmen,

und

Dipl.-Ing. Dr.techn. Helwig Hauser,
VRVis Zentrum für Virtual Reality und Visualisierung,

eingereicht an der Technischen Universität Wien,
Fakultät für Informatik,

von

Dipl.-Ing. Helmut Doleisch,

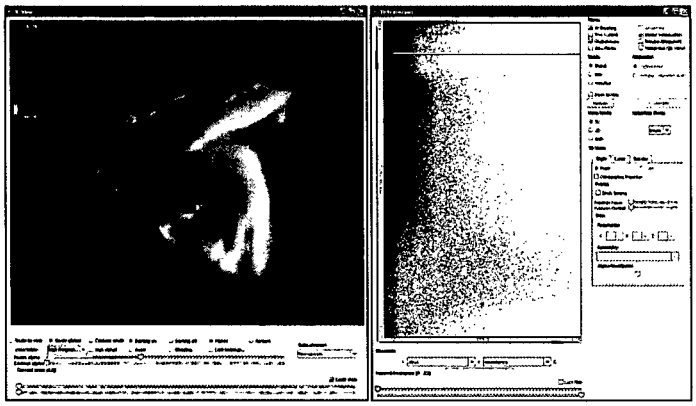
Matrikelnummer 9325735,
Aspettenstrasse 34/26/2,
A-2380 Perchtoldsdorf

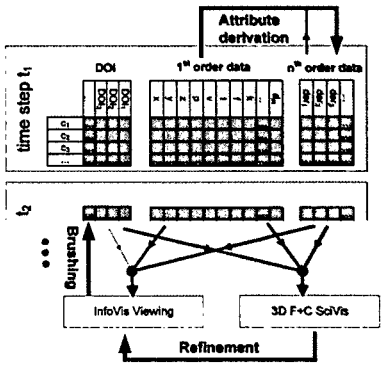
Wien, im Oktober 2004

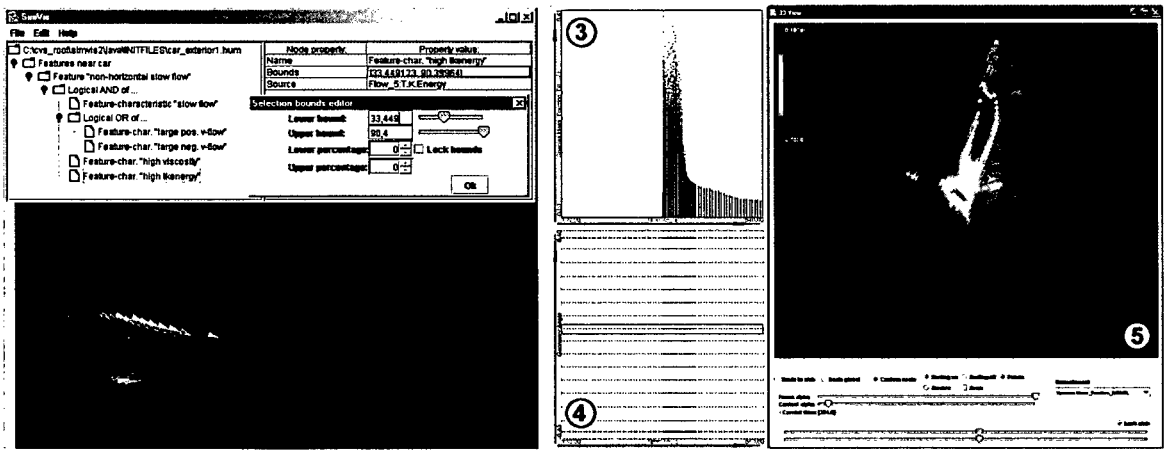
Helmut Doleisch

Visual Analysis of Complex Simulation Data using Multiple Heterogenous Views

Helmut Doleisch, PhD thesis







mailto:Doleisch@VRVis.at
http://www.VRVis.at/vis/resources/diss-HD/

Abstract

Computational Fluid Dynamics (CFD) simulation has become very popular and is used in a wide variety of applications. Applications range from the automotive industry to aerodynamics to environmental and weather simulation, and many more. CFD simulation is popular for several reasons, including that many phenomena can be studied more easily through simulation. Measuring approaches might influence and change flow behavior. Computational simulation speeds up the design and development process of many products.

Typically, CFD simulation results in very large data sets. Results are also usually time-dependent and multi-variate, including many attributes for each simulated point in space and time, e.g., flow vectors, pressure, temperature, mass fraction values of chemical substances, etc. Analyzing such data sets is not an easy task for the engineers, who have to investigate and evaluate the results. Visualization can be used to support the exploration and analysis of these data sets.

Most current visualization methods for data from 3D flow simulation focus either on displaying geometric objects (e.g., streamlines, isosurfaces, etc.), or on feature-based methods employing special feature extraction and tracking techniques. However, these approaches usually do not allow the user to easily and interactivly investigate the multi-dimensional interrelations between different data attributes. The feature extraction process is usually done in a (semi-)automatic way, not allowing for interactive changes of the feature specification.

The central theme of this thesis is to provide a flexible framework for interactive visual analysis of large, multi-dimensional, and time-dependent data sets resulting from flow simulation. In other words, the focus of this work is to develop a framework, which combines multiple, rather well-known concepts from scientific and information visualization, to build a new feature-based visualization framework which is based on user-driven visual analysis. This framework is called *SimVis*.

The major strength of the newly presented visualization approach lies in a balanced combination of several different innovations. These by themselves are not all completely new and some may (to a certain extend) also be found as isolated solutions in other approaches (or in other combinations). Nevertheless, in the combinations proposed here, each component builds an integral part of the framework, which combines different individual solutions to attain maximal flexibility, while still providing solid and stable analysis tools.

The innovations that contribute to this interactive feature specification framework include (1) the combination of views and methods from scientific visualization and information visualization, (2) a sophisticated interaction scheme allowing for fast and flexible information drill-down by means of advanced brushing mechanisms, (3) a fuzzy notion of feature specification and composite specifications, (4) enabling focus+context visualization (especially in the spatial domain of 3D rendering), (5) providing proper access to the special data dimension of time, and (6) coping with interactive visualization of relatively large data sets on standard PCs. Also, with the help of integrating attribute derivation (a mechanism for interactive

calculation of derived data attributes) and advanced brushing mechanisms, the specification of time-dependent features, i.e., features inherently depending on the special data dimension of time, is realized.

Finally two case studies are presented that demonstrate that the framework presented here is indeed generally applicable (e.g., to the automotive industry, aerodynamics, molding, climate simulations, etc.), and how it compares to other solutions and how it adds additional information and value to current methods.

Kurzfassung

Computational Fluid Dynamics (CFD) Simulationen sind in letzter Zeit in immer häufigerem Einsatz in einer Vielzahl von unterschiedlichsten Anwendungsgebieten. Die Anwendungen reichen dabei vom Einsatz in der Automobilindustrie, über Anwendungen im Gebiet der Aerodynamik, bis hin zu Beispielen aus Umwelt-, Wetter- und Klimasimulationen (und vielen anderen mehr). CFD Simulationen werden aus vielen Gründen immer beliebter und öfter eingesetzt, unter anderem, weil Phänomene leichter durch Simulation am Computer untersucht werden können, oder auch, weil auf Computern berechnete Simulationen normalerweise die Design- und Entwicklungsprozesse von vielen Produkten erheblich beschleunigen.

Typische CFD Simulationen erzeugen sehr große Mengen an Ergebnisdaten. Außerdem sind die Ergebnisse normalerweise auch zeitabhängig und multivariat, was bedeutet, dass eine Vielzahl an verschiedenen Datenattributen für jeden Datenpunkt im Raum und für jeden Zeitschritt der Simulation vorhanden ist. Beispiele von solchen Datenattributen sind Strömungsvektoren und -geschwindigkeiten, Druck, Temperatur und Konzentrationen von bestimmten chemischen Substanzen. Die Analyse von solchen Ergebnisdatensätzen ist oft nicht einfach für die Ingenieure, die die Daten untersuchen und bewerten sollen. Dabei kann Visualisierung unterstützend eingesetzt werden.

Die meisten heute verwendeten Visualisierungsmethoden für Daten welche aus einer 3D Strömungssimulation resultieren, verwenden entweder geometrische Strömungsvisualisierungsmethoden (wie z.B.: Streamlines, Isosurfaces, etc.), oder sogenannte merkmals-basierte Methoden, wo zuerst Merkmale (Features) in den Daten extrahiert werden müssen und dann Feature Tracking durchgeführt wird. Allerdings erlauben diese Ansätze normalerweise keine interaktive und einfache Steuerung des Visualisierungsprozesses. Im Speziellen kann nicht interaktiv festgelegt werden, welche Daten gerade von größtem Interesse sind. Die Spezifikation der Merkmale ist normalerweise nur (semi-)automatisch möglich.

Das zentrale Thema dieser Dissertation ist die Entwicklung eines flexiblen Systems für die interaktive visuelle Analyse von großen, multi-dimensionalen und zeitabhängigen Ergebnissen von Strömungssimulationen. Dazu werden mehrere bekannte Methoden und Technologien aus den Bereichen der Visualisierung von wissenschaftlichen Daten (Scientific Visualization, SciVis) und der Informationsvisualisierung (Information Visualization, InfoVis) kombiniert, um daraus einen neuen Ansatz für ein merkmals-basiertes Visualisierungskonzept abzuleiten. Das System, in dem dieser Ansatz exemplarisch angewandt und entwickelt wird, heißt *SimVis*.

Die besondere Stärke des hier neu präsentierten Visualisierungsansatzes liegt in einer ausgewogenen Kombination einer Vielzahl von kleinen Innovationen. Diese alleine sind nicht alle komplett neu, bzw. wurden sie schon in anderen isolierten Lösungen und Ansätzen

verwendet (teilweise in anderen Kombinationsformen). Allerdings stellt die hier präsentierte Form der Kombination dieser Einzellösungen einen neuen Ansatz dar, welcher gleichzeitig maximale Flexibilität auf der einen Seite, und einen stabilen Analyseprozess auf der anderen Seite ermöglicht.

Die einzelnen Innovationen, die zu diesem neuen Ansatz beitragen, beinhalten (1) eine Kombination von Ansichten und Methoden aus SciVis und InfoVis, (2) ein ausgeklügeltes Schema zur Interaktion, basierend auf erweiterten Brushing-Methoden, (3) unscharfe Klassifikationen zur Merkmalsspezifikation (auch zusammengefügte Spezifikationen sind möglich), (4) Fokus+Kontext Visualisierungsmethoden (speziell für die 3D Darstellungen), (5) eine spezielle Behandlung der Zeit, die eine besondere Datendimension darstellt, und (6) das Ermöglichen eines interaktiven Visualisierungsprozesses auch für relativ große Datenmengen auf Standard PC-Systemen. Zusätzlich ermöglicht dieser neue Ansatz durch die Integration von Attributableitungen (eine Methode, um interaktiv neue Datendimensionen, basierend auf Informationen aus den bisher bestehenden Dimensionen, abzuleiten) und erweiterten Brushing-Methoden, eine Spezifikation von zeitabhängigen Merkmalen. Diese Merkmale sind speziell von der zeitlichen Dimension der Daten abhängig.

Abschließend werden sowohl zwei Fallstudien aus dem Bereich der Automobilindustrie präsentiert, als auch die generelle Anwendbarkeit des hier neu entwickelten Ansatzes gezeigt. So lassen sich in SimVis leicht Daten aus den verschiedensten Anwendungsbereichen mit den selben Methoden untersuchen und analysieren, z.B. aus dem Bereich der Aerodynamik, von Klima und Wettersimulationen, von Simulationen von Spritzgussverfahren, aus medizinischen Anwendungsgebieten, usw. Ein Vergleich mit herkömmlichen Methoden zur Visualisierung von Strömungssimulationsdaten zeigt, dass SimVis eine neue, zusätzliche Technologie zur raschen und verständlichen Analyse zur Verfügung stellt.

Contents

Abstract, Kurzfassung iii

Related Publications xi

1 Introduction and Overview 1
- 1.1 CFD Data – Properties and Challenges . . . 1
  - 1.1.1 What is CFD? . . . 2
  - 1.1.2 Why CFD is used? . . . 3
  - 1.1.3 Applications of CFD . . . 4
  - 1.1.4 Data resulting from CFD . . . 4
- 1.2 Visualization – Different Approaches . . . 6
- 1.3 Contribution of this Work . . . 8
- 1.4 Organization of this Thesis . . . 11

2 State of the Art in FlowVis 13
- 2.1 Flow Visualization and Feature Extraction . . . 13
  - 2.1.1 Direct Flow Visualization . . . 15
  - 2.1.2 Dense, Texture-based Flow Visualization . . . 17
  - 2.1.3 Geometric Flow Visualization . . . 18
  - 2.1.4 Feature-based Flow Visualization . . . 18
- 2.2 Multi-Dimensional Data Visualization . . . 26
  - 2.2.1 Multi-Dimensional Data Viewing . . . 27
  - 2.2.2 Focus+Context Visualization . . . 29
  - 2.2.3 Linking and Brushing . . . 29
  - 2.2.4 Analysis and Visualization of Multi-Dimensional CFD Results . . . 30

3 Linking SciVis and InfoVis 33
- 3.1 Dealing with Occlusion in 3D . . . 34
- 3.2 Separating Focus and Context in InfoVis . . . 35
- 3.3 Linking and Brushing in SimVis . . . 36
- 3.4 Smooth Brushing . . . 38
  - 3.4.1 Specifying a Smooth Brush . . . 39
  - 3.4.2 Further Smooth Brushing Results . . . 40
- 3.5 Angular Brushing . . . 42

4 The Feature Definition Framework **47**
4.1 Using a Feature Definition Language 47
4.1.1 Feature Specification 48
4.1.2 Feature Sets 48
4.1.3 Features 49
4.1.4 Feature Characteristics 49
4.2 Interaction 50
4.2.1 Interactive Feature Specification through Brushing 51
4.2.2 Interactive Feature Localization 51
4.2.3 Interactive FDL Refinements 51
4.2.4 Interaction with the Tree Viewer 51
4.2.5 Interactive Data Probing 52
4.2.6 Interactive Management of Views 52
4.3 Visualization and Results from Applications 53
4.3.1 Visualization for Analysis 53
4.3.2 Results from Air-Flow Analysis 54
4.3.3 Results from Catalytic Converter Analysis 56
4.4 Implementation 57
4.5 Discussion 58

5 Time-Dependent Features **59**
5.1 Approaches to Feature-based Flow Visualization 59
5.2 Interactive Specification of Time-Dependent Features 62
5.2.1 Features based on attribute gradients 62
5.2.2 Feature specification relative to data changes 64
5.2.3 Interest which varies over time 66
5.2.4 Features based on stationary attributes 67
5.2.5 Features based on local extrema 67
5.3 Visualization Challenges 68
5.4 Large Data Handling 71
5.5 Application Examples 72
5.5.1 Flood after the burst of a dam 72
5.5.2 Mixing time-shifted flows in an extended T-junction 74
5.6 Discussion and Further Extensions 75

6 SimVis – The Framework **77**
6.1 SimVisBox: Setup and Performance 78
6.2 Data File Format and Data Properties 79
6.3 Software Architecture 81
6.3.1 Feature Definition Framework 81
6.3.2 Views 82
6.3.3 Data Access Layer 86
6.4 Implementation Issues 88
6.5 Performance Optimizations 89
6.5.1 High Level Optimizations 89
6.5.2 Algorithmic Optimizations 89
6.5.3 Low Level Programming Optimizations 90

6.6 Converting Data 92
6.7 Known Limitations and Future Work Plans 93

7 Case Studies **95**
7.1 Visual Analysis of a Diesel Exhaust System 95
7.1.1 Application Scenario 96
7.1.2 Visual Analysis of the Diesel Exhaust System 98
7.1.3 Conclusions and Lessons learned 107
7.2 Visual Analysis of a Diesel Engine 107
7.2.1 Application Scenario 108
7.2.2 SimVis Extensions for Handling Time-Dependent Flow Data based on Time-Varying Grid Geometry 110
7.2.3 Visual Analysis of Diesel Engine FM538 112
7.2.4 Comparison and Conclusions 120

8 Summary **121**
8.1 Combining Multiple Views for Interactive Visual Analysis 121
8.2 Fuzzy Classification 123
8.3 Iterative, Interactive Feature Specification 124
8.4 Time-Dependent Feature Specification 125
8.5 Application Examples and Possibilities 126

9 Conclusions **131**

A Characteristics of Presented Data Sets **133**

B CFD – Data Characteristics **137**

Acknowledgments **143**

Curriculum Vitae **145**

Bibliography **147**

Related Publications

This thesis is based on the following publications:

Helmut Doleisch and Helwig Hauser
Smooth Brushing for Focus+Context Visualization of Simulation Data in 3D,
Proceedings of the 11th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG) 2002, 2002, pp. 147-154.

Helmut Doleisch, Martin Gasser, and Helwig Hauser
Interactive Feature Specification for Focus+Context Visualization of Complex Simulation Data,
Proceedings of the 5th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym) 2003, 2003, pp. 239-248.

Helmut Doleisch, Michael Mayer, Martin Gasser, Roland Wanker, and Helwig Hauser
Case Study: Visual Analysis of Complex, Time-Dependent Simulation Results of a Diesel Exhaust System,
Proceedings of the 6th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym) 2004, 2004, pp. 91-96.

and the following technical reports:

Helmut Doleisch, Helwig Hauser, Martin Gasser, and Robert Kosara
Interactive Focus+Context Analysis of Large, Time-Dependent Flow Simulation Data,
TR-VRVis-2004-024, 2004, VRVis Research Center.

Helmut Doleisch, Michael Mayer, Martin Gasser, Peter Priesching, and Helwig Hauser
Interactive Feature Specification for the Visual Analysis of a Diesel Engine,
TR-VRVis-2004-011, 2004, VRVis Research Center.

Helmut Doleisch and Martin Gasser
SimVis – An Interactive Visualization and Analysis Framework for Large, Time-Dependent, and Multi-Dimensional Flow Simulation Data,
TR-VRVis-2004-027, 2004, VRVis Research Center.

Helwig Hauser and Helmut Doleisch
About SimVis and the Related State of the Art,
TR-VRVis-2004-028, 2004, VRVis Research Center.

During the work on this thesis also the following papers have been published which are related to the here presented work:

Helwig Hauser, Florian Ledermann, and Helmut Doleisch
Angular Brushing of Extended Parallel Coordinates,
Proceedings of IEEE Symposium on Information Visualization, 2002, pp. 127-130.

Robert Kosara, Helmut Doleisch, Martin Gasser, and Helwig Hauser
The SimVis System for Interactive Visual Analysis of Flow Simulation Data
Proceedings of the 2004 Conference "Virtual Product Development" (VDP) in Automotive Engineering, 2004.

Frits Post, Benjamin Vrolijk, Helwig Hauser, Robert Laramee, and Helmut Doleisch
Feature Extraction and Visualization of Flow Fields
State-of-the-Art Proceedings of EUROGRAPHICS (EG) 2002, 2002, pp. 69-100.

Frits Post, Benjamin Vrolijk, Helwig Hauser, Robert Laramee, and Helmut Doleisch
The State of the Art in Flow Visualization: Feature Extraction and Tracking
Journal Computer Graphics Forum (Blackwell CGF), 2003, Vol. 22(4), pp. 775-792.

Robert Laramee, Helwig Hauser, Helmut Doleisch, Benjamin Vrolijk, Frits Post, and Daniel Weiskopf
The State of the Art in Flow Visualization: Dense and Texture-based Techniques
Journal Computer Graphics Forum (Blackwell CGF), 2004, Vol. 23(2), pp. 203-221.

Chapter 1

Introduction and Overview

This chapter gives an overview of the background and motivation of this thesis, as well as of its contribution to the current state of the art. A more detailed discussion of the state of the art together with a review of the most important related works is dealt with in the next chapter of this thesis. This chapter concludes with an overview of the general organization of this thesis.

The major motivation for this thesis is to allow interactive, visual analysis of large, multi-dimensional, and time-dependent simulation data resulting from computational fluid dynamics (CFD) simulation on modern PC-based (and therefore relatively cheap) computer systems. The broad spectrum of application areas for the approaches presented in this thesis range, for example, from the automotive industry, wind, flooding, and avalanche simulation, airplane and spaceship design, to medical applications such as simulating blood flow in human arteries and other related fields. Also in the future data from other related fields, as for example business data originating from financial or insurance applications, customer relationship data, data from stock markets or telecommunication applications and the like, could additionally profit from interactive analysis tools for large, multi-dimensional data sets, as developed during the work presented in this thesis.

1.1 CFD Data – Properties and Challenges

The framework for interactive visual analysis, as presented in this thesis, was initially designed to meet the demands of AVL List GmbH [6], an industrial partner of the VRVis Research Center. The goal was to develop visual analysis tools and methods for simulation data, especially data resulting from CFD simulation. Although most concepts of this work can be transferred more or less directly to other application fields, the work of this thesis focuses on exploring and analyzing such data sets as resulting from CFD simulation.

Below, a short description about what CFD is and how it works, some of its applications, and the resulting data sets is provided. For notes on how general the concepts and technologies presented in this thesis are, and how parts of them can be easily transferred and applied to other fields of multi-dimensional, large data visualization, see chapter 9.

1.1.1 What is CFD?

Computational fluid dynamics (CFD) is concerned with obtaining numerical solutions to fluid flow problems by using computers. The advent of high-speed and large-memory computers has enabled CFD to obtain solutions to many flow problems including those that are compressible or incompressible, laminar or turbulent, chemically reacting or non-reacting [167].

The equations governing fluid flow problems are the **continuity** (conservation of mass), the **Navier-Stokes** (conservation of momentum), and the **energy equations** (conservation of energy). These equations form a system of coupled non-linear partial differential equations (PDEs) [3, 211]. Because of the non-linear terms in these PDEs, analytical methods can only provide very few solutions. In general, these PDEs are often linearized, which is possible either because non-linear terms naturally drop out or because non-linear terms are small compared to other terms so that they can be neglected. If the non-linearities in the governing PDEs cannot be neglected, which is the case for most engineering flows, numerical methods are needed to obtain solutions.

The analytical investigation of linearized equations is an active area of research. Especially for chemically reacting flows and multiphase flows, for example, theoretical developments are still far from being solved [220].

When using CFD for solving fluid flow problems, the differential equations governing the fluid flows have to be replaced with a set of algebraic equations through numerical discretization. These algebraic equations can then be solved with the aid of the computer to get an *approximate solution.* The three most well-known discretization methods used in CFD are the *Finite Difference Method* (FDM), the *Finite Volume Method* (FVM), and the *Finite Element Method* (FEM) [177].

In the discretization step the simulation domain including the boundary of the physical problem is discretized to be covered by a *grid* or *mesh.* Differential equations are replaced by finite difference approximations. In making this replacement, an error which is proportional to the size of the grid elements is introduced. This error can be reduced by increasing the grid resolution to get an accurate solution within some specified tolerance. A trade-off between accuracy and computational complexity has to be made. For more details on grids and related topics see section 1.1.4 as well as Appendix B.

Besides computational fluid dynamics, two older fields of *experimental* and *theoretical fluid dynamics* exist [9, 45]. Experimental fluid dynamics has played (and still plays) an important role in validating and delineating the limits of the various approximations to the governing equations. The wind tunnel, for example, as a piece of experimental equipment, provides effective means of simulating real flows [45]. However, computational fluid dynamics provides an alternative, and often more cost effective means of simulating real flows before building and testing of prototype models. It also sometimes allows testing of conditions unavailable on an experimental basis otherwise (simulating a hurricane or climatic changes, for example) [220].

The role of CFD in engineering has become so strong, that today it may be not only viewed as a new, third dimension of fluid dynamics (the other two being the above stated classical cases of pure experimental and pure theory), but already as *the* most often used method. Due to the development of more powerful computers even in the low-cost PC-based range, as well as mighty super-computers or clustering approaches for multiple computers, further advances in the field of CFD are made constantly. Consequently CFD is now the preferred means of testing alternative designs in most engineering companies, before final, if any, experimental testing takes place [220].

1.1.2 Why CFD is used?

There are many reasons why CFD is employed in such a wide range of applications. Some of the most important reasons are listed below [220].

- Analytical solutions, as employed in theoretical fluid dynamics, exist only for a few of typically simple problems.
- With CFD, problems can be studied, which would not be possible to measure. Imagine, for example, the simulation of a black hole – it would be really hard to measure such a phenomenon.
- Also, measuring devices often influence the behavior of flows or are simply not capable of measuring situations, where closed systems are required. An example would be to measure flow behavior in a combustion chamber of a diesel engine for passenger cars. Here the inclusion of measuring devices both fails to allow closing the chamber completely (at least with the same geometry outline as without the instruments), as well as the devices disturb the normal flow behavior by acting as artificial obstacles.
- Computer models can be used to predict future situations, not measurable at presence. Examples include the simulation of weather or climate systems. Also phenomena, which can not be reproduced in reality, like weather conditions of a special day in the past, can be reproduced and simulated in a quite realistic way.
- In many cases simulating situations on a computer are much cheaper than real laboratory experiments. Typical applications like simulation runs of different settings for crash tests of vehicles or launching of spacecrafts are examples for this class of cases.
- Once a simulation has been setup on a computer, it usually is easier (and thus faster) to change parameters, than with real, experimental test setups.
- The spatial and temporal intervals between individual measuring points is usually limited to a certain distance. Also it would not be possible with experimental fluid dynamics to cover multiple (or at least many different) data attributes per measuring point simultaneously as with CFD. Here often dozens of attributes are available in the resulting data (see also section 1.1.4).

Of course there exist also a couple of disadvantages or at least difficulties with CFD approaches. Some of them are [220]:

- The setting of initial conditions is often very complex, and sometimes such initial conditions to given problems contain a significant level of uncertainty.
- Often processes which are not well understood (e.g. rain formation, some special chemical reactions, turbulence, etc.) have to be parameterized – which is not always easy.
- Typical equations of CFD are partial differential equations (PDE) which require high spatial and temporal resolutions to represent the originally continuous systems.
- Most physically important problems are highly nonlinear – true solutions to these problems are often unknown. Therefore the correctness of the solution is often hard to ascertain, careful and costly validation is required.
- Sometimes a numerical experiment raises more questions than answers are provided.

1.1.3 Applications of CFD

As already mentioned above, CFD is employed in a wide range of application fields in research as well as in product development. Typical application fields include *the automotive industry, aerodynamics applications in general, hydrodynamics, environmental applications, medical and biomedical applications*, as well as *various other industrial and consumer product applications.*

A more detailed overview, together with a description of the various application fields is provided in Appendix B of this thesis.

1.1.4 Data resulting from CFD

As already discussed, the underlying physical problem of a CFD simulation is discretized to be covered by a *grid* or *mesh.* Many different types of grids are known and more or less often used in daily routine when setting up CFD simulation.

The basic structures, that are used to build up grids during the discretization step are **nodes** (or **vertices**, or **points**) and **cells** (or **elements**, or **volumes**) [177]. For simulation carried out in three-dimensional space, a cell is always surrounded by a couple of nodes, which also are the vertices of the boundary faces for the cell. Usually cells are of convex shape, the type of (different) cell shapes which occur in a grid define the type of the grid. Often used cell types in 3D include *tetrahedra*, *hexahedra* (which can be rectilinear or even cubic, for example), *pyramids*, *octahedra*, etc.

There are several different ways of how to categorize the different types of grids. One difference concerns the location of the simulated data at either the nodes of the grid or the cell centers. The two types of grids representing these classes are called *node-centered* and *cell-centered* grids, respectively. Another differentiation is according to the cell types, which are used in composing the grid: grids can be either *structured* or *unstructured.* More details on these classifications are also given in Appendix B of this thesis.

Time aspects for CFD data:

CFD simulation can be either steady-state or time-dependent. Steady-state simulation calculates phenomena which are presumed to be steady over time. If time-varying phenomena are to be simulated, then time-dependent CFD simulation is employed, resulting in time-dependent CFD data. These time-dependent CFD results are stored in data files for several time steps covering the temporal domain. The temporal sampling of the simulation data may be regular, i.e., sampled and stored at regular intervals of time or the temporal sampling may be irregular. In the case of irregular temporal sampling, the data may be stored at a higher temporal frequency during intervals of time that are deemed to be more interesting to the engineer. The subsets of the simulation that are sampled at a faster rate may then be emphasized.

For the different data dimensions of the multi-dimensional simulation results (see also below) two different options according to the existence over time are often encountered. Data for a specific data dimension can either be available for all time steps of a simulation result, or it can be only available for a sub-interval of time, i.e., a sub-set of all available time steps.

When regarding the grids of time-dependent CFD simulations, also three different possibilities exist:

- **Steady grids:** the grids themselves do not change for a time-dependent simulation, only the resulting solutions change over time.

- **Time-varying grids with constant topology:** the grid used can be spatially deformed, but fixed topology remains (neighborhood and structuring of cells). This type of grid is often used to represent geometries, which are expanding or moving over time.
- **Time-varying grids with changing topology:** the grid used is not only changing with respect to spatial deformations, but also the topology of the grid is changing (usually at a few distinct time steps, inbetween these time steps the topology is constant). Such grids are used, if geometries, which are discretized through the grid, are changing drastically, and normal deformation of existing grid topologies is not sufficient to keep a good grid layout over the full temporal domain of the simulation.

Common CFD data characteristics:

Apart from the fact that many different types of grids are used in common CFD simulation, also a few other properties, which characterize CFD results, can be identified and are discussed in the following.

A first property of data sets resulting from CFD simulation is their **multi-dimensional character**. Additionally to the position of each grid element (cell) also a large number of data attributes per cell (data item) is available. Examples for such additional data attributes are flow vectors, temperature or pressure values, or mass fractions of chemical substances, for example. Usually a couple of dozens of additional data attributes are stored per data item.

The **size** of (common) data sets resulting from CFD simulations is influenced by the different characteristics as discussed before:

- the number of grid cells used to build up the discretized version of the spatial domain.
- the number of time steps, for which the CFD results are stored.
- the number of different data dimensions, which are calculated and stored for each time step of the simulation.

Typical data sets sizes range from a few Megabytes (for very simple, steady simulations) to several or even hundreds of Gigabytes (for large sized, complex, time-dependent simulations). For a more detailed presentation of data size properties and sizes see also the Appendix A of this thesis.

Another property of CFD data is, that often the sizes of different cells in a grid differ by a few magnitudes, which is also relevant for many analysis and visualization tasks. Data sets can range from small geometries such as small fluid conduits to mid-range size geometries such as cooling jackets, to large geometries such as automotive exteriors or even environmental outdoor scenarios [112]. The geometric sizes of these grids differ by *six* or more orders of magnitude as well as the sizes of the underlying cells.

For an illustration of the different sizes of cells commonly used in CFD simulation based on complex geometries see figure 1.1. In this figure two views of an intake port system are shown, on the left side an overview and on the right side a close-up view. Intake ports are small valves in a car engine that allow air into the engine's cylinders. When looking at the overview we observe what appears to be four adaptive levels of resolution. But when we zoom in (right side of figure 1.1) we find five adaptive levels of resolution used to evaluate the intake ports themselves. The geometric sizes of the individual cells in this intake port grid differ also by a factor of 1000–2000 [112].

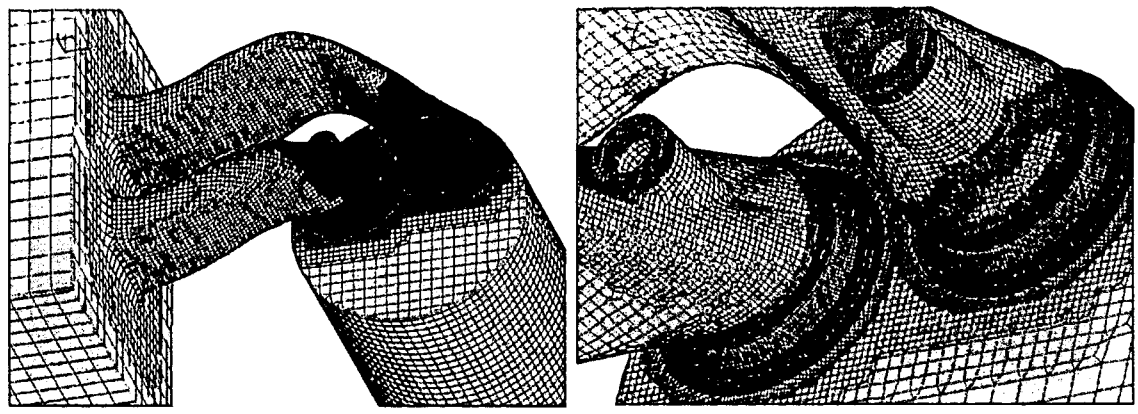


Figure 1.1: The CFD simulation grid of an intake port [112]: overview (left) and close-up view (right). This figure illustrates the large range of cell sizes used in common CFD simulation.

1.2 Visualization – Different Approaches

Once the solution to a CFD simulation case is computed, the next steps include exploring, analyzing, validating and presenting the simulation results. These steps can be supported by visualization and related graphics techniques and therefore are targeted throughout this thesis. Visualization is useful for more than just viewing the computed flow field. It can help with understanding the nature of the problem, with identifying and recognizing interesting relationships of different variables in the simulation output, and also with debugging the simulation process itself.

Visualization is a subfield of computer graphics and deals with the visual representation of data. Visualization aims at supporting the tasks of **exploration**, **analysis**, and **presentation** of (typically) large amounts of data through graphical representations (images, videos, virtual representations in 3D), and therefore, visualization approaches can be differentiated according to how well they fit into the three stages of the visualization process [104].

Visualization for Exploration:

Exploration is usually the first step in data investigation. Before the user can analyze a data set, exploration is carried out, so that the user finds out certain characteristics about the data set, for example, which dimensions are likely to play a major role during the following analysis steps, or which structures are of interest in the given data. Exploration also often serves to check whether the data appears to be valid, and to find and remove any obvious problems, e.g., wrong results (or non-converging results) of a simulation process. Visualization that supports exploration typically should provide tools of maximum flexibility, which can be used interactively.

Visualization for Analysis:

Based on hypotheses which emerged during the exploration phase (or also independently from exploration) the data is now visually analyzed. The final goal is to provide a thorough analysis of all the structures or processes of interest in the data. Verification or falsification of hypotheses can lead to new questions, which also are investigated and analyzed.

Tools which allow flexible analysis of the data must provide some sort of querying possibilities. Therefore, interactivity is a key feature of visualization as part of the analysis process. One example approach is to use interactive brushing (i.e., selection of interesting data subsets) of data items according to certain data attributes (see also section 3.3). In this thesis new tools for interactive visual data analysis are presented.

Visualization for Presentation:

The results and findings gained during the analysis of a data set eventually need to be presented and communicated to others visually. Visualization for presentation usually poses completely different demands on the tools, which should be used for producing such visualizations. Here not the interactivity of an investigation is a primary goal, but rather a high visual quality of the representation of the results. Therefore, tools designed for exploration and/or analysis usually are not directly providing good results for the purpose of presentation, and vice versa.

In practice, the three phases of visualization cannot be separated easily, and many tasks appear in at least two phases of this classification. The framework for interactive visual analysis of flow simulation data, as presented in this thesis, can be used for interactive exploration and analysis, depending on which features of the framework are used primarily. Our framework, however, is only suboptimal for the presentation stage of the visualization process. This is due to the fact that the use and understanding of the interactive and highly flexible visual tools provided needs some (short and basic) introduction usually provided during a training or learning session, to gain maximal support.

With respect to the data, the large field of visualization can be further classified into three main categories: **volume visualization**, **flow visualization**, and **information visualization**.

- **Volume visualization** (VolVis) – the main application field for volume visualization is visualizing data from different medical imaging modalities, e.g., MRI (Magnetic Resonance Imaging) or CT (Computer Tomography). But also in other fields, like material sciences or bio-medical applications, volumetric data has to be visualized. Typical volume visualization techniques include direct volume rendering or isosurface extraction.
- **Flow Visualization** (FlowVis) – vector data, either computed by flow simulation, or measured data using experimental setups, is plotted for the purpose of data investigation. Many different methods are available for the visualization of many different kinds of flows. A detailed overview and discussion of many of those techniques is available in chapter 2.
- **Information Visualization** (InfoVis) – information visualization deals with data that is usually abstract, high-dimensional, and structured in a complex way. Visualization of such data is especially demanding due to the fact, that the user does not have any preconception of how such data could look. The data has usually no spatial layout, which is in contrast to the other two classes (VolVis and FlowVis).

The framework for interactive visual analysis of flow simulation data, as it is presented in this thesis, employs a mix of techniques from flow visualization and also information visualization. More information is available in the following section.

1.3 Contribution of this Work

The goal of this thesis is to present a framework of tools for interactive visual analysis and exploration of large, multi-dimensional, and time-dependent (flow) simulation results.

This thesis introduces the main characteristics of this framework. The approaches included in the framework and presented throughout the remainder of this thesis are designed for interactive exploration and analysis of CFD data sets.

The proposed methodology adds new opportunities for exploration, analysis and presentation of simulation results. The results of our work are not considered as an replacement for existing technologies, nor to be the exclusive solution for the tasks aimed for. Nevertheless, by using the proposed concepts of our framework, additional help for these tasks can be identified. Especially the flexibility offered by an interactive approach is beneficial for most engineers, which are running and supervising CFD simulation cases. The interactive approach as presented in this thesis allows these users to specify their interest during an investigation of the resulting data in a very intuitive way.

The framework for interactive visual exploration and analysis which is presented here is called **SimVis**. SimVis is not only a system (in the form of a research prototype), but rather a technological development. The major strength of SimVis lies in the balanced combination of several different innovations. These by themselves are not all completely new and can (to a certain extent) also be found as isolated solutions (or in other combinations) in other implementations world-wide (see at least partially comparable systems such as XmdvTool [208, 132], XGobi [190, 191], Spotfire [1], IVEE [2], Polaris [187, 186], VisDB [92], etc.). One important inspiration for our research was IBM research work on the visualization of complex simulation data (a beating heart), called WEAVE [61]. This work includes a 3D view for spatial orientation, a scatterplot for visualizing simulation attributes, as well as a histogram and others, which are also visually linked in the context of interactive focussing through brushing (see section 2.2.3 for more details).

Other work which is somehow related to the overall concept of SimVis are linked derived spaces [76] and linking-and-brushing systems [208, 20, 191, 92, 29, and others], for example.

In the following we describe SimVis aspects shortly one-by-one – altogether they contribute that SimVis, as a whole, represents a new technology aimed at supporting an interactive, visual exploration and analysis of CFD data.

Fuzzy classification

In most applications of visualization there are only binary or discrete classifications used to establish a semantic layer on top of the originally unlabeled data [176, 151]. In medical visualization, for example, object segmentation plays an important role, and usually discrete object maps are used to label voxels of either being part of one object or another. Similarly, in flow visualization, also usually a sharp feature extraction process is used to discretely partition the flow domain into portions which represent certain flow features, e.g., a vortex or a recirculation zone.

In SimVis, fuzzy classifications (according to the terminology of fuzzy logic [222]) are used to assign probabilities of class containment. This happens interactively via smooth brushing in information visualization views with respect to what is currently of interest for the user (see section 3.4). Fuzzy logic operations are used to establish a calculus which is based on fuzzy DOI values [101]. This is an important feature of SimVis as it is often not possible to

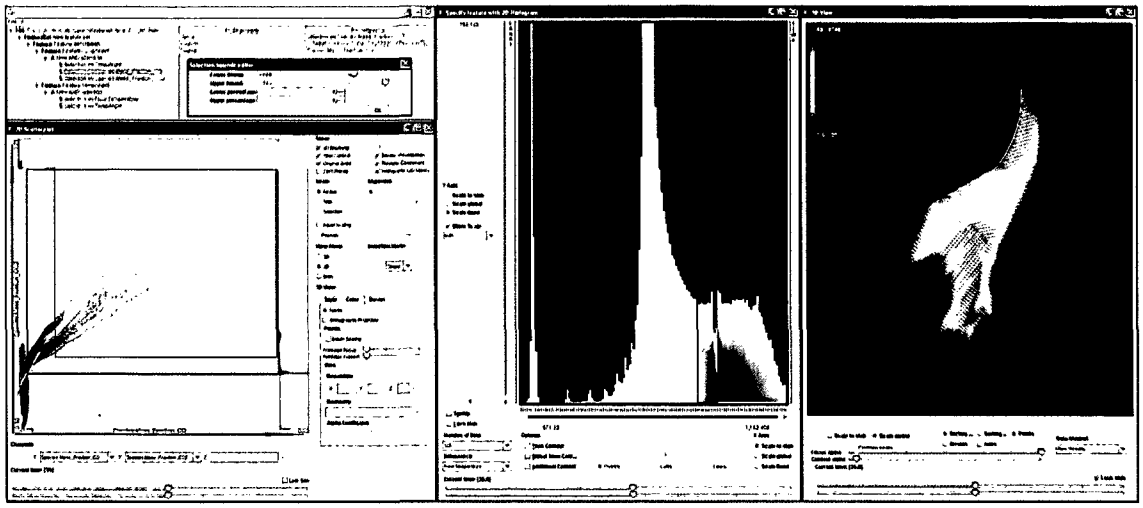


Figure 1.2: A sample SimVis scenario: simulated flow through a diesel particle filter (DPF) is visualized – the flow is shown at the time of 35 secs. after simulation started. The user has reflected his interest in flow regions of heavy oxidation by interactively brushing data items which exhibit a lot of carbon-oxides in the scatterplot (lower left) and then refining this specification to only apply to hot regions (in the histogram, middle). The 3D view on the right shows a focus+context visualization of the DPF with the brushed data items highlighted in color (color shows velocity magnitudes). The upper left view provides a direct and numerical interface to the hierarchical brush definition and all its parameters (more details in the following chapters).

sharply delimit flow portions of interest from all the rest – usually between a certain region of full, i.e., 100% user interest and completely uninteresting portions of the flow (DOI-values of 0) a border region exists for which a gradual change of DOI-values is assumed.

Iterative and interactive feature specification

SimVis utilizes an iterative and interactive approach to feature-based visualization of large and complex data. In SimVis, the setup of synthetic degree-of-interest (DOI) attributes (see chapter 3) is usually started by a simple selection of data items in one view (for example through brushing data values in a scatterplot with respect to two of all available data attributes). After investigating the visual response of this first step (e.g. in the 3D view), iterative refinement is performed to furthermore detail the feature specification in any of the other views (as done in the histogram of figure 1.2 as a second step, for example). The result of such an iterative process is a complex feature specification which is of hierarchical structure and usually involves a set of different data dimensions [33]. In SimVis, this information is explicitly represented to ease user access to feature specifications.

Additionally, the visualization context of all the various parts of such a feature specification also needs to be available when changes are to be made to certain parts of the feature specification. Also, users often want to refine certain parts of a feature specification numerically, especially when certain thresholds carry a specific meaning, e.g., the boiling temperature of water at 100 °C.

The SimVis approach incorporates an explicit representation of all this information, i.e., the hierarchical feature specification with related parameters, called **feature definition language**, together with a separate user interface which enables the direct manipulation of feature specifications [33] (compare to the upper left parts of figure 1.2 or see chapter 4). Such an explicit representation of the feature specification process also immediately enables load- and save-functionality which consequently results in additional advantages such as the opportunity to compare data sets by setting up an analysis for one data set, then saving it, and re-applying the same analysis to another data set.

Multi-view visualization (SciVis–InfoVis combination)

As mentioned before already, the SimVis approach was inspired by IBM research work on the WEAVE system [61], where multiple views (both from scientific visualization (SciVis) and information visualization (InfoVis)) are used in combination to analyze the spatial and high-dimensional data space of simulation results. The multiple-views SimVis approach in conjunction with **linking and brushing** (L&B) allows to combine both directions (see again the figure 1.2 SimVis-example, more details are given in chapter 3). Linking and brushing [20, and others] is a technique where a certain subset of the data is interactively highlighted in one view through brushing and all other views of the same data are updated instantaneously to reflect the data selection in a consistent visual way, e.g., by coloring selected data items red in all views. Typical questions such as where in the flow domain data items adhere to a specific inter-relation of selected data items (brushing in InfoVis views) can be interactively answered easily. Also others such as how data items in a specific location (brushing in the spatial domain) are distributed with respect to certain data attributes can be answered fast. This careful SimVis-combination of visualization approaches from SciVis and InfoVis already resulted in very positive feedback from many engineers, who used the prototype for analysis.

Focus+context 3D visualization

3D viewing is an essential component of the SimVis approach – especially with respect to scientific data, users usually want to be spatially oriented. Up to now, SimVis was successful with rather simple, glyph-based 3D viewing. For every data item in the flow data a glyph (e.g. a small 3D arrow) is drawn with a certain opacity and color. The size of the glyphs is adjusted locally through a transfer function in dependence on a DOI value and globally through a user-defined scaling factor.

One important fact why this approach works so well is the focus+context extension of 3D viewing [66]: according to the feature specification process, where data items are attributed with degree-of-interest values, glyphs are drawn in an emphasized (rather opaque and colored for data items in focus) or in a reduced (rather transparent and in shades of gray) style (right view in figure 1.2). Data items which are associated with fractional DOI values are represented in an interpolated way (interpolated opacity and color). This focus+context rendering of data which is laid out in 3D space enhances the perception on the user side as an efficient part of occlusion control.

Attribute derivation and advanced brushing

Brushing is an intuitive and very effective, but still very simple concept to indicate user interest. One way to characterize this kind of data classification is to understand it as a

shallow, broad-band approach. Major advantages of the brushing approach are:

- that the user is not bound to specific extraction procedures, which usually are linked to specific mathematical formulae, but can select whatever is interesting (especially useful through data exploration),
- that the feature extraction process is easily comprehended by the user (since selections are formulated in explicit terms of the data, no "magic" is going on behind feature extraction), and
- that interactive brushing perfectly fits with an iterative refinement approach where features can be formulated step by step which eases to track down interested regions of the data.

One disadvantage of brushing is that it indeed does not enable the extraction of really complex relations within the flow (which nevertheless often are of great interest). SimVis incorporates two approaches to deal with this limitation of simple brushing [36] (see also chapter 5): On the one hand, SimVis offers advanced brushing mechanisms such as angular brushing [68] which are useful to "dig out" relations within the data which are more complex as compared to the brushing standards. On the other hand, SimVis offers opportunities to interactively derive additional synthetic data attributes on the basis of comprehensible mathematical formulae such as gradient derivation, data smoothing, similarity measures, etc. Through the combination of these two approaches (attribute derivation and advanced brushing) it becomes possible to extract rather complex features from the flow data which are comparable to sophisticated feature extraction processes.

1.4 Organization of this Thesis

The remaining parts of this thesis are organized as follows: Chapter 2 discusses the current state of the art in visualization of data from computational simulation and related research fields. In chapter 3 the fundamental concept of linking scientific visualization and information visualization to enable interactive analysis of large, and multi-dimensional data sets is presented. In this chapter, also two extensions to the standard brushing approach are proposed. Then chapter 4 presents the advanced feature specification framework, used to represent and steer the interactive specification of complex features. After this, chapter 5 dicusses handling of time-dependent data as well as concepts and extensions to define time-dependent features based on attribute derivation and advanced brushing mechanisms.

A short description of a few relevant implementation issues is given in chapter 6. A large part of our research was also devoted to working on many real-world applications mainly coming from the automotive industry. Two detailed case studies are presented in chapter 7. The thesis concludes with a summary of the main contributions, conclusions and implications of this work, acknowledgements, as well as an extensive bibliography.

Chapter 2

State of the Art in Visualization of Flow Simulation Data

This chapter gives an overview of the current state of the art (year 2004) and other work related to this thesis. The overview is split into two major sections. The first one reviews related work in the field of general flow visualization. The second part details on related work in the field of multi-dimensional visualization with special emphasis on the visualization of data resulting from (CFD) simulation.

The first section (discussing the state of the art in flow visualization, with special emphasis on feature-based flow visualization methods, is based on a *State of the Art Report on Flow Visualization* presented at EUROGRAPHICS 2002, entitled "**Feature Extraction and Visualization of Flow Fields**" [150]. Other previously published overview reports and papers related to this chapter include two papers published in the *Computer Graphics Forum* journal [113, 151] and two technical reports [34, 67] published at the VRVis Research Center in Vienna [205].

2.1 Flow Visualization and Feature Extraction

Flow visualization (FlowVis) has been a very attractive subfield of scientific visualization (SciVis) research for a long time. According to the different needs of the users, there are different approaches to flow visualization (see also figure 2.1):

- **Direct flow visualization:** this category of techniques uses a translation that is as direct as possible for representing flow data in the resulting visualization. The result is an overall picture of the flow. Common approaches are drawing arrows (figure 2.2, left) or color coding velocity. These techniques are also called *global techniques*, as they are usually applied to the entire domain, or at least a large part of it. Intuitive pictures can be provided, especially in the case of two dimensions. Solutions of this kind allow immediate investigation of the flow data.
- **Dense, texture-based flow visualization:** similar to direct flow visualization, a texture is computed that is used to generate a dense representation of the flow (Figure 2.2, middle). A notion of where the flow moves is incorporated through co-related texture values along the vector field. In most cases this effect is achieved through filtering of texture values according to the local flow vector. Again these techniques are

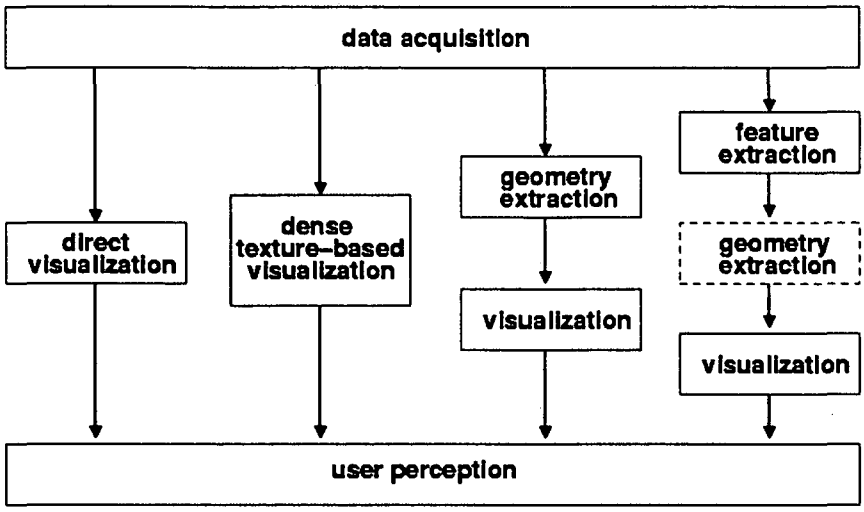


Figure 2.1: Classification of flow visualization techniques – direct (left), texture-based (middle-left), based on geometric objects (middle-right), and feature-based (right) [113].

mainly used in the two-dimensional case or for visualizing flows on surfaces. The results are comparable to the experimental techniques like windtunnel surface oil flows.

- **Geometric flow visualization:** for a better communication of the long-term behavior induced by flow dynamics, *integration-based approaches* first integrate the flow data and use geometric objects as a basis for flow visualization. The resulting integral objects have a geometry that reflects the properties of the flow. Examples include streamlines (Figure 2.2, right), streaklines, pathlines, stream surfaces, time surfaces, or flow volumes. These geometric objects are based on integration as opposed to other geometric objects, like isosurfaces, that may also be useful for visualization. The results of these techniques can be compared to experimental results such as dye advection or smoke injection into the flow.
- **Feature-based flow visualization:** another approach makes use of an abstraction and/or extraction step which is performed before visualization. Special features are extracted from the original data set, such as important phenomena or topological information of the flow. Visualization is then based on these flow features (instead of the entire data set), allowing for compact and efficient flow visualization, even of very large and/or time-dependent data sets. This can also be thought of as visualization of *derived* data.

Figure 2.1 illustrates a classification of the aforementioned classes and Figure 2.2 shows three typical examples. Note that there are different amounts of computation associated with each category. In general, direct flow visualization techniques require less computation than the other three categories, whereas feature-based techniques require the most computation.

The following subsections give a brief overview about the main related works for all four of these flow visualization approaches, with special emphasis on feature-based flow visualization. More detailed reviews and overviews about many more techniques and related works for each of these four approaches are available from several of our survey papers published previously [150, 113, 151].

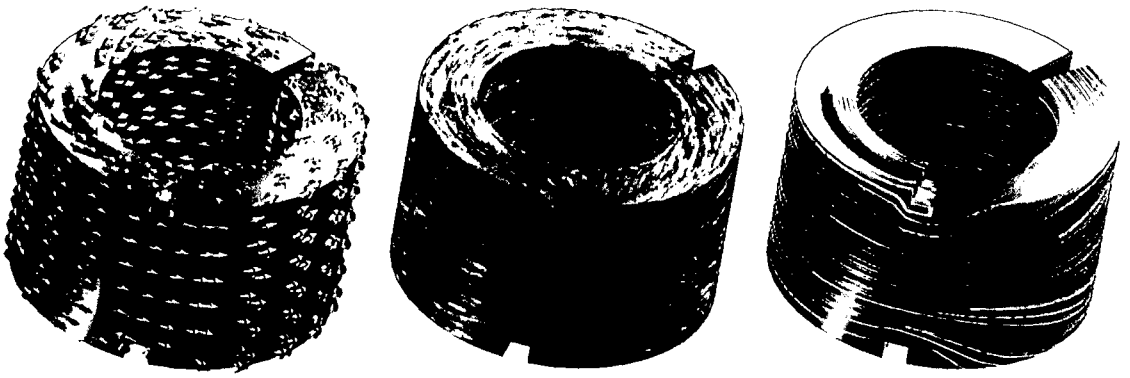


Figure 2.2: An example of circular flow at the surface of a ring to illustrate the classification of flow visualization approaches [113]: direct visualization by the use of arrows (left), dense texture-based visualization by the use of LIC (middle), and a visualization based on geometric objects, here streamlines (right).

2.1.1 Direct Flow Visualization

Direct, or *global*, flow visualization techniques attempt to present the complete data set, or at least a large part of it, at a low level of abstraction. The mapping of data to a visual representation is direct, without complex conversion or extraction steps, as needed for feature-based visualization, for example. These techniques are perhaps the most intuitive visualization strategies as they present the data as is. Unfortunately difficulties arise for time-dependent, as well as also for three-dimensional data sets (due to occlusion and perceptional issues).

A common direct flow visualization technique is to map flow attributes such as velocity, pressure, or temperature to color. In 2D this results in color plots, which are widely distributed and produce very intuitive depictions. Of course, the color scale which is used for mapping the data values to colors must be chosen very carefully and with respect to perceptual issues. One advantage of color coding in 2D is, that it extends also very well to time-dependent data, resulting in changing color plots according to changes of the flow properties over time.

Color coding is also very effectively used for visualizing boundary flows or sectional subsets of 3D flow data. Often it is combined with several other flow visualization techniques for various flow data applications, e.g., in NASA's Field Encapsulation Library [18]. Schulz et al., for example, use color coding of scalars on 2D slices in 3D automotive simulation data [174] as shown in the example in figure 2.3 (middle). Their approach also applies scalar clipping, i.e., the transparent rendering of slice regions not of current interest, which reduces the occlusion problem, when using multiple colored slices.

The natural extension of color coding in 2D (or on slices) is color coding in 3D. This, however, poses special requirements onto rendering due to occlusion problems and nontrivial complexity — volume rendering is needed. Volume rendering is well-known in the field of medical 3D visualization, i.e., volume visualization. However, special challenges, which closely correspond to flow visualization include [150]:

- flow data sets are often significantly smoother than medical data sets — an absence of sharp and clear "object" boundaries (like organ boundaries) makes the mapping to opacities more difficult and less intuitive.
- flow data is often given on non-Cartesian grids, e.g., on curvilinear grids — the complex-

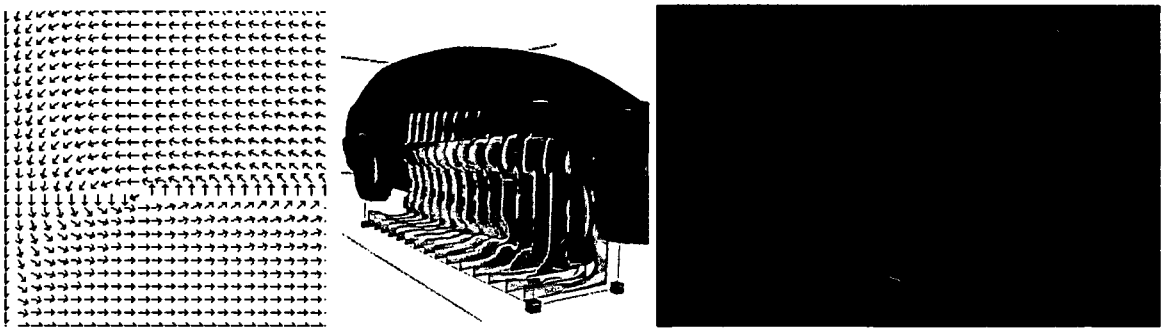


Figure 2.3: Examples of direct flow visualization – the use of arrows in 2D [126] (left), an interactive slicing probe with colored slices and scalar clipping [174] (middle), and direct volume rendering based on resampling [212] (right).

ity of volume rendering gets significantly more tricky on those kinds of grids, starting with nontrivial solutions required for visibility sorting and blending processes.

- flow data is also time-dependent in many cases, imposing additional loads on the rendering process.

In the early nineties, Crawfis et al. [30], as well as Ebert et al. [42] applied volume rendering techniques to vector fields. Little later, Frühauf applied ray casting to vector fields [47]. Westermann presented a relatively fast 3D volume rendering method using a resampling technique for vector field data from unstructured to Cartesian grids [212]. A result from this volume rendering technique is shown in figure 2.3 (right).

Clyne and Dennis [28] as well as Glau [58] presented volume rendering for time-dependent vector fields using algorithms which make special use of graphics hardware. Ono et al. use direct volume rendering to visualize thermal flows in the passenger compartment of an automobile [141]. The support of a system for computational steering is the goal of the work by Swan et al. [189], who also apply direct volume rendering techniques for flow visualization.

Ebert and Rheingans demonstrated the use of nonphotorealistic volume rendering techniques for 3D flow data [40].

Another often used and very natural vector visualization technique is to map a line, arrow, or glyph to each sample point in the field, oriented according to the flow field, as the illustration in figure 2.3 shows on the left side. Klassen and Harrington [100] and Schroeder et al. [173] call this technique a *hedgehog visualization.*

When using 2D arrows on slices from 3D flow data, also effective visualizations can be produced [44]. However, results of such a visualization should be interpreted carefully, as flow components, which are orthogonal to the slice are usually not depicted. These difficulties are basically negligible when talking about boundary surfaces, since in those cases, rarely cross-boundary flows are given. Therefore the use of arrows spread out over boundary surfaces usually is very effective, as used by Treinish for weather visualization [196].

When using arrows in 3D for direct FlowVis, at least two problems are usually apparent: (1) the position and orientation of a vector is often difficult to understand because of its projection onto a 2D screen, and (2) arrow glyphs occluding one another become a problem. The problem of perception can be decreased by using 3D representations of arrows (like a

cylinder plus a cone), the problem of occlusion has to be tackled by using careful seeding strategies for the arrow glyphs (in contrast to dense distributions). In actual applications, arrow plots are usually based on selective seeding, for example, all arrows starting from one out of a few sectional slices through the 3D flow.

Boring and Pang address the problem of clutter in 3D direct FlowVis by highlighting those parts of a 3D arrow plot, which point in a similar direction compared to a user-defined direction [17]. More recent works, e.g., by Heckel et al. [72] and Telea and van Wijk [193], try to solve the problem of cluttering in 3D arrow plots, helping to provide a simple and fast preview of flow fields by applying hierarchical clustering approaches for vector field simplification. Garcke et al. [54, 55] presented a continuous clustering method for 3D arrows used for vector field visualization.

A different approach for direct flow visualization of 2D flows was presented by Kirby et al., who proposed the simultaneous visualization of multiple values by using a layered, hybrid concept [99]. Arrow plots are mixed with color coding to provide visualization results rich of information.

2.1.2 Dense, Texture-based Flow Visualization

Dense, texture-based techniques in flow visualization generally provide full spatial coverage of the vector field. Many different methods of texture-based FlowVis have been published for 2D, as well as also for 3D flows more recently. One of the first presented examples include the *spot noise* technique [213] by van Wijk, where the basic primitive, on which the algorithm operates is a so-called spot (an ellipse) that is distributed and warped along the flow direction. Another early algorithm introduced by Cabral and Leedom is the *Line Integral Convolution* (LIC) [21] method. Here the basic primitive is a noise texture, the properties of which are convolved, or smeared, using a kernel filter in the direction of the underlying vector field. As this algorithm has become very popular soon, many different extensions in several directions have been proposed (see also our state of the art reports mentioned in the beginning of this chapter for more detailed discussions).

A third class of texture-based techniques are so called texture-advection and GPU-based techniques. The primitive in this case is a *moving texel* [133]. Individual texels/texel properties, or groups of texels are advected in the direction of the vector field. Many of the techniques in this category utilize more computation on the GPU (Graphics Processing Unit) – rather than the CPU – in order to realize performance gains.

This group of methods is a very recent research field, most of the techniques, which are classified into this category have been presented during the last 2–4 years. Recent examples include a technique called *Image Based Flow Visualization* (IBFV) by van Wijk [214] which is one of the fastest algorithms for dense, 2D, unsteady vector field representations based on the advection and decay of textures in image space, and a similar dense, texture-based visualization technique on surfaces for unsteady flow called *Image Space Advection* (ISA) by Laramee et al. [114]. IBFV has also been extended to the visualization of flow on surfaces by van Wijk, who presented an extension called IBFVS, IBFV for Surfaces [215], as well as it has been applied to the visualization of 3D flow by Telea and van Wijk [194].

Many more texture-based flow visualization algorithms have been presented over the last years. An extensive overview as well as a detailed discussion of most works is available in our latest state of the art report, published in the journal *Computer Graphics Forum* this year [113].

2.1.3 Geometric Flow Visualization

Geometric FlowVis entails extracting geometric objects for which their shape is directly related to the underlying data. One example, being a natural extension to color coding in is *contouring* in 2D. A contour is a boundary between two distinct regions. Extending contouring from 2D to 3D results in the use of isosurfaces for 3D FlowVis. Special care must be taken with isovalue selection, mostly because of the usually smooth nature of flow data. An example of using isosurfaces for the visualization of flows in unsteady weather data was presented by Treinish [196].

Probably the most often used geometric flow visualization technique is the *streamline* technique. Streamlines are generated through integration of flow vectors along the flow field. When using streamlines, one important aspect is to find the best choice of initial conditions. Since, in general, evenly distributed seed points do not result in evenly spaced streamlines, special algorithms need to be employed. Turk and Banks [200] as well as Jobard and Lefer [85, 86] developed techniques for automatically placing seed points to achieve a uniform distribution of streamlines in 2D vector fields. Verma et al. [202] presented another seed placement strategy for streamlines in 2D flows, which is topology-based. For improving perception of streamlines in 3D, Zöckler et al. presented illuminated streamlines [223]. Bryson and Levit [19] demonstrated interactive seeding of streamlines (and other integral objects) in a virtual 3D environment by using a so-called *rake*.

Additionally many other, different FlowVis techniques for 2D and 3D flow data based on geometric objects have been developed and proposed. A more detailed overview is again available from our state of the art report published at the EUROGRAPHICS 2002 Conference [150].

2.1.4 Feature-based Flow Visualization

Feature-based flow visualization approaches use a higher level of abstraction for the visualization of interesting flow patterns or regions. The visualization shows only those parts that are of special interest to the user, called *features*. Both the definition of what is interesting and the way these features are extracted and visualized are dependent on the data set, the application, and the analysis or research problem.

Features are phenomena, structures or objects in a data set, that are of interest for a certain research or engineering problem. Features can be classified to be either *local* or *non-local*, depending on what information they represent. Local features can be described locally, for example, by means of the Jacobian matrix. Examples include, critical points, low or high pressure areas in the flow, attachment and detachment points and lines, etc. Non-local features are structures, that can not (or only hardly) be described locally, like vortices and their cores, (re-)circulation areas in flows or special topological settings. Here global measures are necessary (like geometric considerations, for example) to detect and describe the features.

There is a number of factors motivating a feature-based approach for flow visualization. First, by extracting only the interesting parts, and ignoring the rest, the information content of the achieved visualization can be increased. Furthermore, by abstracting from the original data, the researcher or engineer is able to focus more on the relevant phenomena or parts of the data. Feature-based flow visualization is also a truly scalable approach to visualizing very large, time-dependent data sets, as a large data reduction can be achieved (in the order of many magnitudes). The original data is then no longer needed, the flow data is described

by the features representing the interesting structures or phenomena in the data. Finally, the objects or phenomena extracted can be simplified and described quantitatively, which allows comparison, tracking over time, feature quantification, and simple mapping to parametric icons, for example.

Feature Extraction:

The first step in feature-based (flow) visualization is feature extraction. The goal of feature extraction is determining, quantifying, and describing the features in a data set. As already mentioned before, many different features are commonly extracted. Although most feature detection techniques are specific for a particular type of feature, in general conventional techniques can be divided into three main approaches: based on *image processing*, on *topological analysis*, and on *physical characteristics*. In this thesis we propose a new feature detection, or specification approach, which is based on an interactive visual analysis of all the data, and discussed in the following chapters.

Image Processing:

The problem of analyzing a numerical data set, represented on a regular rectilinear grid, is similar to analyzing an image data set. Therefore, basic image processing techniques can be used for feature extraction from scientific data. A feature may be distinguished by a typical range of data values, just as different tissue types are segmented from medical images. Thus basic image segmentation techniques, such as thresholding, region growing, and edge detection can be used for feature detection.

From the practical point of view, the fact that CFD simulations are commonly *not* given on regular rectilinear grids, but rather on structured curvilinear or even unstructured grids, poses often a problem. Furthermore, an adaption of most digital filtering techniques which are defined for scalar data, to also work for vector data is not always straightforward.

Vector Field Topology:

A second approach to feature extraction is the topological analysis of flow fields. Vector field topology visualization was introduced by Helman and Hesselink [73, 75]. They present essential information by partitioning the flow field according to its critical points. Lavin et al. present a technique by which they compare 2D vector fields for similarities based upon the characteristics of critical points found in each data set [115]. The goal of their research is the ability to compare computational and experimental flow fields under the same conditions. Batra and Hesselink extend this technique to 3D vector fields [13].

How to properly extract flow topology is a separate issue. Especially when talking about data from flow simulation originating in a (locally) linear computation on a grid, then the extraction of flow topology is non-trivial. Scheuermann et al. worked on higher-order flow topology, i.e., topology of (locally) non-linear flow data [170, 171]. They also investigated improved interpolation schemes for better extraction of flow topology [172].

Tricoche et al. recently presented a topology-based method for visualizing time-dependent 2D vector fields [199]. They perform time tracking of critical points and closed streamlines by temporal interpolation.

Löffelmann et al. worked on the visualization of dynamical systems based on the topology of the flow field [127] (especially near critical points of the systems). An example visualization of the flow near the critical points of the Lorenz-system is shown in figure 2.4, left side. Löffelmann and Gröller use also the results of this topology extraction in 3D dynamical sys-

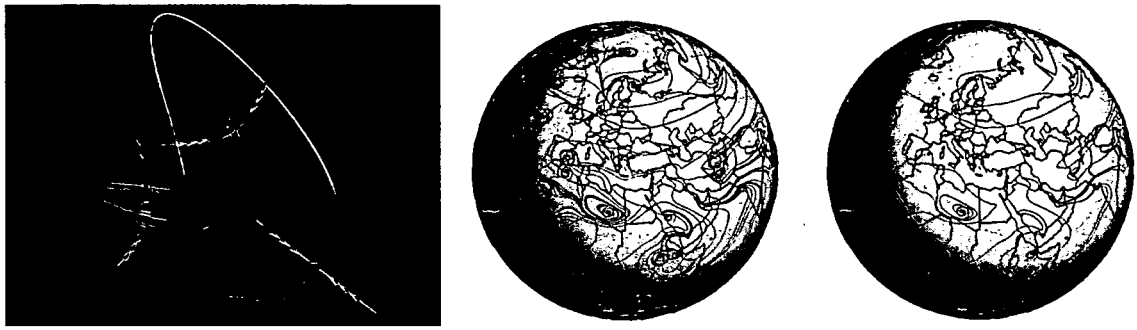


Figure 2.4: Examples of FlowVis based on topology: 3D FlowVis based on the critical point analysis of the Lorenz-system [127] (left), applying multi-levels of topology for FlowVis of meteorological data [120] (middle and right); more critical points for a lower representation level (middle) vs. less critical points for a higher level (right) are shown.

tems flow data for selectively placing streamlets [128]. With this strategy an over-population of phase space (with occlusion problems as a consequence) is avoided.

Another topic of recent research has been the field of **multi-resolution** and **hierarchical flow topologies**. This direction of research originates in the fact, that high-resolution, turbulent flows, as often encountered in CFD simulation data sets, can contain a large number of critical points that clutter the resulting image.

An approach using multiple levels of topology by de Leeuw and van Liere proposes a solution to this problem [119]. Their solution is to use a topological filter called the *pair distance filter* which is defined as the distance between two critical points. By using this filter small topological structures, such as vortices for example, are removed, but the global topological structure of the flow is retained. De Leeuw and van Liere illustrated also the limitations of this first approach using the pair distance filter [118]. As an alternative they proposed a method to calculate the inflow and outflow areas of the critical points, which they apply to data from meteorology [120] (see also figure 2.4 for an example, middle without topology simplification, and right after simplification).

One disadvantage of the methods presented by de Leeuw and van Liere is that they can not handle higher order critical points. This challenge is addressed by Tricoche et al. [197]. Tricoche et al. present also a topological simplification of vector and tensor fields on irregular grids [198]. Wong et al. presented another flow topology simplification approach [218]. In contrast to constantly simplifying the entire vector field, this approach emphasizes keeping full details within certain closed boundaries and eliminating the rest. By adjusting boundary thresholds, a multi-resolution filtering system is achieved. The important factor defining the boundaries is the vorticity of the flow field, thus regions of high vorticity are emphasized. Examples from a climate visualization application are shown.

Physical Characteristics:

The third approach for feature extraction is based on physical characteristics. Often features can be detected by characteristic patterns in, or properties of, physical quantities, e.g. low pressure, high temperature, or swirling flow. The feature extraction techniques discussed in the following paragraphs all are based on this approach, some of them also in combination

with topological analysis or image processing techniques.

Vortex Extraction: Features of great importance in flow data sets, both in theoretical and in practical research, are *vortices*. In some cases, vortices (turbulence) have to be impelled, for example to stimulate mixing of fluids, or to reduce drag. In other cases, vortices have to be prevented, for example around aircrafts, where they can reduce lift.

There are many different definitions of vortices and likewise many different vortex detection algorithms. A distinction can be made in algorithms for finding vortex regions and algorithms which only find vortex cores. Several detailed overviews of many algorithms are available, including the ones by Roth and Peikert [160], by Banks and Singer [12], and by Post et al. [151, 150].

The algorithms for finding vortex regions include besides many others, which are discussed in detail in another overview paper [151], also the following:

- Villasenor and Vincent presented an algorithm for constructing vortex tubes by finding regions with a high vorticity magnitude [203]. Vorticity is the curl of the velocity, i.e., $\nabla \times v$, and represents the local flow rotation, both in speed and direction. They compute the average length of all vorticity vectors contained in small-radius cylinders, and use the cylinder with the maximum average for constructing the Vortex tubes.
- Another idea is to make use of helicity instead of vorticity [124, 221]. The helicity of a flow is the projection of the vorticity onto the velocity, i.e., $(\nabla \times v) \cdot v$. This way, the component of the vorticity perpendicular to the velocity is eliminated.
- Jeong and Hussain define a vortex as a region where two eigenvalues of the symmetric matrix $S^2 + \Omega^2$ are negative, where S and Ω are the symmetric and antisymmetric parts of the Jacobian matrix of the vector field, respectively [82]. This method is known as the λ_2 method.

These methods may all work in certain simple flow data sets, but they do not hold, for example, in turbomachinery flows, which can contain strongly curved vortices [160]. Therefore, some algorithms specifically for finding vortex core lines have been presented, including:

- Globus et al. presented a method for finding core lines by integrating streamlines from the critical points in the velocity field [59].
- It is also possible to use streamlines of the vorticity field [134], but such an algorithm is very sensitive to the starting location.
- Banks and Singer also use streamlines of the vorticity field, with a correction to the pressure minimum in the plane perpendicular to the vortex core [11, 12].
- Roth and Peikert suggest that a vortex core line can be found where vorticity is parallel to velocity [160].
- The same algorithm was presented by Sujudi and Haimes [188], who implemented this algorithm in their pV3 system.
- Recently, Jiang et al. presented a new algorithm for vortex core region detection [83], which is based on ideas derived from combinatorial topology. The algorithm determines for each cell if it belongs to the vortex core by examining its neighboring vectors.
- Kenwright and Haimes also studied the eigenvector method and concluded that it has proven to be effective in many applications [97]. The major drawback of the algorithm is that it does not produce continuous lines. In another paper they also presented a case study, where they identified vortex core lines, spiral vortex breakdowns, vortex bursting and vortex diffusion in an aerodynamics application [96].

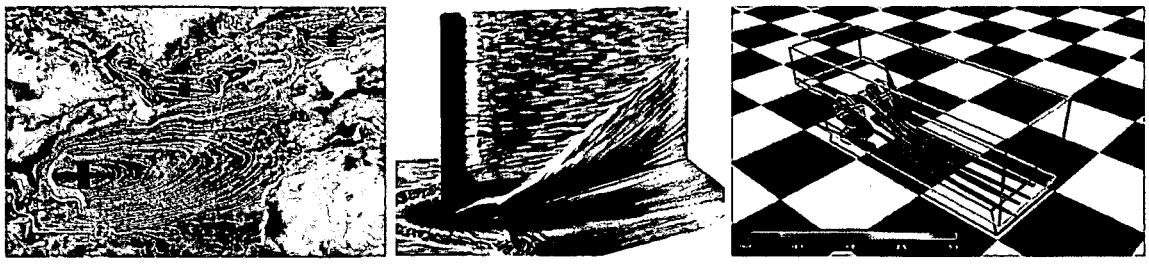


Figure 2.5: Examples of feature-based flow visualization – vortex visualization in Atlantic ocean with ellipses indicating vortices [164] (left), skin-friction on a blunt fin from a flow simulation at Mach 5, visualized with spot noise [121] (middle), and a visualization with streamtubes of the recirculation in the backward-facing-step data set [207] (right).

- Roth and Peikert have also developed a method for finding core lines using higher-order derivatives, making it possible to find strongly curved or bent vortices [161]. Peikert and Roth have also introduced a new operator, the *parallel vectors* operator [145], with which they are able to mathematically describe a number of previously developed methods under one common denominator. Using this operator they can describe methods based on zero curvature, ridge and valley lines, extrema lines, and more.
- Sadarjoen and Post presented two methods for extracting vortices in 2D fields, which are based on geometric criteria [163], as opposed to all the methods discussed above, which use a local criterion for determining a point-to-point basis where the vortices are located. The first is the *curvature center method*, and the second one is the so-called *winding-angle method*, which has been inspired by the work of Portela on classification of vortices [149]. The methods detect vortices by selecting and clustering looping streamlines. Sadarjoen and Post also apply their methods to steady as well as unsteady flows from different application areas, including, for example, flow currents in the Atlantic sea [164] (see also figure 2.5, left side), and a case study of applications in hydrodynamic flows [166]. Pagendarm et al. extended the 2D vortex detection methods of Sadarjoen and Post to 3D flows [142].

Many more methods, as well as detailed descriptions of the here mentioned methods for vortex and vortex core detection are available in two state-of-the-art reports [151, 150].

Shock Wave Extraction: Shock waves are also important features in flow data sets, and can occur, for example, in flows around aircrafts. Shock waves can increase drag and cause structural failure, and therefore are important phenomena to study. Shock waves are characterized by discontinuities in physical flow quantities, such as pressure, density, and velocity. Therefore, shock detection is comparable to edge detection, and similar principles can be used as in image processing. However, in numerical simulations, the discontinuities are often smeared over several grid points, due to the limited resolution of the grid.

Ma et al. have investigated a number of techniques for detecting and for visualizing shock waves [130]. Detecting shocks in two dimensions has extensively investigated [117, 135, 158]. However, these techniques are in general not applicable to shocks in three dimensions. They also describe a number of approaches for visualizing shock waves. The approach of Haimes and Darmofal [65] is to create isosurfaces of the Mach number normal to the shock, using

a combined density gradient/Mach number computation. Van Rosendale presents a two-dimensional shock-fitting algorithm for unstructured grids [158]. The idea relies on the comparison of density gradients between grid nodes.

Ma et al. compare a number of algorithms for shock extraction, including those by Lovely and Haimes [129] and Pagendarm and Seitz [143], and also present their own technique [130].

Separation and Attachment Line Extraction: Other features in flow data sets are separation and attachment lines (surfaces) on the boundaries of bodies in the flow. These are the lines (surfaces) where the flow abruptly moves away from or returns to the surface of the body. These are important features in aerodynamic design because they can cause increased drag and reduced lift [159], and therefore, their occurrence should be prevented or at least minimized.

Kenwright gives an overview of existing techniques for detecting and visualizing separation and attachment lines [95]. Some common approaches are:

- Particle seeding and computation of integral curves, such as streamlines and streaklines, which are constrained to the surface of the body. These curves merge along separation lines.
- Skin-friction lines can be used, analogous to surface oil flow techniques from wind tunnel experiments, as presented by Pagendarm and Walter [144].
- De Leeuw et al. [121] propose the use of texture synthesis techniques to create continuous flow patterns rather than discrete lines (see also figure 2.5, middle).
- Helman and Hesselink can automatically generate separation and attachment lines from their vector field topology [74, 75]. These lines are generated by integrating curves from the saddle and node type critical points in the direction of the real eigenvector. However, only closed separations are found, i.e., the curves start and end at critical points.
- Globus et al. [59] also designed and implemented a system for analyzing and visualizing the topology of flow fields, which is also able to visualize attachment and detachment surfaces and vortex cores.

Kenwright also presented a new automatic feature detection technique for locating separation and attachment lines, which is based on concepts from 2D phase plane analysis [95]. This technique does detect both, open and closed separation lines, in contrast to the above summarized methods based on flow topology analysis, which can only detect closed separations, as open separation does not require separation lines to start or end at critical points. Kenwright et al. [98] also present a second algorithm in addition to the one mentioned above. This second algorithm is based on the parallel vector method, as presented by Peikert and Roth [145].

Other Types of Features: There are other types of features, such as recirculation zones and boundary layers, for example. Work in extracting these features has been presented by Haimes [64] and by Sadarjoen and Post [162] among others. Van Walsum et al. [207] also worked on the visualization of different types of features, an example visualization is shown in figure 2.5 on the right side.

Selective Visualization:

A generic approach to feature extraction is *Selective Visualization*, which is described in several works by van Walsum et al. [206, 207]. According to this approach, the feature extraction

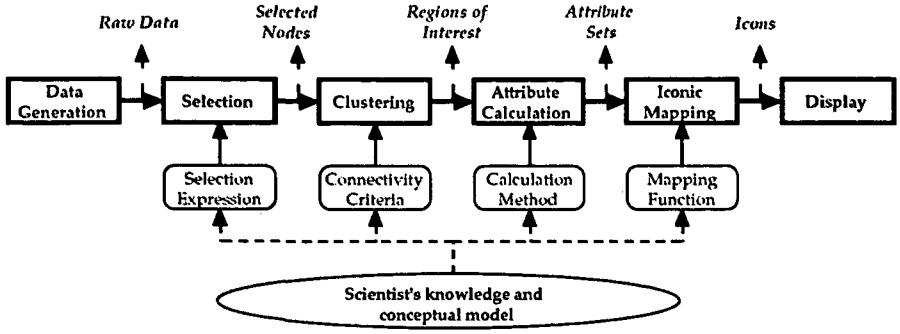


Figure 2.6: The feature extraction pipeline according to Reinders et al. [157].

process is divided into four steps (see also figure 2.6): *segmentation*, *clustering*, *attribute calculation*, and *iconic mapping*. More details on each of these four steps are described in several other publications by Post et al. [151, 150].

Feature Tracking and Event Detection:

In time-dependent data sets, features are objects that usually evolve over time. Determining the correspondence between features in successive time steps, which actually represent the same object but at different time steps, is called the *correspondence problem*. Feature tracking is involved with solving the correspondence problem. The goal of feature tracking is to be able to describe the evolution of features through time. During the evolution of features, certain *events* can occur, such as the interaction of two or more features, or significant shape changes of features. Event detection is the process of detecting such events, in order to describe the evolution of features more accurately.

There are a number of approaches to solving the correspondence problem. Features can be extracted **directly from the spatio-temporal domain**, thereby implicitly solving the correspondence problem. Or, when features extraction is done in separate time steps, the correspondence can be solved **based on region correspondence**, or **based on attribute correspondence**.

Feature Extraction from the Spatio-Temporal Domain:

It is possible to perform feature extraction in 3D or 4D space-time. Tricoche et al. presented an algorithm for tracking of two-dimensional vector field topologies by interpolation in 3D space-time [199]. Bajaj et al. presented a general technique for hypervolume visualization [8]. They describe an algorithm to visualize arbitrary n-dimensional scalar fields, possibly with one or more temporal dimensions. Weigle and Banks extracted features by isosurfacing in four-dimensional space-time [209]. This is conceptually similar to finding overlapping features in successive time steps, as described in the next paragraph about region correspondence. Bauer and Peikert performed also tracking of features in 4D or 5D scale-space [14]. Thereby the data is represented by the 3 spatial dimensions, one temporal dimension, and in the case of a 5D scale-space, also by a measure along a scale axis, which is given by the standard deviation of a Gaussian kernel, with which the data is smoothed.

Region Correspondence:

Region correspondence involves comparing the spatial regions of interest obtained by feature extraction, and trying to match corresponding regions. Correspondence thereby can be found, for example, by using a minimum distance or maximum cross-correlation criterion [63], or by minimizing an affine transformation matrix [87]. As mentioned above, it is also possible to extract features from the four-dimensional time-dependent data set, where the fourth dimension denotes time [209]. The correspondence is then implicitly determined by spatial overlap between successive time steps. This criterion is simple, but not always correct, as objects can overlap, but not correspond, or correspond, but not overlap. Silver and Wang explicitly use the criterion of spatial overlap instead of creating isosurfaces in four dimensions [180, 181, 182, 183]. They prevent correspondence by accidental overlap, by checking the volume of the corresponding features, and taking the best match. This idea is already similar to the class of attribute correspondence, which is discussed next.

Attribute Correspondence:

With attribute correspondence, the comparison of features from successive frames is performed on the basis of the attributes of the features, such as position, size, volume, and orientation, for example. These attributes of the features can be computed during the feature extraction phase (see above), and then be used for description, visualization, but also for tracking of the features.

Samtaney et al. use the attribute values together with user-provided tolerances to create correspondence criteria [168]. Reinders et al. described an algorithm for feature tracking, that is based on prediction and verification [155, 156, 153]. This algorithm is based on the assumption, that features evolve predictably. That means, if a part of the evolution of a feature has been found, a prediction can be made into the next time step.

Event Detection:

After feature tracking has been performed, event detection is the next step. Events are the temporal counterparts of spatial features in the evolution of features. For example, if the path or evolution of a feature ends, it can be of interest to determine why that happens. Different possibilities include that the feature shrinks and vanishes, or that the feature moves to the spatial boundary of the data set and disappears, or that two or more features merge and continue as one, etc. Samtaney et al. introduced the following events: *continuation*, *creation*, *dissipation*, *bifurcation*, and *amalgamation* [168]. Reinders et al. developed a feature tracking system that is able to detect and visualize these and a few further events [156]. They use a different terminology, using *birth* and *death* instead of creation and dissipation, and *split* and *merge* for bifurcation and amalgamation. Furthermore, they can detect *entry* and *exit* events, where a feature moves beyond the boundary of a data set. Finally, for a specific, graph-type feature, the system is able to detect changes in topology, discriminating *loop* and *junction* events.

Visualization of Features and Events:

The final step in the feature extraction pipeline (see also figure 2.6) is the visualization of features. The most straightforward visualization is to show the nodes (or cells) in the data set, that have been selected in the first step of the feature extraction pipeline. Sadarjoen and Post use, for example, crosses at the selected nodes for visualization [165]. Another simple visualization technique is to use isosurfaces. This can be done on the binary data set resulting

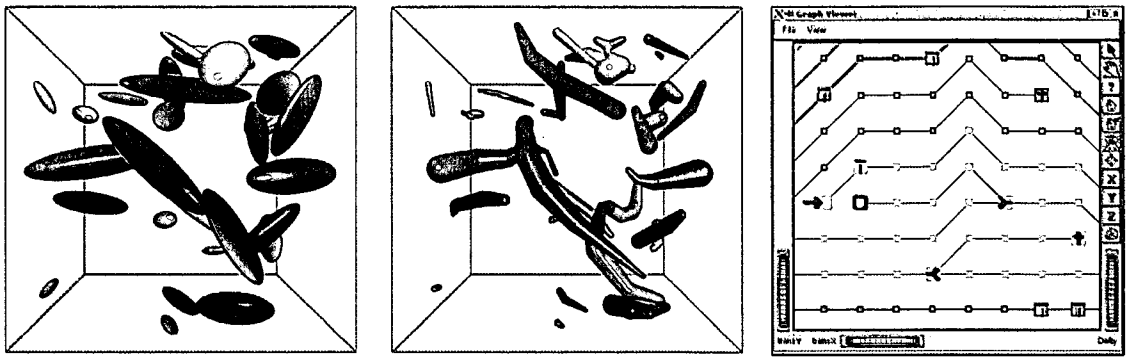


Figure 2.7: Examples for representations of vortical structures: ellipsoid icons (left) and skeleton icons (middle), both from [153]; a sample visualization of events in the graph viewer of Reinders et al. [156] (right).

from the selection step, or, if the selection expression is a simple threshold, directly on the original data set.

If instead of the separate selected points, the attributes are used, that have computed in the feature extraction process, then parametric icons can be used for visualizing the features. For example, if an ellipse fitting of the selected clusters has been computed, there are three attribute vectors: the center position, the axis lengths, and the axis orientations, which can be mapped onto the parameters of an ellipsoid icon (see figure 2.5, left side, for ellipsoid icons representing vortices in a 2D ocean data set [164]). However, for some applications with strongly curved, long, and thin features, for example, the ellipsoids do not give a good indication of the shape of features. Therefore, Reinders et al. presented the use of skeleton graph descriptions for features, with which they can create icons that accurately describe the topology of the features, and approximately describe the shape of the features [154]. See figure 2.7 for a comparison of the use of ellipsoid icons (left) with the use of skeleton icons (middle).

For visualizing the results of feature tracking, it is of course essential to also visualize the time dimension. Reinders developed a player and a graph viewer for showing the temporal evolution of the features, including a visualization for the detected events [156] (see also figure 2.7, right side for an example illustration of the graph viewer. This graph viewer gives an abstract overview of the entire data set, with the time steps on the horizontal axis, and the features represented by nodes, on the vertical axis. The correspondences between features from consecutive frames are represented by edges in the graph, and therefore, the evolution of a feature in time, is represented by a path in the graph.

2.2 Multi-Dimensional Data Visualization

The framework for interactive visual analysis of CFD data, as presented in the following chapters, is very much different from most of the previously described research work on visualization of flow (simulation) data. One such difference is, that we consider not only the flow vector and velocity components in our visualization, but rather the full multi-dimensional domain of the CFD data sets during investigation and analysis processes. We employ *multi-dimensional*

viewing approaches (see section 2.2.1 for related work), *focus+context visualization* (see section 2.2.2), and *linking and brushing* (section 2.2.3) in our setup. In the following, work related to our approach with respect to these three aspects is presented briefly, the concepts as employed in our approach are discussed in the remainder of this thesis.

2.2.1 Multi-Dimensional Data Viewing

In information visualization, one of the basic tasks is to support (interactive) viewing of large, multi-dimensional (or even high-dimensional) data sets. Although no common convention exists about when a data set is considered to be high-dimensional, as compared to "only multi-dimensional", we refer to multi-dimensional visualization when more than the usual 3 spatial + 1 temporal (+ optionally 1 additional scalar valued) dimensions are depicted concurrently. In our eyes, the term high-dimensional data is to be used for data, which consists not only of a couple of dozens of dimensions (as is the case of CFD simulation data, for example), but rather a few hundreds or even many thousands of dimensions. For these high-dimensional data sets, special techniques have to be applied, which are not in the current interest of our research. We are interested in visualization techniques which enable the (simultaneous) viewing of multiple data dimensions, as resulting from flow simulation data.

A few of the already discussed flow visualization techniques are also able to view multiple dimensions (or at least incorporate multiple dimensions into the visualizations). Examples include the works of Kirby et al. [99], who visualized multivalued data from 2D incompressible flows using overlaying concepts from painting, as well as several dense, texture-oriented FlowVis methods, which use opacity maps determined by an additional scalar value, for example.

However, for our works we are more interested in standard viewing techniques for multi-dimensional data sets. One of the oldest, simple, and very useful multi-dimensional viewing tools is the so-called *scatterplot*, which is used to plot data items according to two data dimensions (in the case of 2D scatterplots) in an orthogonal viewing space (see also the next chapters for multiple examples of scatterplots). The mapping of two arbitrary data dimensions to the two axes of the viewing plane can be usually chosen interactively. This allows to compare distributions of different data attributes in a step-by-step manner.

Although scatterplots are an already very old technique coming from the field of statistics (correlations of data items are very well visible in scatterplots), actual applications of scatterplots in the visualization community are rarely published. However, a few recently published examples [61, 60] show how useful they can be in interactive data visualization for multi-dimensional data.

A popular extension of simple 2D scatterplots is the so-called *scatterplot matrix*, where multiple adjacent scatterplots are displayed in one image [27]. Another, natural extension of scatterplots in 2D are 3D scatterplots. Multiple approaches of how to solve the perceptual difficulties of 3D viewing have been proposed, including the works of Becker [15], Kosara et al. [108], Reina and Ertl [152], and Piringer et al. [148]. Some of these works also combine multiple 2D and 3D scatterplot views through linking (see below).

Further related work to 2D scatterplots include, for example, the work on *linked derived spaces* [76] (see below for discussion) and *prosection views* [52].

A different approach for visualizing unstructured multi-dimensional data is the use of glyphs. One of the best known methods for glyph visualization are the so-called *Chernoff-*

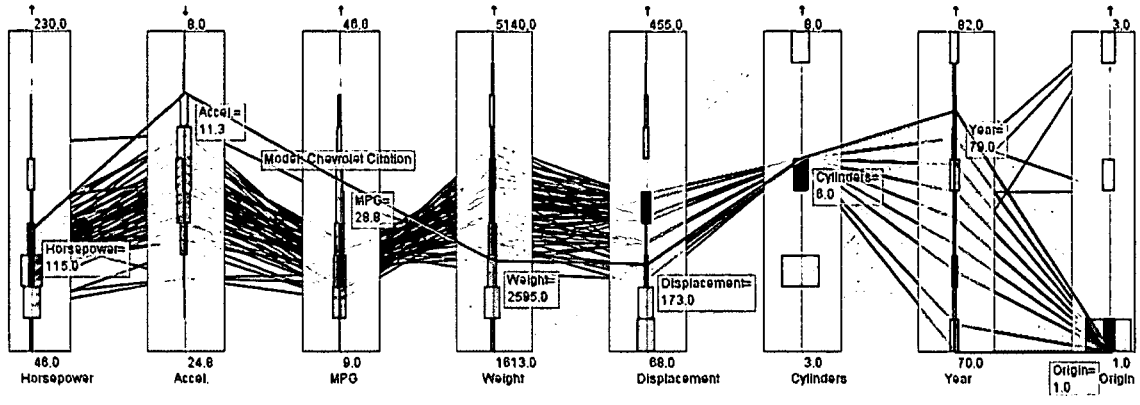


Figure 2.8: *Extended Parallel Coordinates* [68]: a sample view of a multi-dimensional data set about cars [204]: cars with six cylinders were emphasized through brushing, histograms are laid over axes, and one data point is highlighted, showing all details.

Faces [26], where different data dimensions are mapped to different features of a face to build a high-dimensional representation of the data. Other examples of glyphs, which have been proposed in the past include, among many others, *stick figure icons* [147], *star glyphs*, as employed in the XmdvTool [208], *color icons* [92], etc. Also, using three-dimensional shapes to show multi-dimensional data has been presented by Ebert et al. [41].

Another often used visualization approach for visualizing multi-dimensional data are *parallel coordinates* [80, 81, 79]. In this technique multiple (up to a few dozens) of the data attributes available in a multi-dimensional data set, can be viewed simultaneously in one view. For each data dimension shown, one axis is drawn. Thereby the different axes are aligned in parallel to each other. Every data item is represented by one value on each axis (the respective attribute value). These points on the different axes are connected with line segments, thus each data item is represented by a polyline. Parallel coordinates have been extended and used in many areas. Hauser et al. [68] improved on the perceptual problem caused by overdrawing of multiple lines by adding histograms in the form of overlays to the data axes (see figure 2.8 for an example illustration). Hierarchical parallel coordinates have been proposed by Fua et al. [49] for the exploration of large data sets. Higher order parallel coordinates presented by Theisel [195] insert additional "minor" axes between the normal ones.

A further class of approaches uses pixel-oriented techniques to display multi-dimensional data. The basic idea is to map each attribute value to a colored pixel and group the pixels belonging to each dimension of the information space into adjacent areas [89]. Well known examples of how to arrange the pixels on the screen include the *recursive pattern* technique [93], as well as *circle segments* [4]. Also *pixel bar charts* [91] and the extension to *hierarchical pixel bar charts* [90] are related visualization techniques. Those and many more approaches for multi-dimensional data viewing can be found in recent publications. Further references and detailed discussions can be found in an overview by Kosara et al. [105], as well as in a state-of-the-art report on visual data mining by Keim et al. [94], for example.

2.2.2 Focus+Context Visualization

Visualizing huge amounts of data requires advanced methods and techniques which take into account the limited display capacity of the output devices. Focus+context visualization is a well-known approach in information visualization, which tries to enhance the display of large amounts of data: certain data subsets of special interest are emphasized in the depiction, whereas the rest of the data is shown in a reduced style for orientation.

Traditionally, focus+context visualization refers to an uneven distortion of visualization space, assigning more space for the focus, and less space for the context. In computer science this concept has been introduced by Furnas in his work about the *fisheye view* [50].

Since then, many different approaches of how to come up with a focus+context visualization have been presented. According to Hauser [66], there are different possibilities of how to generalize focus+context visualization. Besides conventional approaches, such as, for example, *graphical fisheye views* [169], the *Perspective Wall* [131], the *Hyperbolic Viewer* [111, 110, 137], or the *Magic Eye View* [109], which are based on image space distortion, also the following alternatives of focus+context discrimination in visualizations can be identified:

- using more opacity for visualizations in focus [136, 123]
- using a reduced style for the visualization of the context [71, 70, 31]
- using eye-catching colors for emphasizing the focus [35, 33, 61]
- using a blurred depiction for context data [106, 107]

The interested reader is referred to the works of Leung and Apperly [122], Keim et al. [94], and Hauser [66] for overviews and further details on many focus+context visualization techniques.

2.2.3 Linking and Brushing

Brushing is an interactive selection process, directly based on the visual display of the data. When brushing, the user actively marks a subset of the data set in a view as being of special interest, i.e., in focus, using a brush-like interface element. Brushing therefore is an effective method to actively (and explicitly) discriminate data points in focus from context subsets [216].

In addition to standard brushing [16, 208], several useful extensions to brushing have been proposed. Multiple brushes and composite brushes [132], and the use of non-binary DOI functions for smooth brushing [35] extend the available toolset for interactive DOI specification (the following chapters of this thesis present these in more details). Also, more complex brushes like those designed for hierarchical data [48], or such using wavelets [217] or relative information between different data dimensions [68] have been proposed. Recently, a new brushing concept has been proposed, which models the brushing process and techniques in dynamic data visualization. This method is called compound brushing [25].

Linking is a process, which communicates the selected data from one to other (linked) views [20], usually providing interactive updates of all (linked) views, to show a consistent state of focus and context discrimination. Linking is mostly used in combination with brushing (and vice versa), as the combination of these two concepts yields more possibilities for interactive data viewing (and analysis).

Through the concept of linking, the simultaneous use of *multiple views* on the data is supported. Thereby, multi-dimensional viewing is enabled, if different data dimensions are shown in each linked view. For visualizing multi-dimensional data, the use of multiple (linked)

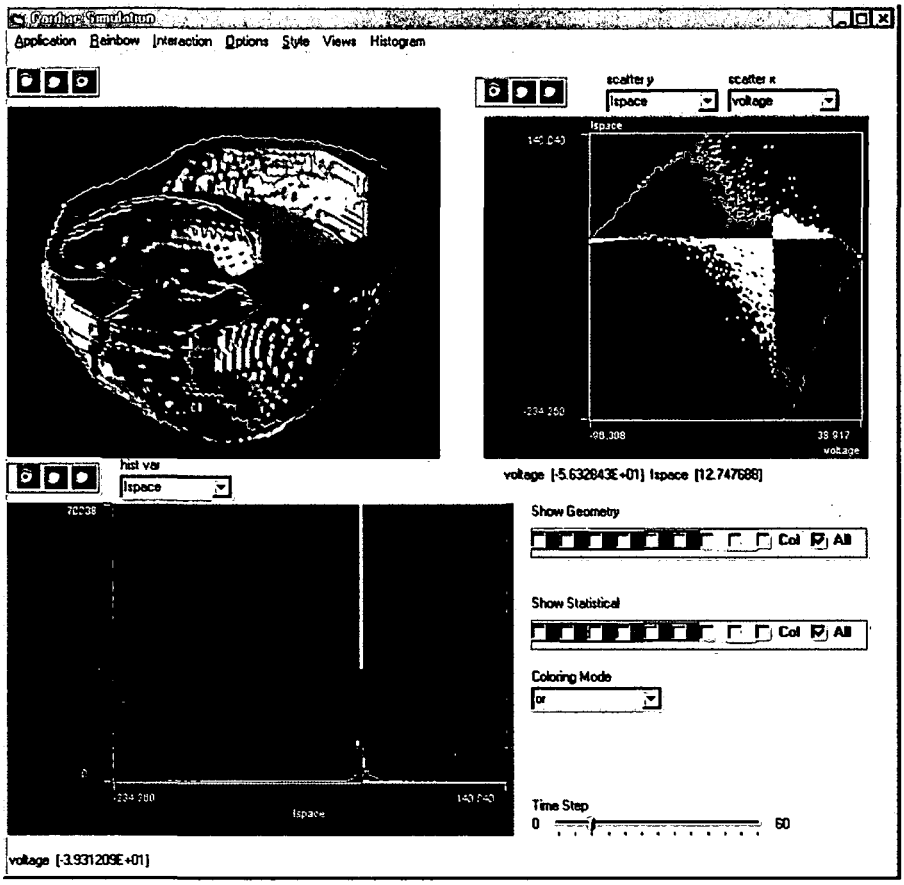


Figure 2.9: Sample screenshot of the WEAVE system during the investigation of multi-dimensional data from the simulation of a (beating) human heart. Multiple, linked views from InfoVis and SciVis are used, including a scatterplot, a histogram, and a 3D SciVis rendering view.

views has become very popular. Even though the principle idea looks rather simple, it is not easy to design a multiple views system being as flexible and efficient as possible. Therefore the publication of very useful guidelines by Baldonado et al. [10] is very valuable.

Further references on linking and brushing, as well as on approaches using the multiple views concept can be found in overviews by Kosara et al. [105] as well as by Keim et al. [94].

2.2.4 Analysis and Visualization of Multi-Dimensional CFD Results

Above we have already discussed many approaches for flow visualization, which mostly are also capable for visualizing results from CFD simulation. What has not been discussed up to here, is work, which is, at least to a certain extent, similar to our approach, in that sense that also multiple views are used to visualize the multi-dimensional character of these CFD data sets.

One approach which was presented by Henze [76] is called *linked derived spaces.* In this approach Henze uses multiple 2D views, called portraits, which have a certain similarity to

scatterplot views, as they also show distributions of different data dimensions. The views, however, are not standard scatterplot views, but rather extended, to show also the connectivity of the different data points plotted in the plane. All views are linked, and through interactive brushing, the user can highlight a part of the data via coloring in all views consistently. Unfortunately this approach only deals with CFD data given in 2D and has not been extended to 3D flow simulation data.

Another approach, which also was the inspiration for our work as presented here, is a system called WEAVE [61], which was presented by Gresh et al. In this approach, which deals with multi-dimensional data from a heart simulation, (discrete) brushing can be performed in InfoVis views, which are linked to a 3D SciVis view through coloring of the selected regions on surfaces. An example screenshot of this system is shown in figure 2.9, where a 3D SciVis view, a scatterplot view, and a histogram view are used for interactively investigating the multi-dimensional data set.

Chapter 3

Linking Scientific Visualization and Information Visualization

Data from CFD simulation comes in multi-variate form: per simulation cell (or vertex of a cell) a number of different flow attributes is given, usually including, for example, the flow vector and velocity, pressure, temperature values, and others. Classic scientific visualization (SciVis) approaches which are employed to visualize CFD data focus on the visualization of data values in the context of their spatial embedding (e.g., [18, 189, and others]). Questions of how to render 3D data distributions meaningful to the 2D screen [146] or of how to encode flow data in a way which communicates the important properties of the data best, are treated in traditional, direct 3D flow visualization [113].

Information visualization (InfoVis), on the other side, deals with the visual presentation of data values without a special emphasis on spatial relations [22]. Scatterplots, parallel coordinates, pixel-based techniques, etc. (there is a lot of related work [105, 152, 81, 79, 89, and many others], see also section 2.2 of this thesis), are used to bring data values themselves in relation to each other. Thereby inter-relations between the different data items are visually displayed and investigated.

As both fields (SciVis and InfoVis) have their specific strengths, a natural extension for benefitting from the strengths of both approaches would be to use a hybrid or combined approach. The work presented throughout the remainder of this thesis originates from this consideration of combining both approaches. The spatial dimensionality of CFD data (usually given in 3D) requires methods which clearly show, **where** in the geometrical context of the data something interesting happens. The high dimensionality of the multi-variate attribute space (including also the temporal component) demands for methods which answer the questions of **what** and **why** it is happening (and **when**).

Following the idea of Donna Gresh and others at the IBM Watson Research Center, who presented their research work on the visualization of complex simulation data called WEAVE [61], we started to develop our own framework for the interactive analysis of large, high-dimensional, and time-dependent simulation data. WEAVE is a system, which combines multiple views from SciVis and InfoVis to allow efficient and fast visual analysis of simulation data of a beating heart (see also section 2.2.4).

The framework presented in this thesis also combines multiple (different) views from SciVis and InfoVis, but extends the previous work in several directions: (1) we allow an **arbitrary number of InfoVis and SciVis views to be linked** (see next sections for linking details),

(2) we include **advanced brushing methods** (sections 3.4 and 3.5), (3) a **flexible feature specification framework** (chapter 4), allowing to interactively define complex features is integrated, (4) and extensions of the technology to **handle time-dependent and large data sets**, as well as to allow the specification of **unsteady features** (chapter 5) are provided.

3.1 Dealing with Occlusion in Scientific Visualization

When rendering truly three-dimensional information in scientific visualization (SciVis), such as, for example, medical data from computer tomography, it is very important to decide *how* to render the data. But even more important, the question of *what* parts of the data should be displayed needs to be addressed [70]. Because of occlusion, not all of the data can be shown concurrently.

In volume rendering, usually a so-called *transfer function* is employed, which assigns an opacity value to each of the data items [146]. A compositing procedure is then used to produce volume visualization results through image synthesis. The design of a transfer function is a difficult challenge, which strongly depends on the specific goals of the visualization process. At the IEEE Visualization conference in 2000, the most recognized approaches have been discussed in an interesting panel [146].

Another approach which recently has become very popular is to use feature-based techniques for visualization of large data sets [151]. Feature-based visualization is characterized by focussing on essential parts of the data, instead of showing all the data in the same detail at the same time. This kind of visualization gains increasing importance due to bigger and bigger data sets which result from computational simulation, so that not all of the data can be shown simultaneously. For feature-based visualization, proper feature extraction methods are essential.

Up to now, feature extraction usually is done (semi-)automatically [150] with little interactive user intervention, often also as a preprocessing step to the actual visualization. But for interactive analysis, the question of what actually is (or is not) considered to be a feature refers back to the user: depending on what parts of the data the user (at an instance of time) is most interested in, features are specified accordingly. Therefore, flexible feature extraction requires efficient means of user interaction to actually specify the features. The work presented in this thesis aims at providing such a flexible framework for feature-based visualization.

A comparison of direct 3D visualization and a feature-based visualization of flow simulation data is shown in figure 3.1. On the left side, a full direct visualization of the data coming from a simulation of a flooding after the break of a dam is shown (see chapter 5 for a discussion of this application case). In this visualization for each data item (representing a cell of the underlying grid) a colored and smoothed point icon (using the ARB point sprite extension of NVidia [140]) is rendered with full opacity for every cell. As can be seen, this naturally leads to occlusion of all inner structures of the flow. In the middle of figure 3.1, direct rendering of all data items for the same data set is applied, using a reduced opacity for each cell. In this case the inner flow structures are not occluded any more, but again not clearly visible, due to lack of contrast information. On the right, a feature showing regions which exhibit flooding through water is shown. This time perception of the inner flow structures is possible, but the rest of the data is disregarded (showing only the flooded area). This example illustrates the challenges, that have to be met when visualizing data in 3D.

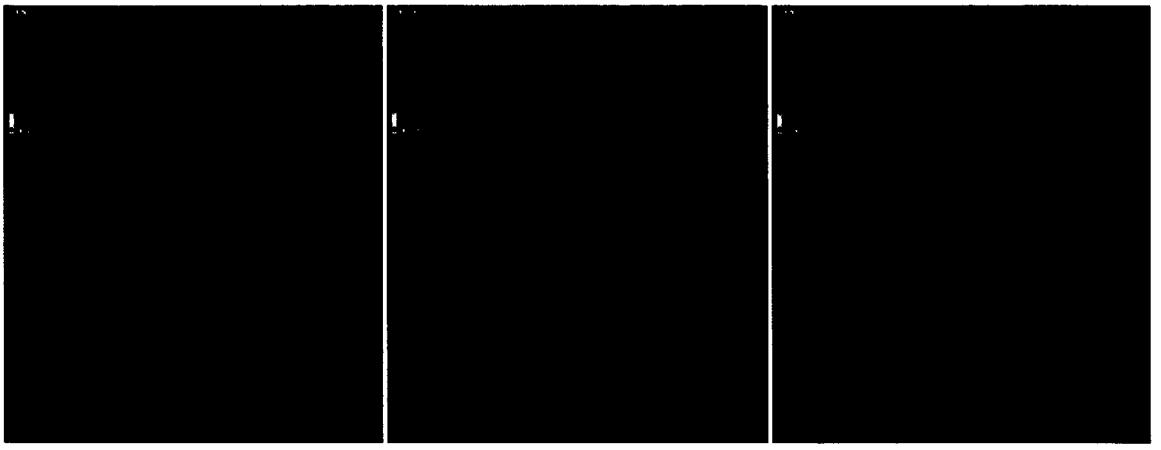


Figure 3.1: Comparing direct and feature-based visualization for data from the simulation of a flooding after the break of a dam (see chapter 5 for a discussion of this application case). *Direct* (left and middle): for each data item (cell of the grid) rendering of smoothed point icons (using the ARB point sprite extension of NVidia [140]) is employed. Thereby every cell is represented in the same style. On the left full opacity is assumed for each data item, leading to occlusion of inner structures. In the middle, transparent rendering is used, leading to unclear perception of flow structures. *Feature-based* (right): the regions which exhibit flooding through water have been selected as a feature.

3.2 Separating Focus and Context in Information Visualization

Apart from scientific visualization as discussed above, information visualization also deals with data sets of large size and increased dimensionality. Again, when dealing with large and high-dimensional data sets, also in InfoVis, the simultaneous display of all the data items usually is impossible. Therefore, *focus-plus-context* (F+C) techniques are often employed to show some of the data in detail, and (at the same time) the rest of the data, at a lower resolution, as a context for orientation [22, 51]. Thereby the user's attention is directed towards the data in focus (e.g., through visual enlargement), whereas the rest of the data is provided as context in reduced style (translucently, for example). This is especially useful when interacting with the data, or when navigating through the visualization.

To discriminate data in focus from context information, often a so-called *degree of interest* (DOI) function can be used [66], assigning an importance measure for each data item. For the definition of the DOI function, several different techniques are described in literature, including implicit techniques (e.g., focus specification through dynamic querying [178]) as well as explicit techniques, such as interactive object selection [70] or brushing [16, 208]. When brushing, the user actively marks a subset of the data directly in a view as being of special interest, i.e., in focus, using a brush-like interface element. Brushing therefore is an effective method to actively (and explicitly) assign values between 0 and 1 to data-points of interest [216]. For a discussion of different methods of brushing, the reader is referred to chapter 2 of this thesis, as well as to an overview by Keim et al. [94]. In our framework, also brushing in InfoVis views is used to specify a DOI function (see next section).

3.3 Linking and Brushing – A Fundamental Concept of the SimVis System

In the following the basic concept of **linking and brushing** as a central part of the SimVis approach is discussed in more detail. Special emphasis is put on the combination through linking views from InfoVis and SciVis.

Linking several views [20] to interactively update all changes of the data analysis process in all views simultaneously is a crucial property for making optimal use of multiple (different) visualization views. **Brushing**, as already mentioned above, is the process in which the user can interactively highlight, select, or delete a subset of elements with regard to visualization by using some appropriate brushing tool. Brushing is an intuitive and very effective, but still very simple concept to indicate user interest. One way to characterize this kind of data classification is to understand it as a shallow, broad-band approach. Data items are marked with respect to data attributes which explicitly are represented in the data, or relatively simple combinations thereof, no complex inter-relations between data attributes are considered as, for example, through sophisticated feature extraction processes where specific aspects of the data are considered which only implicitly are available.

In SimVis, *brushing is associated with linking*, as is often the case in multi-view information visualization systems. Unlike standard multi-view systems known from InfoVis, such as, for example, the XmdvTool [208, 132], XGobi [190, 191], Spotfire [1], IVEE [2], and many more, our approach combines multiple views from both, InfoVis and also SciVis. We consider this approach to be especially effective for the interactive, feature-based visualization and analysis of (flow) simulation data, as it improves the understanding of the data in terms of their high-dimensional character as well as the data relation to the spatial layout.

The result of a brushing operation in any of the InfoVis views linked in the SimVis framework, i.e., a notion of user interest, is reintegrated within the data in the form of a synthetic data attribute $DOI_j \in [0,1]$, in the following called a *degree of interest (DOI) attribution* of the data (compare to Furnas [51]). Thereby the value of 1 represents data "in focus", 0 is used for context information).

In the current version of the framework, scatterplot views (both 2D and 3D scatterplots [148]), different versions of 2D and 3D TimeHistograms [103], and parallel coordinates views are available for InfoVis viewing and interacting. The user can interactively choose, which and also how many of each of these views he or she wants to use during the interactive analysis session. This allows maximal flexibility with respect to high-dimensional viewing of the data sets.

The SciVis views are currently limited to one type of 3D SciVis view, which shows glyphs for each cell center (or node) of the cells of the underlying spatial grid. Again multiple 3D views can be opened, each showing a different feature for example, by linking them to different InfoVis views.

An example setup to illustrate the linking and brushing approach of the framework is shown in figure 3.2. Here one 3D SciVis view (left) and two scatterplot views (middle and right) are used, to view data from a simulation of hot inflow into a T-junction of three pipes (two inlets and one outlet). For more details on this data set see Appendix A of this thesis. In this setup, one brush has been applied to the scatterplot view in the middle. All data items which exhibit high temperature values (horizontal axis) and medium velocity values (vertical axis), have been selected through interactive brushing. The DOI function has assigned the

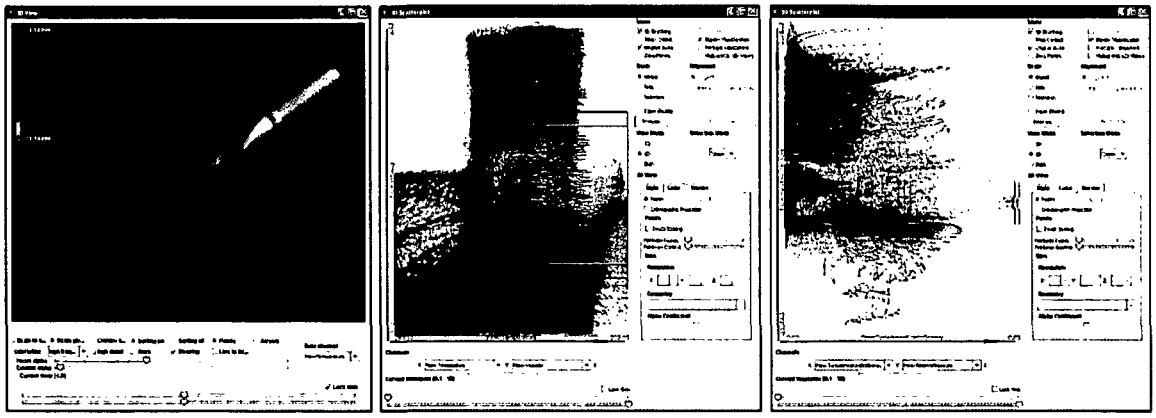


Figure 3.2: An example for the basic concept of *linking and brushing*, as integrated in the SimVis approach: Multiple views from SciVis and InfoVis are used to view different aspects of the multi-dimensional flow data, brushing in a InfoVis view (middle) yields focus+context visualization of the data items in all other linked views.

value of 1 (i.e., maximal interest) to all data items, which are contained in the brushed subset. By linking this view with the other two views, the discrimination of focus from context is also conveyed by these two other views.

In the second scatterplot (right) the data distribution with respect to the attributes of pressure (vertical axis) and turbulent kinetic energy (horizontal axis) is shown. As the current focus has been defined by the brush in the first scatterplot view, it remains unchanged, and thus the distribution of data items belonging to the focus and context subsets of the data set is visible clearly in the second scatterplot as well, yielding additional information about this interactively specified feature (hot, medium velocity flow regions). From this view it can be told, that the brushed regions belong to two clusters according to the pressure values contained, namely exhibiting either high or medium pressure distributions.

The 3D SciVis view on the left shows the spatial geometrical outline of the data, as well as a focus+context visualization of the flow, with the interactively specified flow feature in focus (more opaque, colored, and larger glyphs, representing the individual cells). In our framework a 3D F+C visualization of flow features is realized by displaying parts of the data which are *in focus*, i.e., the flow features, rather opaque and in an enhanced way (larger glyphs, colored), whereas the rest of the data (the context) is shown translucently and diminished (gray-scale and smaller glyphs).

This way of 3D rendering of 3D flow simulation data also could be (in theory) interpreted as an extension of the multi-dimensional transfer functions as recently introduced for volume rendering [102]. These transfer functions are also taking into account additional data attributes such as data gradients and curvature information.

Through this consistent highlighting of features across the different data dimensions (shown in different linked views), it becomes relatively easy to visually gain insight and comprehend the complex simulation data.

As for the focus+context visualization in the 3D SciVis views, there are two further extensions which enhance the effectiveness of the visualization. One option is to use different types of glyphs, representing each data item (cell) of the simulated data. We currently use either (more or less) transparent balls (see figure 3.3, left side for an example), or three-dimensional

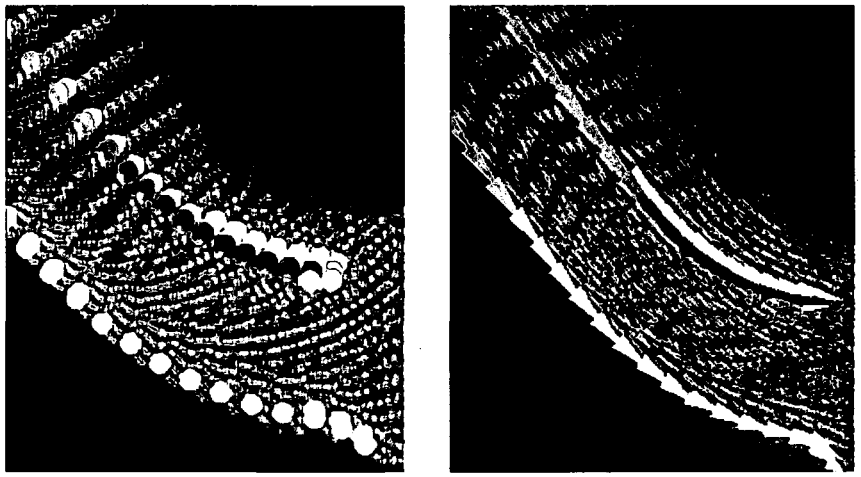


Figure 3.3: Illustrating different glyph styles for representing the cells in the 3D focus+context visualization: 3D balls (left) vs. 3D arrows (right).

arrow glyphs can be rendered for each data item (figure 3.3, right side). The second option allows to flexibly map a data attribute via coloring the glyphs according to the values of this data attribute for each data item for the focus parts of the visualization. In figure 3.2 the colors in the 3D F+C visualization indicate temperature values of the respective feature shown, for example. This allows to integrate an additional dimension into the visualization.

3.4 Smooth Brushing

After the presentation of the basic linking and brushing concept as it is included in our SimVis framework, this section now describes a first extension to the standard binary brushing approach, namely *smooth brushing*. Parts of this section have been published in an extended form at the *WSCG Conference in 2002 in Plzen, Czech Republic* [35].

In most applications of visualization (with a few exceptions [125]) there are only binary or discrete classifications used to establish a semantic layer on top of the originally unlabeled data. Especially in the medical visualization field, for example in object segmentation, usually only sharply bounded objects are separated. In contrast to this, flow simulation data in general often exhibits a rather smooth distribution of data values along spatial dimensions (and also with respect to time).

To cope with the special characteristic of smoothly distributed data in flow simulation results, we use a new brushing mechanism, called **smooth brushing**, which results in a DOI function that continuously maps to the $[0, 1]$ range. The hereby derived DOI function also can be interpreted as a fuzzy set describing a "degree of being *in focus*" [222]. For scientific visualization, the DOI function, as specified in an InfoVis view, corresponds to modulated opacity values in 3D rendering (through linking).

Although smooth brushing is a fairly straight-forward extension of simple, discrete brushing, we do not see a lot of similar approaches in the visualization domain. Linking-and-brushing approaches usually also build on a sharp delimitation of selected data items versus all the rest. Traditional focus+context visualization, however, often builds on the concept of a gradual change between areas in focus and the context [51, 169, 88, and others].

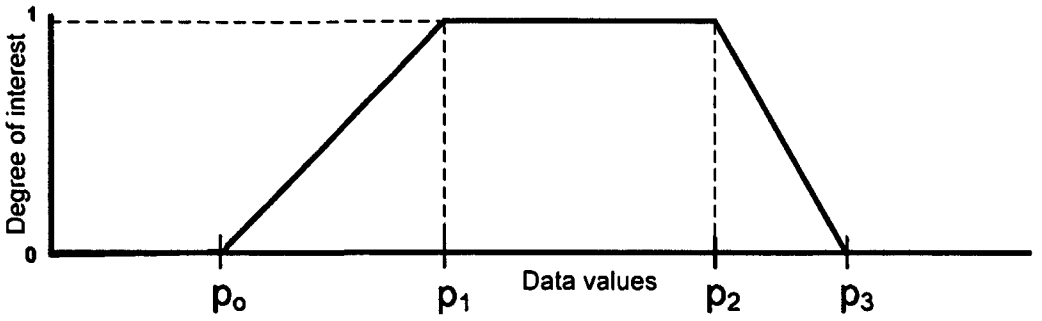


Figure 3.4: A 1D smooth brush results in a trapezoidal DOI function.

3.4.1 Specifying a Smooth Brush

A 1D smooth brush results in a trapezoidal DOI function as defined below and illustrated in figure 3.4.

$$DOI_j(d_i, p_0, p_1, p_2, p_3) = \begin{cases} 0 & \text{if } d_i \leq p_0 \\ \frac{d_i - p_0}{p_1 - p_0} & \text{if } p_0 < d_i \leq p_1 \\ 1 & \text{if } p_1 < d_i \leq p_2 \\ 1 - \frac{d_i - p_2}{p_3 - p_2} & \text{if } p_2 < d_i \leq p_3 \\ 0 & \text{if } d_i > p_3 \end{cases} \tag{3.1}$$

As obvious from this equation, and the respective figure, a one-dimensional smooth brush is determined by two separate regions. The inner region of the brush (100% interest) is defined by the interval $[p_1, p_2]$. The outer region of a smooth brush is divided into two regions, $]p_0, p_1[$ and $]p_2, p_3[$, respectively. These regions represent a gradually increasing/decreasing interest. To enable a flexible DOI specification through brushing, the two border regions of the smooth brush (and thereby the slope of the trapezoidal DOI function) can be chosen independently of each other, allowing a differentiation of neighboring (smoothly increasing/decreasing) interest influence on the focus+context visualization.

When using a smooth brush interactively in a (2D) scatterplot view, the concept of the 1D smooth brush is extended to support the specification of smooth brushes of higher dimensionality. The different DOI values resulting from each 1D DOI function (defined by brushing the values of a single data attribute) are logically AND-combined. This is realized by using concepts from fuzzy logic technology (T-norms and T-conorms, see chapter 4 for more details on the combination of fractional DOI values from multiple brushes).

Figure 3.5 shows an example of a 2D smooth brush applied to a scatterplot view (right side) and the resulting focus+context visualization in a linked 3D SciVis view. The data shown in this figure comes from the 3D simulation of flow through a catalytic converter. The left view shows a scientific visualization with small 3D arrows representing data items. Velocity is mapped to color, with red corresponding to high velocity. The right view is a scatterplot view where velocity values (vertical axis) are plotted against values of static pressure (horizontal axis). A non-binary DOI function was defined by smooth brushing a cluster of data items exhibiting high velocity and rather high pressure. The two rectangles in the scatterplot denote the inner and outer regions of the smooth brush.

The DOI function of this example is used to modulate the opacity as well as the color of the 3D arrows in the rendering view – display elements not in focus are drawn in a gray-scale

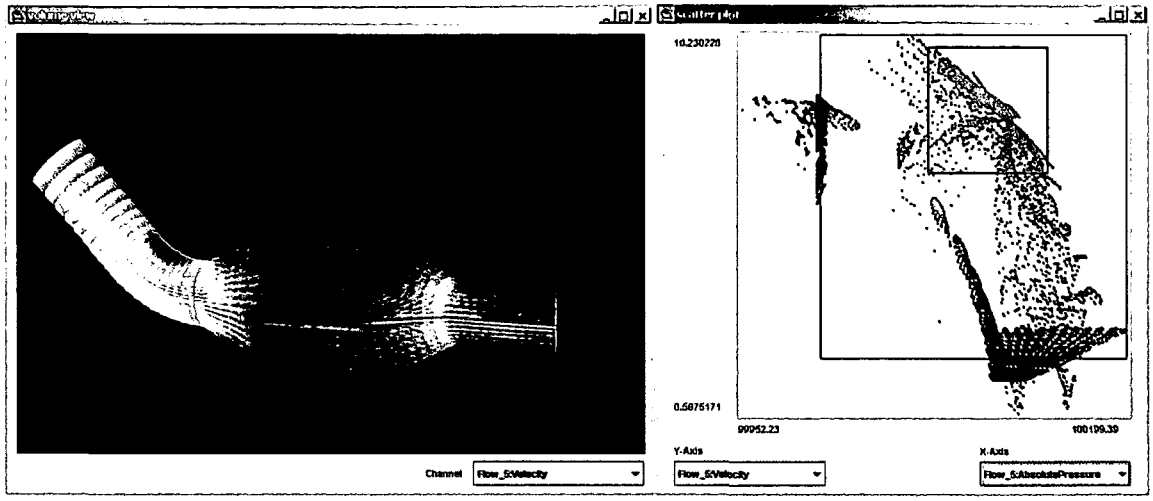


Figure 3.5: *Smooth Brushing and Linking:* simulation data of a catalytic converter is shown. Right side: a scatterplot, where a non-binary DOI function was defined by smooth brushing a cluster of high velocity/high pressure data; Left side: a 3D SciVis view, employing opacity/color modulation for the 3D arrow glyphs.

fashion. One can see that mainly the inlet of the catalytic converter (upper left part in the 3D view) exhibits values of high pressure and high velocity. Through the smooth brush parts of the outlet are also of partial interest, since values there are not much different to the core focus of this visualization, and thus are also partially influenced by the F+C mapping.

In the scatterplot view, the boundary region of non-zero DOI values is visualized by a second rectangle enclosing the 100% brushing region. This second rectangle can also be resized interactively, providing the possibility to separately define the region of interpolation of DOI values between 0 (context) and 1 (full focus) in each of the four directions. The induced DOI value is reflected in the color intensity of the points which represent the data items in the scatterplot.

In addition to the modulation of opacity and color of the 3D glyphs (balls or arrows), as visible in figure 3.5, left side, also the size of the glyphs is modulated according to the DOI value. This additionally emphasizes the subsets of the data in focus.

Besides two-dimensional binary (see figure 3.6(a)) and smooth (figure 3.5, right) brushing, also the brushing of multiple regions in scatterplots is possible in SimVis. Figure 3.6(b) shows an illustration, where two regions of interest are specified through interactive brushing. This allows to put distinct data clusters or subsets of the data into focus simultaneously. In chapter 4 extensions, which also allow to include other logical combinations, are discussed in more detail. There it is also shown, that the combination of multiple smooth brushes is possible in SimVis, too, but requires special care with respect to a reasonable combination of fractional DOI values.

3.4.2 Further Smooth Brushing Results

In figure 3.7 two joining flows (from the left and from above) in a T-junction are shown. Figure 3.7 (b) shows the scatterplot corresponding to figure 3.7 (a). A low velocity/low

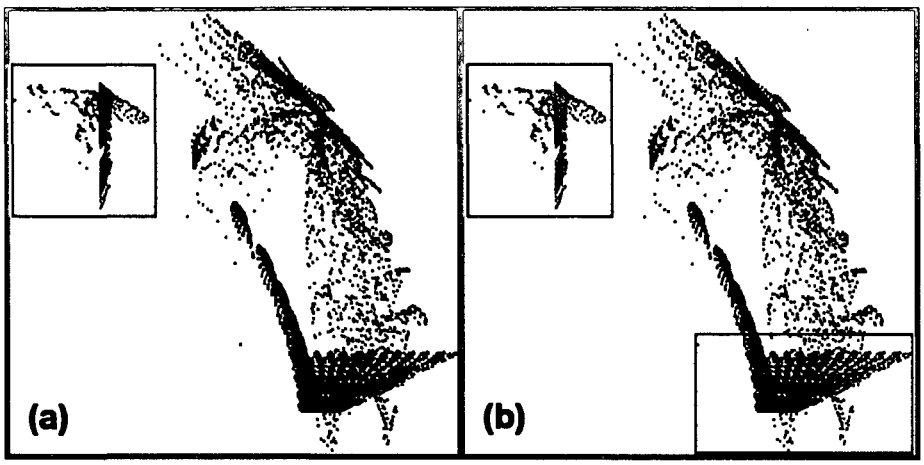


Figure 3.6: A scatterplot brushed once (a) and twice (b) to specify features of different complexity (an illustration).

pressure subset has been brushed (smoothly), to focus on the recirculation area of the mixing flows. In the lower two images the same views are shown, but with different attribute-to-axis mappings: The y-axis of the scatterplot now maps to "turbulent kinetic energy", and the color mapping of the transfer function used for focus visualization is determined by the same attribute. Note the persistence of the DOI function as the result of the last brushing step. After re-mapping data attributes to different axes available in the views, the focus is still the same, and thereby relations to other data attributes become visible.

Figure 3.8 presents another example, i.e., the simulation of the ventilation system in a car. This time, only a 2D slice of cells from the front to the back of the car is considered during analysis. The inlet of the ventilation system is close to the red region on the left side, where maximum velocity is observed. The data has been brushed in a histogram of velocity values, shown on the right side. Relatively high velocity values have been brushed (smoothly), to specify the DOI function. Again the transfer function of the features in the 3D view is defined to color highest velocity values red and relatively small velocity values green (which are not in focus and thus hardly visible here).

Results from a third example are shown in figure 3.9. Here the difference between standard (binary) brushing and smooth brushing for smoothly distributed flow simulation data is illustrated. Cloud formation in the simulated hurricane Isabel (which struck the US eastern coast in 2003) is brushed in a scatterplot view. In the upper row of this figure, a binary brush has been specified in the scatterplot, selecting regions with clouds (horizontal axis) and low pressure (vertical axis). In the accompanying 3D view, the cloud formation in the center of the hurricane, and parts of two fronts are visible as the selected feature. In the lower row of figure 3.9 the same brush has been extended into a smooth brush, selecting lower cloud and higher pressure values. The accompanying 3D view shows, that there are larger regions with cloud formation and that the fronts, identified in the step before, are connected to the center of the hurricane.

Further results and video-sequences (especially about smooth brushes) are available at *http://www.VRVis.at/vis/research/smooth-brush/*. For the simulated hurricane example, many more results (including also several video-sequences) are available at

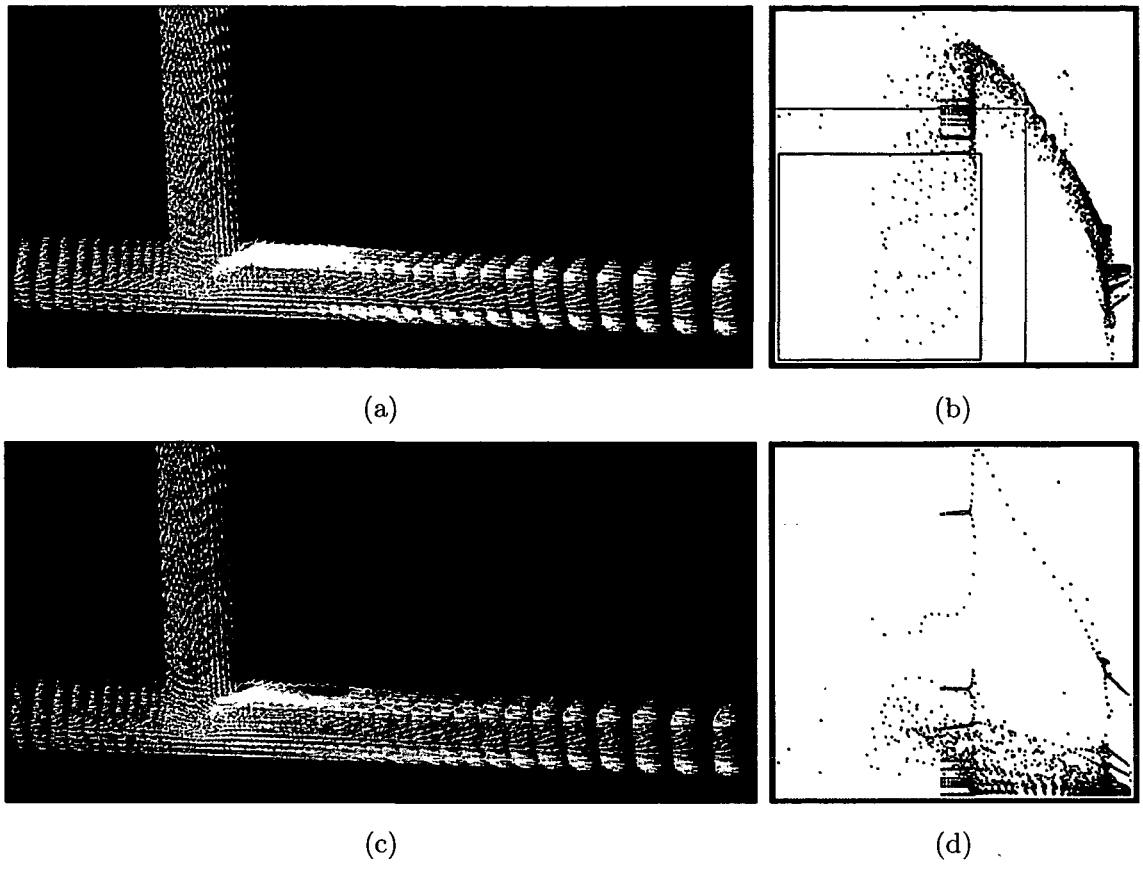


Figure 3.7: Smooth brushing of a slow/low pressure subset (the recirculation area) in a T-junction (a) and (b); changing the attribute-to-axis mapping and attribute-to-transfer function mapping to turbulent kinetic energy on the previously brushed information (c) and (d)

http://www.VRVis.at/SimVis/Isabel/.

3.5 Angular Brushing

As already discussed above, brushing effectively enables the user to (explicitly) specify his or her focus. From section 3.3 we also know, that in the SimVis system multiple different InfoVis views can be used to view the high-dimensional data items, and to interactively specify features through brushing for focus+context visualization.

In parallel coordinates [81, 79, 49, etc.] brushing is usually accomplished by marking a certain data subset of interest according to values within a single data dimension. For an example of how a brush can be used to selectively highlight data items in a parallel coordinates view see figure 3.10.

In our application of visually investigating and analyzing multi-dimensional CFD data sets, we found standard brushing of parallel coordinates very useful, however, we needed to go further. The requirement for some cases was to specify a focus not only in dependence

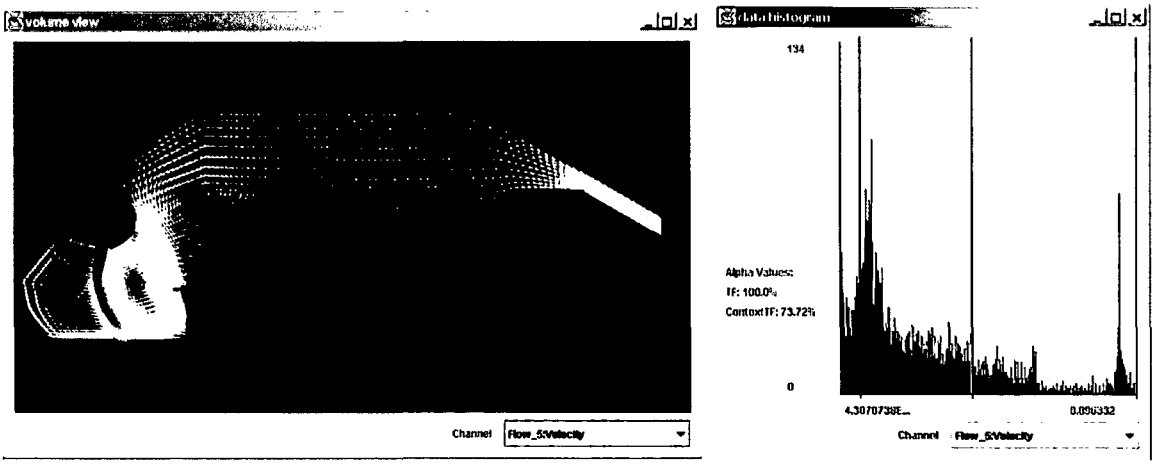


Figure 3.8: Smooth brushing in a histogram (right) and linking to the 3D view (left): a 2D slice of the simulation of a car ventilation system, fast flow in focus.

of one data attribute, but to do so with respect to at least two of them – like brushing all those data items which feature a z-velocity larger than the respective x-velocity, for example. The consequence was to introduce a new brushing technique which we call **angular brushing** [68]. This new brushing extension was presented at the *IEEE Symposium on Information Visualization 2002.*

One strength of parallel coordinates is its effectiveness in 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 respect to two of the data dimensions. This way, it is not only possible to see whether data items exhibit high or low values in single dimensions, but also their relation can be understood. This can be seen by investigating the slope of lines in-between two axes of interest. Positive and negative correlations are visible, the only assumption, that has to be made is, that the user must have set up a proper mapping of data values to the axes.

This feature of parallel coordinates (of conveying also relative information) has been exploited, when extending standard brushing to angular brushing. In addition to standard brushing, which primarily acts on one of the (parallel) axes, we enable the space in-between axes for brushing interactions, as well. The user can interactively specify a subset of slopes which then yields all those data items to be marked as part of the current focus, which exhibit a matching correlation between the brushed axes. See figure 3.11 for an example, where all negative slopes have been marked in-between the second and the third axis.

Angular brushing as described above is a useful extension to standard brushing for 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 of logical operators (see also chapter 4 of this thesis), angular brushes allow the selection of rational properties with respect to two of the data dimensions. When compared to composite brushes in the context of a scatterplot, we clearly see that brushes based on logical operators select subsets which are aligned with

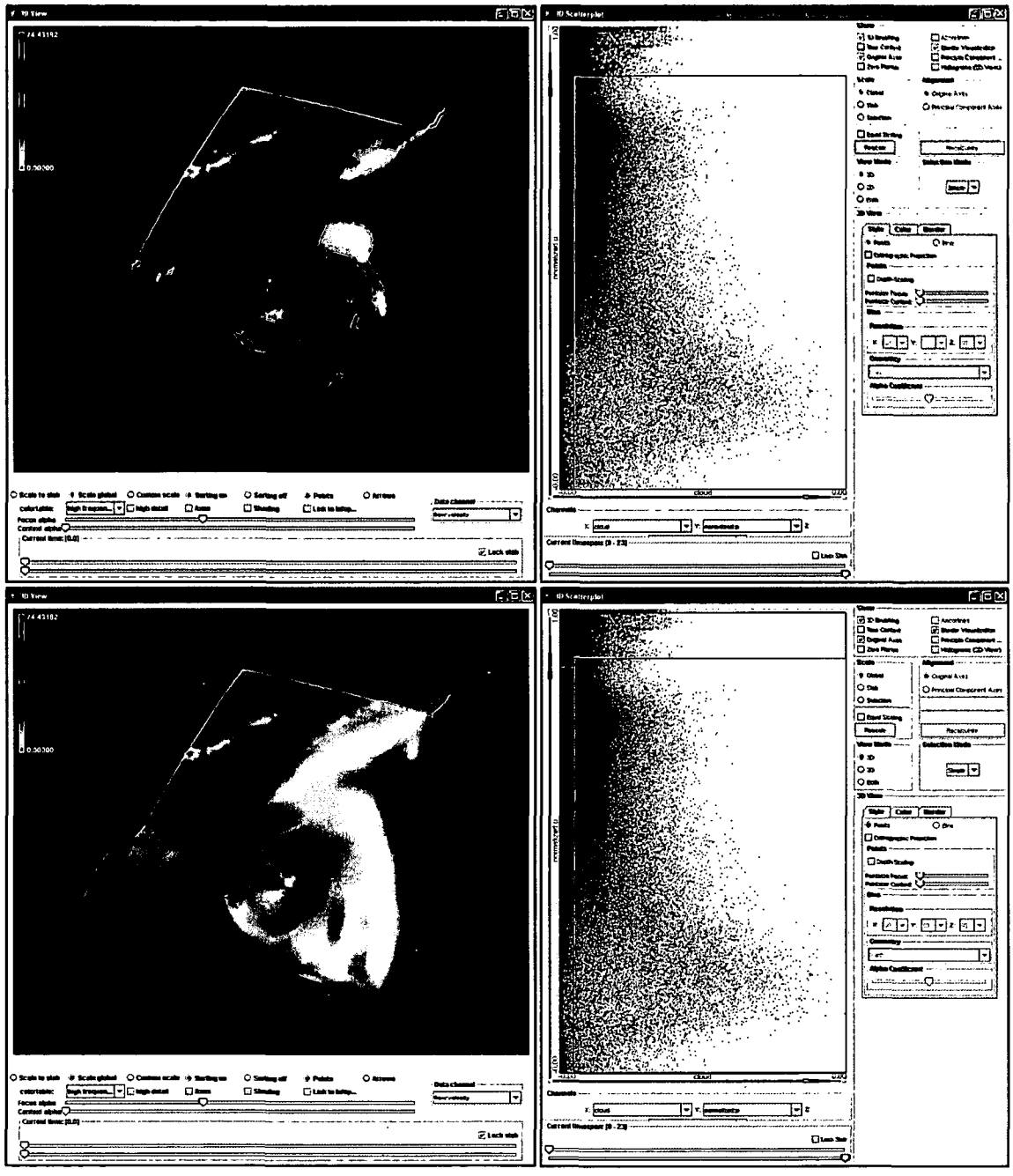


Figure 3.9: Comparing the effect of standard (binary) brushing and smooth brushing for smoothly distributed flow data. Cloud formation for the simulated hurricane Isabel (which struck the US eastern coast in 2003) is brushed in a scatterplot view. Upper row: binary brush, selecting regions with clouds and low pressure values. Lower row: the same brush extended into a smooth brush, selecting all but no clouds and also higher pressure values.

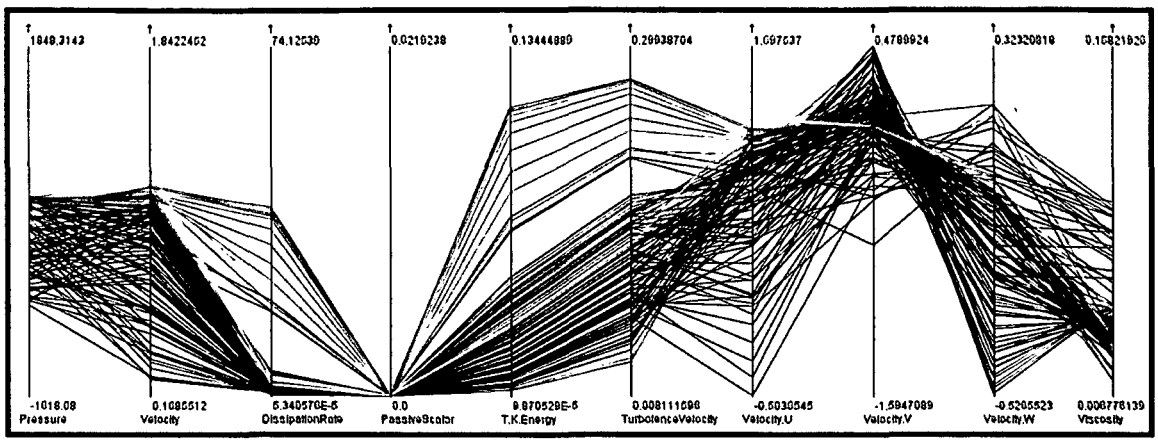


Figure 3.10: The data attributes (not all dimensions shown) of the T-junction example from figure 3.7 is shown in parallel coordinates. The data items, as brushed in the scatterplot view in figure 3.7(b) are highlighted.

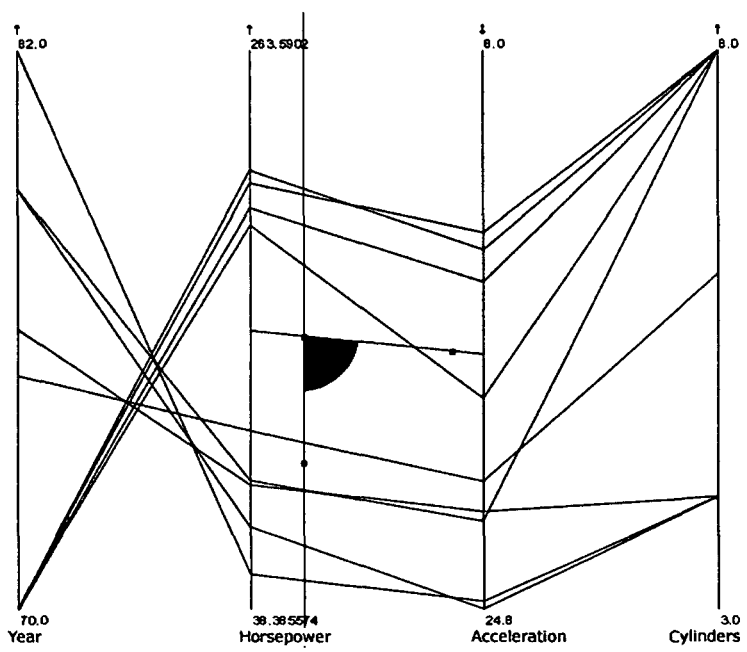


Figure 3.11: Reading between the lines: whereas most line-segments go up in-between the second and the third axis (visualizing a positive correlation of data values there), just a few go down – those have been emphasized through angular brushing.

the display axes, whereas angular brushes select subsets which are aligned with the diagonals when visualized in a scatterplot (see also figure 3.12 for a comparison).

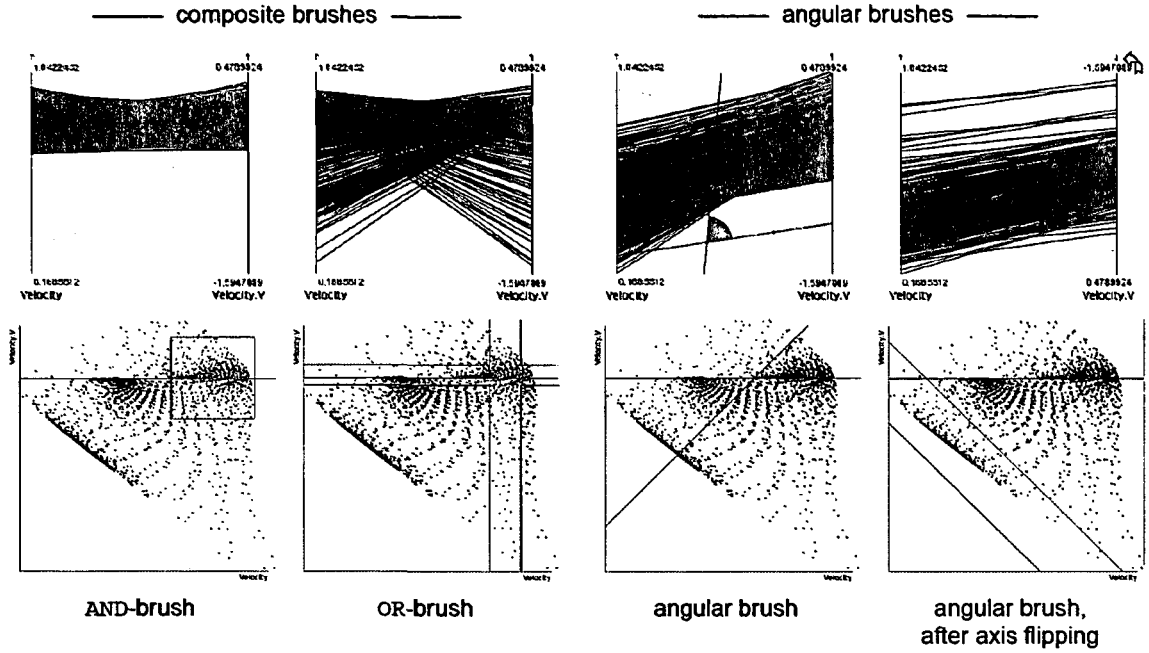


Figure 3.12: Comparing composite brushes (on the left) and angular brushes (on the right), using parallel coordinates (upper row) and scatterplots (lower row): composite brushes address "Horizontal/vertical" features, whereas angular brushes emphasize "diagonal" features.

Chapter 4

The Feature Definition Framework

When using feature-based visualization, one of the most difficult questions is how to extract or specify the features. Up to now, feature extraction usually is done (semi-)automatically [150, 151] with little interactive user intervention, often also as a preprocessing step to visualization. In interactive data analysis, the question of what actually is considered to be a feature generally refers back to the user: depending on what parts of the data the user (at an instance of time) is most interested in, features are specified accordingly. Therefore, flexible feature extraction requires efficient means of user interaction to actually specify the features, and therefore tools supporting such an interactive feature specification are needed.

In the SimVis framework (smooth) brushing of data values in InfoVis views is employed to interactively and graphically separate focus and context. But when analyzing simulation data, one very often encountered problem is the limited flexibility of current brushing and interaction techniques. Brushing is usually restricted to simple combinations of individual brushes, as well as missing support of high-dimensional brushes due to the tight coupling of GUI interactions and the representation of the brush data itself. For fast and flexible analysis of the usually large and high-dimensional simulation data, complex and also high-dimensional brushes are necessary. In this chapter, we present a formal framework, that is very closely coupled to the data, allowing to define and handle such brushes interactively (see section 4.1). Parts of this chapter have been published in an extended form at the 5^{th} Joint IEEE TCVG – EUROGRAPHICS Symposium on Visualization (VisSym 2003) [33]. After the presentation of the formal feature specification framework, advanced interaction techniques for specifying features (section 4.2) are discussed, and a few application examples are shown in section 4.3. Finally, a short overview about implementation details is given and a brief discussion of the feature definition framework concludes this chapter.

4.1 Using a Feature Definition Language

When dealing with results from computational simulation, usually very large and high-dimensional data sets are investigated. Previous work already showed, that interactive specification of features with tight reference to the actual data attributes is very valuable for visualization of such data sets [61, 35]. For a fast and flexible analysis of these results, powerful and intuitive tools are needed – the here described approach provides flexibility in terms (a) of multiple options to differently view the data, and (b) a wide range of user interactions to construct and adapt feature specifications. Whereas previous work mainly focussed on

viewing, we mainly improve on interaction in this chapter.

To generalize the specification of features (enabling feature descriptions which are portable between data sets, for example) and to also formally represent the state of an analysis session, e.g., to allow for loading/saving of interactive visualization sessions, we present a compact language for feature specification, i.e., a **feature definition language**, here called FDL for short.

```
<FSpecData>:     Reference to data,
                 <FeatureSets>          // mult. feat. sets possible
<FeatureSets>:   <FeatureSet>+          // only one active at a time
<FeatureSet>:    <Feature>+             // implicit OR of features
<Feature>:       <FeatureChar>+         // implicit AND of f.-chars.
<FeatureChar>:   <SimpleFChar>          // RxR-map (one channel->DOI)
                 || <ComplexFChar>      // induces recursion
<SimpleFChar>:   Reference to channel,  // direct data access
                 mapping2DOI-info       // info for DOI calc.
<ComplexFChar>: AND <FeatureChar>+      // AND-comb. of subsequ.chars.
                 || OR <FeatureChar>+   // OR-comb.  --"--
                 || NOT <FeatureChar>   // NOT of subsequent char.
```

Figure 4.1: *Feature definition language*: sketch of its structure.

A sketch of the FDL-structure is presented in figure 4.1. Here the different key components of this language are shown, namely the feature specification itself (root), feature sets (level 1), features (level 2) and feature characteristics (level 3). In the following subsections, these four layers of the FDL are discussed in more detail.

4.1.1 Feature Specification

The description of a feature specification usually is closely coupled to a data set (the one that is to be analyzed). Alternatively, it could also be portable to similar data sets, when data semantics coincide. In the regular case, a feature specification therefore has a reference to the source data set, as well as to one or multiple feature sets (see below).

Our FDL is realized as an XML [219] language, which makes it easy to handle as well as the FDL-files, resulting from storing the specification process, readable. This also allows to save feature specifications as files, and load them again at any later point in time to resume an analysis session. Additionally, XML-files can be edited using a text-editor, which allows to re-adjust feature specifications also on a file level.

The explicit representation of feature specifications in the form of FDL-files makes using feature specifications on other data sets possible. Of course, care has to be taken that only data channels are referred to, which are available in all these data sets. With portable feature specifications it is possible to generate general feature definition masks, which can be applied very easily (and interactively adapted, if necessary). Examples for employing this functionality were used for comparing different simulation runs of two applications presented in chapter 7 of this thesis.

4.1.2 Feature Sets

A feature set subsumes an arbitrary number of features which all are to be shown simultaneously (realized via an implicit logical OR-combination). Within each single view, always

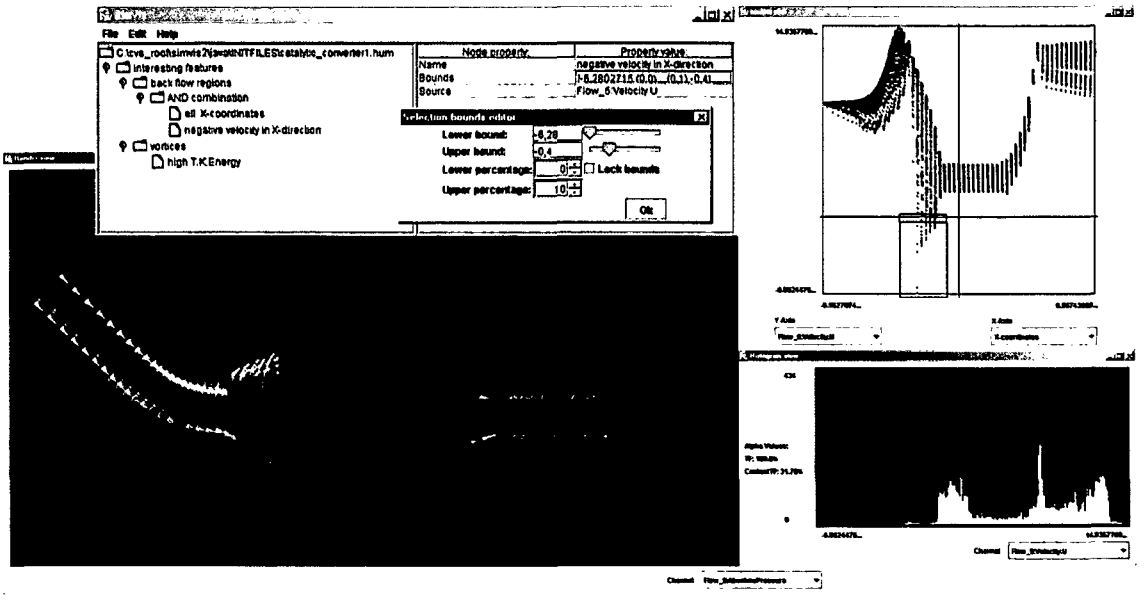


Figure 4.2: *Flexible Feature Specification:* simulation data of a catalytic converter is shown, two features have been specified based on our feature definition language, using different views for interaction and visualization.

only one feature-set is used for focus – context discrimination, all the other feature-sets are inactive. Multiple feature sets can be used to interactively switch foci during an analysis session or to intermediately collect features in a "repository" feature set, not used at a certain point in time but saved for later use. Multiple views can be used for simultaneously showing different feature sets (one per view).

4.1.3 Features

Features are specified by one or multiple feature characteristics (see below). The DOI function related to each feature is built up by an (implicit) AND-combination of all DOI functions of all associated feature characteristics. Multiple features are used to support named feature identification as well as intuitive handling of interesting parts of the data by the user. A feature can be moved or copied from one feature set to any other.

In figure 4.2 two distinct features have been specified, one being backflow and the other vortices. The latter one has been characterized through brushing high values of turbulent kinetic energy, whereas the first feature is described through a logical combination of two separate feature characteristics.

4.1.4 Feature Characteristics

Feature characteristics can be either simple or complex. Whereas simple feature characteristics store brushing information with respect to one data attribute to derive a DOI function, complex feature characteristics imply a recursion in the form of a tree of logical combinations.

Simple feature characteristics store a reference to the data channel which it is based on, as well as information about how the data of this channel is mapped to a DOI function (being

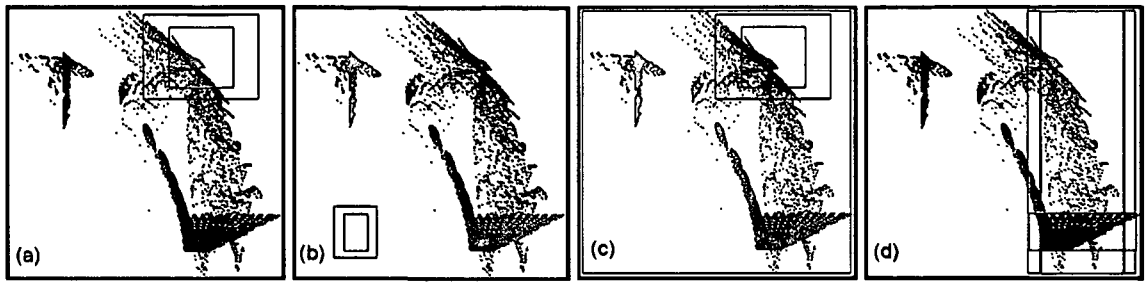


Figure 4.3: four examples of 2D brush types which users found useful during interactive analysis (catalytic converter example, pressure [x] vs. velocity [y]): (a) "high velocity and high pressure" (logical AND), (b) "low velocity or low pressure" (log. OR), (c) "all but high vel. and high pressure" (NOT-AND), and (d) "high pressure but not low velocity" (SUB = AND-NOT).

the output of this characteristic). Especially the possibility to directly interact with the data attributes by specifying feature characteristics and modifying them interactively is very intuitive and straight-forward. In figure 4.2 a simple feature characteristic named "negative velocity in X-direction" is shown in the selection bounds editor. Simple feature characteristics support discrete and smooth brushing (via specifying percentages of the total brushing range, where the DOI-values decrease gradually).

Complex feature descriptions on the other hand provide logical operations (AND, OR, NOT) to combine subsequent feature characteristics in an arbitrary, hierarchical layout. For combining smooth brushes, which can be interpreted as fuzzy sets, fuzzy logical combinations are used, usually implemented in form of T-norms and T-conorms [101]. We integrated several different norms for the above mentioned operations. By default, we use the minimum norm (T_M) in our implementation: this means, when doing an AND-operation of two values, the minimum value is taken, and for the OR-operation the maximum respectively.

In figure 4.3, four examples of 2D brush types, which users found useful during interactive analysis sessions, are shown: The data displayed in the scatterplot views comes from the catalytic converter application shown in figure 4.2 (which is also explained in more detail in section 4.3), pressure (x-axis) vs. velocity (y-axis) values are plotted. The shown brush type examples include: (a) "high velocity and high pressure" (logical AND), (b) "low velocity or low pressure" (log. OR), (c) "all but high vel. and high pressure" (NOT-AND), and (d) "high pressure but not low velocity" (SUB = AND-NOT).

4.2 Interaction

One main aspect of analyzing results from simulation is that investigation is often done interactively, driven by the expert working with the visualization system. Therefore, interaction is one of the key aspects [77] that has to be considered when designing a system which should support fast and flexible usage (as described previously). Especially the task of searching for unknown, interesting features in a data set, and extracting them, implies a very flexible and intuitive interface, allowing new interaction methods. In the following subsections, we categorize the main types of interaction which our system supports. Note that these interactions are designed to meet users' most often requested requirements for such an analysis tool.

4.2.1 Interactive Feature Specification through Brushing

The first type of interaction that has to be considered when designing an interactive analysis tool for exploring simulation data, is **brushing**. In our system, interactive brushing of data visualization is possible in all views except for the 3D SciVis view, which is used for 3D focus+context visualization of the feature specification results (see section 4.3). Brushing is used to define feature characteristics in the FDL interactively. As has already been described in the previous chapter, SimVis allows the user to specify also non-discrete brushes.

4.2.2 Interactive Feature Localization

Another very often used type of interaction is **feature localization**, which is usually requested in the context of simulation data, that has some spatial context. When analyzing this kind of data, the first interest is often, **where** features of specific characteristics are located in the spatial context of the data. Interactively defining and modifying features in different views, coupled with linking, the specification immediately results in a 3D rendering which provides fast localization of the features in the spatial context of the whole data set. For an example see figure 4.4 (a)-(c), where the backflow regions are interactively localized to be in the entrance of the catalytic converter chamber.

4.2.3 Interactive FDL Refinements

After having defined multiple features via brushing and having localized them, often interactive refinement of these features is the next step. Refining the feature specification can be either done by interactive data probing (see below) or by imposing further restrictions on the feature specifications, e.g., by adding additional feature characteristics to the actual state of a feature. One example of such an interactive FDL refinement is shown in figure 4.4 (d)-(f). As a first step (first row), all parts of the data, that exhibit backflow, have been selected, defining a feature that spans over two distinct regions in the spatial domain. In the refinement step (second row) a logical AND-combination of the first feature specification (a) with a new selection in a second scatterplot (showing two other data attributes) is performed (e). Thereby only those back-flow regions of the data are put into focus, which exhibit a general velocity above a specified threshold (f).

4.2.4 Interaction with the Tree Viewer

Interaction with a tree viewer (see figure 4.2, left upper window) as a GUI for FDL is another very useful way to adapt or extend feature sets and features, as well as their characteristics. The tree viewer provides standard GUI elements, such as text fields for manual input of numbers or range sliders, for example. Naming of the different nodes of the FDL, as well as editing all the feature characteristics, and also the management of the tree structure (through copy, delete, or move of the different nodes and subtrees) are the most often used interactions in this viewer. It strongly depends on the nature of users of whether mouse-interactions or keyboard-input are preferred when specifying features. Sometimes, in the case of well-known thresholds, for example, the keyboard-input to the tree viewer is faster and more accurate than mouse-interaction to an InfoVis view.

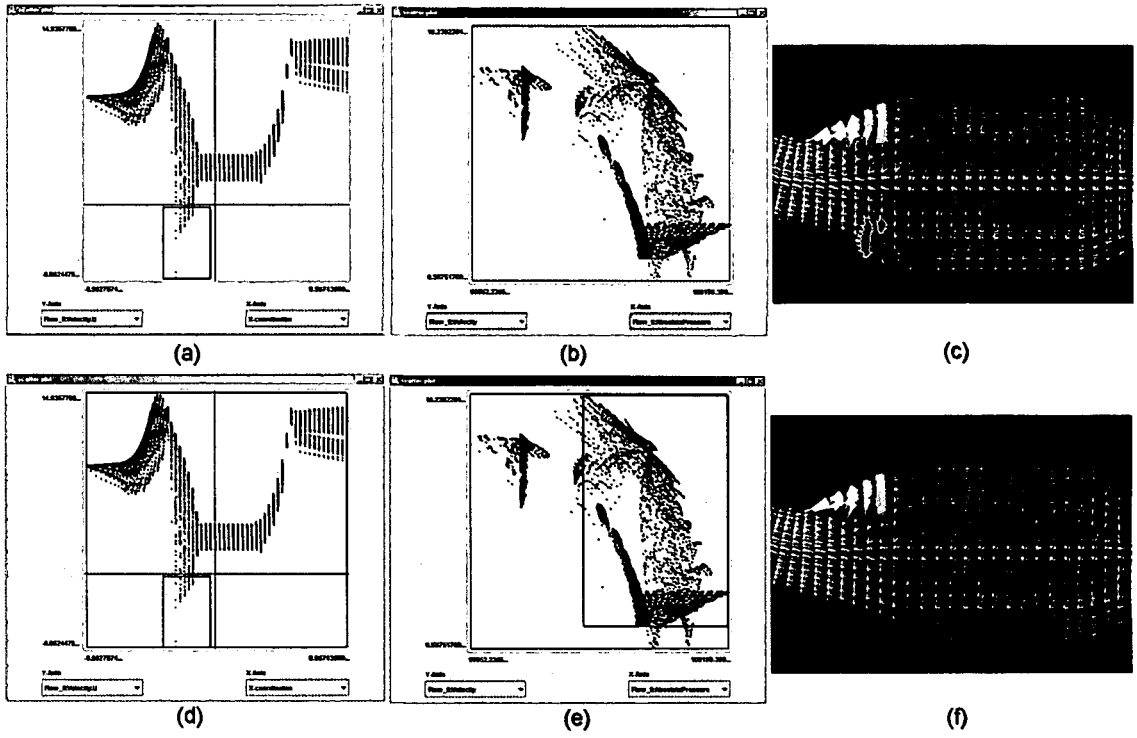


Figure 4.4: *Interactive feature specification and refinement:* (a)-(c): first step: defining backflow in a catalytic converter (see also figure 4.2) in a scatterplot (a) by selecting negative x-flow values, direct linking to a second scatterplot (b) and the 3D view (c). (d)-(f): second step: AND-refinement with a new selection in the second scatterplot (e), linking of the interaction via feedback visualization (color of points according to newly calculated DOI values) to the first scatterplot view (d). Now only the backflow region is selected, that exhibits general velocity above a specified threshold (f).

4.2.5 Interactive Data Probing

Another form of interactively exploring features is using a data-probing approach. Thereby, after having specified a feature (via brushing, for example) the one or other feature characteristic can be changed interactively (e.g., by using a range slider). In all linked views (showing the same data and showing different data attributes) immediate feedback of DOI changes can give new insights into different data aspects. Especially for exploratively investigating value ranges and better understanding of associated patterns in the data sets, this interaction metaphor is very useful.

4.2.6 Interactive Management of Views

One key aspect of a system which provides multiple, different views of one data set, is the interactive management and linking of these views. Our system supports an arbitrary number of InfoVis views (currently scatterplots, histograms, and parallel coordinates), as well as SciVis views. Views can be opened and closed at any point in time without distracting the feature specification. In the InfoVis views, the mapping which assigns data channels to the axes can

be changed interactively. In the 3D SciVis view the mapping of a data attribute to rendering properties (color and/or opacity) via transfer functions can be interactively modified, too. Additionally, the different axes of all available views can be linked (and unlinked) interactively, allowing rapid updates in multiple views.

4.3 Visualization and Results from Applications

After having discussed our feature specification framework as well as the important role of interaction for analysis of simulation data, now general aspects of visualization during analysis are presented below. Then, two different application examples are described in detail. For high quality versions of the images presented here, as well as for additional examples and movies which illustrate the interactive behavior of working sessions with our framework, please refer to *http://www.VRVis.at/vis/research/fdl-vis/.*

4.3.1 Visualization for Analysis

When visualization is used to support analysis of large, high-dimensional data sets, the use of multiple views, as well as of flexible views (with respect to data dimensionality) is very important. SimVis supports an arbitrary number of each type of InfoVis views, as well as SciVis views. When interactively working with data, two types of views in a multiple views setup can be distinguished: **Actively linked views** are the views, which are primarily used for interaction purposes, i.e., for specifying the features, whereas **passively linked views** are primarily used for F+C visualization of the data, providing interactive updates.

3D SciVis views

The 3D SciVis views of SimVis are used as passively linked views for providing a F+C visualization and interactive feature localization. The F+C discrimination is mainly accomplished by using different transfer functions for focus and context parts (and interpolating in-between, for smooth F+C discrimination). The transfer functions in use do not only specify color and opacity, but also the size of the glyphs, that are used to represent single data items (see figure 4.2, lower left window for a 3D SciVis view, showing a smooth F+C visualization).

Two main tasks of this F+C visualization can be identified. The support for feature localization and the visualization of data values through color mapping. Feature localization, as already described in section 4.2, plays a major role in interactive analysis based on features. By using a F+C visualization, the user attention is automatically drawn to the more prominently represented foci, i.e., the features. Value visualization is another very useful task of visualization in this view, and it is accomplished by coloring glyphs according to the associated data channel.

Of course, interactive user manipulation of rendering parameters (opacity, size of glyphs, or zoom and rotate) are necessary, very useful, and support the analysis task, too.

InfoVis views

Apart from supporting interaction, the InfoVis views are very valuable for visualization purposes, too. They visualize different data distributions and also give visual feedback of focus – context discrimination. Points in the scatterplot views, for example, are colored according to

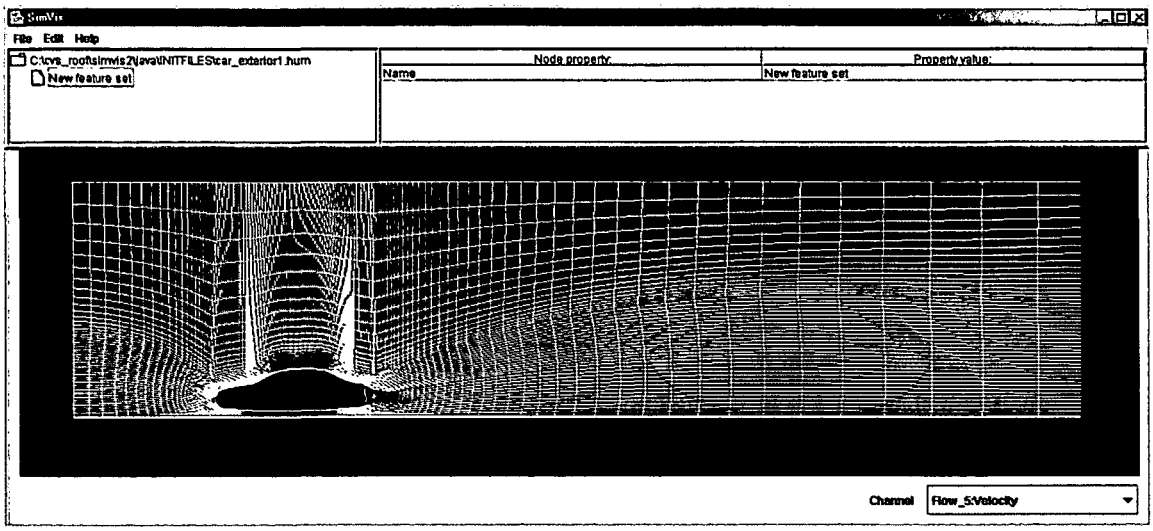


Figure 4.5: *Air-flow around a moving car:* After loading the data set, an empty feature set is created, and the spatial layout of the data is shown, overall velocity information is mapped to color (green denotes low, red high velocity).

the DOI value of the associated data item. Fully saturated red points are shown for data in focus, whereas the saturation and lightness of points decreases with decreasing DOI values, respectively (see figure 4.3 for examples).

In the InfoVis views it is especially useful that the mapped data attributes can be changed interactively. Mapping spatial axis information to one of the scatterplot axes, for example, is very intuitive in our applications (see below). Additionally, using several scatterplots, comparable to a (reduced) scatterplot matrix, often adds information about the data and internal relations of different data attributes.

4.3.2 Results from Air-Flow Analysis

We now want to give a step-by-step demonstration of how a typical analysis session works, especially to show the importance of interaction when analyzing simulation data.

(1) In a first step, a data set is loaded: in our example, results from air-flow simulation around a car (just on one central slice, from the front to the back of the car) are shown. To also cope with 2D-slices of 3D-data, we adapted our 3D-rendering view accordingly. It should be noted, that the general flow direction in this application is in X-direction, past the car from front to back. Before a tree viewer is opened automatically, an empty feature set is generated for preparation of an analysis session. A SciVis view is then opened interactively, to show the general spatial layout of the data (see figure 4.5 for the initial view setup). In this figure the unstructured grid of the data set is shown, overall velocity information is mapped to color (green denotes low, red relatively high velocity values).

(2) As a first start into feature specification (focussing on non-horizontal, slow flow at this step of the analysis) a scatterplot is opened, showing V-velocity (vertical component of overall velocity values), mapped to both axes. In this scatterplot an OR-brush is used to select relatively large positive V-flow, as well as relatively large negative ones, too. Then the

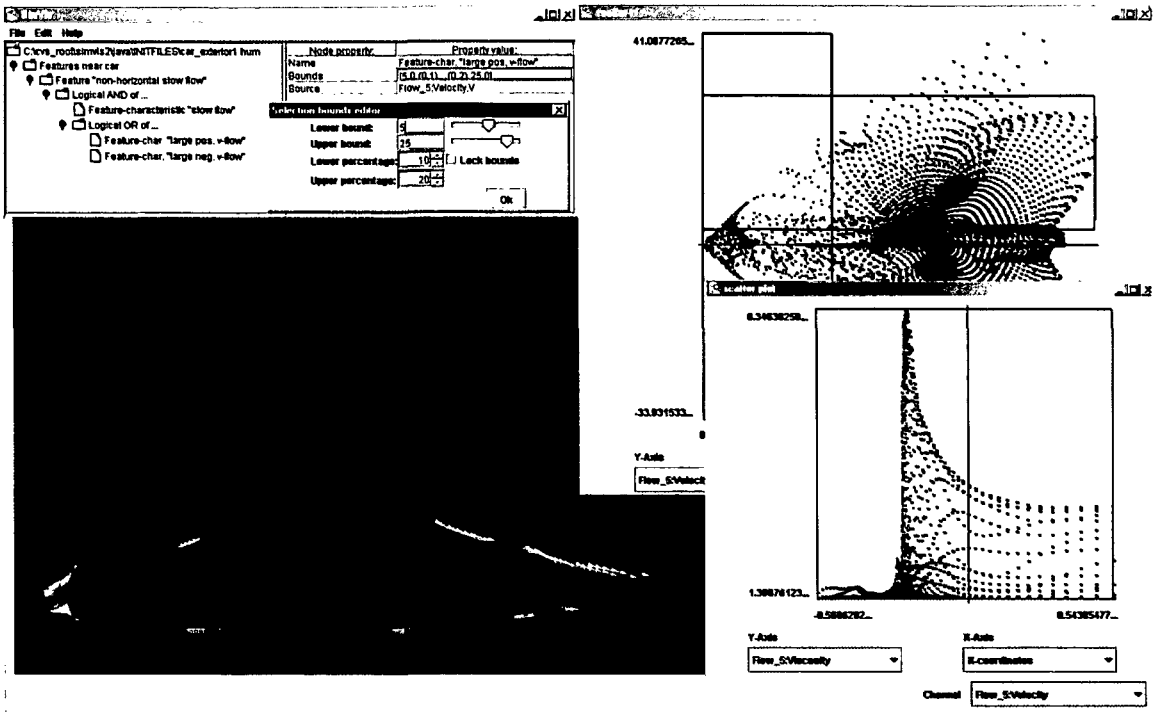


Figure 4.6: *First step of analysis (non-horizontal slow flow):* a tree viewer showing the current feature specification in the upper left (interaction panel for adjusting a simple feature characteristic shown), a scatterplot used for feature specification in the upper right (velocity vs. V-Velocity component), the SciVis view for F+C visualization in the lower left, a second scatterplot for visualization of focus –context distribution (X-coordinates vs. viscosity).

x-axis of the scatterplot is changed to show overall velocity and an AND-refinement is done to limit the feature specification to slow flow (see figure 4.6, upper right view).

To furthermore visualize the feature specification up to this step, a second scatterplot is opened, showing feature and context distribution with respect to the spatial X-coordinates and viscosity (mapped to y-axis of the view, see figure 4.6, lower right). In an interaction panel of the tree viewer, the restriction of V-velocity components is further adapted, to meet the user's needs (see figure 4.6 for a screen capture after this step).

(3) A further AND-refinement, restricting the feature specification to "high viscosity" values is added by using the second scatterplot. As a result of this step, only features behind the car are part of the new focus (see figure 4.7).

(4) Yet another AND-refinement, further restricting the feature specification to high values of turbulent kinetic energy (a value also computed by the simulation), is performed in the tree viewer (see figure 4.8). This clips away parts of the previously selected features, leaving only the parts that exhibit stronger rotational behavior.

(5) To get a better idea of the vortical structures induced, interactive probing on one part of the feature specification (positive V-velocity) is performed. When limiting the focus to negative V-flow only, the down facing parts of the upper as well as of the counterrotating, lower vortex become visible (see figure 4.9).

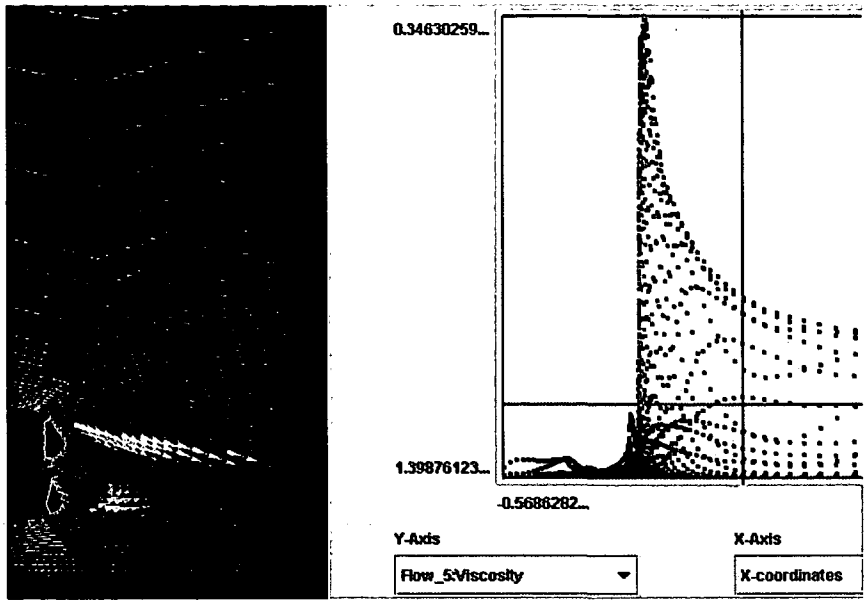


Figure 4.7: *Step 2 of analysis:* AND-refinement, restricting feat. spec. to high viscosity values in the second scatterplot view. Only features behind the car are part of focus now.

4.3.3 Results from Catalytic Converter Analysis

A second example presented here is an application, where the data comes from a simulation of a catalytic converter from automotive industry. The results of another analysis session are shown. The data is given on an unstructured grid in three spatial dimensions, and has 15 different data attributes for each of the approximately 12000 cells of the grid.

The data set and a corresponding feature specification is shown in several views in figure 4.2. The data set consists of basically three spatially distinct parts, the flow inlet on the left hand side, the chamber of the catalytic converter (middle), and the flow outlet on the right-hand side (see figure 4.2, left lower window for a 3D SciVis view). The other views shown in figure 4.2 include: the tree view for handling the FDL (including a pop-up window for changing the brush properties on the x-component of the velocity), a scatterplot (right upper window) plotting x-velocity vs. x-coordinates for each data point, and a histogram, showing the distribution of x-velocity values over the data range.

Two distinct features have been specified using the InfoVis views and the FDL tree viewer. The first feature defines all backflow regions in the data set (with negative x-component of the velocity, as general flow is in x-direction). Two such regions are identified at the entrance of the chamber, a weaker one at the bottom of the catalytic converter, and a stronger one at the top. The second feature description defines all regions, with high turbulent kinetic energy, these are the regions, where vortices are appearing usually in the flow. As can be seen, two vortex cores are easily separated from the rest of the data at the inlet and outlet of the catalytic converter in this case.

Both, the vortices and the backflow regions have been brushed smoothly, to show some information about the gradient of the values in the 3D rendering view. Note, that the coloring in the 3D view is mapped from another data channel, namely data values of absolute pressure.

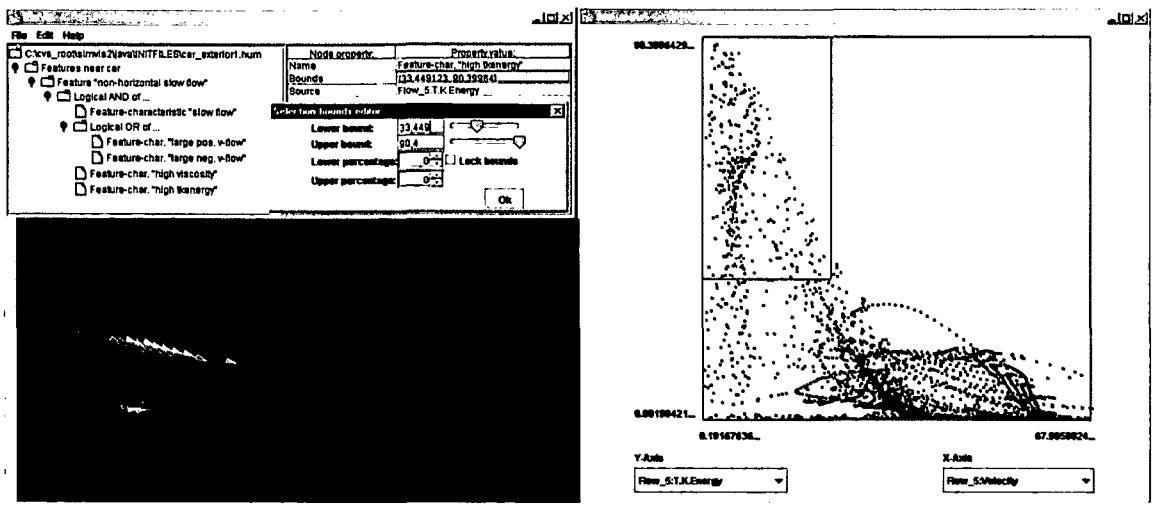


Figure 4.8: *Step 3 of analysis:* another AND refinement, further restricting to high values of turb. kinetic energy, performed in the tree viewer. Only parts with strong rotational component are in focus.)

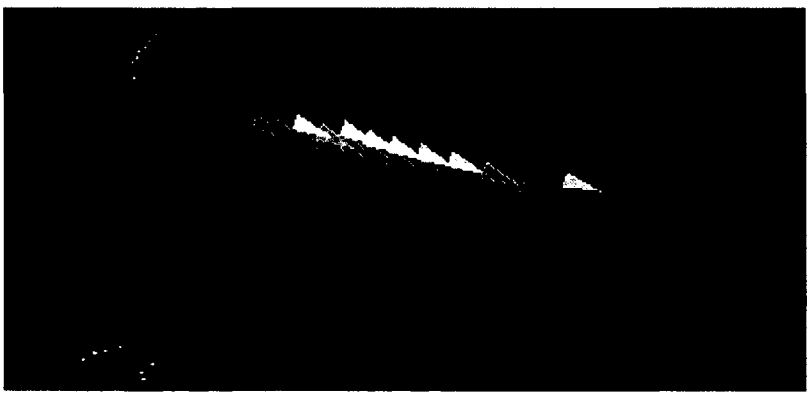


Figure 4.9: *Step 5 of analysis:* interactive probing of V-velocity reveals different behavior of vortical structures, only down facing parts are shown here.

This allows to visualize an additional data dimension for all the data, that was assigned to be in focus beforehand. In the here applied color mapping, green denotes relative low values of absolute pressure, and red corresponds to relative high values.

4.4 Implementation

Implementation details about the software of the presented prototype as well as a hardware description of the PC-platform, on which the results were achieved, are available in chapter 6 of this thesis. The size of the data sets shown is in the range of 20.000 to 60.000 cells, where 15 to 50 data attributes are associated to each cell of the grid. Handling of larger data sets, as well as of time-dependent flow data together with more application examples are presented in the following chapters of this thesis. More details on the data sets as shown here are available

in Appendix A.

The cells of the data are organized in unstructured grids. For the rendering of these grids a visibility algorithm was implemented, based on the XMPVO algorithm [179] presented by Silva et al.

When designing the presented FDL, several considerations were taken, including for example: ease of implementation (close to the visualization system and the data), allow for manual input by the user (preferably ASCII-based, with semantics), verification should be possible (to check for invalid definitions), and many more. To meet all these design considerations, is was decided to use the XML language [219] for storage of the FDL and as interface to other applications. For writing and reading feature specifications to and from FDL-files, the Apache Crimson parser (delivered with the SUN Java SDK) is used, but any other validating XML Parser could also be used. We use a **DTD** (Document Type Definition) for the verification of the FDL trees. The purpose of a DTD is to define the legal building blocks of an XML document. It defines the document structure with a list of legal elements.

4.5 Discussion

The presented framework for flexible and interactive, high-dimensional feature specification for data from computational simulation is aimed at supporting the analysis of large and especially high-dimensional simulation data. For this a feature-based F+C visualization is a good approach, to guide the user and allow interactive analysis. For F+C visualization interactive focus specification is very useful, if real-time updates of multiple linked views are available. Actual features in simulation data often only are captured with a complex type of specification (hierarchical specification, multiple data channels involved). This is why we believe, that using a simple language to define features hierarchically, namely our feature definition language, helps to extract and manage features during an interactive analysis session. In combination with using multiple InfoVis views (for data examination and feature specification) and SciVis views (for F+C visualization of the interactively extracted features) it is a very useful approach.

Chapter 5

Time-Dependent Features

After having presented and discussed the interactive, feature-based approach of focus+context visualization for steady simulation data in the previous chapters, this chapter extends the work to the interactive analysis of time-dependent simulation data sets. Parts of this chapter have been published in an extended form in a technical report at the VRVis Research Center [36], which is currently (2004) in submission.

Visualization of time-dependent simulation data, such as for example the large data sets resulting from unsteady CFD simulations is one of todays most challenging tasks in scientific visualization. We extended our framework for interactive analysis of flow data built on a feature-based approach with special emphasis on the definition and integration of **time-dependent features**. We consider time-dependent features to be those flow features which are inherently dependent on time. For the integration of this new type of features we propose the integration of *attribute derivation* into our interactive data visualization tools to allow interactive specification of complex time-dependent features based on temporal relations in the data. Again the result of the interactive specification, this time of time-dependent features, is linked to all views, and a focus+context style of visualization is realized.

In the following different approaches to feature-based flow visualization are discussed. We then propose a hybrid approach to feature-based flow visualization for time-dependent flow simulation data and shortly recall some important characteristics of our visual flow analysis tool. Sections 5.2, 5.3, and 5.4 focus on recent extensions to our system which have become necessary to deal with large and time-dependent flow data.

5.1 Approaches to Feature-based Flow Visualization

Gaining thorough insight into flow data is difficult because not all of the information within the data also is explicitly represented as first-order data attributes. For example, a flow feature like a vortex is usually not accessible through an explicit data dimension. In fact, one can interpret the properties of a data set as being represented in a *layered feature space*, where deeper layers represent rather complex relations in the data (e.g., a vortex or a shock wave) and more shallow layers correspond to data properties which are explicitly represented in the form of data attributes. Figure 5.1 illustrates this layered feature space with the first-order data attributes forming the upper most layer and the more complex flow features being considered as the "deep treasures" in the feature space.

Feature-based visualization requires that the user has general access to data properties, either in the form of first-order data attributes or from deeper layers in the feature space.

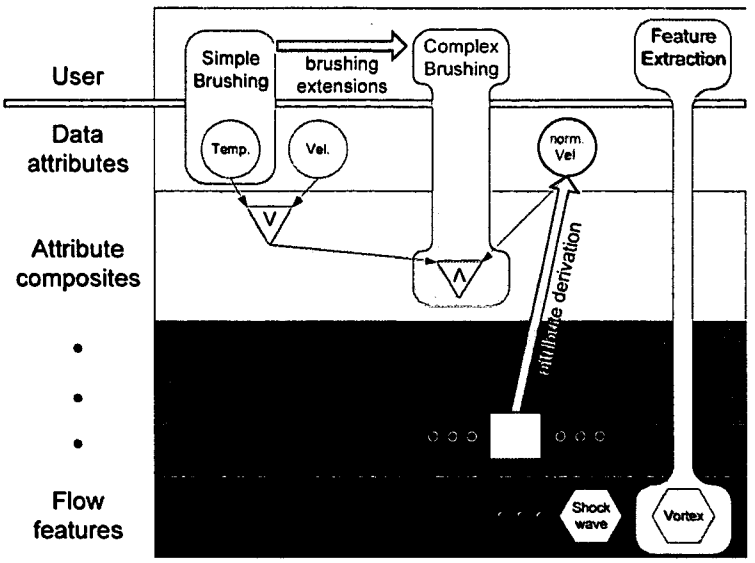


Figure 5.1: The layered feature space: attribute derivation and extended brushing form a hybrid, feature-based visualization approach.

An intuitive and easy-to-use approach is to support interactive brushing [208] of first-order data attributes, for example, marking all those data items with high values of temperature as the feature of current interest. A set of advanced interaction techniques has been recently proposed to ease the interactive brushing of first-order data attributes (see chapters 2, 3, and 4 of this thesis for details).

Additionally, there are other advanced approaches to extract the "deep treasures" from feature space, i.e., to access and visualize complex relations within the data. Usually sophisticated computations are used to extract those parts of the data which correspond to the respective flow features. In chapter 2 we detail on related literature. Special challenges include non-local relations, computationally expensive extraction methods, and numerically unstable feature specifications. Often, feature extraction techniques pose non-negligible challenges for the user to also understand what actually goes on behind the feature extraction process. In many cases the extracted flow features are explicitly represented in the visualization, for example, by the means of boundary surfaces or glyphs. Often the rest of the data is not shown at all yielding significant compression rates of several orders of magnitude [151].

In this chapter we propose a hybrid approach to feature-based visualization of simulation data. We exploit the advantages of brushing as well as of feature extraction. On the one side we "lift up" data properties in the layered feature space by **attribute derivation** (related to feature extraction), i.e., for every data item additional (synthetic) data attributes (second- and nth-order data attributes) are computed through mathematical combinations of first-order data. This happens, for example, through the application of linear filters or the use of formulas describing physical relations between first-order data attributes. On the other side we propose **extended brushing mechanisms** which allow to access data properties even in the deeper regions of the layered feature space. This way, logical combinations of first- as well as of higher-order data attributes can be used to formulate more complex relations than what otherwise would be possible with simple brushing alone.

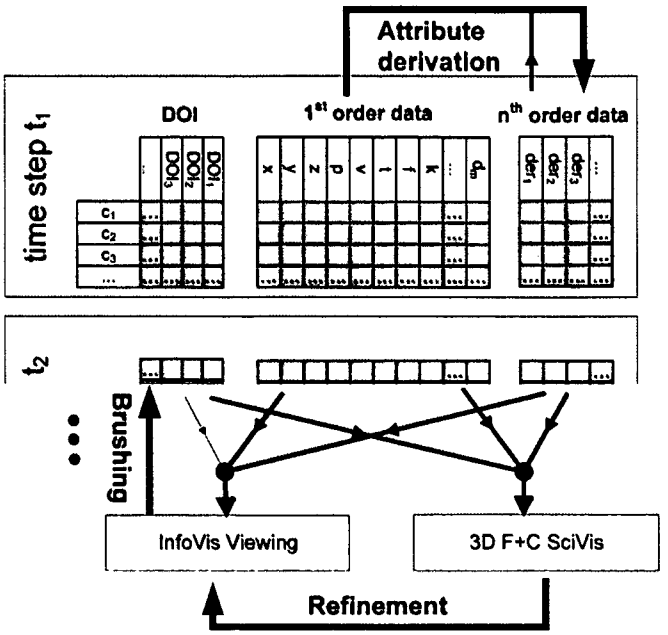


Figure 5.2: Scheme for interactive specification and visualization of unsteady flow features: interplay of *attribute derivation*, *interactive brushing* and *iterative refinement* within the SimVis system.

One advantage of our approach is that since flow features are formulated in the terms of data attributes, it is easier for the user to understand what the actual result of the feature extraction is. For example, attribute derivation is used to represent local temperature maxima as an additional, synthetic data attribute. Complex brushing is then used to select all those local temperature maxima which are in the vicinity of a certain region of interest and which come together with large values of velocity.

This hybrid approach provides a large amount of flexibility in the course of feature specification, improves the user's comprehension of the process of feature extraction, and is well-aligned with focus+context (F+C) visualization, where features are visually enhanced in the 3D scientific visualization (colored and rather opaque) as opposed to the rest of the data which is not shown at all or added as a context for improved user orientation and navigation in a reduced style.

Figure 5.2 illustrates the interplay of attribute derivation, interactive brushing, and iterative refinement with respect to data representation and visualization. Through attribute derivation n-th order data attributes are added to the database, whereas through interactive brushing synthetic DOI attributes are generated. InfoVis views are used for visualization and interactive feature specification (through brushing), 3D F+C visualization (SciVis) provides linked spatial reference for interactive data analysis.

In the following sections the extensions to our system necessary to meet the demands of time-dependent CFD data are presented. Extensions to the feature specification (for the definition of time-dependent features) are realized through *attribute derivation* and *extended brushing* (section 5.2). For the feature-based F+C visualization, extensions to the visualization of flow data with special respect to time-dependent features are presented (section 5.3).

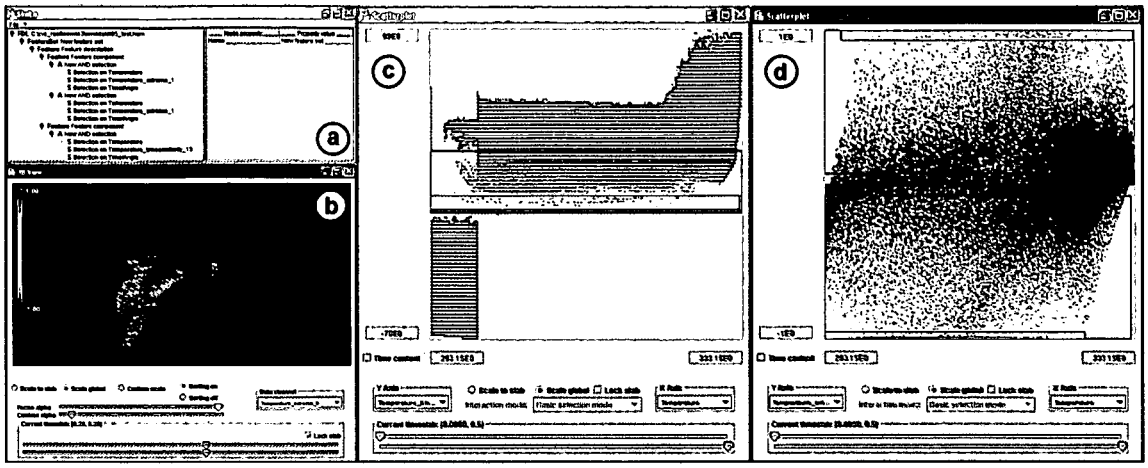


Figure 5.3: Typical setup of a working session with the interactive feature specification framework: a tree viewer is used to manage the feature specification [33] (a); a 3D rendering view, used to visualize the spatial structure of data using a focus+context visualization approach (b); two scatterplots, showing interactively brushed and refined feature specification parts (c+d). For more details on the data set see Appendix A.

Performance improvements to the previously presented framework, which became necessary to handle reasonably large data sets, are also described (section 5.4). Afterwards we present sample results from two applications (section 5.5) before conclusions are drawn.

5.2 Interactive Specification of Time-Dependent Features

When analyzing data sets from large and time-dependent CFD simulation, interactive visualization is very useful. Flexibility with respect to feature specification also is of great utility for the user. For time-dependent data it is necessary to support the specification of features which are inherently dependent on time. In the following we call features, which cannot be directly extracted or specified in single time steps of unsteady CFD data, *time-dependent features.*

As a result from an informal user study with application engineers we identified five types of important time-dependent features. In the following subsections we discuss these feature types one by one. In the SimVis system attribute derivation is issued through the use of a separate interface – the user chooses which data attribute to derive from as well as the type of derivation, and adjusts derivation parameters. For following analysis steps through interactive brushing in InfoVis views all data attributes (1st-order as well as n-th order, i.e., derived) are equally available.

5.2.1 Features based on attribute gradients

Often users are interested in changes of data attributes d_i with respect to time. So we provide the user with access to attribute gradients ($d_i'(c,t) = \mathrm{d}\, d_i(c,t) \,/\, \mathrm{d}\, t$). Due to the discrete representation of time in the form of time steps, three types of gradient approximations are commonly used: forward, backward, or central differences. In the SimVis system, central

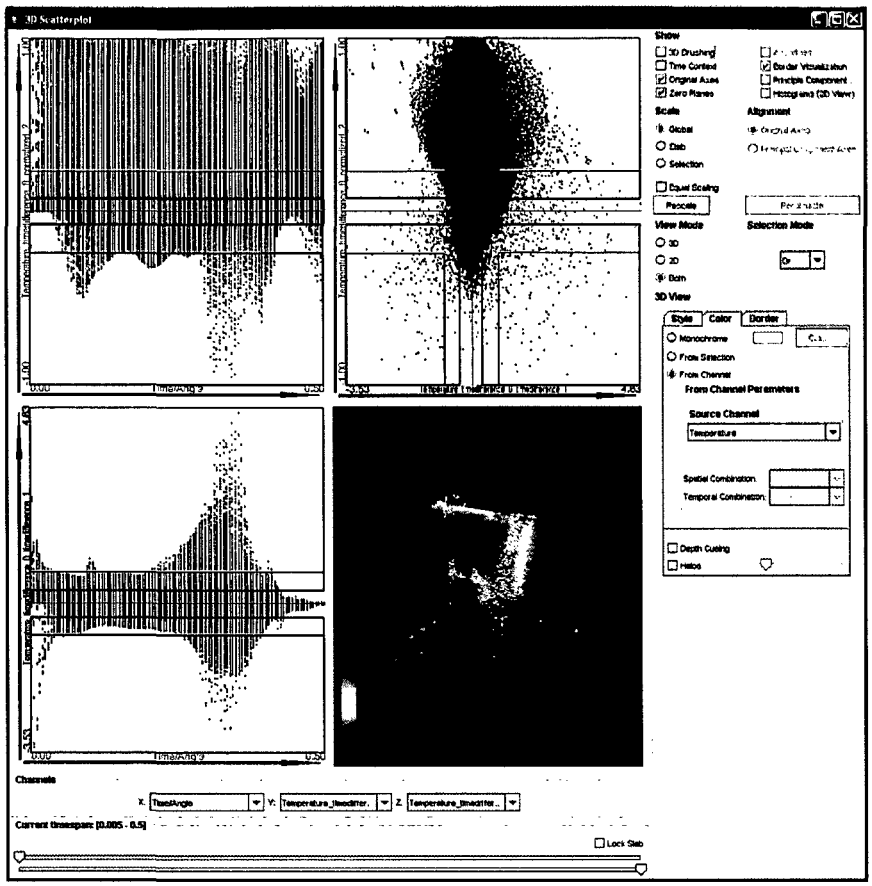


Figure 5.4: Specifying a feature in a 3*2D+1*3D scatterplot view: normalized first-order gradients (green axis) and second-order gradients (blue axis) of temperature values are plotted over time. Higher values of both gradients are brushed in the right upper 2D scatterplot view. For the F+C visualization in 3D see figure 5.5.

differences according to the following formula are used as a useful compromise between data smoothing and frequency amplification as a side-effect of data derivation.

$$\Delta\, d_i(c, t_k) = \frac{d_i(c, t_{k+1}) - d_i(c, t_k)}{2(t_{k+1} - t_k)} + \frac{d_i(c, t_k) - d_i(c, t_{k-1})}{2(t_k - t_{k-1})} \tag{5.1}$$

With this type of gradient approximation we compensate for cases which exhibit unevenly spaced time steps in the simulation.

Gradients are used to detect patterns of changes. One example is a search for large temperature gradients when investigating the burning front in a combustion chamber.

Iterative attribute derivation also is an important functionality of our SimVis system. Applying gradient estimation repeatedly, for example, results in second- or higher-order differences. Figure 5.4 gives an example of how data gradients of different orders are used for feature specification. In a 3*2D+1*3D scatterplot view [148] the simulation time (red axis), normalized first-order gradients of temperature (green axis), and second-order gradients of temperature (blue axis) are plotted against each other for the flow data set from figure 5.3.

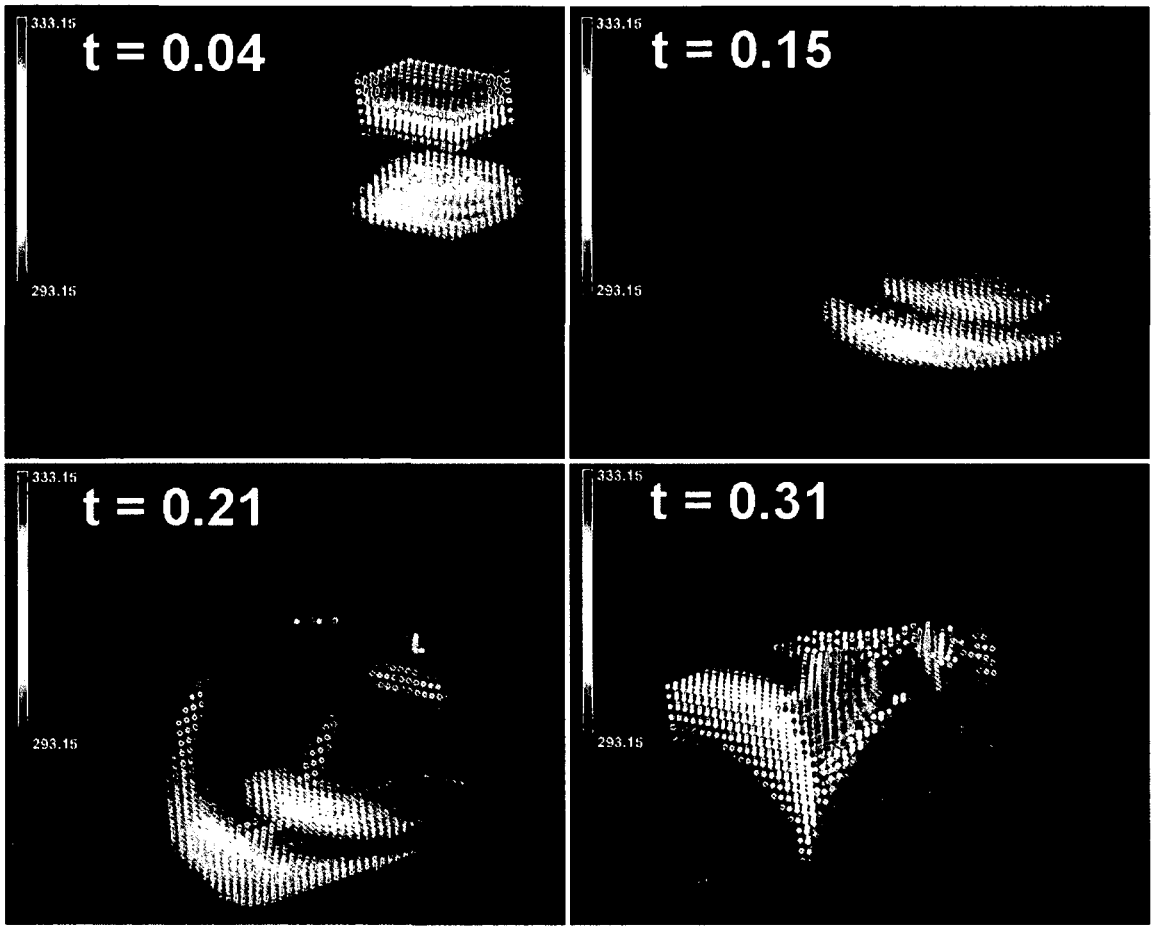


Figure 5.5: Regions of the hot inflow-front in the data set shown in figure 5.3 visualized over time. This feature was specified in the setup of figure 5.4.

A complex brush is used to investigate flow regions in front and behind a moving hot inflow front (see figure 5.5).

Another case of iterative attribute derivation is to filter data (with respect to time) in a preprocessing step to another attribute derivation. We include smoothing of data according to

$$\Gamma d_i(c,t) = \frac{\sum_{t-w \le \tau_j \le t+w} d_i(c,\tau_j) \cdot G((t-\tau_j) \cdot 4/w)}{\sum_{t-w \le \tau_j \le t+w} G((t-\tau_j) \cdot 4/w)} \tag{5.2}$$

in our system, where t denotes the time of the current data item of interest. As a filter a Gaussian lobe with $\sigma = 1$ is used from $[-4,4]$, scaled to time-interval $[-w,w]$. For other purposes, of course, $G()$ also can be exchanged with any other filter kernel easily.

5.2.2 Feature specification relative to data changes

Often the min-max range of data values with respect to certain data attributes changes over time – the meaning of what is relatively hot, for instance, changes with the distribution of temperature values over time. When a relative measure d_rel_i of a certain data attribute d_i

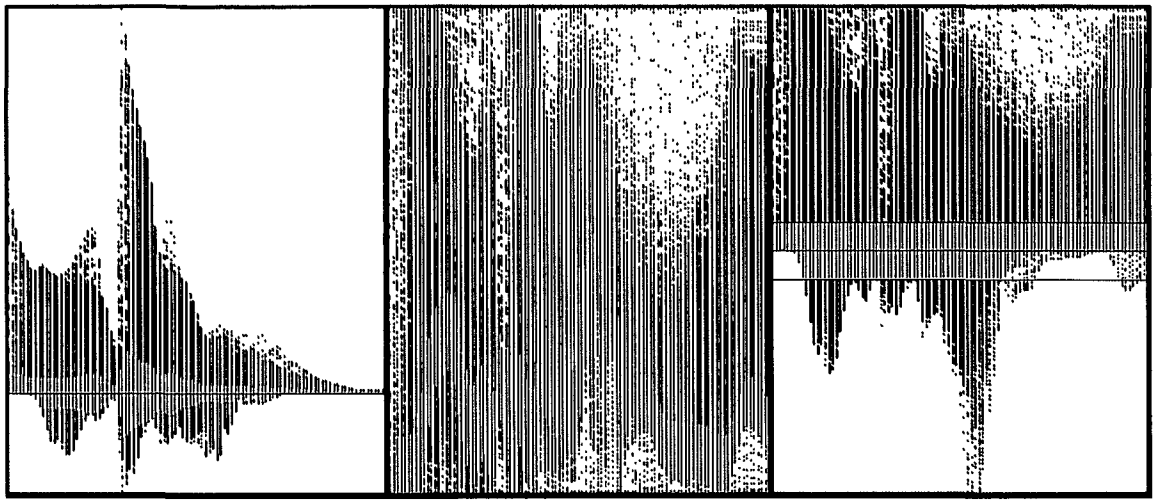


Figure 5.6: Comparison of gradients (left), simple normalized gradients for each time step (middle) and zero-preserving normalized gradients (right) of temperature distributions over time. Data is coming from the extended T-junction example presented in section 5.5.

is available with respect to the data range of the data attribute at the respective time step t then time-step-relative features are easily defined:

$$d\_\mathrm{rel}_i(c,t) = d_i(c,t) / \left(\max_c d_i(c,t) \; - \; \min_c d_i(c,t) \right) \tag{5.3}$$

An example are relatively high temperature-gradients (realized as a feature based on temperature changes) in the case of hot inflows into an extended T-junction (figure 5.15).

For the specification of features relative to data changes, local data normalization with respect to time is necessary. In SimVis, data normalization can be applied to any data attribute available during investigation. Two conceptually different versions are available: the local data range of each time step can be simply mapped to the unit-interval as shown in equation 5.3. Alternatively, normalization can be done to the $[-1,1]$ interval such that zeros in the input data also map to zeros in the output as described by the following four cases ($p \geq 0, k \geq 0$):

$$\begin{aligned}
1: &\ [+p,+k] \text{ with } 0 \leq p < k \text{ is mapped to } [+p/k,+1] \\
2: &\ [-p,+k] \text{ with } p \leq k > 0 \text{ is mapped to } [-p/k,+1] \\
3: &\ [-p,+k] \text{ with } p > k \geq 0 \text{ is mapped to } [-1,+k/p] \\
4: &\ [-p,-k] \text{ with } p > k > 0 \text{ is mapped to } [-1,-k/p]
\end{aligned}$$

Figure 5.6 illustrates the two different versions of normalization and how they compare to each other. In the most left scatterplot, temperature gradients are shown on the vertical axis vs. time steps on the horizontal axis. Simple normalization of the gradients is presented in the middle view, whereas zero-preserving normalization is shown on the right.

With the help of this type of attribute derivation, *relative brushing* of certain data attributes can be realized very easily. After data normalization of a specific data attribute, a brush can be specified in the normalized data range, always selecting the same relative data range per time step. Figure 5.6 shows how gradients relatively close to zero have been brushed in the right scatterplot, the other views are visually linked.

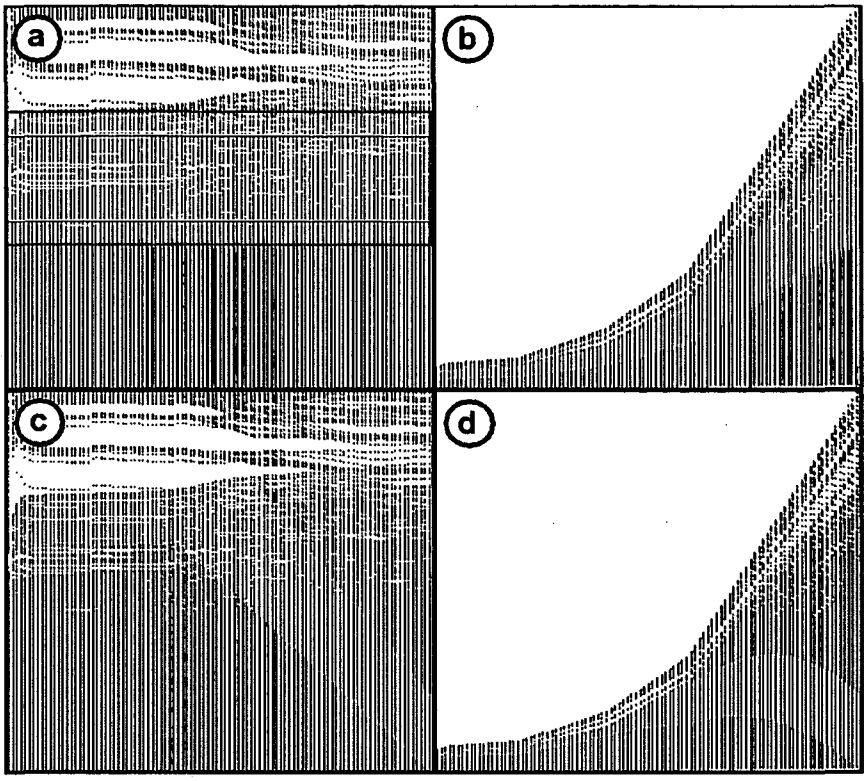


Figure 5.7: Comparing feature specification relative to data changes (relative brushing, (a) and (b)) with interest which varies over time (interpolating brushes, (c) and (d)). The different scatterplots show: normalized velocity values for each time step ((a) and (c)) and original velocity values over time ((b) and (d)). The time domain is always on the x-axis, the data is taken from the sample data set shown in figure 5.3.

5.2.3 Interest which varies over time

Due to simulation processes with certain temporal phases of the flow, users are interested in features whose specifications vary from phase to phase. The difference to feature specification relative to data changes is here, that also the definition of the (relative) interest changes over time and not only the data range of the attribute. When providing an interpolation scheme for feature specifications over time, then a continuous change of feature specifications between certain key time-steps can be realized. Interpolation between the DOI specifications for two different key time-steps is enabled by extending the DOI definition presented in the equation 3.1 in chapter 3 to

$$DOI\_\mathrm{var}_k(d_i, s, \mathbf{p}_s, u, \mathbf{p}_u) = DOI_j(d_i, \mathbf{p}_s + \frac{t-s}{u-s} \cdot (\mathbf{p}_u - \mathbf{p}_s)) \tag{5.4}$$

where $DOI_j(d_i, \mathbf{p}) = DOI_j(d_i, p_0, p_1, p_2, p_3)$ and s and u denote the time of the two corresponding key time-steps with $s \leq t \leq u$.

An example of an interest which varies over time is the definition of moving brushes in the spatial domain when working with data sets based on time-varying grids (see section 7.2.2 in chapter about case study reports of this thesis). Here parts of the geometry are interpolated

from key-geometry to key-geometry. With an interpolation scheme for brushes, spatially moving parts of the geometry can be kept in focus for further feature refinement in the respective areas.

To illustrate the difference between feature specification relative to data ranges and an interest which varies over time, a comparison is presented in figure 5.7. Data is shown from the example presented in figure 5.3, in all four scatterplots time is shown on the x-axis, (a) and (c) show normalized velocity values, (b) and (d) plot original velocity values on the y-axis. In figure 5.7(a) a relative brush is specified, selecting relatively high velocity values. The resulting DOI distribution when viewed for the original velocity values is visualized in (b). In (c) a time-varying interest has been specified, brushing relative high values of velocities for the first temporal phase of the data set, and then decreasingly lower relative values for the rest of the time. The resulting DOI-definition with respect to the original velocity values is shown in figure 5.7(d).

5.2.4 Features based on stationary attributes

In contrast to the interest in changing phenomena, users are sometimes also interested in subsets of the flow, where certain attributes remain stationary (at least to a certain extent) over a longer period of time. We attribute each data item (at each instance of time) with the number of time steps stat_count$_j$ (equ. 5.5), for which a certain data attribute d_i remains stationary with respect to a user-defined threshold. Features with respect to these stationary attributes in certain subsets of time t can then be easily defined.

$$\begin{aligned} \text{stat\_count}_j(d_i(c,t),\varepsilon) = \max_{s\le t\le u} \| \{ s,\ldots,u \} \| \,| \\ \forall \tau \in \{s,\ldots,u\}\colon\ d_i(c,t)-\varepsilon \le d_i(c,\tau) \le d_i(c,t)+\varepsilon \end{aligned} \tag{5.5}$$

Figure 5.8 illustrates three different examples of how the calculation of *measures for stationary flow attributes* (stat_count$_j$) is performed. In case (a) no temporally neighboring data value for cell c_l at time t_u is in the specified ε-neighborhood. In this case stat_count$_j(d_i(c_l,t_u),\varepsilon)$ is assigned the value 1 – this means that the data value is not stationary at all, but stays only for one time step in the specified data range. Case (b) illustrates an example, where the data values of cell c_k stay for a longer time within the specified data range (here for 5 preceding and 5 following time steps). Hence stat_count$_j(d_i(c_k,t_u),\varepsilon)$ is assigned 5+1+5=11. A special case of our calculation scheme is shown in case (c). If the temporal boundary of the simulation is reached (the first or the last time step) before attribute values leave the ε-neighborhood of d_i, then we tag stat_count$_j(d_i(c_j,t_u),\varepsilon)$ by representing it as a negative number. We distinguish this case since it must be assumed that stat_count$_j(d_i(c_j,t_u),\varepsilon)$ is even greater in such a case as attribute values potentially stay within the ε-neighborhood for an even longer time span (outside the simulation time frame). Here in case (c) stat_count$_j(d_i(c_j,t_u),\varepsilon)$ is assigned -13.

An example for interest in features based on stationary attributes is the investigation of all those regions of a cooling jacket for a diesel engine where a stationary high temperature of the cooling fluid is given.

5.2.5 Features based on local extrema

Often it is of special interest where and when local extrema occur in a data set, e.g., when temperature reaches a temporary maximum in a specific location of the simulation grid. In

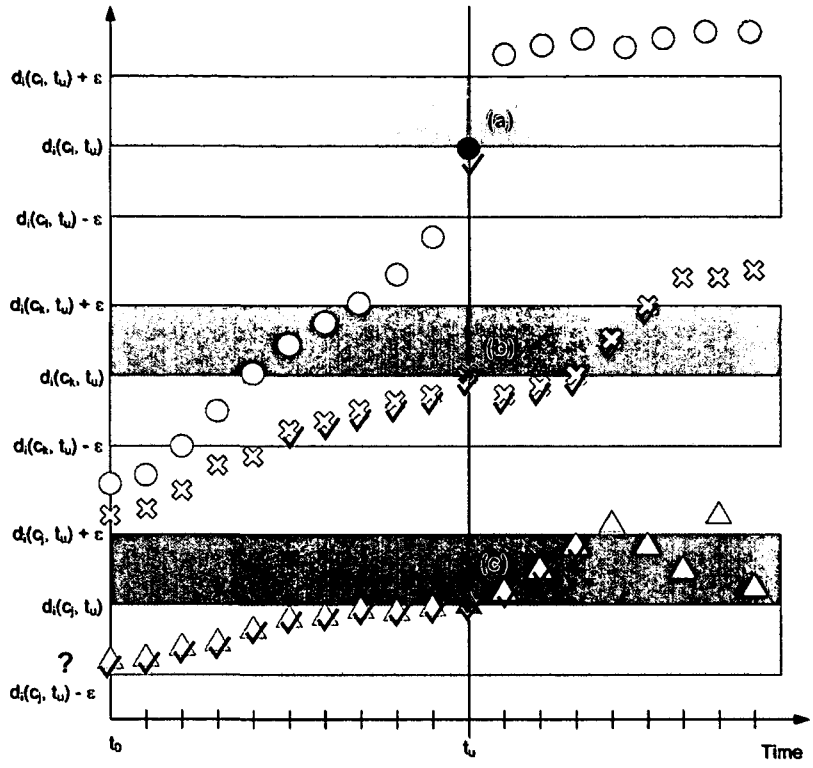


Figure 5.8: Illustrating different cases for the calculation of stationary attribute values.

such cases not only the maximal/minimal value of the respective data attribute is of interest, but also the spatial location as well as the point in time when the local extremum happens. Below we describe how DOI values can be computed to represent local data extrema for each cell c.

$$d\_\text{lmax}_i(c,t) = \min(DOI_j(\Delta\Gamma d_i(c,t), -\varepsilon_1, -\varepsilon_0, \varepsilon_0, \varepsilon_1), \\ DOI_j(\Delta\Delta\Gamma d_i(c,t), -\infty, -\infty, \gamma_0, \gamma_1)) \tag{5.6}$$

In this equation $\Gamma d_i(c,t)$ represents smoothed data values as defined by equation 5.2, Δ denotes gradient calculation and $DOI_j()$ represents a subset selection. The boundary representations ε_0 and ε_1 as well as γ_0 and γ_1 influence how sharply local maxima are specified with $\gamma_0 < \gamma_1 \leq 0$ and $\varepsilon_1 > \varepsilon_0 \geq 0$.

For calculating local minima, only the second argument of min() in equation 5.6 has to be replaced by $DOI_j(\Delta\Delta\Gamma d_i(c,t), \gamma_1, \gamma_0, -\infty, -\infty)$ with $\gamma_0 > \gamma_1 \geq 0$.

One example where local maxima are of special interest is again in a cooling jacket for a diesel engine. Here it would be easy to find all those moments in time, when temperature values of the cooling fluid are at a maximum (and above some limit).

5.3 Visualization Challenges

All the above described unsteady flow features can be interactively specified in the InfoVis views of our system. If attribute derivation is performed during analysis, the resulting derived

data attributes can be used in any of the linked views of our framework just as first-order data. In the following we describe how SimVis views were extended to cope with time-dependent flow data.

Temporal Focus+Context visualization

When dealing with unsteady data sets, usually all data attributes are provided for multiple time steps (usually for all of them, if no pre-selection is performed before the analysis). When viewing the data, two straight-forward options are to either show the data of all time steps simultaneously, or to interactively select one time step for visualization.

In the SimVis system the user can choose from which specific time steps data is visualized in each view by adjusting two sliders. Usually both sliders are bound to each other, so data from one time step of choice is shown. Alternatively, a from-until span of time steps can be selected for visualization, also enabling the concurrent visualization of data from multiple (or all) time steps. An example is shown in figure 5.3. In the 3D SciVis view on the left side, only the data from time step 14 is selected for rendering, whereas in the two scatterplots on the right side, data from all time steps is shown.

For better control of time step selection we added another commonly used *VCR-like interface panel* to our system. It enables synchronized animation of all views in the system through simultaneously updating the temporal focus, i.e., data from which time steps to show. It allows to step through the time steps interactively in both directions of time, to animate at a fixed step rate, and to rewind and loop animation sequences. For the generation of short repeated animation sequences of especially interesting time spans, all options for animations can be restricted to a specific time interval.

Besides time interval specification to specify a temporal focus, we also enable visualization of the temporal context, i.e. showing data from all the remaining time steps, not in the current user interest, in the InfoVis views. We follow the F+C visualization concept for this purpose. Data from selected time steps is shown as focus (black points in a scatterplot view), whereas data from the temporal context (not selected time steps) is diminished (in gray-level style), to get an impression of the overall data distribution. This demands, that the user has to pay attention to the fact, that when performing brushing in the InfoVis views, the brush action is only performed on the focus of time, i.e., on data from the selected time steps. Figure 5.9 shows an example, where only data from a small time interval is in focus.

In SimVis, all views can also display time values on all axes as if time would be first-order data. Examples include the mapping of time onto the axes of scatterplots (see figure 5.7) or the dependence of color maps on time values when rendering in the 3D view (to represent temporal changes in the case of showing data from several time steps).

Scaling the data visualization

In time-dependent visualization, ranges of data attributes usually vary between time steps. When visualizing data from different time steps (or time intervals), this has to be considered. We provide different scaling options in each view: **local scaling** to the data range of the currently visualized time step (or time interval), **global scaling** to the data range of all time steps, **scaling to a specific selection** (scaling to a brush), and **user-defined scaling** (interactive scaling).

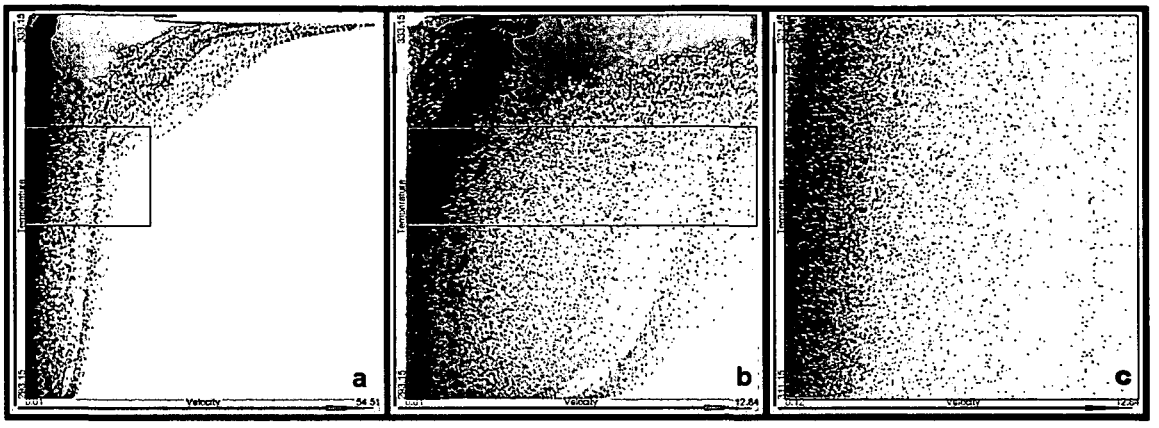


Figure 5.9: Scatterplot views illustrating three different types of view scaling for data from the data set shown in figure 5.3. General velocity vs. temperature is plotted. (a) global scaling to full data set time range; (b) local scaling to the data range of the time interval in focus; (c) scaling to the previously brushed selection (see other views).

Figure 5.9 illustrates this. Here velocity (x-axis) vs. temperature (y-axis) values from the flow data of figure 5.3 are plotted in three scatterplots. Figure 5.9a shows global scaling for the full time range, figure 5.9b scales to the data ranges of a selected time interval (time steps 48 to 52 out of 100 available time steps), and figure 5.9c shows the data distribution scaled to the brush, shown in the first two examples.

Multi-Level Focus+Context visualization

As SimVis utilizes a multi-view approach to focus+context visualization, feature specification is possible in any of the InfoVis views. SimVis not only provides the opportunity to specify multiple features in multiple views, but especially supports the interactive specification of one complex feature in two or more views. Degree-of-interest values which stem from (smooth) brush operations in one or more views can be AND-/OR-/SUB-combined based on fuzzy logic operations [33].

For visualization this means that different levels of focus and context have to be considered. Usually we differentiate four levels (see also figure 5.3 for an example):

- most important (first-level focus) is the DOI attribution which represents the complex feature itself (after logical combinations). In the InfoVis views shades of red are used to represent data items in first-level focus.
- within each of the InfoVis views which are used to specify a certain aspect of the complex feature, the brushed data items are considered as second-level focus. Shades of yellow are used to visually enhance the second-level focus.
- all data items from the time interval which has been used as the temporal support for feature specification are considered as first-level context in the visualization and visualized in black.
- all the rest of the data, i.e., data items outside the selected time interval, is considered as second-level context for visualization and visualized in gray (or left out entirely).

The above described visual discrimination of the different levels of focus and context is set up least-important level first, i.e., first-level focus is drawn over second-level focus, and so on.

3D rendering features

The rendering of data from single time steps in the 3D rendering view in a step-through or animated mode is straight forward. In contrast, 3D rendering of data from a time interval is a more complicated case. In feature-based rendering, the concurrent visualization of data from multiple time steps allows to show the temporal evolution of features in one single image. Another effect is, that by rendering a few time steps simultaneously, outliers, i.e., parts of features which are present only for a short time, are smoothed out and thereby deemphasized.

When rendering data from a time interval in the 3D SciVis view, all selected time steps are rendered into one final image. We render the cells sorted (back to front) and for each cell all time steps are rendered and composited. Each time step (out of n) is rendered using a reduced transparency $\beta = 1 - \sqrt[n]{1-\alpha}$, so that the overall transparency α (for all time steps) stays the same, whether only one time step is rendered or several. Figure 5.14 shows a result of such a 3D rendering of several time steps.

5.4 Large Data Handling

Data sets from CFD simulation are usually large. The size of the data is influenced by the number of time steps, the number of data attributes at each cell of the grid, and the number of grid cells themselves. Loading such large data sets and handling them during visualization and interaction usually is not an easy task, therefore optimized data access policies are very important.

The data sets which we currently visualize with the SimVis system have about 100 time steps, 20–50 data attributes and 100,000 to 1,000,000 cells. File sizes of such data sets range from several hundred megabytes to a few gigabytes. The main extensions to the SimVis system for supporting an efficient handling of large data sets are outlined below. Further details are discussed in chapter 6 and in a separate technical report [56].

The first extension was to enable lazy loading (*activating*) of data. This is realized by following a *Virtual Proxy* pattern [53] for the implementation of the data attribute classes. Due to the fact that it is very expensive in terms of system resources to leave all data attributes activated, a caching algorithm based on a LRU (least recently used) queue has been implemented. First-order data attributes, which can be re-read from the simulation output at any time, are not swapped out to another external file for later reuse as opposed to data resulting from attribute derivation.

Besides the memory management optimizations, performance optimizations have also been included in SimVis. The first version of the software was strictly single-threaded, resulting in a blocking behavior in certain cases as 90% of the computing time was spent in the calculation of DOI values and mouse interaction continuously triggered a recomputation of the DOI values. Our solution to maintain an interactive response of the user interface elements (without blocking), and thus to enable interactive changes of feature specifications, was to decouple the DOI computation from the user interface by using two threads. The DOI computation thread notifies the registered views after finishing the computation to update the viewing information accordingly. DOI computation time is additionally reduced by only recalculating parts of the DOI hierarchy, which are affected by recent user interactions.

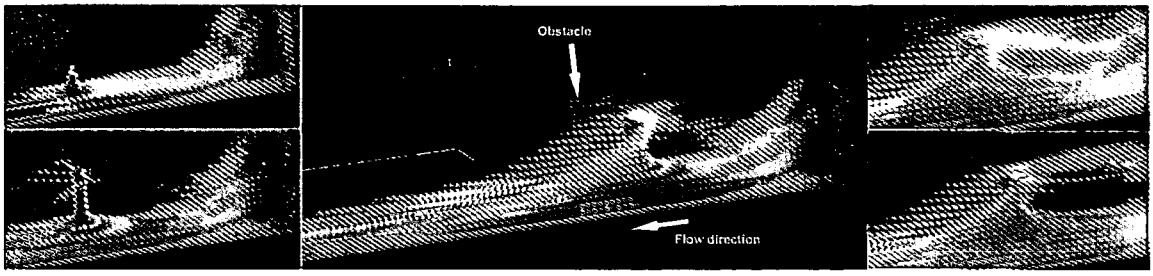


Figure 5.10: Unsteady feature showing the temporal sequence of the impact of a flood after the burst of a dam (situated at the right side) on a close-by object.

To further speed up computationally intensive processes, we also investigated hardware-dependent features. For more details on this investigation and further optimizations implemented in the SimVis system, see chapter 6 of this thesis.

5.5 Application Examples

In the following subsections we briefly sketch two examples of flow analysis using our time-dependent feature specification setup. Further results, as well as a short video, especially demonstrating the interactive behavior of the framework, and the process of how to specify complex, unsteady features are available on the project home page [184]. Two other, also more detailed application descriptions are available in chapter 7 of this thesis and in the two related case study reports [38, 37]. In these case studies a subset of the unsteady features which are discussed in this chapter already proved to be very useful in real-world analysis scenarios.

5.5.1 Flood after the burst of a dam

The first example demonstrates the investigation of a CFD simulation of the burst of a dam and the resulting flooding of the surrounding area. The data set, which can be seen in figures 5.10 and 5.12 is made up of a grid of approximately 90,000 cells, exhibiting 28 data dimensions which are organized in 48 time steps. The flood is simulated as a two-phase flow comprised of water and air. The interesting region in this simulation is the region around an obstacle which is located closely to the bursting dam (on the right side in figure 5.11). In figure 5.10 a time-dependent feature has been specified to investigate how long the flood actually impacts this object. Also the height-level of the impact in different phases of the simulation can be clearly seen from this visualization.

The feature has been specified by smoothly brushing values of high cell fraction of water (high in cells where water is). The specification has then been refined by restricting the selection to represent only cells which stay at a high level of water cell fraction longer than just a few time steps (using the stationary attribute derivation of water cell fraction). The latter restriction eliminates all cells, where just for one or two time steps transient flooding occurs, i.e., we focus only on those parts which are seriously affected by the flood. The features in focus are colored according to the water cell fraction values, yellow denoting medium cell fractions (partly mixed with air) and red showing regions of water only. The temporal evolution in this figure is from upper left to lower right image.

Figure 5.11: RTVR image showing volume rendering of high water cell fraction areas at time step 19 of the simulation of a flood resulting from the burst of a dam.

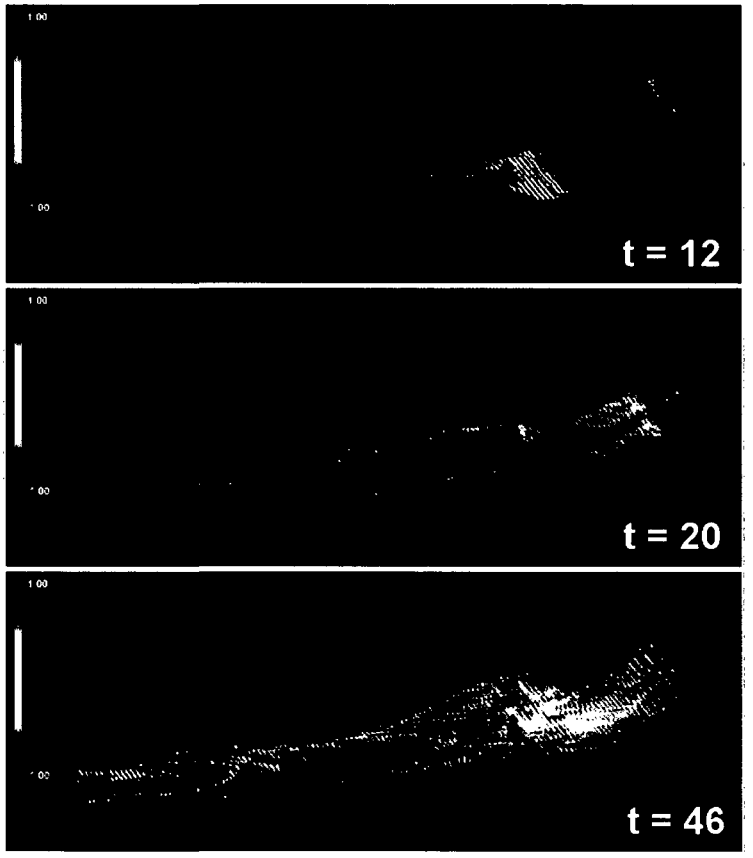


Figure 5.12: Unsteady feature showing the temporal sequence of the impact of a flood after the burst of a dam on a nearby obstacle. Local maxima (red) and minima (green) of volume fraction values of water in the 2-phase flow are brushed.

What can be clearly seen is the recirculation zone in front of the obstacle. It is also interesting that the simulation yields a long-term high level of flooding. For details on the setup of the feature and an animation of the resulting feature, please refer to the accompanying video [184].

SimVis also has been connected to RTVR [136], a real-time volume rendering system. So we are also able to visualize features as specified with our framework with high-speed, real-time volume rendering [69]. As the RTVR allows only rendering of regular grids, the exchange of data between SimVis and RTVR works via on-demand hardware-software balanced resampling of the DOI values into a regular grid, based on an algorithm by Weiler and Ertl [210].

Figure 5.11 shows an RTVR volume rendered image of a feature describing areas of high water cell fraction at time step 19. The outline of the data set is rendered using an NPR contour rendering style, available in the RTVR library.

In figure 5.12 another unsteady feature is visualized for three different time steps of this data set (12, 20, and 46 seconds after simulation start). This feature shows the local maxima (red) and minima (green) for regions with significant volume fraction of water. As can be clearly seen, flow fronts are usually preceded/followed by alternating minima/maxima of volume fraction values, respectively.

5.5.2 Mixing time-shifted flows in an extended T-junction

Figure 5.13 shows a second application example, namely an extended T-junction, consisting of two inlets (one on a lower level, right side, the other on a higher level, front) and one outlet (left side, upper level). Near the front inlet, an obstacle is blocking the flow. This data set consists of approximately 32.000 cells on an unstructured grid, each cell having 18 data attributes. The temporal domain spans 100 equidistant time steps.

The data set has two time-shifted inflows from the two inlets, one starting at time step 0 from the right inlet, injecting warm fluid into the T-junction, the other one from the front inlet, starting at time step 33, injecting hot flow. The two flows mix in the chamber around the obstacle.

Figure 5.14 shows a 3D rendering of temperature values for the data of all 100 time steps of the data set. The visualized feature is specified by high velocity values smoothly brushed for the middle part of the data set (right inlet and left outlet have been neglected). The image shows, which parts are in focus and for how long (the more opaque, the longer in focus). The color codes global temperature values over the whole data, with red coding high values and green showing low temperature (initial temperature in the whole data set). When comparing this visualization to the three renderings of data from single time steps in figure 5.13, the following difference can be observed: the 3D rendering of data from a longer time interval gives a general overview, which regions are in focus for longer periods of time. It smoothes out regions being in short-term focus, like the green, low temperature area in the upper image of time step 16 shown in figure 5.13. Figure 5.13 shows time steps 16, 36, and 60 using the same feature specification as in figure 5.14.

Another unsteady feature is presented in figure 5.15. The feature represents regions, where the temperature gradients are high and the rate of change of gradients (second derivation) is either very high, or very low. This shows regions, where temperature changes very much, but the change itself is only short-term. This corresponds to the front and backside of the flow fronts. By color coding according to the change of gradients, the distinction between positive and negative changes of gradients becomes visible in the 3D view. Images 1 and 2 of this figure show the first flow front, whereas images 3 to 6 show the second front. Note, that this is captured by the specification of one feature (not temporarily or spatially connected), although the two flow fronts represented by this feature exhibit different temperature ranges.

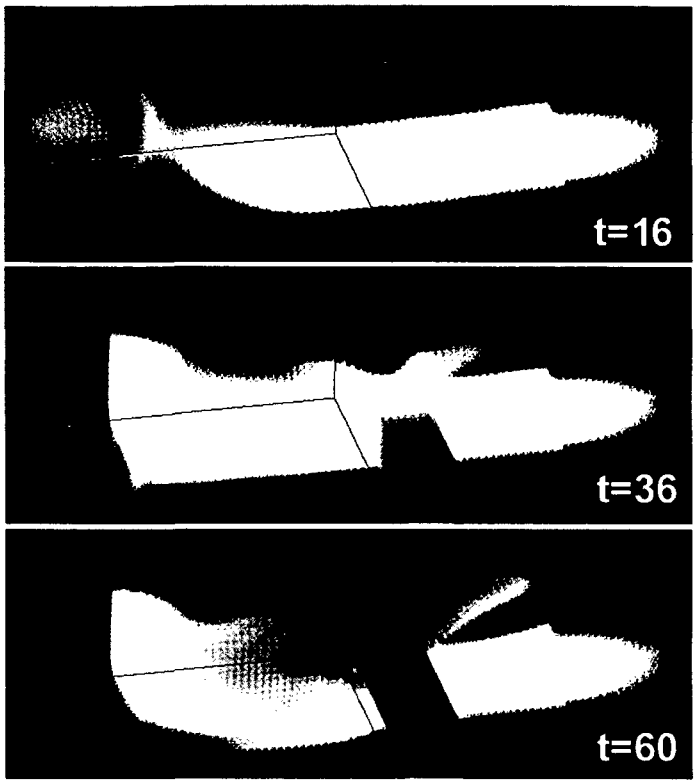


Figure 5.13: Time steps 16, 36, and 60 of the extended T-junction data set, showing the same feature as in figure 5.14.

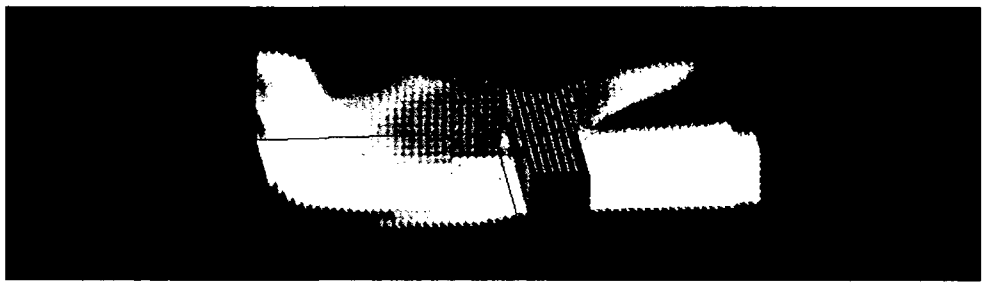


Figure 5.14: 3D rendering of temperature data for all 100 time steps in the high velocity areas. More opaque feature areas are long-term in focus. This unsteady feature shows high velocity regions smoothly brushed in the middle part of the T-junction.

5.6 Discussion and Further Extensions

Extensions to the SimVis framework for interactive visual analysis of CFD simulation data, especially focussing on the demands of *time-dependent CFD data*, have been presented in this chapter. Extensions to the feature specification process (for the definition of time-dependent features) are realized through *attribute derivation* and *extended brushing* mechanisms. For the feature-based F+C visualization, extensions with respect to the visualization of time-

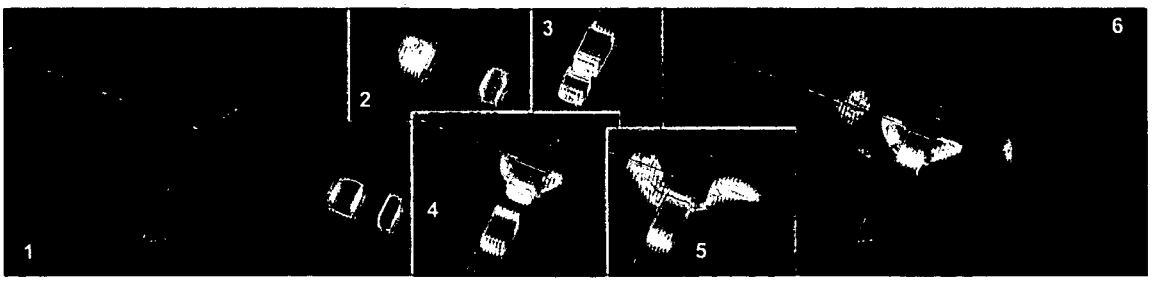


Figure 5.15: The regions around the two different flow fronts are selected by using a single unsteady feature specification, using first and second derivations of temperature values (see text for details).

dependent flow data were presented. Performance improvements to the SimVis system which became necessary to handle reasonably large data sets (as dealt with in the commercial application of our industrial partner) were described shortly (and will be discussed in more detail throughout the next chapter of this thesis).

Attribute derivation, to enable access to information which is inherently dependent on time, as well as supporting data-range independent, relative data assessment are new tools which support the specification of unsteady flow features in time-dependent data sets. Given such flexible tools for interactive visual exploration and analysis of large data sets, the user gains improved insight into CFD data.

Also, the extension of conventional 3D rendering of data from singular time steps, as was previously employed, to showing data from longer time intervals concurrently by enabling 3D visualization of multiple time steps out of such time intervals brings new findings. Thereby it becomes possible to show in a single image the temporal evolution (up to a certain degree) of unsteady feature characteristics.

Further extensions to our framework regarding the temporal component of simulation data include the possibility to handle data based on time-varying grid geometries. A case study, showing how to handle such data sets has been investigated [37], and a detailed report about this case study is available in section 7.2 of this thesis.

Future work will include to extend the here presented attribute derivation methods for specifying time-dependent features to cope with the special challenges of data based on time-varying grids. Using position-based methods for feature extraction will have to be included, as opposed to cell-based methods. An example would be to calculate gradients for absolute or relative spatial positions rather than cell-based temporal gradients.

Chapter 6

SimVis – The Framework

In this chapter the current state of the SimVis framework is presented from an implementation point of view. Special emphasis is put on the discussion of the design of the data layer and a couple of performance optimizations of the highly interactive parts of the SimVis system.

In the previous chapters of this thesis different parts of the SimVis framework have been presented and discussed in detail. Special focus has been put on explaining the multi-view visualization environment, which combines different methods and views from scientific visualization (SciVis) and information visualization (InfoVis). A typical SimVis setup for an analysis session consists of multiple views, each showing different attribute distributions or correlations (see figure 6.1 for a sample setup, data from the simulation of a ski jumping application is shown).

As described in the previous chapters already, the system supports (smooth) brushing in all of the InfoVis views and linking of multiple views to specify complex feature definitions. Linking between different views is realized by distributing degree-of-interest (DOI) values for each data item according to the respective brushing actions in the different views. Feature specification is handled by representing it explicitly in a feature description language (FDL), which can also be used to store and resume analysis sessions, or for comparing results from similar simulation runs. In the 3D SciVis views, focus+context (F+C) visualization is supported, showing the regions containing currently brushed data items in focus (more opaque, colored) and the rest of the data for context orientation (more transparent, gray-style).

To demonstrate the powerful opportunities for data analysis by using our concepts for interactive visualization, we have put together a demo-workstation and a proof-of-concept implementation. This combined hardware-software setup is called SimVisBox. The purpose of this chapter is to give insight in challenging problems and respective solutions we have developed during the realization of the SimVisBox with special respect to large data handling and speed-up optimizations.

The remainder of this chapter is structured as follows: section 6.1 shortly presents the hardware setup and gives a performance evaluation for the current version of our SimVisBox. After this, section 6.2 gives an overview about the data formats used. Then the software architecture is presented shortly (section 6.3), implementation issues are discussed (section 6.4), and performance optimizations are described in more detail (section 6.5). At the end a few known limitations are mentioned and future work plans are presented in section 6.7.

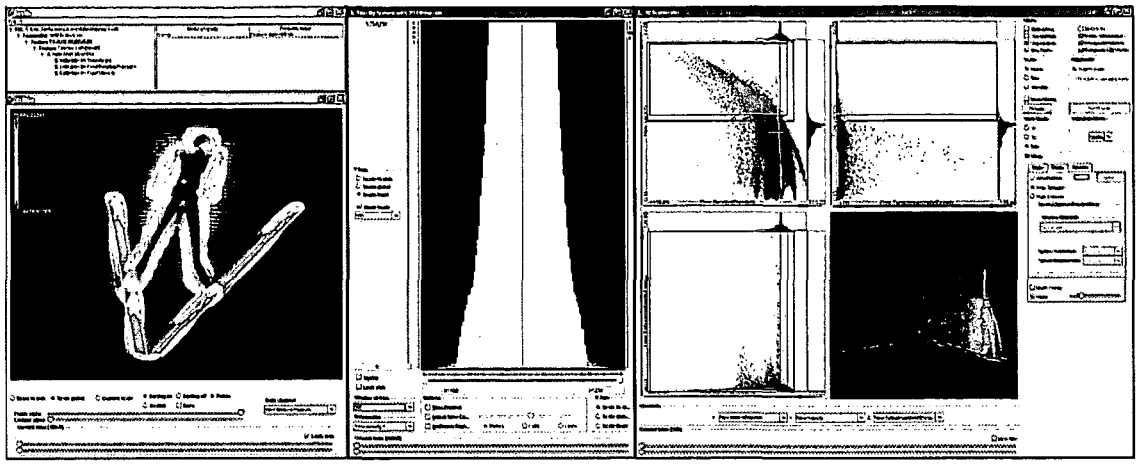


Figure 6.1: Typical setup of an analysis session with SimVis. Multiple views are used to explore attribute distributions and 3D context.

6.1 SimVisBox: Setup and Performance

The current version of the SimVisBox (as of July 2004) consists of the following parts: a *PC-based computer system* (hardware) and our current *software prototype of the SimVis visualization system.* The software part is described in deeper details throughout the next sections of this chapter. Here a short summary of the (current) hardware setup and a performance evaluation are presented.

Hardware Setup

The current hardware setup of the SimVisBox has the following technical details:

- OS: Windows XP Professional
- Intel Pentium 4, 3GHz
- 333 MHz FSB
- Hyperthreading support enabled
- NVIDIA Geforce FX 5950
- 2GB RAM
- 2x80GB S-ATA RAID0

The SimVis software runs also on less powerful PCs, but it has to be mentioned, that the size of memory and the speed of the processing unit, as well as the speed and texture memory of the graphics card influence the possibilities of what size of data sets still can be shown interactively.

SimVis runs on Windows-based Intel processor PC-platforms from the Pentium III upwards. Required is an installation of Java JDK1.3 or higher, and a minimum of 512MB of memory. NVIDIA graphics cards [139] from the Version GeForce3 upwards are supported, but the system has also been tested recently on ATI graphics cards [5], like the ATI Radeon 9800, successfully.

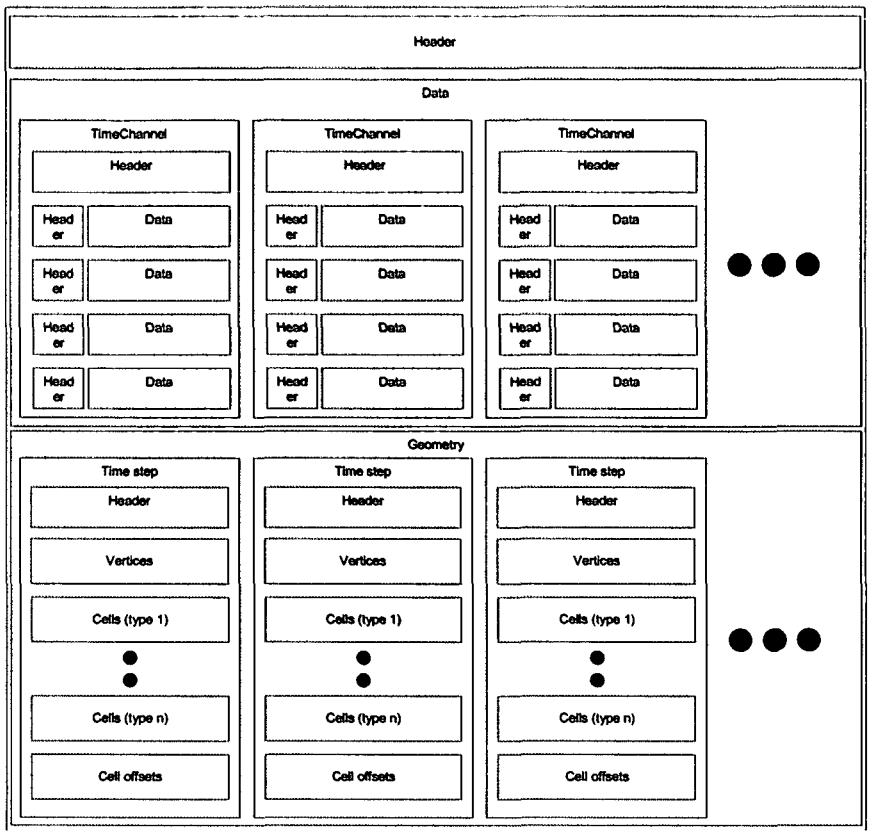


Figure 6.2: The structure of the HUM file format.

Performance Evaluation

With the hardware setup of the SimVisBox as described above, we are able to interactively handle and visualize data sets of the following sizes:

- number of cells (per time step) $N_c < 3.000.000$
- number of time steps N_t is generally not bounded, but $N_t < 1.000$ is usually assumed
- **limiting factor**: due to memory limitations, currently $N_c * N_t < 15.000.000$ is required
- the number of data attributes N_d per data item (or cell) is theoretically unlimited, due to the caching algorithms implemented, but usually not more than 50–60 data attributes (including derived data attributes) are needed.
- simultaneous viewing of < 15 data attributes is assumed

In the future we plan to further improve the performance of the system, especially the limiting factor of $N_c * N_t < 15.000.000$ should be extended by one or two orders of magnitude.

6.2 Data File Format and Data Properties

To allow for easy import of data from different customers as well as to optimize our visualization algorithms, we have developed our own proprietary file format and data access layers

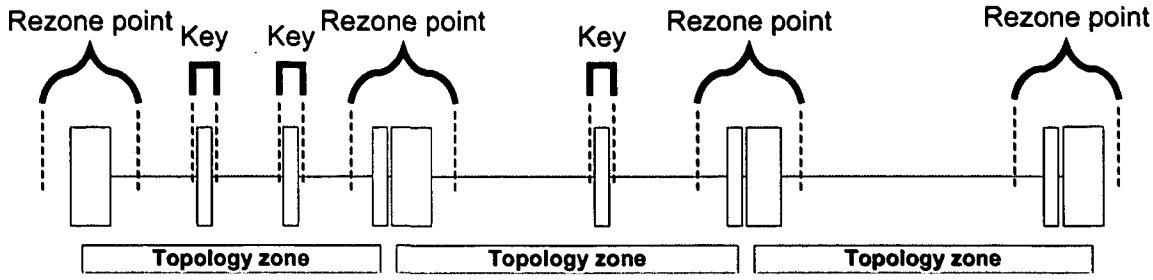


Figure 6.3: Moving Meshes: Our concept for handling simulation data based on time-varying grids.

in the SimVis software. The file format is called **HUM** (**H**igh performance **U**nstructured **M**esh). It is used both, for storing the data resulting from the simulation as well as the corresponding geometry information (for which the calculation is performed during the simulation). The HUM file format (see illustration in figure 6.2) is divided in three parts: a header, specifying general information about the data set, a data block, storing the simulation results for the different data attributes simulated over the different time steps (we are dealing with time-dependent simulation data), and a geometry block, holding the geometry (and topology) information. For a more detailed description of the format, see the description of the internal data representation (which is closely related to the file format) in section 6.3.3.

The use of our own file format allows fast input of new data being provided in different data formats. Often the only necessary extension is to provide a converter application for new data formats. The description of the use of one data conversion application for data from our main industrial partner (AVL List GmbH [6]) is available in section 6.6.

Moving meshes: Support for Time-Varying Grids

A recent extension to SimVis includes the ability to handle data sets based on time-varying grid geometry/topology. We therefore extended the HUM format, to handle geometries, where vertex positions and the adjacency structure (i.e., cell topology) can change over time. In our format, vertex positions are only stored for key geometries that define the beginning and the end of a linear motion sequence. All other vertex positions can then be derived by linear interpolation.

We have also exploited the fact that changes to the adjacency structure occur less frequently than changes to the vertex positions by storing connectivity information only once for a segment with constant topology.

Figure 6.3 illustrates our concept: *rezone points* cut the moving mesh into *topology zones*. *Rezone points* consist of two *key geometries* at different resolutions and represent correspondence between zones of different topology. *Topology zones* are segments with constant topology. Within a *topology zone*, an arbitrary number of *key geometries* can be defined, but every *topology zone* consists of at least two *key geometries*.

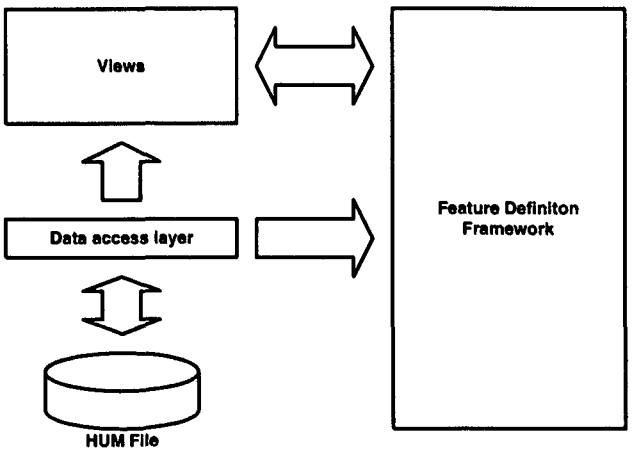


Figure 6.4: Main software components of the SimVis prototype system.

6.3 Software Architecture

Figure 6.4 shows the data flow between the main software components. The *Data Access Layer* implements transparent data access and caching. The *Views* and the *Feature Definition Framework* perform bidirectional communication, relying heavily on the data access services. These three main components of the system are now described in a little more detail.

6.3.1 Feature Definition Framework

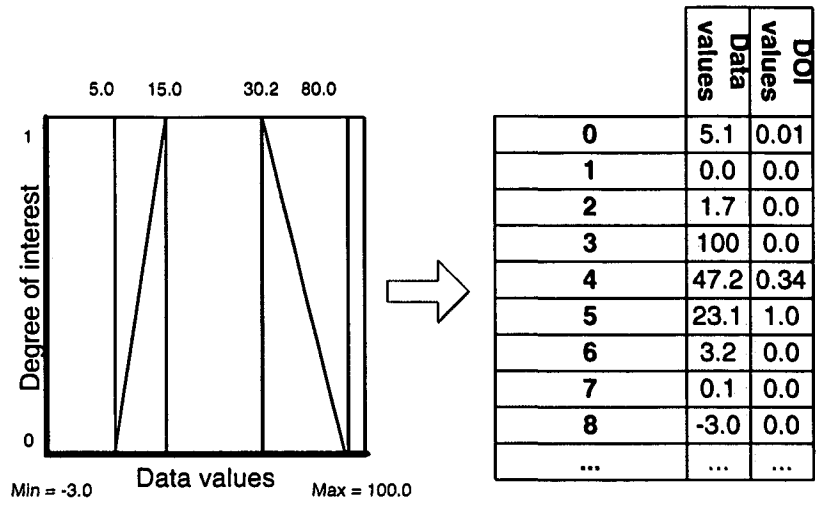
The feature definition language (FDL) framework is built around a hierarchical tree structure which logically combines simple one-dimensional selections to more complex feature specifications following a descriptive and modular approach.

Each *SimpleSelection* (forming a leaf of the tree) defines a trapezoidal DOI function (see figure 6.5) that can be interpreted as a fuzzy membership function. From each node in the tree, a degree of interest function can be derived in a bottom-up fashion by logically combining DOI functions derived from SimpleSelections to selections of higher dimensionality.

To provide better manageability of the code, the functions for computing the DOI values have been centralized in a separate object by employing a *Visitor* pattern [53]. Changes to leaves of the tree induce toggling of a *dirty* flag along the path to the root, followed by asynchronous execution of the recursive DOI recomputation code for the *dirty* tree nodes, starting with the root of the tree. Figure 6.6 illustrates how the individual nodes in a simple FDL tree would interact after a manipulation of a SimpleSelection.

Logical operations are currently realized as minimum T-norm (fuzzy AND), maximum T-conorm (fuzzy OR), and the complement on one (fuzzy NOT) [101]:

$$d1 \wedge d2 = min(d1, d2)$$
$$d2 \vee d2 = max(d1, d2)$$
$$\neg d = 1 - d$$



	Data values	DOI values
0	5.1	0.01
1	0.0	0.0
2	1.7	0.0
3	100	0.0
4	47.2	0.34
5	23.1	1.0
6	3.2	0.0
7	0.1	0.0
8	-3.0	0.0
...	...	...

Figure 6.5: On the lowest level of the feature definition framework, each n-dimensional data point is mapped to a DOI value, according to its value in one dimension.

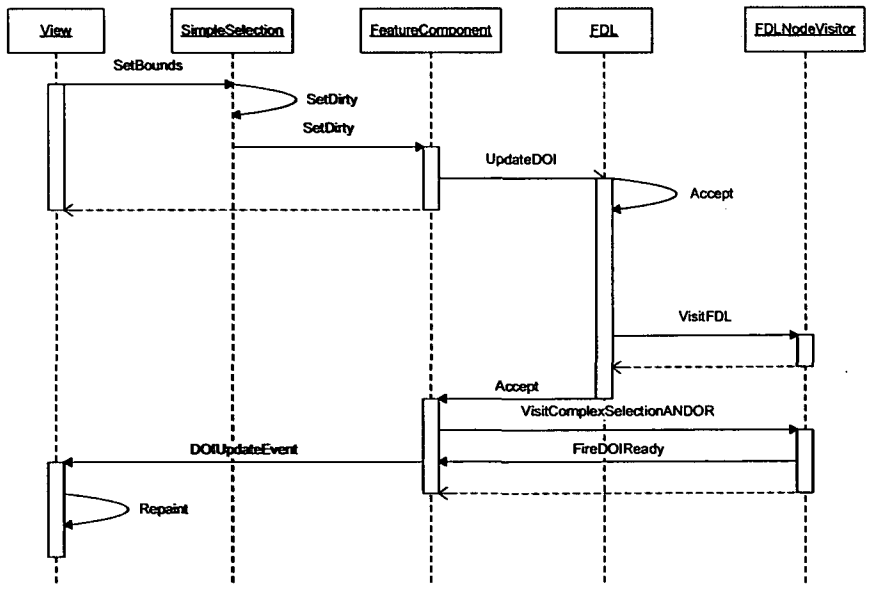


Figure 6.6: Views alter simple selections and trigger the recomputation of the DOI.

6.3.2 Views

Each feature specification stores a reference to the view which created it and views can be opened and closed on demand. Changes to tree nodes fire events that are propagated up the tree, resulting in an update of each view associated to a tree node in the path from the affected

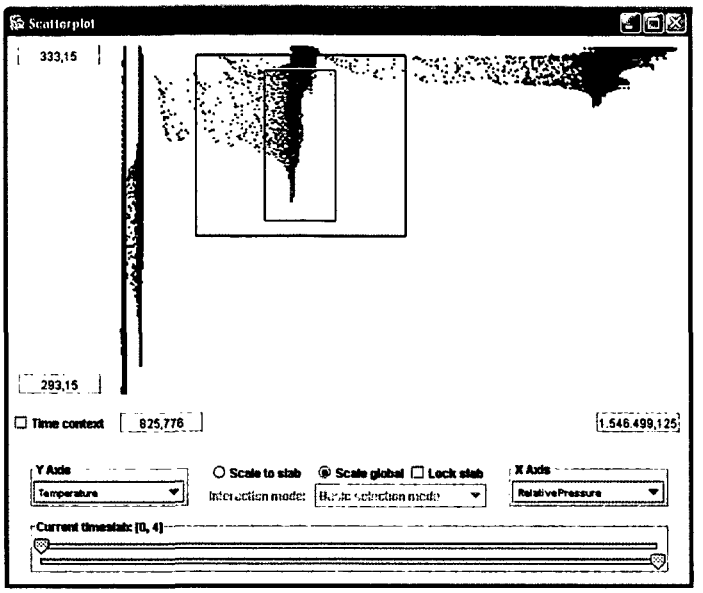


Figure 6.7: A cluster in the combined distribution of *temperature* and *pressure* is (smoothly) brushed in a scatterplot view.

node to the root. The views act as *Observers* [53] that are notified whenever changes occur in the observed objects (feature specifications). Hence the feature specification hierarchy can be seen as a hierarchical hub, forwarding update requests only to those views that are affected by it and realizing implicit linking of all views that belong to the current subtree.

In the following, a short overview of some of the views that we already integrated into our system, is given.

Scatterplot

A (2D) scatterplot shows two-dimensional projections of the data space. Two components of an n-dimensional data point are interpreted as coordinates in a Cartesian coordinate system. Figure 6.7 shows a scatterplot that plots the *temperature* values of the data points (on the Y-axis) versus their *relative pressure* values (on the X-axis). Points with medium relative pressure and high temperature have been classified as features with a smooth brushing widget in this example.

Combined 2D/3D Scatterplot

A combination of 2D and 3D scatterplots, which allows multiple different modi to view distributions of data attributes, is also available in the SimVis system [148]. Besides offering single 2D or 3D scatterplot displaying (plotting 2 data attributes against each other, as described above, or 3 data attributes in a XYZ-orthogonal coordinate system), it also offers a combined mode. In this mode, the displaying window is subdivided into a 2-by-2 matrix of four views, as shown in the sample image in figure 6.8.

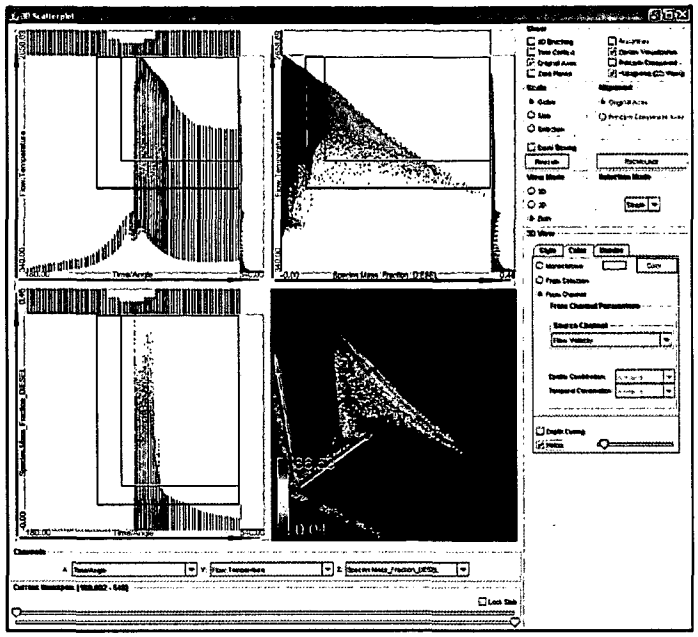


Figure 6.8: A combined 2D/3D scatterplot view: 3 data attributes are plotted against each other (right, lower sub-view); plots showing all 3 possible data distributions of two of the variables each are arranged in the remaining three sub-views.

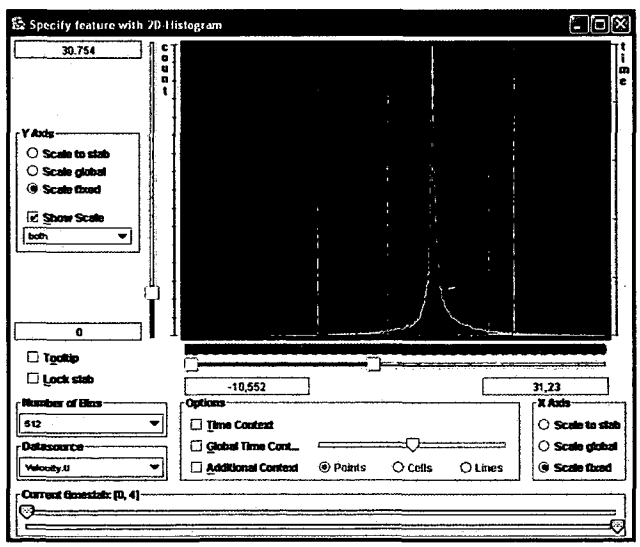


Figure 6.9: Peaks in the distribution of *velocity in x-direction* are selected in a histogram.

Histogram

A histogram counts the frequencies of occurrences of data values in a particular data dimension. Since this is not directly possible for continuous data, the data range has to be segmented into bins (small subranges) that represent a discretized version of the data space

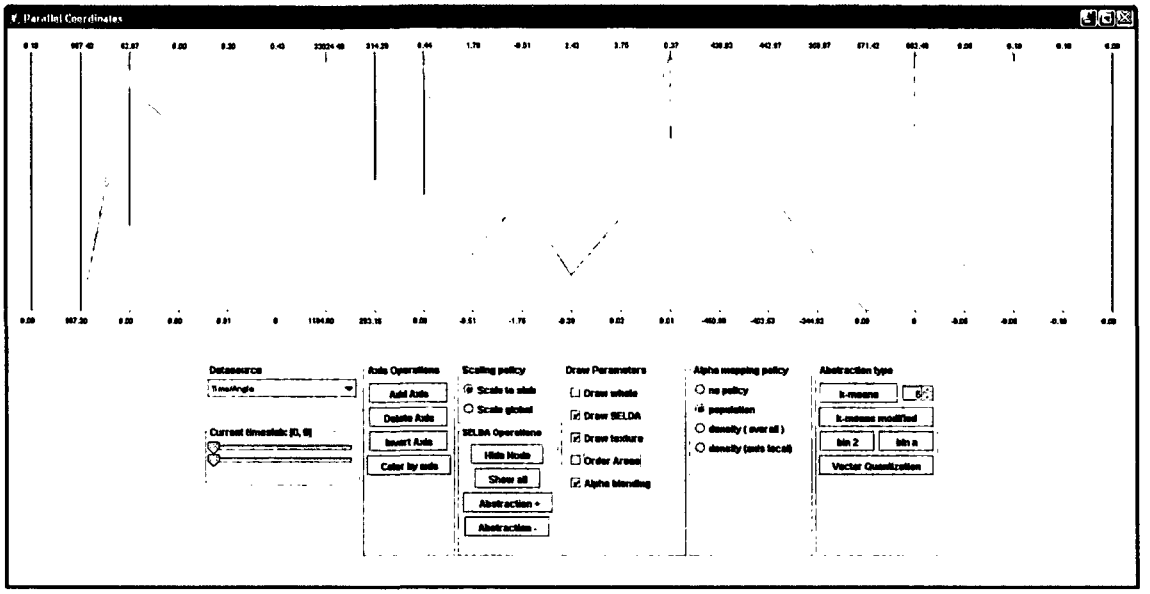


Figure 6.10: A parallel coordinates view showing 6 clusters of a data set in 23 data dimensions.

(see figure 6.9 for an example histogram view).

We implemented *TimeHistograms* as described in Kosara et al. [103]. TimeHistograms offer special support for time-dependent data, like visualization of trends or three-dimensional histograms with time as third axis (see also next chapter for sample figures).

Parallel Coordinates

A parallel coordinates view has been also included in the SimVis system, allowing interactive visualization of the multi-dimensional character of data items. As parallel coordinate views are problematic (at least for interactive visualization of large data amounts), we included a clustering process, which then can be used to interactively visualize the clusters (and properties of them) in the parallel coordinates view (see figure 6.10 for an example).

3D View

The 3D view is, as opposed to the InfoVis views, an extended SciVis view. It shows the distribution of the data points in 3D space but also gives hints about their position in data space via customizable color coding (e.g., the temperature range of points can be mapped to a color gradient from green to red).

Since continuous DOI functions are supported, the opacity of the displayed points is modulated by the corresponding degree-of-interest values. Data points with high DOI values are drawn opaque while data points with a low DOI value are shown more translucently, discriminating data *in focus* from *context* data.

In figure 6.11, a 3D view shows regions of hot and fast flow in a diesel exhaust system [38] (see also section 7.1).

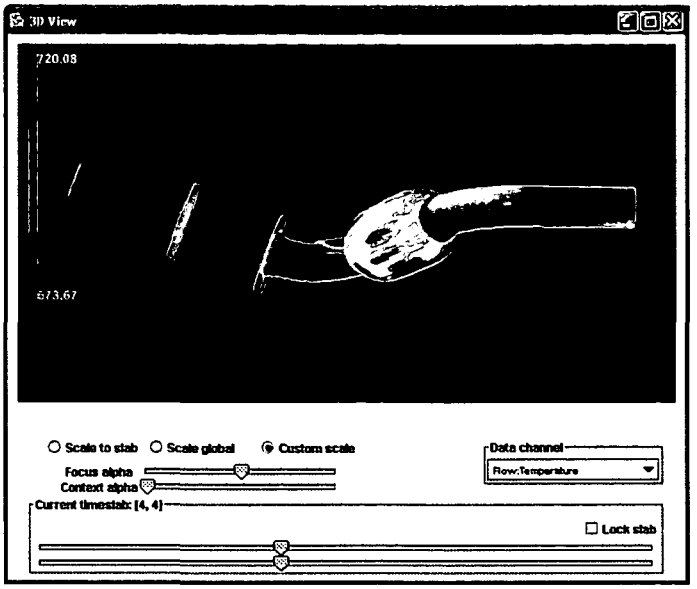


Figure 6.11: A 3D SciVis view, showing regions of hot and fast flow in a diesel exhaust system [38].

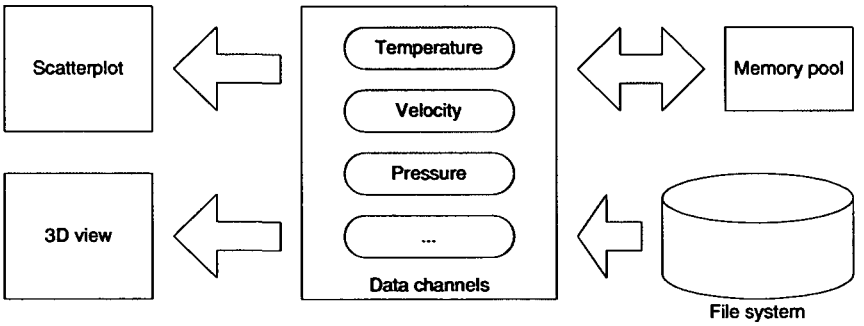


Figure 6.12: Overview of the data access layer in SimVis.

6.3.3 Data Access Layer

The data access part of a visualization application is of particular importance for the efficient operation of the software. Figure 6.12 illustrates the main system components of the SimVis system and the data flow between them.

In our data representation, data from one attribute and for each time step is stored in a *Channel*. The channel class encapsulates an array of data values and provides bounds-checked accessor functions for those values. Multiple *Channels* (representing one data attribute for multiple time steps) can be collected in a *TimeChannel*. A *TimeChannel* therefore organizes individual time steps, represented as channels. Basically, one data dimension can be associated with a *TimeChannel* (see figure 6.2).

Channels are implemented following a *Virtual Proxy* pattern [53], making transparent lazy loading (*activating*) of data possible. Figure 6.13 illustrates the associated components for the data activation process.

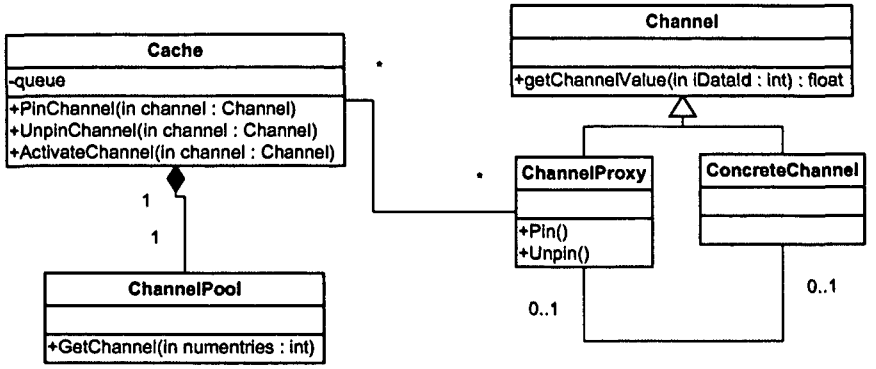


Figure 6.13: The classes associated with the caching functionality.

Due to the fact that it is impossible (or at least very expensive in terms of system resources) to leave all channels activated, a caching algorithm based on a LRU queue has been implemented. The following requirements were the cornerstones for the design of the caching subsystem:

- Minimize the use of the page replacement system provided by the OS
- Minimize the number of harddisk read accesses
- Minimize allocation/deallocation of large memory blocks

Activated *ChannelProxies* are stored in a global queue, with the last accessed proxy always at the head of the queue. Whenever a *ChannelProxy* is accessed, the caching layer ensures that it is either moved to the front of the LRU queue or activated and then inserted at the front of the queue. If the channel has been already activated, no change is necessary.

The activation function itself tries to get the least recently used *ChannelProxy* from the queue and reuses its *ConcreteChannel*. If the channel stores read-only data from a file, the *Cache* can just reload the data if it is needed. Generated data (e.g., data that has been derived from existing data by applying a filter) is swapped out to another file if needed.

If the queue is empty – because all data channels are currently in use – the caching layer allocates an empty *ConcreteChannel* from the *ChannelPool*. The acquired *ConcreteChannel* is assigned to the *ChannelProxy*, which is then loaded with the requested data. In the following, the caching algorithm is sketched:

```
if (not QueueIsEmpty()) {
    LRUProxy = RemoveLastQueueItem();
    SwapOut(LRUProxy);
    EmptyChannel = GetChannel(LRUProxy);
    AssignChannel(aProxy, EmptyChannel);
}
else {
    EmptyChannel = GetChannelFromPool();
    AssignChannel(aProxy, EmptyChannel);
}
SwapIn(aProxy);
```

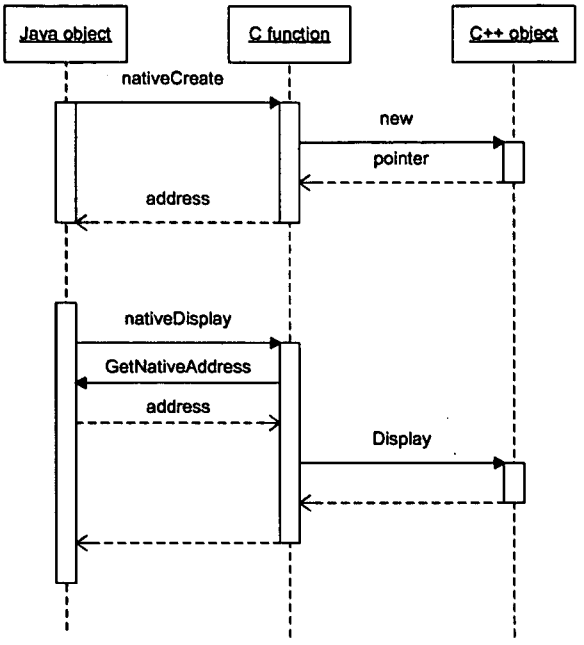


Figure 6.14: Binding of Java to native code.

6.4 Implementation Issues

The proof-of-concept implementation of the SimVis system has been realized in C++ (Visual C++ .NET) and Java (j2sdk 1.4.2). We combined the advantages of C++ – mainly the high optimizability of the code – and the advantages of Java – rapid user interface design – as good as possible.

Essentially, all performance-critical parts are implemented in C++. These parts include low level data access, OpenGL rendering, DOI calculation, and data filtering. The user interface components are implemented in Java.

Java-native Binding

The binding of native C++ object instances to their corresponding Java objects is implemented with JNI [84] and an adapted proxy [53] pattern. All Java objects that have a native counterpart, act as proxies by just forwarding requests and caching values for succeeding read operations.

Basically, our concept is simple: A Java object stores the address of its corresponding C++ native object. Whenever a method of this object is called, another method with the same signature, but marked as *native*, is called; this method is implemented in plain C (since JNI is a C API). In this C function, a pointer to the corresponding C++ native object is constructed from the address stored in the Java object and the native object is accessed by first casting the untyped address to the native object type and then calling the member function. The obvious drawback of this solution is the large coding overhead for the C wrapper functions (however, this could be easily avoided with a code generation utility). In figure 6.14, the functionality of the Java/C++-binding is shown in the form of a sequence diagram.

Since it was a requirement to use OpenGL for the rendering part of the views, we needed a way of how to natively render to Java UI components. Since the JNI API did not expose native UI handles from the AWT before the arrival of JDK1.3, the first prototype of the system used the GL4Java API [57]. Due to the development stop of this OpenGL implementation we soon decided to move to a less proprietary solution. The JAWT interface (Java Abstract Windowing Toolkit native interface) introduced calls that returned handles to native windowing resources with the release of JDK1.3, which made it possible to create OpenGL rendering contexts [7].

6.5 Performance Optimizations

In the following we present a couple of performance optimizations which have been implemented. We differentiate the discussion into three parts: *high level optimizations*, *algorithmic optimizations*, and *low level programming optimizations.*

6.5.1 High Level Optimizations

The first version of the software was strictly single-threaded, resulting in a blocking behavior in certain cases as 90% of the computing time was spent in the calculation of the DOI values and mouse interaction continuously triggered a recomputation of the DOI values. Large data sets left the system completely unusable because mouse interaction was impossible.

Our solution to maintain an interactive response of the user interface elements (without blocking), and thus enable interactive changes of feature specifications, was to decouple the DOI computation from the user interface by using two threads. The following pseudo code fragment illustrates the algorithm:

```
function Recompute(Node) {
    CancelRunningThreads();
    SetPathToRootDirty(Node);
    RunInThread(RecomputeRec(Root));
}

function RecomputeRec(TreeNode) {
    for each (child of TreeNode) {
       if IsDirty(child) then
           RecomputeRec(child);
    }
    ComputeValues(TreeNode);
    SetClean(TreeNode);
}
```

During the recursive depth-first traversal of the feature specification tree, the DOI computation thread notifies the registered views after finishing the computation of each completed node to update the viewing information accordingly.

6.5.2 Algorithmic Optimizations

An additional optimization came with the algorithm in the previous section: Computation is further reduced by recalculating only tree paths of the DOI hierarchy, which are affected

by recent user interactions (only the path from the updated node to the root is set dirty, see figure 6.6).

Additionally also the actual calculation of different DOI values is optimized. The fundamental DOI functions in the simple selections are computed by calculating the value of the trapezoidal function for each data value in the corresponding data dimension. A straightforward implementation would loop through all the values and calculate the DOI value for each of them. Since the trapezoidal function can be divided into 4 regions we can split up the algorithm according to these four regions. In a preprocessing step, we sort the data along each dimension v_k. This can be efficiently done with double-indexing via a permutation array p_k, so that $v_k[p_k[i]] \leq v_k[p_k[i+1]]$ is true for each dimension. Now we can perform binary search on each data dimension and find the indices of the bounds. Then we can construct the trapezoidal function with two linear interpolations and one constant period.

```
Brush(DOI, {lowerSoft, lower,
                upper, upperSoft}) {

    IndexLowerSoft = FindFirstGreaterThan(
        SortedData,lowerSoft);
    IndexLower = FindFirstGreaterThan(
        SortedData,lower);
    IndexUpper = FindLastSmallerThan(
        SortedData,upper);
    IndexUpperSoft = FindLastSmallerThan(
        SortedData,upperSoft);

    InterpolateZeroToOne(
        DOI,IndexLowerSoft,IndexLower-1);

    SetToOne(DOI,IndexLower,IndexUpper);

    InterpolateOneToZero(
        DOI,IndexUpper+1,IndexUpperSoft);
}
```

6.5.3 Low Level Programming Optimizations

Here we shortly report on two low level programming optimizations that have been included in the current version of the SimVis system: the first one concerns again the *DOI computation*, which is a major factor concerning performance of the interactive feature specification, and the second one deals with *3D rendering* optimizations for the visualization in the 3D SciVis view.

DOI computation

Due to the fact that the DOI computation subsystem is based on componentwise vector arithmetics, it was possible to highly optimize that part of the software by taking into account standard optimization techniques (loop unrolling, code inlining) as well as special features of the hardware.

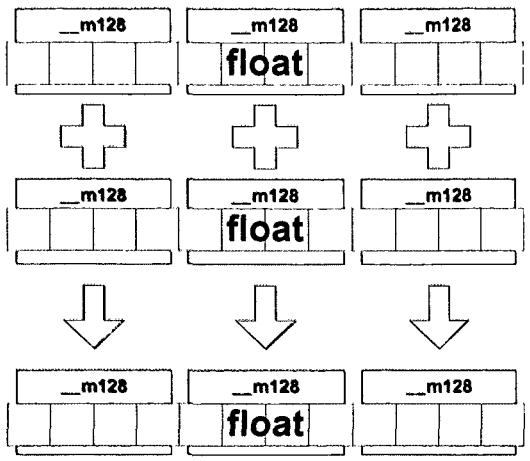


Figure 6.15: Usage of SSE: always four floats are packed into one operation.

	No SSE	SSE
Without HT	10.36ms	2.81ms
With HT	8.45ms	5.08ms

Figure 6.16: Comparison of SSE and HT optimization for the DOI calculation.

Intel processors from the Pentium III upwards support a SIMD (Single Instruction, Multiple Data) instruction set (Streaming SIMD Extensions, SSE [185]), which is – like its predecessor MMX – optimized for the operation on four-dimensional vectors, but is not limited to integer data (as MMX is). SSE introduces a set of 8 additional 16 byte registers to the CPU architecture. Each of these registers can take a vector of 4 float values, which are then addressed with a special set of instructions. We only used additions, multiplications and min/max operations. By processing the DOI arrays in chunks of 4 floats (see figure 6.15), it was possible to optimize the hierarchical computation of the DOI function.

On the other hand, the Pentium IV also supports the *hyperthreading* technology [78] which offers thread-level parallelism, enabling multi-threaded applications to make better use of CPU resources as compared to single-threaded applications.

We at first tested both solutions (SSE and HT) for their suitability in the special case of calculating combinations of data coming from different large float arrays (as it is the case for the performance-critical parts of the DOI calculation). As a benchmark we used the following case: the minimum T-norm (logical AND) of arbitrary DOI values from two arrays with 50,000 entries each is calculated (similar to the combination of brushes in SimVis [33]). Figure 6.16 gives averaged results for multiple tests with the different setups (with and without SSE, with and without hyperthreading). Using (only) the SSE instruction set gives the maximum speed-up for our case. When adding hyperthreading (by using different threads for calculating parts of the combinations, for example), no further speed-up can be achieved. Compared to using only SSE extensions, the increased cache miss numbers due to using different threads even slow down computations significantly in most cases. We therefore optimized the calculation

of the DOI function based on the SSE extensions. The reason for the near-optimal benchmark results for SSE (see figure 6.16) is presumably the linear layout of the data that fully exploits cache coherency and thus drives the CPU at full load.

3D rendering

Due to the large amount of points to be rendered for one time step, simple immediate mode rendering of GL_POINTS is too slow. Through the employment of *Vertex Arrays* it is possible to render mesh geometries with 1.000.000 geometry cells and more at interactive frame rates. We also used the *Vertex Buffer Objects (VBO)* extension [201] that provides fine grained control over the vertex array memory management and allows for tuning of the caching policies for the vertex/color/index data arrays according to their update frequencies.

Since we modulate the opacity of the rendered fragments with the DOI function, the hardware z-buffer cannot be used and rendering in back-to-front order is necessary to support hardware based alpha blending. The sorting is performed with an hybrid QuickSort/HeapSort algorithm that has running time $O(n \cdot log(n))$ in the worst case (IntroSort, see [138]). Although the sorting performance decreases when very large geometries are rendered, data sizes of up to 250.000 geometry cells can be rendered interactively ($\sim$ 15 frames/second) with this algorithm. Of course, resorting is only necessary and performed when the viewing direction changed. Therefore, the sorting algorithm has been evacuated to a separate thread, and thus progressive sorting is enabled. If the user interaction does not allow sorting in real-time, the correct sorting results are provided after the user interaction has stopped, and the calculations have been finished by this separate thread.

6.6 Converting Data

SimVis employs its own data format throughout the whole system (see also section 6.2 for more details). This allows using all visualization techniques included in the SimVis framework for different data sets having different types of data formats. The interface between external data formats and our internal data format has to be provided via a data format converting tool.

In the current version a data converting tool for data from our industrial partner company AVL List GmbH [6] is available. This converting tool is presented in the following. Providing converting tools for other similar data from CFD simulation is rather straightforward, and therefore simple to include. Converting other, more abstract data formats will also be possible in the future.

All data from CFD simulation from our partner company AVL are calculated using SWIFT [192] or FIRE [46], AVL's computational fluid dynamics (CFD) software packages. The grids used for the discretization of the flow domain are generated with AVL's automatic mesh generator, called Advanced Fame Hybrid [43].

The converting tool for the resulting data sets works as follows:

1 The path to the file holding the mesh data has to be chosen in the first line of the file dialog window (see figure 6.17).

2 Next the path to the file holding the CFD simulation results has to be specified in the second editing field of the dialog window.

3 In the third editing field the default name for the output file where the converted data shall be stored to, is displayed and can now be altered by the user.

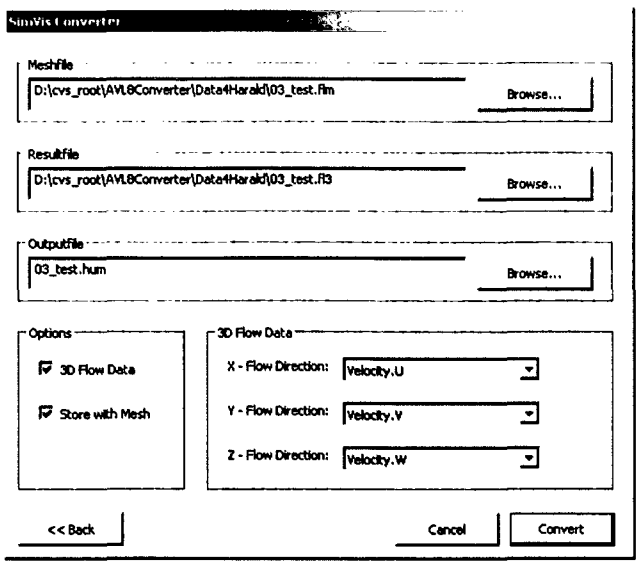


Figure 6.17: GUI for the data converting tool for AVL data sets.

4 In the bottom of this user dialog, additional options for the converting process can be selected: a checkbox should be disabled, if not 3D flow data results are to be converted (which is very rarely the case in AVL data); if the data is 3D flow data, the three attribute names of the velocity components for the flow have to be specified (if different from the default values displayed); another option is to disable to convert any mesh information, thus converting only the abstract data per mesh cell without any spatial information.

After the specification of all these values and options, the converting process is started. Converting the data can take a few minutes, especially for larger data sets, depending on the hardware of the computer where the conversion is carried out. Crucial thereby is a large amount of memory, to reduce IO costs for swapping parts of the data by the system.

6.7 Known Limitations and Future Work Plans

The current version of the SimVisBox as presented and discussed here has of course a few known limitations. These, together with an outlook of some future work plans and directions, are discussed shortly in this section, now.

Known Limitations

Besides all the support for visual analysis of simulation data as offered by the different methods available in our SimVis system, also a few limitations are known, which have not been solved up to now. These limitations include:

- The already mentioned limiting factor from section 6.1: due to memory limitations, currently the number of cells N_c times the number of time steps N_t in the resulting data set must be below (approximately) 15.000.000.
- Most attribute derivations included in the current version of the software can only work with time-dependent data based on steady grid geometry/topology. Only the

calculation of normalized data attributes (normalized per time step) is possible on grids with varying geometry/topology.

- In the current version of the SimVisBox, measuring tools are missing. Such measuring tools (measuring distances of different points in space, distance to the boundary geometry, measurements of flow integration over time through defined cross sections, etc.) are often very important to allow a complete analysis of data resulting from CFD simulations.

Future Work Plans

Future work plans include finding solutions to the known limitations as mentioned above (make larger data possible, allow attribute derivation for data based on time-varying grid geometry/topology, include some standard measurement tools in SimVis), but also some further interesting and important improvements. A few of them include:

- How to deal with missing and/or uncertain data values? Currently there is no strategy of how to plot data items in a scatterplot, if one of the two attributes, that shall be plotted against each other, is not available for the current time step, for example. Also techniques of how to convey the degree of uncertainty for a data item with respect to the two data attributes, that shall be plotted against each other shall be researched. There are multiple other challenges, when dealing with missing data values or data values having an uncertainty value associated with them (how sure is the result of the simulated data, etc.).
- How to better visualize non-numerical or categorical data (data with only a small number of possible values) in some of the data attributes. One example in CFD data for a categorical data dimension is the temporal dimension. Only a (usually) small number of discrete values of different time steps are available in the data. When plotting one other data attribute (e.g. velocity) vs. the time steps in a scatterplot, normal plotting results in cluttering of data points along lines (of distinct, individual points in time).
- Currently the feature representation is very basic. More advanced representations shall be researched, leading also to a link to other, well known feature detection and representation techniques. First spatially connected parts of features will be computed. An additional panel will provide more information about these detected feature components (which are connected in space, per time step). Properties like relative or absolute volume of the connected feature regions are displayed, individual cells belonging to each connected feature region could be identified and displayed in the form of a linked table view. Also manual tracking of connected feature regions by specifying a feature sequence over time (identifying the connected feature regions of each time step, that belong to each other) could be installed.
- Better suitable visualization techniques for presentation (especially for the 3D view) are required. This would help to increase the acceptance of analysis results by managers, who decide on product development based on the results of simulation, which have been produced by engineers. Techniques including integration-based flow visualization methods (streamlines, texture-based techniques, etc.) or surface- and volume-based methods, possibly also in clever combination with the feature specification results (represented in form of a DOI attribution) would definitely improve the illustrative character of visualizations as needed for presentation purposes.

Chapter 7

Case Studies

This chapter of the thesis presents two case studies which have been investigated and worked out with the visualization framework as presented and discussed in the previous chapters. Both cases together show the powerful opportunities of our approach, especially for the purpose of interactive exploration and analysis of results from CFD simulation.

The first study (section 7.1) presents the analysis of a comparison of two related cases of a diesel exhaust system for passenger cars, where some important questions about these two different cases are addressed [38]. This case study has been presented at the 6^{th} Joint IEEE TCVG – EUROGRAPHICS Symposium on Visualization (VisSym 2004) [38].

The second study (section 7.2) presents another case study, where two versions of a diesel engine for heavy trucks are analyzed and compared [37]. This case study includes the handling of data sets based on time-varying grid geometries and is currently in submission.

7.1 Visual Analysis of Complex, Time-Dependent Simulation Results of a Diesel Exhaust System

In this section we present a case study describing an interactive analysis session with our visualization framework, as presented in the previous chapters of this thesis. We show, how the here presented visualization techniques provide engineers with a high degree of interactivity and flexibility for analyzing simulation results. The results of this case study demonstrate how users benefit from using these techniques during their routine analysis, as well as for exploring new phenomena. A comparison of two related cases of a diesel exhaust system for passenger cars is presented, and some important questions about these cases are addressed.

In general the analysis of data which results from computational simulation is both, challenging and complex. Additionally it is also important to speed up simulation-cycles, which leads to shortening design and development times of new products. An interesting application field where computational simulations play a major role for innovative developments is in simulating catalyst processes in a diesel exhaust system in the automotive industry.

In the previous chapters we have presented our framework for interactive visual analysis of CFD simulation data, called SimVis. We have shown that combining methods from SciVis and InfoVis provides powerful possibilities for visualizing multi-dimensional simulation results computed on 3D grids. In contrast to our system, common commercial systems for post processing and visualization of simulation results usually provide only views of 2D slices or surfaces and very limited interaction possibilities. Three different types of tools are mainly

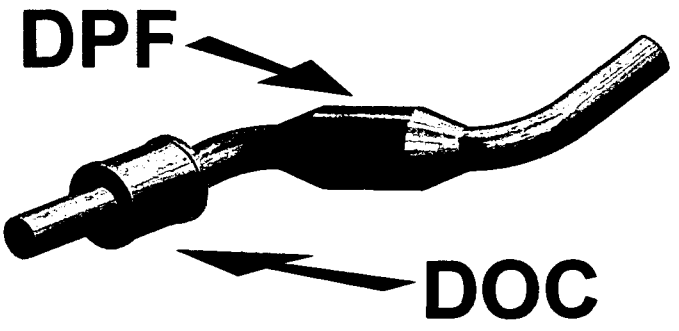


Figure 7.1: A diesel exhaust system consisting of a diesel oxidation catalyst (DOC) and a diesel particulate filter (DPF).

used: planar cuts, views of surfaces or iso-surfaces onto which specific attributes are color-mapped.

7.1.1 Application Scenario

The application presented in this case study is a diesel exhaust system for passenger cars powered by diesel engines. We now shortly describe such a diesel exhaust system, and afterwards discuss several important questions of engineers who work on the development of such a system.

Diesel Exhaust System

For passenger cars powered by diesel engines, layout and design of the *diesel oxidation catalysts* (DOC), used to reduce hydrocarbons and CO emissions, are standard applications. Because of decreasing legal limits for particulates, *diesel particular filters* (DPF) are in the focus of recent research [32]. In this case study a diesel exhaust system (see figure 7.1), consisting of a diesel oxidation catalyst and a diesel particulate filter, is analyzed.

The DOC is still the only catalyst technology that demonstrates the required robustness and durability and has been commercially established in a number of light- and heavy-duty applications. At sufficiently high exhaust temperatures, diesel oxidation catalysts can provide very effective control of hydrocarbons and CO emissions, with reduction efficiencies in the order of 90%.

The emission reductions in the DOC occur through chemical oxidation of pollutants, which can be described in the following chemical reactions (see also figure 7.2):

$$\mathrm{CO} + \tfrac{1}{2}\mathrm{O_2} \rightarrow \mathrm{CO_2} \tag{7.1}$$

$$\text{hydrocarbons} + \mathrm{O_2} \rightarrow \mathrm{CO_2} + \mathrm{H_2O} \tag{7.2}$$

$$\mathrm{CO} + \mathrm{NO} \rightarrow \mathrm{CO_2} + \tfrac{1}{2}\mathrm{N_2} \tag{7.3}$$

$$\mathrm{H_2} + \tfrac{1}{2}\mathrm{O_2} \rightarrow \mathrm{H_2O} \tag{7.4}$$

$$\mathrm{NO} + \tfrac{1}{2}\mathrm{O_2} \rightarrow \mathrm{NO_2} \tag{7.5}$$

The DPF traps the diesel particulates (*soot*) of the exhaust gas in its filter material. Over time the collected particulates block the filter which would negatively affect the engine operation.

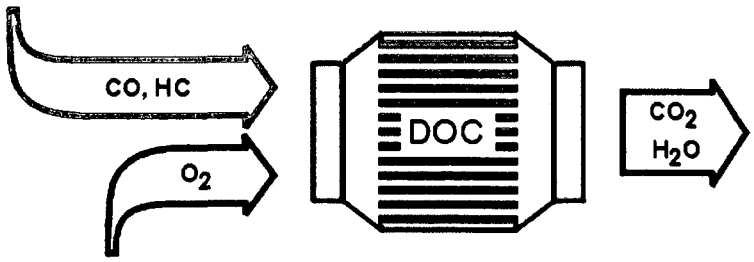


Figure 7.2: Emission reductions in the DOC [32]

Therefore, diesel filter systems have to provide a way of removing particulates from the filter periodically to restore its soot collection capacity (filter regeneration). This is attained by oxidizing the collected particulates at high temperatures to gaseous products, primarily CO_2. To achieve this high temperature in the DPF, the engine operates on rich conditions, which means that there is more fuel injected than necessary for a complete combustion. Due to the incomplete combustion, the amount of CO and hydrocarbons increases in the exhaust gas, which oxidize in the DOC and thereby heat the gas. This raises the temperature of the DPF itself and the oxidation of the particulates starts. This leads to additional heat in the DPF. To achieve a high efficiency, the temperature in the DPF should increase to more than 600°C.

The switching to rich engine operation conditions for filter regeneration should be performed by the engine management system without any impact to the driving behavior, such that this procedure should be transparent to the vehicle drivers. The goal of the filter regeneration is the complete oxidation of soot, which unfortunately causes a higher pollutant concentration of CO.

We analyze the results of the simulation of the diesel exhaust system during DPF regeneration. There are several advantages of simulation over real test applications. One is, that the simulation can be performed in an early phase of development. Improvements can be made before the first real application is built, which minimizes the time-to-market. Another point is, that checking devices can influence the application, e.g., a feeler gauge for temperature measurement of the exhaust gas in the DOC damages the DOC and changes the fluid flow. An additional advantage of simulation is, that all physical values for all calculated geometric points are available, while the measurement usually displays only one physical value at one (or a few) positions. By this account the simulation is also used for better understanding of dynamics processes.

All in this case study analyzed results have been simulated using SWIFT [192], AVL's computational fluid dynamics (CFD) software. These CFD simulations generate large amounts of data, including many different flow attributes like velocity, pressure, turbulence kinetic energy, temperature, etc. For the diesel exhaust system, with the DOC and the DPF, additionally an catalyst module is used, which simulates the equations 7.1 to 7.5. For the DPF the following chemical reactions are used for the simulation of the soot oxidation:

$$C_{soot} + O_2 \rightarrow CO_2 \tag{7.6}$$

$$C_{soot} + \tfrac{1}{2} O_2 \rightarrow CO \tag{7.7}$$

With this module the following data is available: the mass fractions of CO, CO_2, H_2O, C_3H_6, O_2, N_2, NO, and NO_2, and the mass of soot. Note that for simplicity of the simulation, C_3H_6

is taken to represent all hydrocarbons in the fuel. Furthermore, the heat transfer is calculated and, as a result, the temperature of the solid part of the DPF is available per cell.

Application Questions

During real testing of a diesel exhaust system only some questions can be answered, like the degree of conversion of pollutants and the temperature of the exhaust gases. But it is hard to analyze why something is happening. Therefore, computational simulation can help to find more answers. The main problem of analyzing simulation results is to handle the large amounts of data and displaying the data clearly.

The main goal of the regeneration of the DPF is a complete oxidation of soot. As during the regeneration phase of the DPF fuel consumption and emission of pollutants increases, the duration of this phase should be as short as possible. Therefore, the oxidation should be fast, which usually causes high thermal stresses, especially in the DPF.

From this goal (complete oxidation of soot) and the requirement of short regeneration times, the following application questions are of high interest to engineers developing such DPF systems:

1. *Does the soot oxidize completely?*
   And if not, what are the reasons?
2. *Where and how fast does the soot oxidize?*
   What influences the actual behavior of the oxidation?
3. *When is the regeneration phase completed?*
4. *How high is the thermal stress of the DPF?*
   Where do high thermal stresses occur?

To get an overview about the behavior of the DPF during the regeneration phase, different boundary conditions for the simulation are chosen. We here present two different cases with the following boundary conditions:
First case: At the inlet boundary the mass flow is 0.05 kg / s and the mass fractions of CO, C_3H_6, CO_2, H_2O, O_2, N_2, NO, and NO_2 lie in the typical range of a diesel engine which operates on rich conditions. The outlet is defined with static pressure. The wall temperature is fixed with 300°K. This is a rather cold diesel exhaust system. With these wall conditions a filter regeneration shortly after starting the engine is simulated.
Second case: Outlet and wall conditions are the same as in the first case. For the inlet boundary the mass flow is equal but the mass fraction of CO and C_3H_6 is 30% lower than in the first case.

7.1.2 Visual Analysis of the Diesel Exhaust System

To analyze the simulation results in the here presented application we employ SimVis for a visual analysis of two different cases of the simulation (see section 7.1.1). During exploration of the data sets, we identify four interesting application questions. By analyzing these questions we demonstrate the behavior of the DPF during this regeneration phase. Results of the visual analysis of these four questions are now presented. For larger versions of the images as presented here, as well as for some more results and additional animation sequences, please refer to *http://www.VRVis.at/vis/research/diesel-case/.*

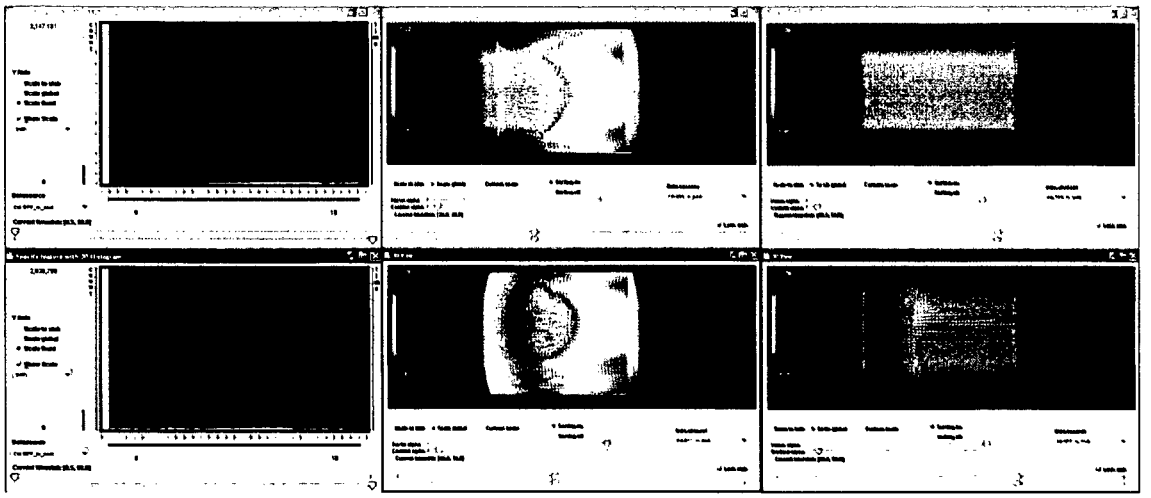


Figure 7.3: Visualization of soot mass in the DPF after 30 sec. (middle) and after 50 sec. (right). Histograms (left) and 3D views for both cases are shown.

1) Does the soot oxidize completely?

Complete soot oxidation is the main goal of the simulation of the DPF regeneration phase. In the case of incomplete soot oxidation, analysis of cause for this behavior are of highest interest. To analyze the soot oxidation process, first soot mass values are investigated. In two histograms, the soot mass values are displayed for all time steps of both cases (see section 7.1.1). A brush is performed in the upper histogram view (see figure 7.3), selecting all data items of case one having a soot mass value greater than 0 (selected data is colored red in SimVis InfoVis views). Note that we always show the views for the first simulation case (with more CO and C_3H_6 at the inlet) above the views of the second case for comparison of results throughout this section. To ensure correct comparison and analysis of the results of both cases, all feature specifications performed in the first case are saved and loaded again for the second case (using the FDL of SimVis), thus the same parameters for the brushes are guaranteed. In figure 7.3 two different time steps are shown in the 3D views in the middle and right, where soot mass values are color-mapped in the focus parts of the above described brushed regions.

As clearly seen, in the first case there is almost no soot left after 50 seconds, but in the second case the soot does not oxidize completely, there are still fractions of the soot in the front part of the DPF after 50 seconds. To explore the reason for that, the temperature values in the DPF are displayed for the features as defined above (non-zero soot mass). This is shown in figure 7.4 for 20 and 30 seconds after simulation start. As it can be seen for the first time step (after 20 seconds), the temperature in the first case is much higher than in the second one. The reason for that is, that more hydrocarbons and CO oxidize in the DOC in the first case due to higher C_3H_6 and CO mass fraction at the inlet. The oxidation generates more heat, which results in a higher fluid temperature. The warmer fluid heats the DPF and the oxidation of soot starts quickly. In the second case the incoming fluid has a lower temperature and the oxidation of soot is not as effective as in the first case. The soot oxidation produces heat too, which will be transported downstream along the fluid flow and heat the rear of the DPF (see figure 7.4, right side). As a result the effectiveness of the soot

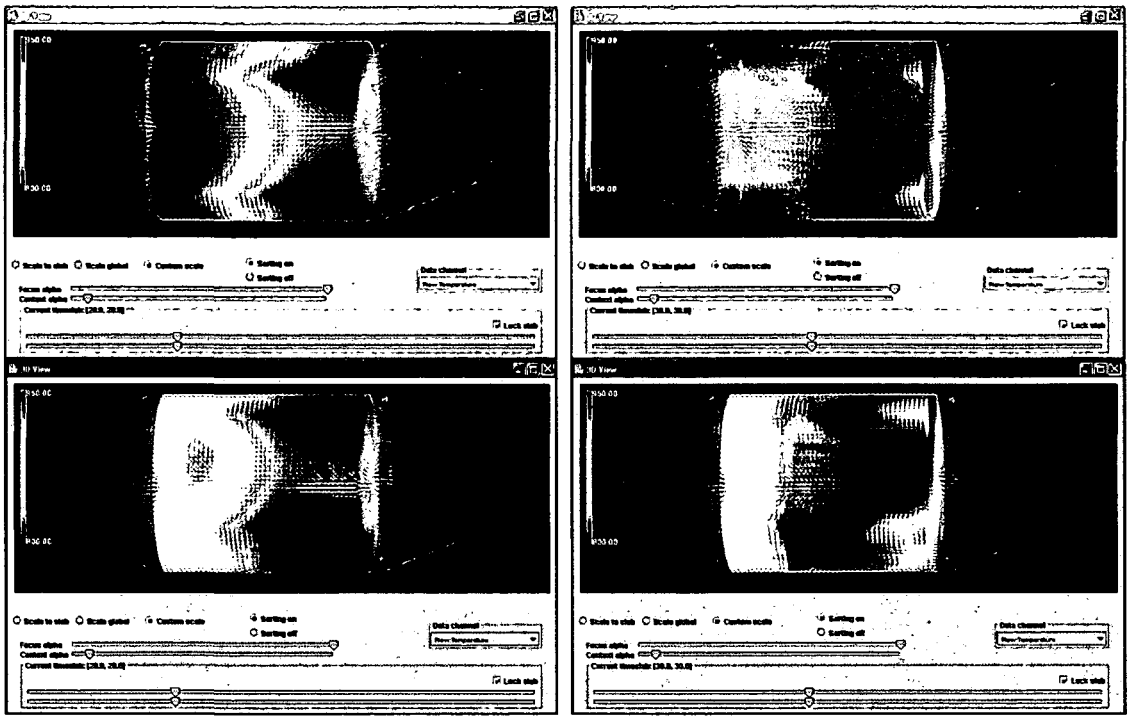


Figure 7.4: Temperature values in the DPF are color mapped to the features (regions of non-zero soot mass) after 20 sec. (left) and 30 sec. (right). Visualizations from case 1 (upper row) and case 2 (lower row) are compared.

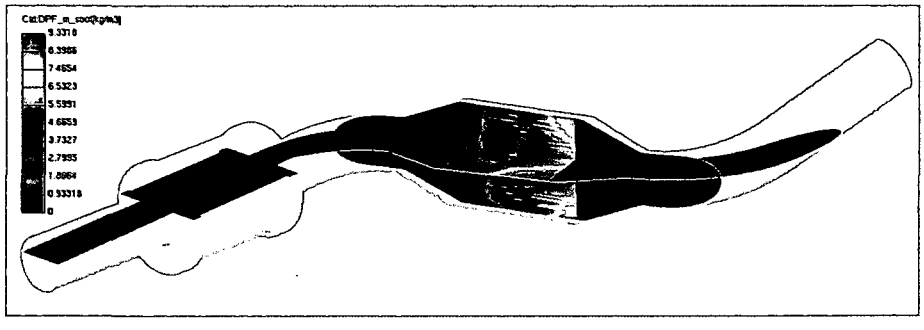


Figure 7.5: Typical visualization of soot mass in common commercial visualization tools.

oxidation increases in the rear, and the soot oxidizes completely in this region. To sum up the first question, a high mass fraction of C_3H_6 and CO is needed for a complete soot oxidation.

Analyzing this with a common visualization software needs many slices to get the same information, because of the three-dimensional behavior of the fluid flow and the chemical reactions. Figure 7.5 shows a typical visualization of the soot mass with a common visualization software for simulation results. With the displayed two slices it is not possible to get a fully three-dimensional impression of the soot distribution. Therefore more slices would be required. Additionally to the possibility to view the data in 3D in SimVis, interactive analysis through brushing (with immediate feedback in the 3D view) and working on multiple

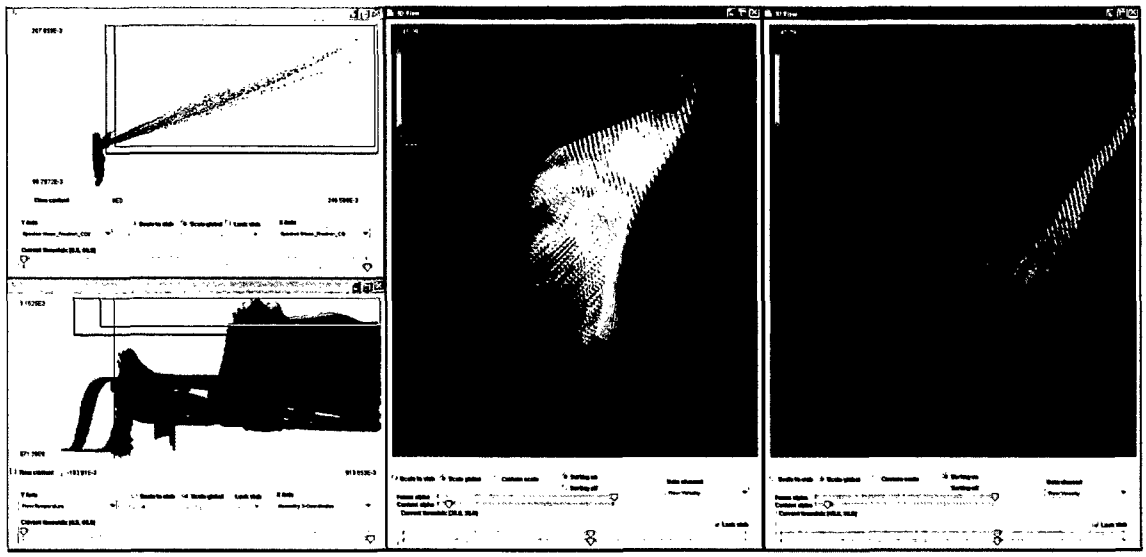


Figure 7.6: Velocity in cells with high CO and CO_2 mass fraction and high temperature after 35 sec. (middle) and 45 sec. (right).

dimensions simultaneously, improve and speed up the analysis process compared to common visualization software tools.

2) Where and how fast does the soot oxidize?

For this question only the second case is of interest because of the incomplete soot oxidation. Due to soot oxidation, CO_2 and CO are generated in regions of high temperature. To display these areas, it is necessary to specify a complex feature. In a first scatterplot the mass fraction of CO_2 (Y-axis) is plotted against the mass fraction of CO (X-axis). A smooth brush selecting higher values of both attributes is applied (see figure 7.6, upper scatterplot view). Then, in a second step, to refine the first 2D brush, a second scatterplot showing general fluid temperature (Y-axis) vs. the spatial X-coordinates of the data set (X-axis) is used. Note that the spatial X-axis of this data set correlates very well with the main flow direction through the whole data set (from left to right in figure 7.1). Here, high values of fluid temperature in the region of the DPF are brushed, leading to a 3D, composite brush. The two scatterplots as well as two different time steps in the 3D views are shown in figure 7.6. In the 3D views, velocity values are color mapped. Visualization shows, that the CO and the CO_2 production are not symmetric. As can be seen in the right 3D view, the velocities in this region are relatively small (colored green), which seems to be the reason for this asymmetric temporal behavior of the oxidation process.

For a better understanding, the oxidation progress has to be analyzed. This can be done by displaying only cells with a high mass soot gradient. As the amplitude of the gradient values changes over time, we need a method to select relatively high gradient values with respect to the maximum gradient value for each time step, to get interesting cells for each time step separately. Therefore differences of the soot mass values over time are calculated and normalized. With normalized differences we can easily select the upper 20% of difference values per time step, for example.

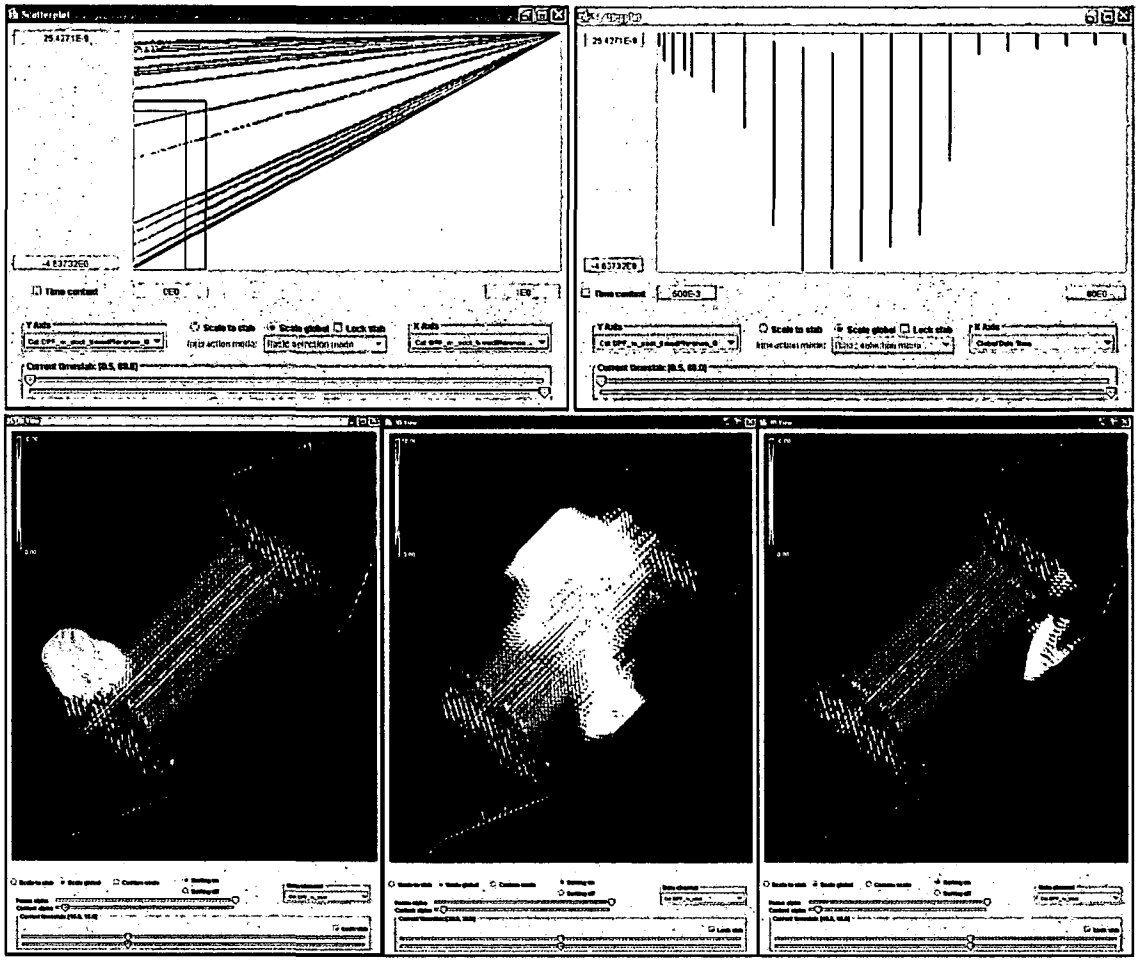


Figure 7.7: Soot mass in cells with high oxidation rate after 15, 30, and 40 sec. (from left to right)

After calculating the differences of the soot mass values and normalizing them, a scatterplot showing the normalized mass soot gradients (X-axis) and the mass soot gradients (Y-axis) is used. Low mass soot gradients and low normalized mass soot gradients are brushed in this scatterplot (see figure 7.7, upper left). Note that central differences are used for the approximation of gradients, which in the case of mass soot oxidation are negative, thus the lower gradient values are selected. For better illustration and better exploration a second scatterplot showing the mass soot gradients (Y-axis) and the time domain (X-axis) is shown in figure 7.7. Here the changing range of mass soot gradients becomes visible very intuitively. The vertical cluttering of data items results from the discretization of time on the X-axis, each streak denoting one time step in the temporal domain. For color mapping in the three 3D views, the mass soot values are taken into account.

From figure 7.7 it is clear, that the region with the highest soot oxidation rate after 15 seconds is not symmetric (left 3D view). The highest oxidation rate at the beginning of the soot oxidation is in the area with the highest temperatures (see also figure 7.4, left). The

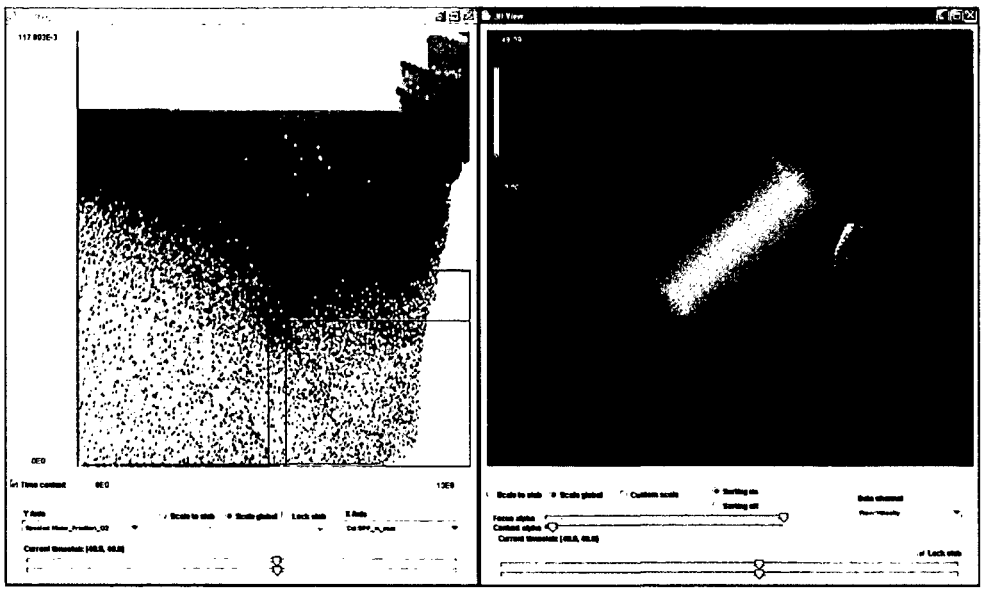


Figure 7.8: Velocity values in cells with high soot mass values and low mass fraction of O_2 after 40 sec.

asymmetric temperature distribution originates in the asymmetric flow field, due to the bend of the geometry between the DOC and the DPF. The amplitude of the oxidation rate is rather low at the beginning, which can be seen in the right scatterplot of figure 7.7. The asymmetric behavior is continued also at later time steps (see figure 7.7 middle to right). That means, that in the region displayed in the right most 3D view of figure 7.7 the oxidation starts later. This happens, although there seems to be enough heat for a fast oxidation (see figure 7.6, which shows the regions with high temperature). A possible reason for low oxidation rate is that there is not enough O_2 for oxidation. To check this, a scatterplot showing mass fraction values of O_2 (Y-axis) vs. the soot mass values (X-axis) is used (see figure 7.8). High soot mass values and low mass fraction values of O_2 are brushed, the color mapping of selected regions in the 3D view shows velocity values.

The area of high soot mass values with low mass fraction of O_2 is the same as the area where the oxidation starts later. So the low mass fraction of O_2 causes the later soot oxidation start. The reason for less O_2 is, that the velocity is low and thus only little oxygen can flow into this region. To answer the second question, the soot oxidation is executed asymmetrically, because of the asymmetric fluid flow.

Analyzing this question with common commercial visualization software would be quite tedious, because usually there is no possibility to display a combination of more quantities in one slice. Another advantage of SimVis is, that it is easy to define gradients or calculate other derived data attributes for all quantities and normalize them afterwards.

3) When is the regeneration phase completed?

The regeneration phase is completed, when there are no more chemical reactions. That means that the species mass fractions do not change anymore. A possibility to check this is to use histograms. For this purpose we can use the time-dependent 2D histogram as well as

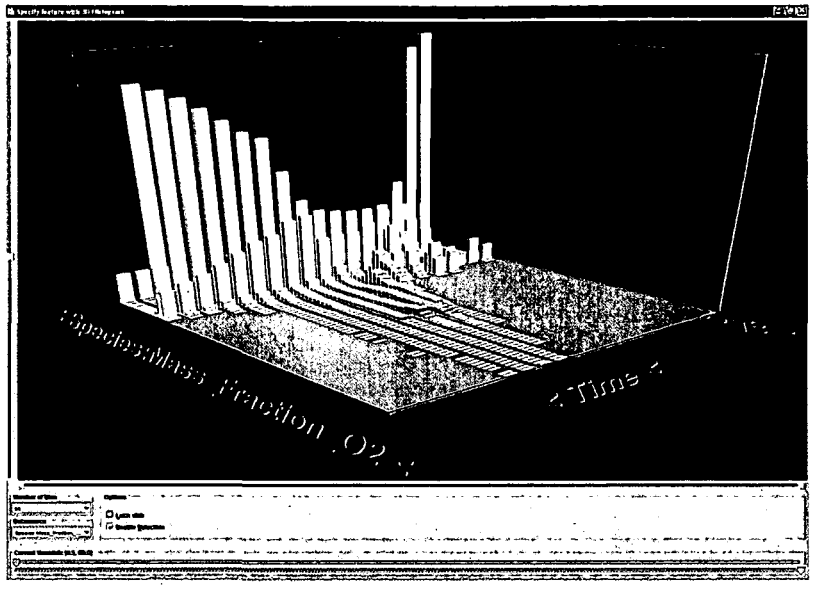


Figure 7.9: 3D visualization of a histogram showing the temporal evolution of mass fraction values for O_2.

time-dependent histograms in 3D (time vs. one attribute).

Figure 7.9 illustrates the number of cells with the same range of oxygen mass fraction over time. In this case the O_2 mass fraction is still changing after 80 seconds (which is the end time of the simulation for this case). This means, that there are still chemical reactions at this time. If we want to find the point in time, when the regeneration phase is completed for the cases simulated, we have to pursue the simulation for a longer time. Therefore we recalculated the simulation of case two with the regeneration phase assumed to last for 120 seconds. Two visualizations of this new case are shown in time-histograms [103] in 2D in figure 7.10.

Figure 7.10 shows 2D histograms of the O_2 mass fraction values (upper view) and the velocity values (lower view) with a global time context. This means that in addition to the normal 1D histogram (white vertical bars) showing data counts for time step 38 seconds after the simulation start, also in the background a texture visualizes the counts for each bar for every available time step. In this new time-histograms time runs vertically from bottom to top of the views, where in the background for each bar and each time step a lightblue rectangle is drawn. The opacity of each rectangle (thus the visibility) denotes the number of counts for this rectangle. The current time step shown in the 1D histogram is marked by a green bar (horizontally).

From these two views on the newly calculated data set, it can be seen, that the mass fractions of O_2 are nearly steady after 120 seconds, and even the flow field (the behavior of which lags a little behind the chemical processes) is nearly quasi-steady at this time. The time-histogram for the velocity values of the flow field shows the influence of the oxidation, which causes higher velocities. The oxidation starts approximately 15 seconds after the start of the regeneration phase, and has its maximum after 45 seconds. Then the influence decreases because of the decreasing oxidation rate (see lower view, figure 7.10).

To sum up the analysis of the third question, the chemical reactions are completed after

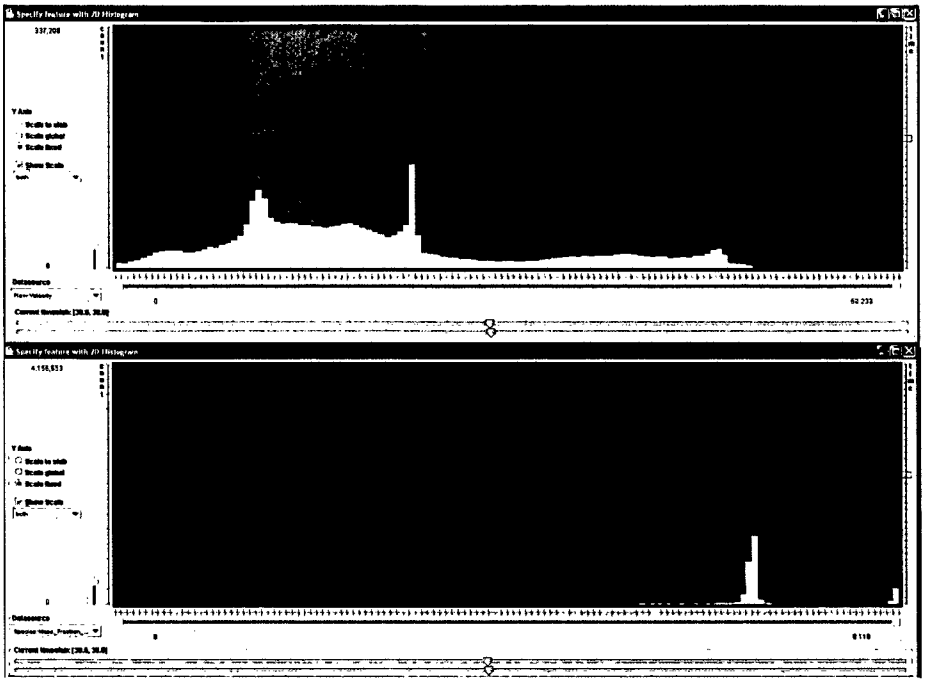


Figure 7.10: 2D time-histograms of mass fraction of O_2 and velocity values with time-context at background [103].

120 seconds. That means, that the regeneration phase needs at least 120 seconds. Displaying and analyzing this question with common commercial visualization software packages in use for simulation result analysis is usually not possible.

4) How high is the thermal stress of the DPF?

Here we again compare the two cases as described in section 7.1.1. One measure for thermal stress is the temperature amplitude of the DPF. For analyzing this, a scatterplot with the solid catalyst temperature (Y-axis) and the spatial X-coordinates of the data set (X-axis) is used. High values of solid catalyst temperature in the region of the DPF are brushed (see figure 7.11). As can be seen from this figure, in the first case the area with high solid catalyst temperature is bigger and the maximum value is higher than in the second case. So in the first case the thermal stresses in the DPF are higher than in the second case.

Additionally thermal stresses are caused by temperature fluctuation. Thus we search for regions with high heating or high cooling properties. To analyze this we open a new scatterplot, showing differences of solid catalyst temperature values (Y-axis) against temperature values of the flow (X-axis). Here we define two new brushes: high positive difference values of solid catalyst temperature give high heating factors, whereas high negative difference values of solid catalyst temperature represent strong cooling capacities (see the two brushes in the scatterplot views of figure 7.12). In the 3D views of figure 7.12, the green-colored regions represent areas exhibiting cooling gradients, whereas the red-colored regions show areas with heating gradients. As can be seen from the two time steps shown in figure 7.12, gradients of both, heating and cooling, are higher for the first case, thus thermal stresses are also higher in this case.

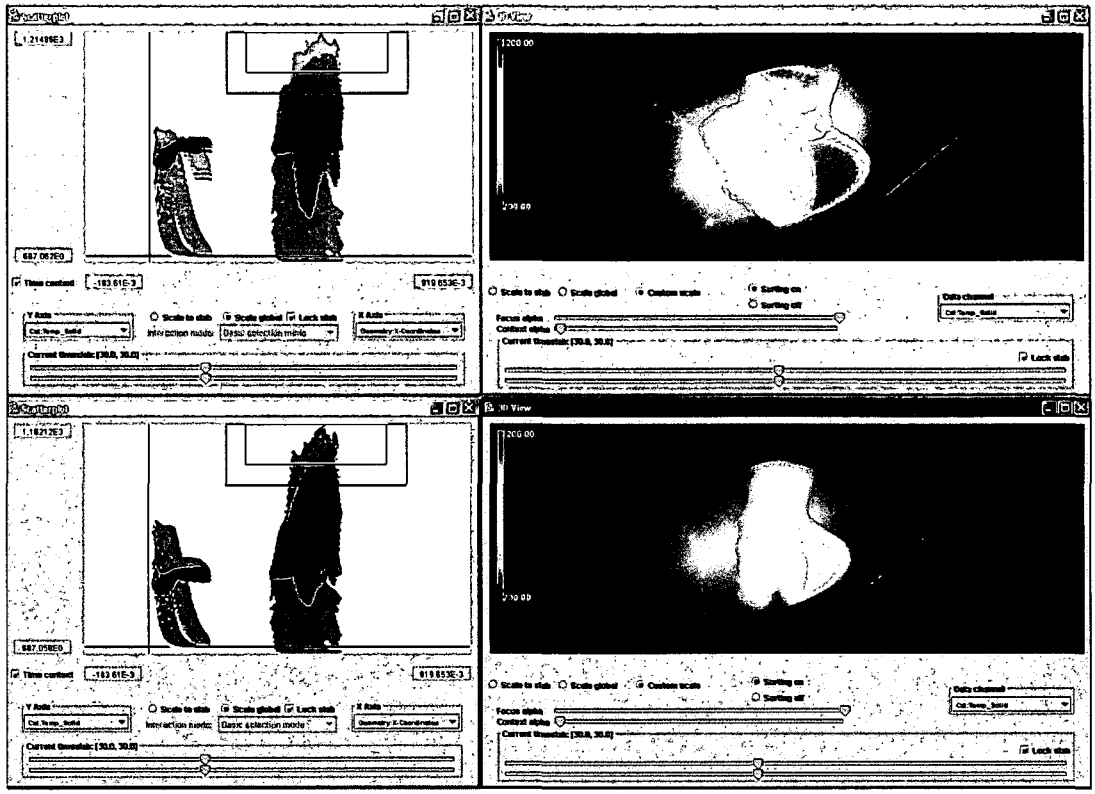


Figure 7.11: High solid catalyst temperature after 30 sec.

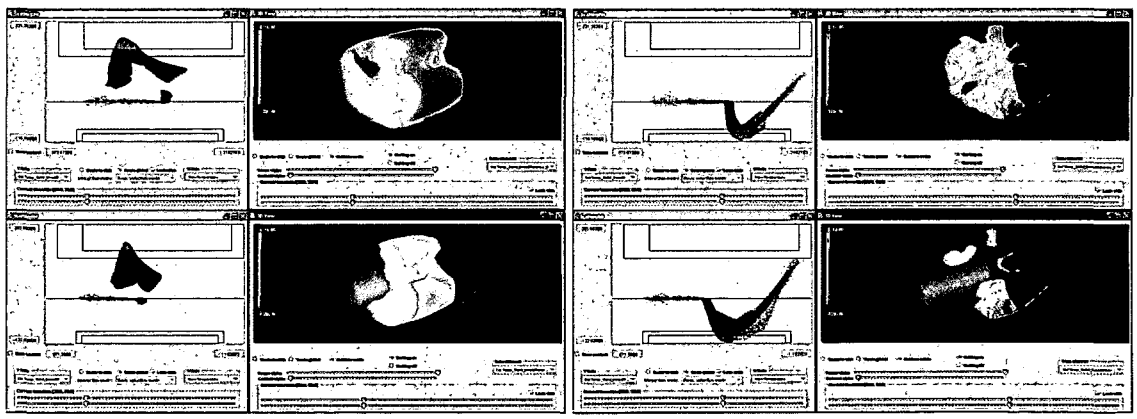


Figure 7.12: Time differences for solid catalyst temperature after 20 sec. (left) and after 50 sec. (right)

The analysis of this fourth question has shown, that the thermal stresses in the first case are higher than in the second one, due to higher heating gradients and higher solid catalyst temperatures in the DOC especially in the beginning of the regeneration phase.

Performance Issues

The presented case study was carried out on a system consisting of the following components: The hardware used was a standard PC-based system (Intel Pentium4, 2.8GHz, 2GB RAM, 2x 40GB HD, RAID 0 (Striped Set)) with a NVidia GeForceFX 5900 graphics card. The software is our Java-C++ based SimVis visualization environment for high-dimensional, time-dependent simulation data.

The two data sets investigated during this case study each consisted of approximately 260,000 cells with 37 attributes per cell. The temporal dimension of the simulation results is 0 to 80 seconds after the start of the DPF regeneration. We used 20 different time steps out of this interval for visualization and analysis. With these settings, loading all available time steps into a scatterplot, for example, requires drawing approximately 5.2 million points. Fortunately, due to algorithmic optimizations, and a simple but effective data handling framework, we are able to handle analysis sessions with multiple views for data sets even a magnitude larger than the ones presented here, while still providing full interactivity.

7.1.3 Conclusions and Lessons learned

In the course of this case study, we have shown how different features of our visualization approach are used to effectively bring forward the exploration and analysis of a diesel exhaust system. Complex application questions have been investigated, providing new insight into the data, which would not have been possible with standard technology (at least not in such an efficient manner). The here described system (SimVis) is especially useful when multiple data attributes are to be considered during the analysis of a specific question (e.g., why oxidation of soot is not symmetric). To actually work out this case study, SimVis needed to be extended by several means.

Lessons learned from this case study are that interactive visualization with the opportunity to interactively drill down into certain aspects of the data (through brushing as provided by SimVis) effectively supports the investigation of simulation data. Also the clear formulation of features in terms of the feature definition language proves to be very useful, not at the least when it comes to comparing different data sets side by side and porting of feature specifications from one data set to the other becomes handy. In comparison to standard analysis tools for simulation data, the visual linking between the visualization of attribute space (through InfoVis views) and physical space (through 3D SciVis) also proves to be very useful. Thereby it is easy and intuitive to answer questions like "where in grid do certain conditions (with respect to the data attributes) arise" or "in what way can certain subsets of the grid be characterized (in terms of the data attributes)", etc.

7.2 Interactive Feature Specification for the Visual Analysis of a Diesel Engine

In this section we demonstrate the application of our visualization approach to the analysis of CFD data from the simulation of combustion in a diesel engine. This is another interesting and recently very active application field for interactive analysis of unsteady flow data coming from CFD simulation.

In the following we describe how we extended and adapted the SimVis system to work out this case, e.g., by extending it to also analyze time-varying grid geometries and support polar

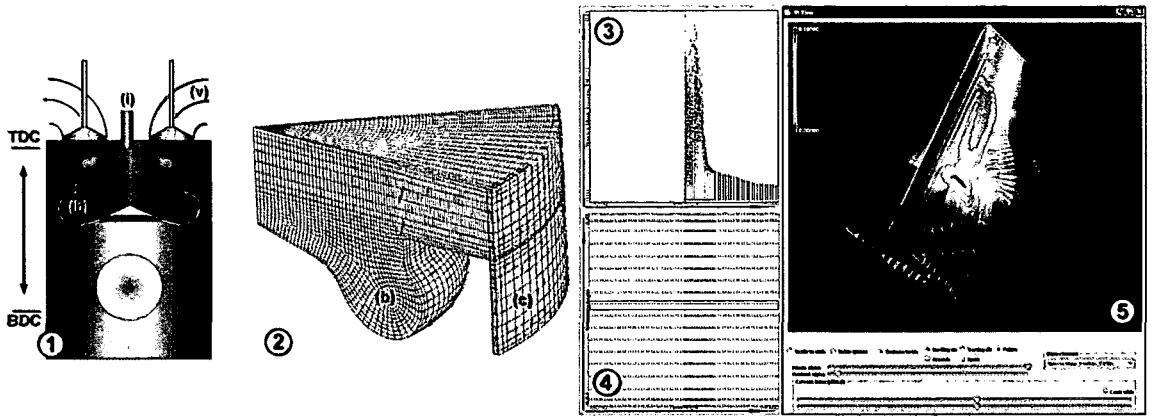


Figure 7.13: Visual analysis of the diesel engine FM538: geometrical layout (1) and underlying grid layout for the simulation of one time step (2); scatterplots showing the amounts of diesel vapor over time (3) and polar coordinates vs. time (4); amount of diesel vapor displayed on a sectional slice in focus in a linked 3D view (5).

coordinates. We especially focus on the description of how researchers and engineers use the SimVis approach to investigate their concrete user questions. We also assess the advantages and disadvantages of our feature extraction and visualization approach in the context of this industry-level application.

7.2.1 Application Scenario

The application presented in this section is the visual analysis of combustion in a diesel engine called FM538. We first describe this diesel engine and afterwards discuss the investigation of concrete questions of engineers who work on the development of diesel engines at our industrial partner.

Diesel Engine FM538

The FM538 is a single-cylinder diesel engine which is used by our industrial partner for research and development purposes (see figure 7.13(1)). The dimensions of the FM538 are that of a heavy duty engine as it is used in a truck. Engines like the FM538 usually are turbo-charged and work with direct diesel injection.

Figure 7.13(1) illustrates the geometrical layout of the diesel engine. The blue area represents the calculation volume for the simulation. The injector (i) with eight nozzles lies aligned with the cylinder axis and the shape of the cylinder head is assumed flat. The geometry of the intake valve (v) leads to the generation of a strong swirl flow (flow around the cylinder axis) which consequently leads to a proper mixing of fuel and air as necessary for an effective combustion process. For simplifying the simulation (and the underlying grid geometry), the generation of swirl flow through the intake valves is replaced by the assumption of a solid body swirl as an initialization of the simulation at the moment of the intake valve closing.

For the simulation rotational symmetry is exploited and only one of the eight injection nozzles is taken into account. The computational mesh therefore only covers a 45 degree cake of the combustion chamber. As the geometry of the cylinder head is assumed flat (the valve

pockets in the piston are neglected), a compensation volume (c) is placed at the cylinder outside (see figure 7.13(2)). This is necessary to conserve the correct compression ratio in the combustion chamber, given by the volume at the *Top Dead Center* (TDC, time step of upmost position of the piston) divided by the volume at the *Bottom Dead Center* (BDC) of the moving piston.

Another very important factor for the performance of the engine is the shape of the piston bowl (b). It is especially designed to conserve (or even increase) the swirl flow in the combustion chamber. During the compression stroke (from 180 to 360 degree crank angle (deg CA)) the rotating air is pressed from the cylinder into the bowl. As the diameter of the bowl is smaller than that of the entire cylinder, the angular velocity of the rotating load is accelerated, constrained by the conservation law for rotational motions. This acceleration of swirl flow is needed for a proper mixture formation during the fuel injection, which here starts at 360 deg CA (TDC) and ends at 396.5 deg CA. Due to the high pressure and high temperature level in the combustion chamber at TDC the fuel mixture ignites and combustion starts usually at about 1-2 deg CA after the injection starts. During combustion pollutants (NO_x) and soot are generated.

In the following sections, we analyze results of the simulation of this diesel engine from 180 to 540 deg CA. The simulation results of this case study have all been simulated using FIRE [46], AVL's CFD software for combustion processes. CFD simulation generates large amounts of data, including a multitude of different flow attributes like velocity, pressure, turbulent kinetic energy (TKE), temperature, etc. For the diesel engine, additionally a spray module, which is modeling the fuel injection, and a combustion module are used. The combustion is simulated by oxidizing the fuel to carbon dioxide and water vapor. Additionally the generation of NO_x and soot is calculated. With these modules the following additional data attributes are available as a result from the simulation process: Liquid density and mass of spray, the combustion reaction progress variable, the mass fractions of CO, CO_2, diesel, H_2O, H_2, N_2, O_2, NO_x and soot. The combustion reaction progress variable is a measure for the progress of the combustion process, defining how much fuel has already been burnt. If it reaches its maximum value of 1, all fuel ideally has been burnt.

Application Questions

During real-world testing of diesel engines only some questions can be answered, like what is the overall pressure in the combustion chamber or how much soot is in the exhaust gas. Usually it is hard to analyze why something is happening. CFD simulation can help to gain more insight into the physical and chemical processes inside the combustion chamber. The main challenge of analyzing simulation results is the need to handle large amounts of data and to display the data intuitively.

The simulation discussed in this section calculates a half cycle of a four-stroke diesel engine – from 180 to 540 deg CA. The simulation can be divided into three phases: the *swirl flow*, the *mixture formation*, and the *combustion*. Accordingly, we search answers to the following questions:

1. *Flow analysis during the compression stroke:* The simulation of the diesel engine starts at the moment when the valve is closed with a predefined swirl. Up to 360 deg CA (TDC) the upwards moving piston compresses the gas in the combustion chamber. During this period the turbulent kinetic energy, the velocity, and the swirl velocity are

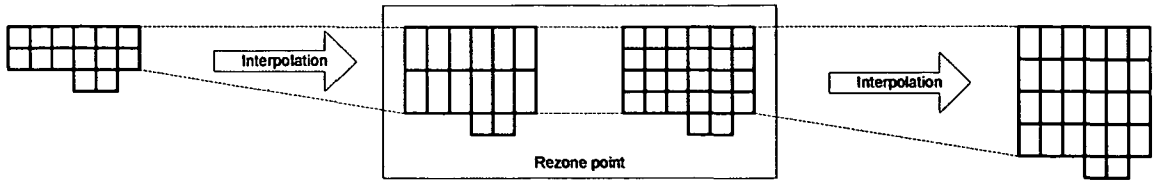


Figure 7.14: Illustrating the concept of rezone-points for time-varying grid geometries. At each rezone-point two topologically different grids are available, inbetween rezone-points grid-geometries are interpolated to each other.

of special interest. The swirl velocity is the rotational velocity component with respect to the cylinder axis. An important question arising during the analysis of this phase is: Is enough turbulence generated to yield a sufficient mixture formation?

2. *Analysis of fuel–air mixture formation:* The fuel injection starts at 360 deg CA (TDC) and ends at 396.5 deg CA. Several questions are important for analyzing the mixture formation during injection. How deep penetrates the injected fuel the combustion chamber? Does the liquid fuel evaporate properly and is the mixing between fuel and air ok?

3. *Analysis of the combustion progress:* After a sufficient amount of fuel has evaporated the combustion starts automatically through auto-ignition. Important questions during the analysis of the combustion phase include: Where are the regions, where fuel combustion remains incomplete and what are the reasons for this incomplete burning? How much soot is caused by the combustion process for a special setup, and what factors influence the amount of soot?

Here in this section we discuss the investigation of results from two different simulations of the same diesel engine for comparing different layouts. The two cases only differ with respect to the injection systems. The fuel spray at the nozzle outlet is cone-shaped. The experimentally determined cone angle depends on the geometry of the nozzle, the pressure of the fuel in the nozzle, and the pressure in the combustion chamber. For the injection system of the first case the opening angle of the cone is 20 degrees. For the injection system of the second case the cone angle of the spray is 10 degrees.

7.2.2 SimVis Extensions for Handling Time-Dependent Flow Data based on Time-Varying Grid Geometry

Before we actually discuss the interactive analysis of the diesel engine described above, we first briefly discuss the nature of the data, which is based on time-varying grid geometries. A categorization of time-dependent data according behavior of the underlying grid was already introduced in section 1.1.4 of this thesis. Especially in the automotive industry, simulation often has to deal with moving parts, and accordingly the computational grids adapt their spatial layout over time, as, for example, also in this case study.

The data, we are dealing with here, is divided in the temporal dimension into different *zones*. For each zone the topology of the underlying grid does not change, and linear interpolation is assumed for each grid position between two (topologically identical) grid layouts. These grid layouts are stored for each *rezone-point*. At a rezone-point 2 grid geometries are

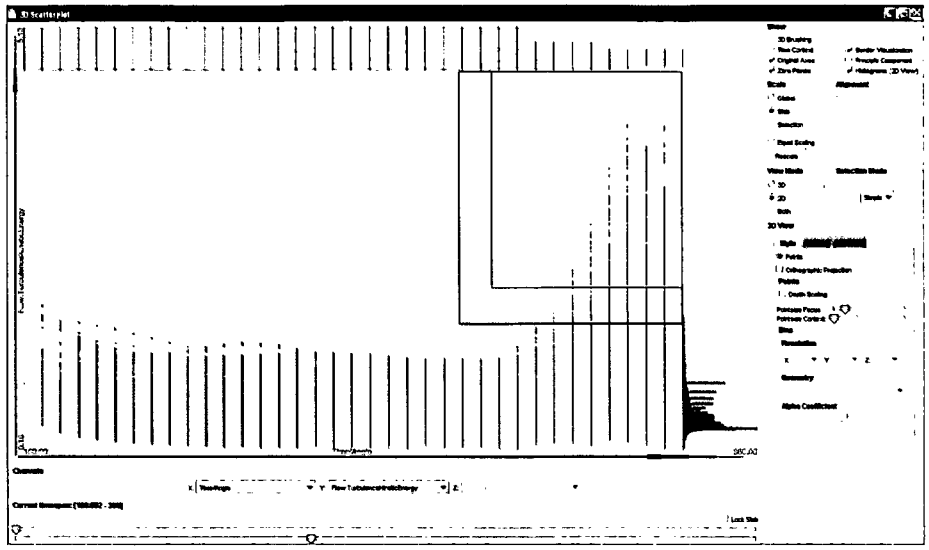


Figure 7.15: Interactive brushing of all data items which exhibit high values of TKE in a 2D scatterplot. Only data for the compression stroke (180 to 360 deg CA) is shown.

available, allowing a switching from one topology to another (see also section 6.2). During simulation, the interface (data exchange) between the two grid layouts at a rezone point is accomplished with interpolation.

The two simulation data sets of the diesel engine described here each consist of six different zones. Three zones are representing the compression stroke (180 to 360 deg CA), and three are used for the power stroke (360 to 540 deg CA). Figure 7.14 illustrates the concepts of rezone-points in the application described here. Through linear interpolation, the grid of the combustion chamber is stretched during the power stroke and has to be refined from time to time, to keep a useful grid geometry for the simulation.

When including data from time-varying grids into the SimVis system, the first challenge is to handle a different number of cells at different time steps. Especially in the 3D view, we not only have to deal with different numbers of cells but also with the fact, that cell positions change over time. Another challenge is that data set sizes of the grid geometries are generally a magnitude larger for data sets with time-varying geometry. In our case, not only the geometries at the rezone-points are important, but rather cell positions for each available time step in the simulation result are needed to allow interactive viewing in 3D as well as loading spatial references into the InfoVis views. We interpolate cell positions for each time step during loading of the data, connectivity information can be retrieved from the rezone-point geometries.

For the investigation of the application described here we extended SimVis also by supporting the use of polar coordinates, especially useful for data sets made up of curved, round, axis-aligned geometrical subparts, as is the case for diesel engine applications. Thereby sectional subsets of the rotationally symmetrical data set can be brushed interactively for further investigation (see figure 7.13(2)).

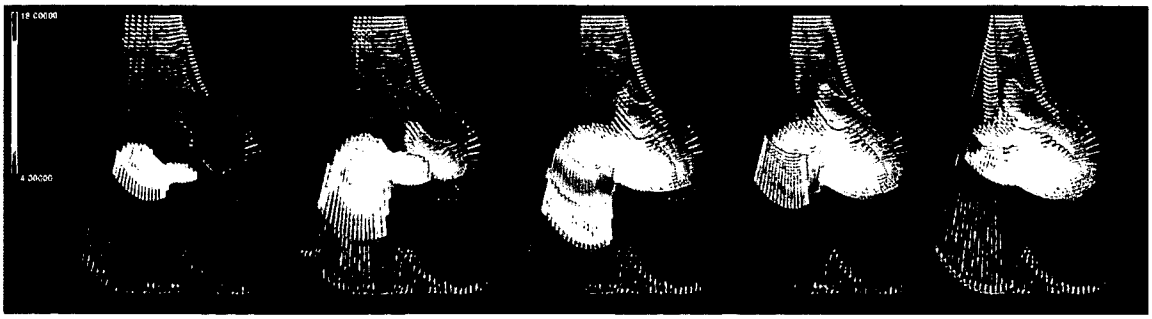


Figure 7.16: Interactively specified feature during the compression stroke of a diesel engine, showing swirl velocity values for areas which exhibit high levels of turbulent kinetic energy. Results from 340 to 360 degree crank angle are displayed.

7.2.3 Visual Analysis of Diesel Engine FM538

We discuss now results of the visual analysis of the two different simulation cases of the diesel engine FM538. For analyzing the complex behavior of the combustion process in such a diesel engine, we first explore the flow behavior during the compression stroke, then we analyze the mixture formation process, and finally we compare the combustion progress of the two cases. For larger versions of the images as presented here, as well as for further results and additional animation sequences, please refer to *http://www.VRVis.at/vis/research/diesel-engine/.*

Flow Analysis During the Compression Stroke

The flow which is created during the compression stroke, influences the mixture formation and also subsequently the combustion. For a proper mixture formation areas with high values of turbulent kinetic energy (TKE) are needed, at least in places where the fuel is injected. This forces the diesel droplets to break up and to evaporate. Additionally the swirl which is generated by special intake ports will enforce this behavior. Up to 360 deg CA there is no injection. As the two cases only differ with respect to the injection system, the analysis of the compression stroke yields the same results. For exploring the general flow behavior during this phase, regions of high TKE are of interest. Therefore all data items exhibiting high values of TKE are brushed in a scatterplot (see figure 7.15).

In a linked 3D SciVis view a F+C visualization of all brushed data items for several time steps from the end of the compression stroke, i.e., from 340 to 360 deg CA, is shown (see figure 7.16). The focus parts in this visualization are colored with respect to the swirl velocity values for each data item. It can be seen, that high TKE values are generated at the crossover from the Top Dead Center (TDC) gap to the piston bowl (at 340 deg CA). The TDC gap is the gap between the piston and the chamber head. The swirl velocity values at that time have a modest level and increase up to 18 m/s at 350 deg CA. Further on, up to 360 deg CA the flow in the TDC gap carries the region with high TKE values to the piston bowl, where it is needed for proper mixture formation. To analyze the reasons for the changes in the flow, regions with high velocity are of interest. As the data range of velocity values varies significantly over the whole simulation cycle, a method for selecting all data items with relatively high velocity values with respect to the data range of each time step (crank angle) is needed.

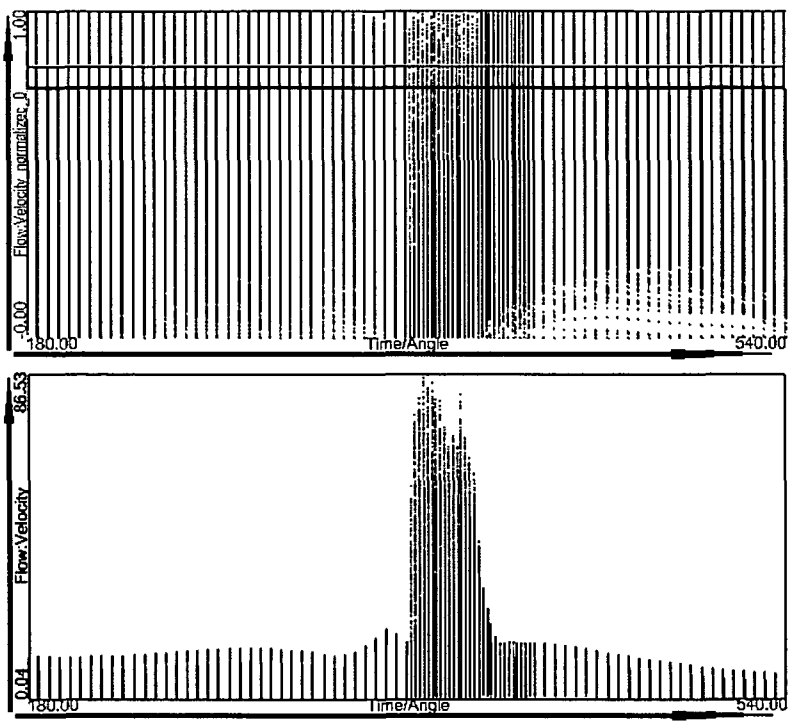


Figure 7.17: Defining a relative brush: high normalized velocity values are brushed in the upper scatterplot, leading to a relative selection of data items with high velocity values for each time step.

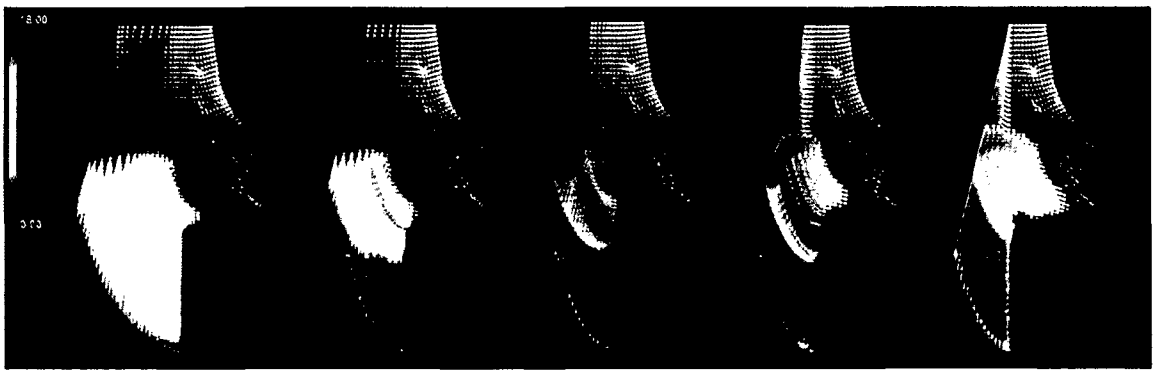


Figure 7.18: Interactively specified feature during the compression stroke of a diesel engine, showing swirl velocity values for areas exhibiting relatively high levels of velocity values. Results from 335 to 360 deg CA are displayed. Flow from the TDC gap to the piston bowl is depicted.

We normalize velocity values for each time step, possible in SimVis through the capability to calculate derived data attributes. When displaying these normalized velocity values in a scatterplot, we can easily brush data items which exhibit velocity values in the upper 20% of the data range for each time step (see figure 7.17, upper view). In a second, linked scatterplot

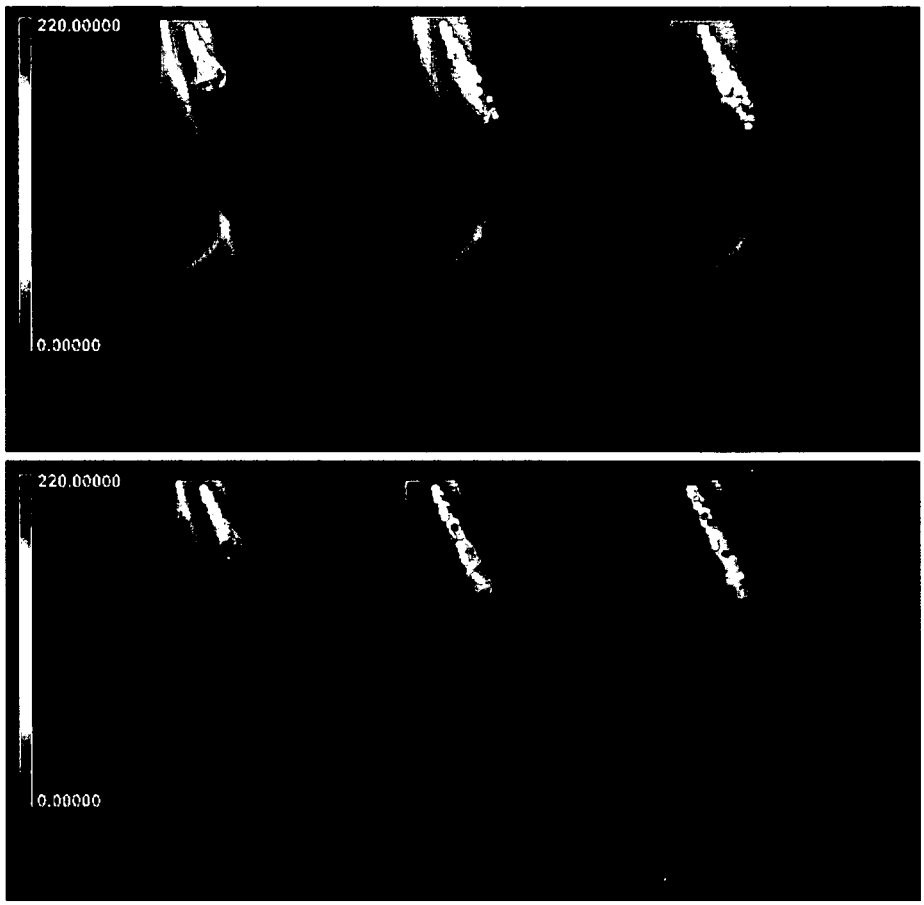


Figure 7.19: TKE values displayed for the regions of higher values of liquid spray density. Comparison of results from case 1 (upper row) and case 2 (lower row) at 362, 366, and 370 deg CA.

view (figure 7.17, lower view) the attribute distribution of original velocity values over time (crank angles) is shown for illustrating the advantage of specifying a relative brush (brushed data items are colored red in SimVis InfoVis views).

We again visualize the brushed regions in a linked 3D SciVis view, which allows us to easily localize the regions which exhibit a relatively high level of velocity values. In figure 7.18 time steps from 335 to 360 deg CA are shown, swirl velocity values are again used to define the coloring of the feature parts in focus. The general flow from the TDC gap to the piston bowl at the end of the compression stroke is depicted.

Analysis of Fuel–Air Mixture Formation

We now analyze the mixture formation for both cases. The mixture formation starts with the injection of liquid diesel at 360 deg CA. First we take a look at the liquid diesel spray. We brush all data items having high values of liquid spray density for data from the first simulation case. As the breaking up and evaporation of diesel droplets is influenced by high TKE values,

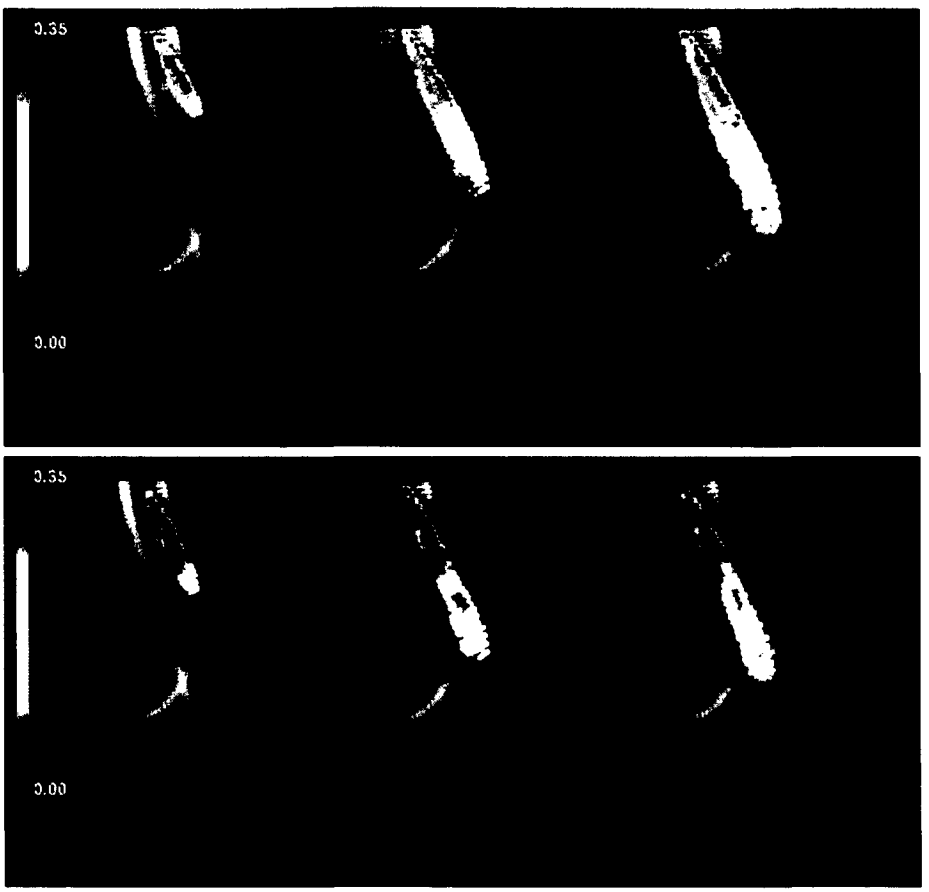


Figure 7.20: Diesel vapor mass fraction values shown in regions exhibiting high values of TKE and velocity. Comparison of results from case 1 (upper row) and case 2 (lower row) at 364, 368, and 372 deg CA.

this attribute is used for coloring the specified focus in a 3D view (see figure 7.19).[1] To ensure a correct comparison of both cases, all feature specifications that are performed in the first case are stored to an external file [33] and re-applied to the second case.

Figure 7.19 shows, that the liquid spray of case 1 is wider and causes a shorter penetration in the chamber. In this case the liquid spray is touching the piston surface at 362 deg CA, which leads to higher soot formation. The soot formation process is analyzed in more detail in the next section.

For a proper mixture formation, the diesel vapor should have high TKE and velocity values. Therefore we brush all data items exhibiting high TKE and velocity values in a scatterplot. In a linked 3D view we visualize the brushed regions via coloring the focus according to the respective mass fraction values of diesel vapor. In figure 7.20 we see a 3D F+C visualization of the simulation results at 364, 368, and 372 deg CA for both cases. It can be seen, that there is almost no diesel vapor near the nozzle outlet. This is because the injected liquid diesel needs some time for evaporation. When comparing the results of both

[1]Note that we always show the views for the first simulation case (with the wider spray cone angle) above the views of the second case for comparison of results throughout this section.

cases it becomes obvious, that in case 2 the region with high TKE and high velocity values has in general more diesel vapor. In case 1 there is almost no diesel vapor at 364 deg CA as opposed to case 2, where the evaporation starts earlier and therefore results in a better mixture formation. The reason for this can be found in higher TKE values caused by the liquid spray, as previously shown in figure 7.19. In general this visualization series shows, that for both cases the region with high values of TKE and velocity contains enough diesel vapor, assuring a good mixing behavior between fuel and air.

7.2.3.1 Analysis of the Combustion Progress

As described before, a considerable amount of liquid fuel has to evaporate before combustion can start. After this, auto-ignition occurs which initializes the combustion process. It is also important to consider that the processes of mixture formation and combustion are running simultaneously (at least to a certain extent of time), and thus influence each other.

At first we explore, how the fuel combustion progresses, and if there are regions, where not all fuel is burnt. In the extended 2D/3D scatterplot view we plot crank angle values (X-axis) vs. mass fraction values of diesel vapor (Y-axis) vs. the combustion reaction progress variable (Z-axis) of data from case 1. The 3D scatterplot, together with the respective 2D scatterplots for all axis combinations, is shown in figure 7.21. It can be seen (e.g., from the lower left 2D scatterplot), that at the end of the simulation of case 1 (540 deg CA) not all fuel has been burnt completely. The same holds for the second case, if inspected in a similar view.

For the analysis of where and why unburnt fuel remains at the end of the simulation, we interactively brush all data items having a mass fraction of diesel vapor above zero together with high values of the combustion reaction progress variable (see figure 7.21). This highlights all those regions where the combustion is almost finished. Figure 7.22 shows these regions in a F+C style of visualization for both simulation cases. Mass fraction values of O_2 are used for coloring the focussed regions. In the visualization of the first shown time step (at 396 deg CA) both cases show a concentration of diesel vapor together with progressed combustion in the bowl of the chamber. The reason why the diesel vapor can not burn completely is that there is almost no oxygen in this region (visible through coloring the focus with respect to mass fraction values of O_2). The result is an incomplete combustion of diesel vapor in the bowl for both cases. By comparing the results for the two cases, we notice, that for case 2 there is more unburned fuel in the bowl than for case 1, especially also at the end of the simulation. This is contradictory to our previously found result, that the mixing formation for the second case performs better. To analyze these conflicting facts, we investigate the differences in the flow behavior for both cases in the following step.

To do so, we want to analyze the flow behavior by using sectional slicing of the spatial layout in a 3D view. We take advantage of the possibility to calculate polar coordinates via attribute derivation from the spatial references in our system. For the diesel engine application we are interested in polar coordinates around the z-axis (aligned with the moving-action of the piston). In figure 7.13(4) we show a scatterplot where crank angels (horizontal axis) vs. polar coordinates around the z-axis (vertical axis) are plotted. By interactively brushing a horizontal sub-section in this plot, we can focus on a specific spatial section in the data set. In a linked 3D view (figure 7.13(5)) we see the interactively brushed sectional plane in the data. For coloring mass fraction values of diesel vapor are used for data items on this plane. In a second scatterplot (figure 7.13(3)) the distribution of the amount of diesel vapor over time is shown.

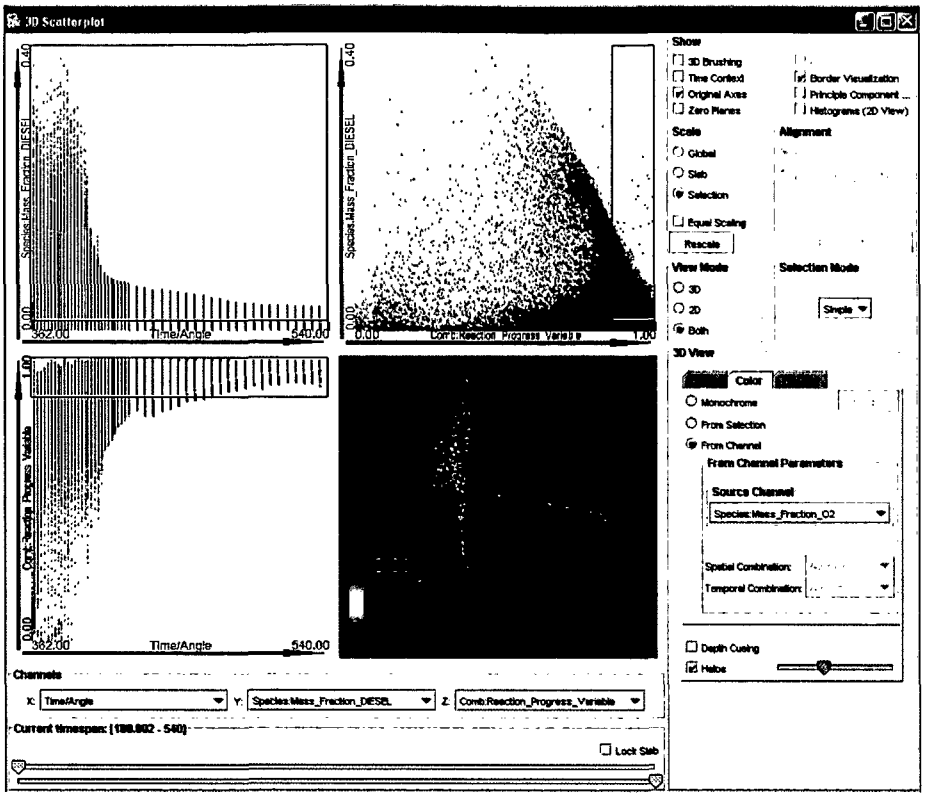


Figure 7.21: Combined 2D/3D scatterplot [148] used for specifying a feature represented by high values of the combustion reaction progress variable and non-zero mass fraction values of diesel vapor.

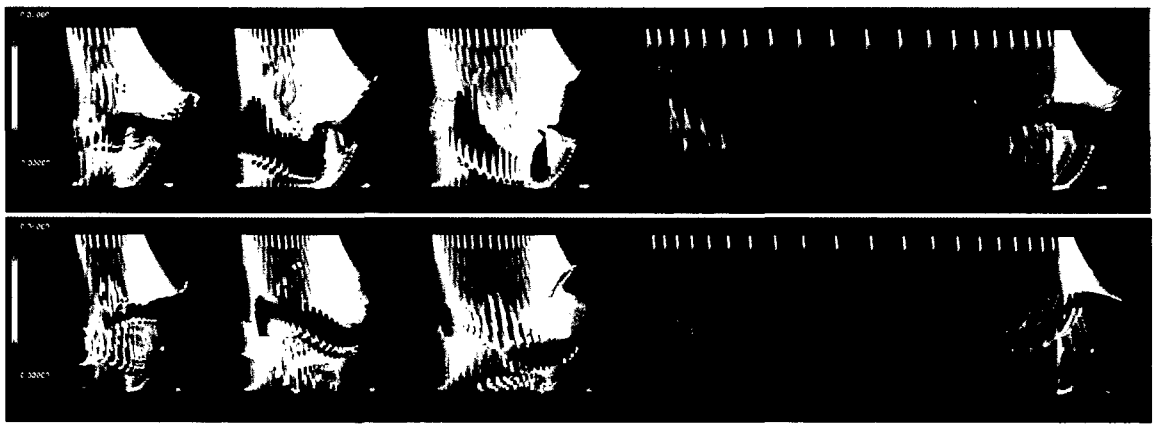


Figure 7.22: Mass fraction values of O_2 for regions with high values of the combustion reaction progress variable and non-zero mass fraction values of diesel vapor. Data from case 1 (upper row) and case 2 (lower row) is compared for 396, 408, 418, and 540 deg CA.

When reducing the context in the 3D view, and using arrow glyphs for showing the flow direction and velocities, we can get a very good impression of the flow behavior in one sectional

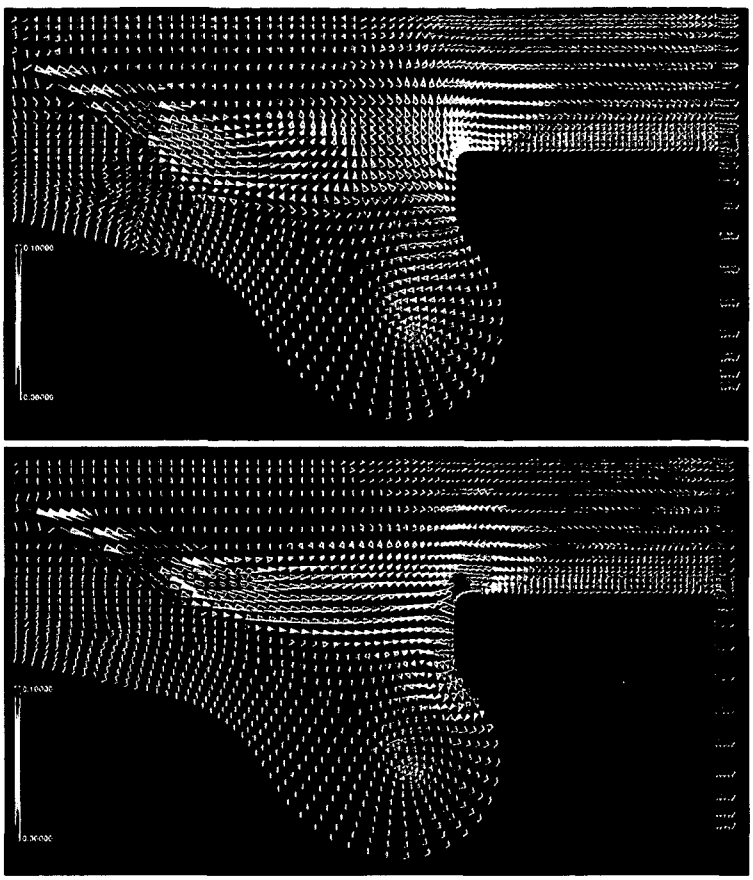


Figure 7.23: Comparing the flow behavior for both cases at 384 deg CA. The amount of diesel vapor is visualized for the sectional slice as defined by the brushing action shown in figure 7.13(4).

slice of the data. Figure 7.23 shows a comparison of the flow field at 384 deg CA for both cases. Again, the focus is colored with respect to the amount of diesel vapor. A fast inspection of different sectional slices is intuitively realized by interactively brushing different subsets of polar coordinates in the linked scatterplot view. In figure 7.23 we see, that in the second case the diesel vapor has a deeper penetration in the direction of the bowl than in the first case. Differences in the flow patterns for the two cases are visible as follows: For case 1 the main flow carries the diesel vapor up in the direction towards the chamber head. In case 2 the main flow direction in this region is horizontal (due to higher velocity and TKE values). As a result, diesel vapor is carried to the vertical piston wall where parts of it are re-directed into the bowl. Because of lower circulation in the bowl during the power stroke, there is almost no mixing between diesel vapor and oxygen. This leads to incomplete diesel combustion. From this point of view, case 1 is preferred.

Another important issue during the analysis of the combustion process is the production of soot. For quickly investigating this question, we brush high values of soot mass fraction. When visualizing values of the soot formation rate in areas with a high amount of soot, the formation of soot can be observed (see figure 7.24). A positive soot formation rate denotes

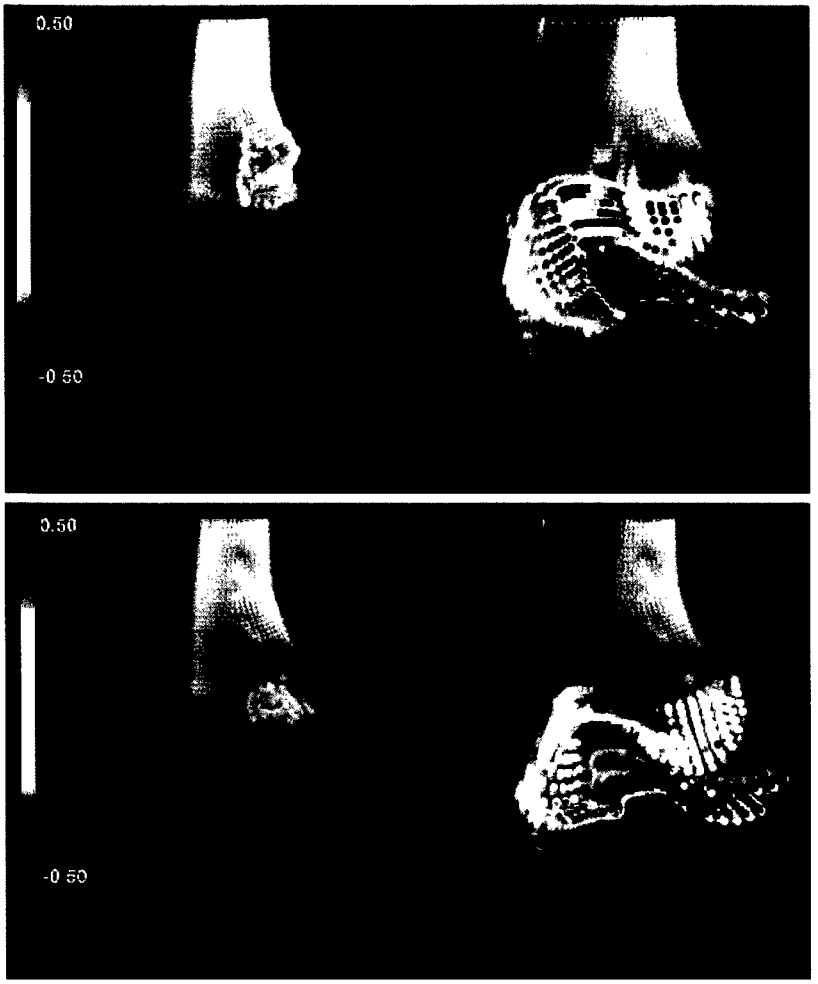


Figure 7.24: Soot formation at 368 and 386 deg CA: soot generation (red), soot burning (green) and constant levels of soot (yellow) are shown for areas of high soot concentration.

production of soot, a negative value means that soot is burnt. Through user-defined control over the scaling for the coloring, we can achieve that generation and dissipation of soot can be shown clearly in 3D. In figure 7.24 regions in red show places, where soot is produced, green regions are areas where soot is burnt. When comparing the two cases, we see that in the first case more soot is produced at 368 deg CA (left images) than in the second case. This is due to the liquid spray reaching the piston surface as mentioned earlier (see also figure 7.19). At 386 deg CA (right images) more soot is generated in case 2. The reason is, that in case 2 the diesel vapor reaches the vertical piston wall (see also figure 7.23), where a relatively low temperature of the wall leads to higher soot formation due to reduced soot burning.

Performance Issues

The visual analysis of the application presented here was carried out on the same system, as the previous case (see section 7.1). The two data sets investigated during this case study

are based on time-varying grids, each consisting of approximately 30,000 to 50,000 cells per time step with 44 attributes per cell. The temporal dimension in the cases described here is provided in terms of crank angles. Data for 92 different crank angles out of the whole time domain (180 to 540 deg CA, resembling to one half cycle of a four stroke diesel engine) was investigated during the analysis process. With these settings, loading all available time steps into a combined 2D/3D scatterplot (see figure 7.21), for example, requires handling and displaying of approximately 14.5 million points. Due to algorithmic optimizations, and a simple but effective data handling framework, we are able to handle analysis sessions with multiple views for data sets even a magnitude larger than the ones presented here, while still providing full interactivity.

7.2.4 Comparison and Conclusions

In this case study we demonstrated the interactive visual analysis for the application of combustion in a diesel engine. The exploration and analysis process is greatly supported the interactive focus+context visualization approach as presented in contribution sections of this thesis. By applying our techniques to the data set of this application, trying to answer several important user questions, we have shown the effectiveness and intuitivity of our visualization approach.

SimVis offers new methods which extend standard visualization capabilities of common industrial visualization tools. It allows to get a quick overview over the three-dimensional flow patterns and relations in the spatial domain through linking 3D SciVis views to InfoVis views. In the InfoVis views interactive analysis through brushing (with immediate feedback in the 3D view) and working on multiple dimensions simultaneously, is possible, improving and speeding up the analysis process compared to common visualization software tools. Additionally, working with SimVis allows to define synthetic data attributes through attribute derivation based on existing data attributes (e.g. normalizing data, using different coordinates for spatial brushing, etc.) and to display combinations of several quantities simultaneously in different InfoVis and SciVis views, which is usually not possible with common visualization tools for CFD data. Finally, the intuitive formulation of features in terms of a feature definition language proves to be very useful, especially when it comes to comparing different data sets side by side, as porting of feature specifications from one data set to another can be applied.

Chapter 8

Summary

Visualization of complex results from computational fluid dynamics (CFD) simulation has become a very active field of research. With increasing computational power of computing systems both the frequency of CFD simulation being used as well as the complexity of the calculations (and consequently also the complexity of the results) has increased. The complexity is especially caused by the multi-dimensional, as well as the time-dependent character of the simulation. Traditional ways of exploring and analyzing the resulting data sets are often no longer feasible due to the increased complexity of the results. Therefore, supporting engineers in their analysis process by providing tools for visual analysis is a natural extension to previously employed investigation methods, which worked primarily on a pure numerical data output without any graphical tools.

In this thesis a new approach to feature-based visualization of complex and large flow simulation data is presented. The new approach employs a clever combination of methods from scientific visualization (SciVis) and also from information visualization (InfoVis). After a short summary of this new approach and a discussion of the different concepts of the framework, we also give a short overview about the wide field of applications, which have already been explored or are targeted by our approach.

8.1 Combining Multiple Views for Interactive Visual Analysis

Our approach to interactive visualization of large, multi-dimensional, and time-dependent flow simulation data, resulting from simulation in a 3D spatial context is realized in a research prototype, called *SimVis*. SimVis provides the following technological features: Time-dependent flow simulation data which is given on grids of different kinds is visualized by the use of multiple views, including scatterplots, histograms, and 3D views, for the purpose of interactive visual analysis of the data.

The data sets we are analyzing usually consist of hundreds of time steps and are given on unstructured grids, often also with time-varying grid geometry/topology. Additionally a large number of data attributes per data item (e.g., flow vector, pressure, temperature, etc.) is available.

In the SimVis framework, the different views from InfoVis and SciVis are combined and used in a multiple-view setup. One key feature of the SimVis approach is the *support of interactivity* for the investigation of CFD data sets. Additionally, analysis and exploration of the large data amounts, is based on a visual interface to the data. Thereby a transparent

visualization process enables the user (i.e., engineers who are analyzing the simulation results) to understand better and faster complex relations of data items. Also, when compared to traditional visualization approaches for CFD simulation data sets, the SimVis approach is mainly targeted towards interactive analysis and exploration tasks, and not so well suited for presentation issues.

The major strength of the SimVis approach lies in the balanced combination of several different innovations. These by themselves are not all completely new and can (to a certain extent) also be found as isolated solutions (or in other combinations) in other implementations world-wide (see at least partially comparable systems such as XmdvTool [208, 132], XGobi [190, 191], Spotfire [1], IVEE [2], Polaris [187, 186], VisDB [92], and others). One important inspiration for our SimVis research was IBM research work on the visualization of complex simulation data (a beating heart), called WEAVE [61]. This work includes a 3D view for spatial orientation, a scatterplot for visualizing simulation attributes as well as a histogram and others, which are also visually linked in the context of interactive focussing through brushing. Other work which is somehow related to the overall concept of SimVis are linked derived spaces [76] and linking-and-brushing systems [208, 20, 191, 92, 29, and others], for example.

Linking and brushing [20, and others] is a technique where a certain subset of the data is interactively highlighted in one view through brushing [208, 35] and all other views of the same data are updated instantaneously to reflect the data selection in a consistent visual way, e.g., by coloring the selected data items red in all views. This concept of linking and brushing is usually employed in multiple-view setups, which often consist of multiple InfoVis views only. These InfoVis views can be either all of the same kind (e.g. several scatterplot views, etc.) or they also can be of different kinds (e.g. combining scatterplot views with histograms, parallel coordinate views, and so on). A combination of multiple InfoVis views in conjunction with SciVis views for additionally also showing the spatial structure of data sets, however, is a relatively new and seldom used concept.

In our approach the linking between all the different views is accomplished through the representation of user interest in the form of so-called degree-of-interest (DOI) attributions of the data. The DOI attributions are defined through a DOI function, which is interactively constructed through brushing in the InfoVis views of the framework. Thereby, for each data item a DOI value is specified, which is then used to separate data items in focus and context. This discrimination into parts of the data, which are of current interest (focus) and the rest of the data (context) is then used to accomplish a focus+context (F+C) visualization in the 3D SciVis views.

Focus+context visualization is often used in InfoVis for visualizing very large data sets. The data is split into the focus and context parts respectively, usually also using an interactive specification process, and then the visualization is adapted in that way, that the focus parts are presented in a more prominent way, with the context shown less prominently. By using a F+C visualization approach, the disadvantages of too much cluttering on the screen for visualizing very large data sets can be reduced, but on the other hand the orientation of how the essential parts of the data (focus) are related to the rest of the data is not lost.

Hauser [66] recently showed how this concept can be extended in several ways to also be used in SciVis approaches for efficient visualization of large data sets. In our approach we use color and opacity, as well as style for focus – context discrimination in the 3D visualization. The focus parts of the data are shown using a colored transfer function, and in a more opaque style [35]. Also we use larger glyphs for representing each data item (cell of the grid) in the

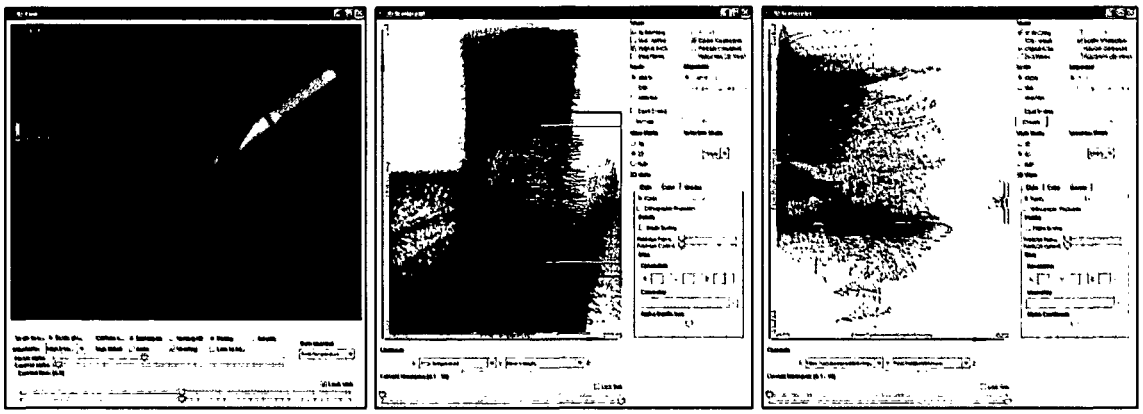


Figure 8.1: An example for the basic concept of *linking and brushing*, as integrated in the SimVis approach: Multiple views from SciVis and InfoVis are used to view different aspects of the multi-dimensional flow data, brushing in a InfoVis view (middle) yields focus+context visualization of the data items in all other linked views (for more details see text).

visualization. Thus the attention of the user is drawn mainly to these parts of the data. The opacity of the glyphs which represent the context data items (which are only shown in gray-scale and by the use of smaller glyphs) can be interactively reduced or increased, giving the user control over how interested he or she is in an overview visualization or rather a detailed focus visualization.

An example of the linking and brushing concept, as well as of the F+C visualization included in the SimVis framework is shown in figure 8.1. Here, one 3D SciVis view (left) and two InfoVis (scatterplot) views, that are used for an analysis of a CFD simulation result of two mixing fluids (one hot and one cooler one) in an extended T-junction of three pipes, are shown. Data items, exhibiting high temperature values and medium velocity values have been brushed interactively in the first scatterplot view (middle). The corresponding focus regions are clearly visible in the 3D SciVis view, the color of the focus parts represents the temperature values at each data item (red corresponds to relatively high temperature). In the second scatterplot view (right) the distribution of the data items with respect to the data attributes of "Turbulence Kinetic Energy" (horizontal axis) and "Relative Pressure" (vertical axis) is visualized. Here also the focus – context discrimination (red points are in focus) helps to get better insight into the relations between the various data attributes.

8.2 Fuzzy Classification

In most applications of visualization there are only binary or discrete classifications used to establish a semantic layer on top of the originally unlabeled data [176, 151]. In medical visualization, for example, object segmentation plays an important role, and usually discrete object maps are used to label voxels of either being part of one object or another. In flow visualization, also usually a sharp feature extraction process is used to discretely partition the flow domain into portions which represent certain flow features, e.g., a vortex or a recirculation zone.

In SimVis, we utilize fuzzy classifications (according to the terminology of fuzzy logic [222])

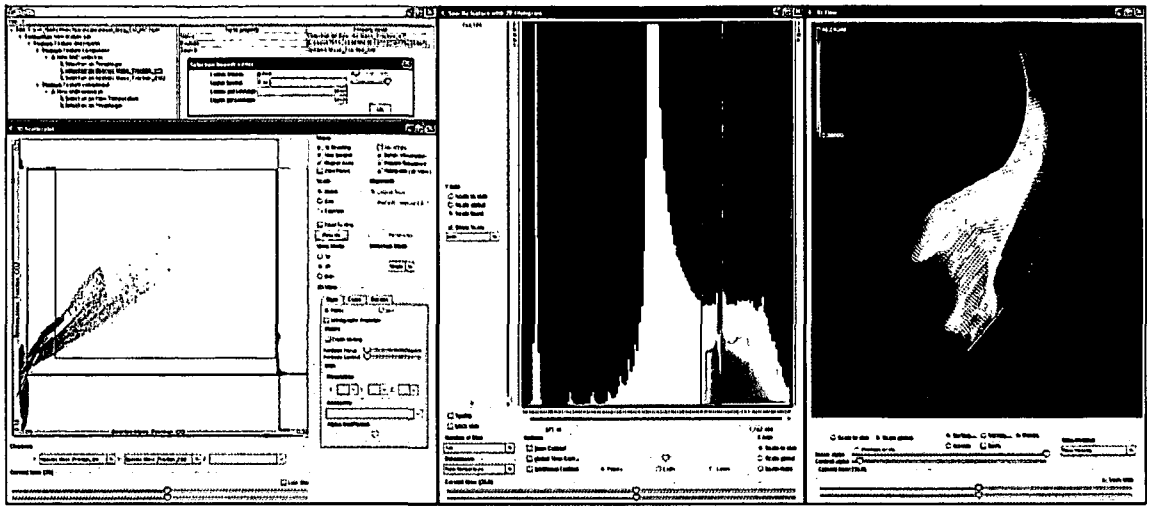


Figure 8.2: A sample SimVis scenario: simulated flow through a diesel particle filter (DPF) is visualized – the flow is shown at the time of 35 secs. after simulation started. The user has reflected his interest in flow regions of heavy oxidation by interactively brushing data items which exhibit a lot of carbon-oxides in the scatterplot (lower left) and then refining this specification to only apply to hot regions (in the histogram, middle). The 3D view on the right shows a focus+context visualization of the DPF with the brushed data items highlighted in color (color shows velocity magnitudes). The upper left view provides a direct and numerical interface to the hierarchical brush definition and all its parameters (repeated figure, see also figure 1.2).

to assign probabilities of class containment. This is accomplished by employing *smooth brushing* [35] as an extension to standard brushing methods in SimVis. By smooth brushing, the user specifies a non-discrete DOI function, allowing to smoothly discriminate data items in focus from those in context. This is an important feature of SimVis as it is often not really possible to sharply delimit flow portions of interest from all the rest – usually between a certain region of full, i.e., 100% user interest and completely uninteresting portions of the flow (DOI-values of 0) a border region exists for which a gradual change of DOI-values is assumed.

For an example of smooth brushing of see again figure 8.1. The interactively brushed region in the scatterplot in the middle was specified using smooth brushing, thus the resulting focus+context visualization in the 3D SciVis view also better reflects the smooth nature of the simulated data.

8.3 Iterative, Interactive Feature Specification

In addition to allowing singular brushing events in the InfoVis views of the framework, SimVis utilizes also an iterative approach to feature specification and feature-based visualization of large and complex data. In SimVis, the setup of synthetic DOI attributes is usually started by a simple selection of data items in one view (for example in a 2D scatterplot with respect to two of the data attributes as also done in the figure 8.2 example in the lower left scatterplot).

After investigating the visual response of this first step (e.g. in the 3D view), iterative

refinement is performed to furthermore detail the feature specification in any of the other views (as done in the histogram of figure 8.2, for example).

The result of such an iterative process is a complex feature specification which is of hierarchical structure and usually involves a set of different data dimensions [33]. To build up the hierarchical structure of the feature specification out of the non-discrete DOI functions of each brushing step, fuzzy logic operations are used to establish a calculus which is based on fuzzy DOI values [101].

In SimVis, this information about the (hierarchical) feature specification together with all relevant related parameters is explicitly represented in the so-called *feature definition language* to ease user access to feature specifications. A separate user interface is also available, which enables the direct, e.g., numerical manipulation of feature specifications [33] (compare to the upper left parts of figure 8.2).

Such an explicit representation of the feature specification process also immediately enables load- and save-functionality which consequently results in additional advantages such as the opportunity to compare data sets by setting up an analysis for one data set, then saving it, and re-applying the same analysis to another data set. This has already been applied for two different recent case studies very successfully [38, 37].

8.4 Time-Dependent Feature Specification

In addition to iterative feature specification for the specification of more complex features as compared to only using simple brushing, another extension of our approach was to allow specification of *time-dependent features*. We consider time-dependent features to be those flow features which are inherently dependent on time.

For the integration of this new type of features we integrate *attribute derivation* into our interactive data visualization for the interactive specification of complex time-dependent features based on temporal relations in the data.

Together with the previously mentioned advanced brushing and iterative feature specification methods, attribute derivation is used in a hybrid feature-based visualization approach (see also figure 5.1 in chapter 5). On the one side we "lift up" data properties in the layered feature space by **attribute derivation** (related to feature extraction), i.e., for every data item additional (synthetic) data attributes (second- and nth-order data attributes) are computed through mathematical combinations of first-order data. This happens, for example, through the application of linear filters or the use of formulas describing physical relations between first-order data attributes. On the other side we propose **extended brushing mechanisms** which allow to access data properties even in the deeper regions of the layered feature space. This way, logical combinations of first- as well as of higher-order data attributes can be used to formulate more complex relations than what otherwise would be possible with simple brushing alone. This can be especially useful for the definition of time-dependent features.

We included the specification of five types of time-dependent features in the SimVis framework based on this hybrid approach, namely (see chapter 5 for more details):

- *features based on temporal attribute gradients*,
- *feature specification relative to data changes*,
- *features based on interest which varies over time*,
- *features based on stationary attributes*, and

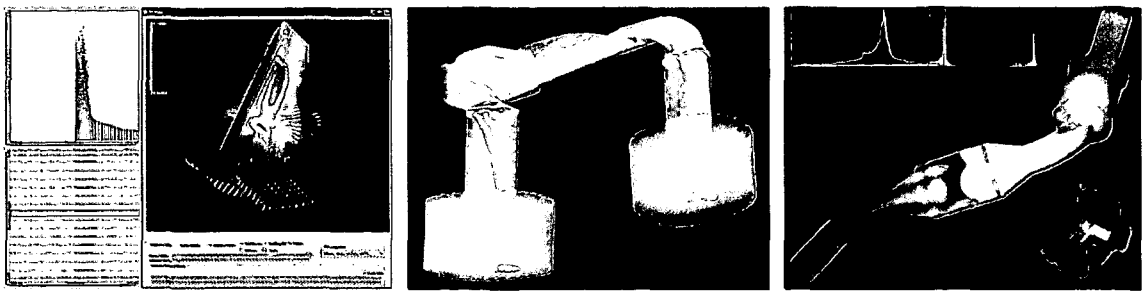


Figure 8.3: Applications of CFD simulation in the automotive industry: analyzing the combustion process in a diesel engine for heavy trucks (left); exploration of flow behavior in a ventilation system (middle); analyzing the soot burning process for a diesel particulate filter in a diesel exhaust system for passenger cars (right).

- *features based on local extrema with respect to time.*

A couple of case studies [38, 37, etc.] already showed how this new type of time-dependent features can provide additional help when investigating time-dependent CFD simulation results.

Besides this new hybrid approach for time-dependent feature specification, providing a proper access to the special dimension of time has also been dealt with in our approach. Here mainly three methods have been taken into account:

- time has been established as a "normal" data dimension, just as all other data attributes are, so that time can be used in SimVis views just as any attribute dimension [36]. This way, for example, the distribution of one data attribute against time can be plotted in a 2D scatterplot.
- SimVis views have also been consistently extended by an additional interface component to allow for a special handling of time [36]. This way the user selects with sliders from which time steps data is shown in a specific SimVis view – the same range selection on the time axis is also used as the reference for brushing.
- the framework has been extended to allow handling of data sets which are based on time-varying grid geometries [37]. For these data sets, different grid geometries are available for temporal subsets of the simulation, allowing deformation and rescaling of different simulated spatial parts (see also chapter 7).

8.5 Application Examples and Possibilities

We have recently shown in two case studies [38, 37], as well as in many of our other works, that the SimVis approach is generally applicable.

Our original goal was to provide an analysis and exploration technology for results from CFD simulation, especially from the automotive industry. But recently we also employed SimVis technology for the analysis of data from other application fields, and thereby proved the generic nature of our approach. In the following we give a short overview about application fields from which we recently analyzed data, as well as an outlook to further application domains.

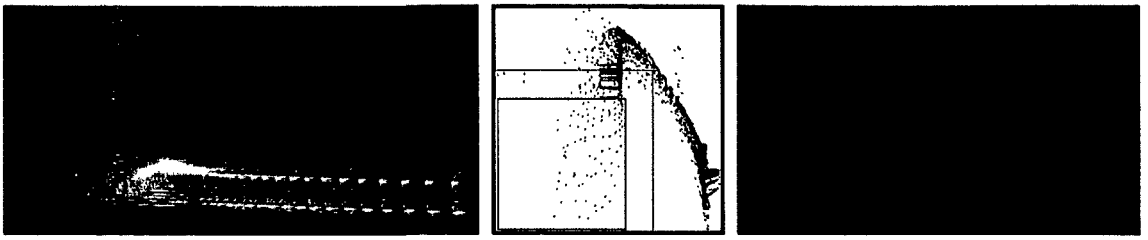


Figure 8.4: Analyzing mixing fluids in two different T-junction layouts: smooth brushing of a recirculation area in a simple T-junction (left), a scatterplot view was used to specify the recirculation feature (middle); analysis of mixing hot and cool fluids in an extended T-junction, only hot flow regions are shown (right).

Applications from the Automotive Industry

CFD simulation is used in the development process of many parts of passenger cars or heavy trucks. Sample applications include the design of combustion chambers [37], exhaust gas systems [38], catalytic converter applications [35], ventilation and air conditioning systems [35], aerodynamics design of cars [33], and many more. Example images from analysis of data from the above mentioned application examples with our SimVis framework are shown in figure 8.3.

Mixing Fluids

Mixing fluids are a phenomenon which is often researched using CFD simulation. We analyzed, for example, different mixing behaviors of hot and cool flows in several versions of different T-junction layouts [35, 36]. Two example images of the analysis of these T-junctions are shown in figure 8.4.

Wind and Air Flow Simulation

Wind and air flow simulation is a very important application field of CFD simulation. Air flow can be observed in many different applications, e.g., in the automotive industry (ventilation and air conditioning systems [35]) or in the simulation of room climate conditions. Also the simulation of fire falls into this category. Wind simulation recently has become of great importance. We analyzed, for example, data from a hurricane simulation [39] (see also figure 8.5 for an example). Also, simulating the influence of different architectural layouts of (large) buildings on the wind conditions in neighboring regions has been used and results of such a simulation were analyzed in our framework (see figure 8.5, showing wind distributions around the UN headquarter in Vienna, Austria).

Water Flow

Simulating water is a very natural application field for CFD simulation. The SimVis technology has been employed for exploration and analysis of results from flooding simulation [36]. Figure 8.6, left side, for example, shows two results of a time-dependent analysis of the impact of a flooding onto a nearby object after the break of a dam.

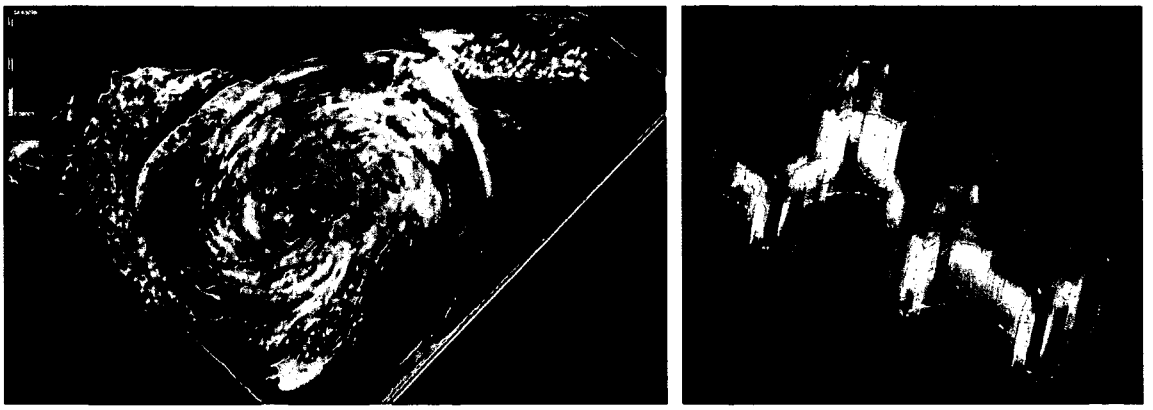


Figure 8.5: Wind and air flow analysis: analyzing results from the simulated hurricane Isabel, which struck the US Eastern Coast in 2003, clouds are shown (left); results from wind simulation are explored in a 3D SciVis view of SimVis, showing the wind velocity (red corresponds to relatively high velocities) for wind around the UN headquarter in Vienna, Austria (right).

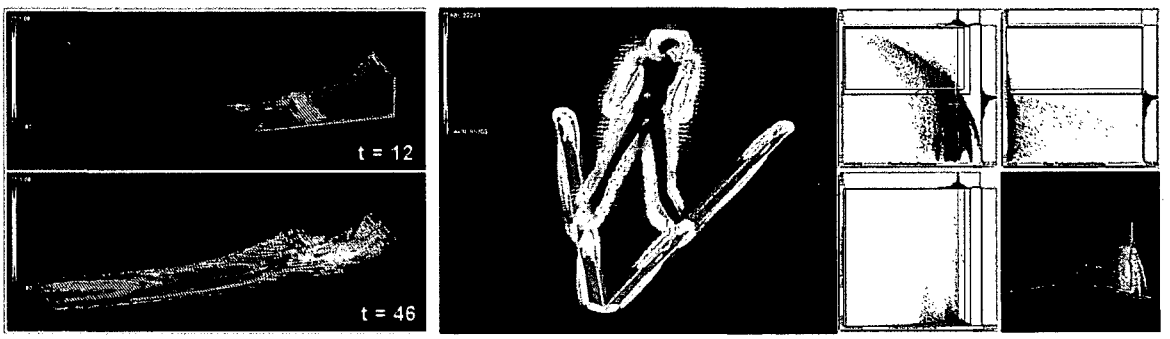


Figure 8.6: Analysis with SimVis: analysis of flooding and its impact onto a nearby object after the break of a dam (left); exploration and analysis for determining better aerodynamics in a ski jumping application (right).

Applications in Sports

Simulation is applied in applications of sports, especially for the purpose of improving aerodynamic behavior in speeding competitions. SimVis has been recently used to analyze the results of a simulation of ski jumping, which currently is developed at the Technical University of Graz, Austria, together with our industrial collaboration partner, AVL List GmbH [6] (see also figure 8.6, right).

Medical Applications

In medical applications, several application fields for employing CFD simulation have already been identified (see also chapter 1). We recently analyzed data from the simulation of blood flow in the junction of three blood vessels (see figure 8.7).

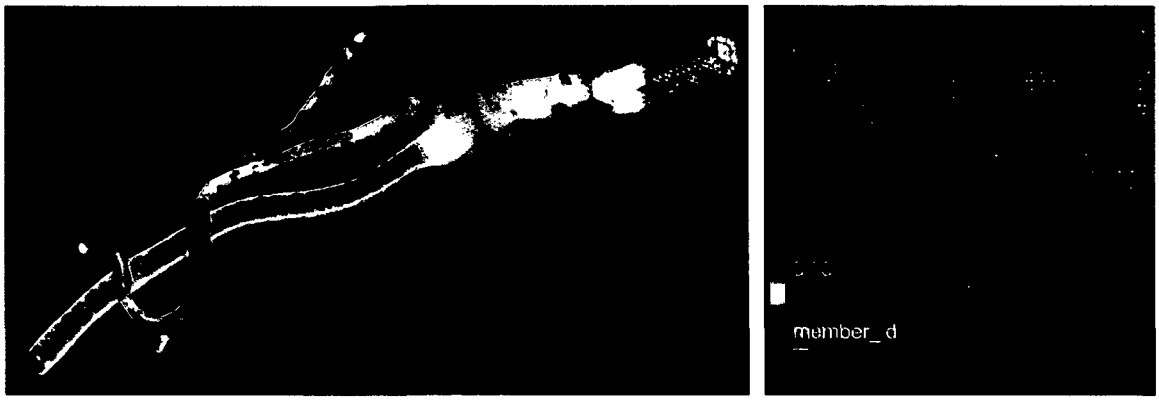


Figure 8.7: Analysis with SimVis: analysis of blood flow through the junction of three blood vessels (left); analyzing financial data coming from a stock market in our framework (right).

Financial Business Applications

As the SimVis approach employs many different concepts and technologies from the field of information visualization, it was straight forward, to also employ the framework for more abstract data, coming from different sources. We, for example, analyzed financial data coming from a stock market (see figure 8.7, right) with SimVis. Here, of course no spatial data component is available, but nevertheless all InfoVis views can be used to view different data attribute distributions (over time) and thereby analyze complex relations in the data sets. This example shows, how general the SimVis approach is.

Further Possible Application Domains

As the above examples have shown, many more application fields for our *framework for interactive visual analysis* are suitable. Data sets from different kinds of simulation (like climate simulation, simulation in molding applications, etc.) are very similar to the applications mentioned above, and therefore can be easily investigated and analyzed with the SimVis technology.

The last example (analysis of data from a stock market) has also proven, that, with some restrictions, even the analysis of more abstract, e.g., financial data or CRM (customer relation management) data, is easily possible with the presented framework.

Chapter 9

Conclusions

One of the main concerns of recent visualization research has been to come up with approaches and solutions that are applicable in more than just a few specialized cases. But how to characterize the necessary requirements for the design of approaches that are likely to be used by a wide-spread community of users? Such an approach must definitely include concepts like flexibility, generability, scalability, as well as intuitive usability in a well-balanced combination.

The framework for interactive visual analysis of large, multi-dimensional, and time-dependent flow simulation data presented in this thesis was designed exactly with this idea in mind, namely trying to provide an visual exploration and analysis approach, which is as general as possible, and at the same time, very flexible and scalable to large data sets.

As already has been shown in several case studies as well as other application oriented project setups, the approach for interactive visual analysis implemented in the SimVis framework, is at least for data coming from the field of 3D, time-dependent, multi-variate simulation (e.g. CFD simulation, but also others) a very general approach. It generally is applicable for the exploration and analysis of data which exhibits a spatial component (in 2D or 3D), a multi-dimensional character (different data attributes per data item, which can also be of different kind, e.g., scalars, vectors, etc.), and a temporal dimension. Examples from many different fields, including, for example, applications from the automotive industry, aerodynamics, weather and climate modeling, environmental simulations, material sciences, or medical applications, can be handled in the same conceptual framework with little or no adaption.

The presented framework is also very flexible, as the analysis usually adapts to the users interest (as it is defined interactively). Through visual feedback after each analysis step and the possibility to iteratively refine an analysis, the user is included in the generation of analysis results.

From collaboration with our industrial partners we know that interactive exploration and analysis of simulation data often follows one of three characteristic interaction patterns. One type of exploration is to search for places in the 3D simulation where certain feature characteristics are present (feature localization). In the SimVis system the user can brush features in InfoVis views and concurrently localize the respective feature in the 4D (3D+time) flow domain. Additionally users also investigate multi-dimensional data properties by specifying a feature in one InfoVis view and at the same time analyzing (through view linking) the DOI distribution with respect to other data attributes in additional InfoVis views. Furthermore users are often interested in inspecting the values of selected data attributes with respect to certain spatial subsets of the flow domain. In the SimVis system the user can load spatial

and temporal references of the data also into InfoVis views – brushing these kinds of data attributes yields features which are (at least in parts) specified by their location in space and/or time.

As a result, the intuitive usability of the SimVis framework is guaranteed by the support of these three characteristic patterns of interaction. Thereby, also the comprehensibility of the feature specification is much better than for conventional feature extraction and tracking methods.

All in all, interactive visual analysis, especially for large, multi-dimensional, and time-dependent flow simulation data based on multiple, linked views, is a new technology, which can provide additional ways of getting quick insight into complex behaviors of flows.

This new approach has the above mentioned advantages, which make it a very useful approach, especially for the case of quick exploration and interactive analysis tasks. However, there are also a few shortcomings or disadvantages of the newly presented visual analysis approach. One is, that frequent focus changes during interaction with the data can lead to frequent DOI updates, which result in a potentially computationally expensive process of focus – context discrimination. A second disadvantage as compared to conventional feature extraction techniques is, that with the general approach as presented, not all kinds of features can be specified interactively. This is due to the limited feature complexity, that can be separated by interactive brushing in the current state of the framework. One more shortcoming is, that the current approach is very well suited for interactive exploration and analysis, but only suboptimal for presentation purposes. This is partly due to the fact, that InfoVis methods are abstract and understanding results of these techniques require at least some basic learning, and partly also due to the fact, that the primary focus of our research so far was not fully directed towards an optimization of 3D Focus+Context visualization. Here future research should improve on this shortcoming to increase the acceptance level of the approaches.

Appendix A

Characteristics of Presented Data Sets

What follows now, is a short description about the data sets presented and used in the example illustrations of this thesis. For acknowledgements according to data sets courtesy regarding the different data sets please have a look at the Acknowledgements section.

In table A.1, a overview about different data sets according to different data size measures is provided. This table can be roughly divided into three sections (top to bottom) representing three classes of data sets, which have been analyzed with SimVis so far. The first class (data sets (1) to (4)) represents *steady-state CFD simulation results*, these were the first test cases used during the development of the basic linking and brushing approach, as well as during the specification of the feature definition language framework. The second class (data cases (5) to (15)) represents *time-dependent CFD simulation results*, which are already more complex, and most of them are real-world application examples. The third class (data sets (16) to (17)) represents also *time-dependent CFD simulation results*, but this time based on *time-varying grid geometry*. The columns of the table provide the following information (from left to right):

- *Data/Case*: in brackets a number is assigned to each discussed data set, to allow easy referencing in the following descriptions. Also a short name for each data set is provided.
- *no. of cells #c*: this column provides the number of cells of the underlying grid used in the simulation. For time-dependent data sets this means that the number of cells which are available at each of the time steps is provided. Therefore, for the cases based on time-varying grid geometries, the minimum and maximum number of grid cells per time step are provided, meaning that all the grids used for each time step have a number of grid cells, lying in this interval. For the other time-dependent data sets (class 2), this number is constant over all time steps.
- *no. of time steps #tst*: the number of time steps, for which results are available, is provided in this column.
- *no. of data attributes #a*: for each data item is a number of data attributes available (for time-dependent data sets again per time-step).
- $\#c \cdot \#tst$: this column gives the resulting value for a calculation of the number of cells times the number of time steps, which gives the number of data values available per data attribute. This is an interesting value, as many visualization operations in SimVis are somehow related to working on such a number of data values. For example, when

Data/Case	no. of cells #c	no. of time steps #tst	no. of data attributes #a	#c · #tst	#c · #tst · #a
(1) Car-Interior	4.150	1	10	4.150	41.500
(2) T-Junction	5.400	1	12	5.400	64.800
(3) Catalytic Converter	9.600	1	13	9.600	124.800
(4) Car-Exterior	9.344	1	15	9.344	140.160
(5) Blood	162.585	5	15	812.925	12.193.875
(6) Ventilation	306.526	6	18	1.839.156	33.104.808
(7) 05_test	17.120	100	23	1.712.000	39.376.000
(8) Extended T-Junction	30.930	100	17	3.093.000	52.581.000
(9) Hurricane 2	3.125.000	1	24	3.125.000	75.000.000
(10) UNO	651.747	7	17	4.562.229	77.557.893
(11) Flooding	76.505	48	28	3.672.240	102.822.720
(12) Hurricane 1	200.000	24	24	4.800.000	115.200.000
(13) Diesel Exhaust System	262.674	20	38	5.253.480	199.632.240
(14) SkiJump	704.900	20	16	14.098.000	225.568.000
(15) Secret Data Example	150.124	600	6	90.074.400	540.446.400
(16) Engine	min: 3.260 max: 4.590	in all: 40	17	172.960	2.940.320
(17) Diesel-Engine	min: 25.956 max: 50.796	in all: 91	45	4.042.836	181.927.620

Table A.1: Characteristic numbers for data sets analyzed with the SimVis system.

showing data from all time steps in a scatterplot view, this number determines the number of points plotted in the view.

- *#c · #tst · #a*: the number provided in this column gives the total number of data values (for all cells, time steps, and data attributes), that are available in the data set for analysis and visualization. This however is not such an important number, as rarely the whole data (data for all time steps **and** for all data attributes) is handled (or visualized) simultaneously.

In the following each of the 17 data cases from the table is described and discussed briefly:

(1) Car-Interior: this data set shows one vertical front-to-back slice of the interior of an automobile cabin (one slice of cells, not 2D). An inlet in the front is used to simulate a ventilation system, the outlet for this model is in the rear of the car. A visualization of the data showing velocity values through color mapping is available in figure A.1, left side.

(2) T-Junction: a 3D data set, three pipes are joint in a so-called T-Junction (see also figure 8.4, left side, for example). Two inflows (coming from the left and from top) join into one outflow (leaving to the right).

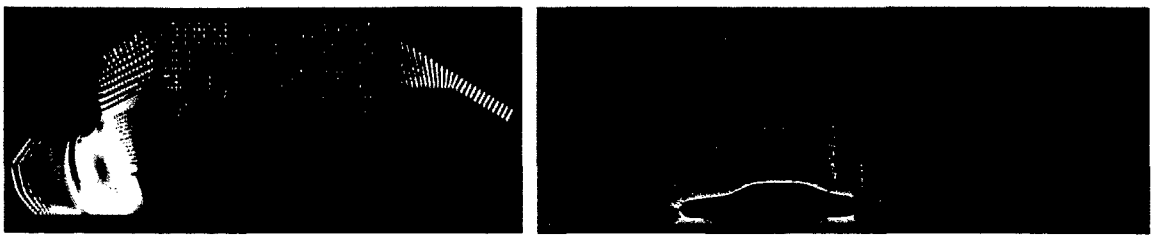


Figure A.1: Two data sets from the automotive industry: simulation of a ventilation system in the cabin (left) and flow around a moving car (right).

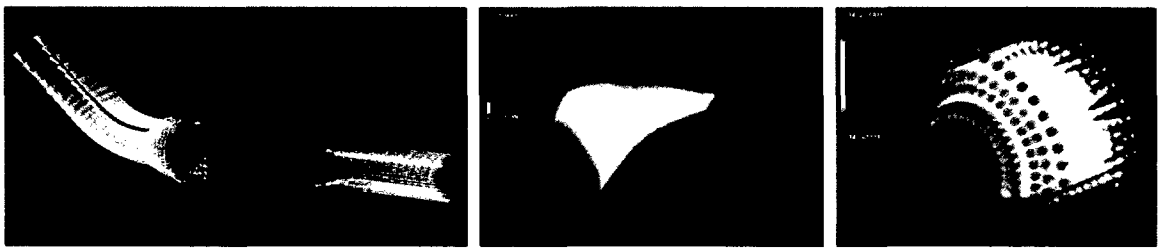


Figure A.2: Catalytic Converter data set (left), a test data set for time-dependent hot inflow (middle), and a test data set for a simulation of combustion in a diesel engine based on time-varying grids (right).

(3) Catalytic Converter: a data set used for simple (steady) simulation of a catalytic converter application. The 3D model is obtained by griding half of the geometry, as the application is assumed to be symmetric. This is a test case for real catalytic converter applications, such as the catalytic converter which is a part of the "Diesel Exhaust System" data set (13). A sample visualization showing regions of high turbulent structures is shown in figure A.2, left.

(4) Car-Exterior: this data set represents one vertical front-to-back slice through a car, where the flow around a moving car is simulated. A view on the grid is shown in figure A.1, right, where velocity is color mapped onto the grid edges. The simulation is steady, a flow inlet on the left side of the grid is assumed to simulate the moving behavior of the car.

(5) Blood: this data set stems from a simulation of blood flow in a junction of three blood vessels (see also figure 8.7, left side). In this data set, especially the large differences in spatial grid resolution distances are a challenge for analysis and visualization.

(6) Ventilation: a part of a ventilation system for passenger cars, as shown in figure 8.3, middle. One inlet (at the top) and two outlets (bottom), into which the flow is split.

(7) 05_test: this is a test data set, which was used during development of the time-dependent feature specification setup. A hot inflow coming from the top right (see figure A.2, middle) and going to the bottom, left is simulated.

(8) Extended T-Junction: this data set is an extended T-Junction version, which is more

complex than the one simulated for data set (2). Apart from the fact, that it is representing a time-dependent simulation, also the geometry of this example is more complex. In the center of the mixing chamber in front of the inlets is an obstacle, which is used to create turbulence for achieving a better mixing behavior. Additionally the incoming flows are differently timed: a first warm inflow from the right (see figure 8.4, right) is started at the beginning of the simulated temporal domain. Then after 35 time steps, a second, hotter inflow from the front pipe is joint.

(9) Hurricane 2: a data set resulting from the simulation of a real hurricane named Isabel, which struck the Eastern US coast in September 2003. This data set is a down-sampled version of a larger simulation, which was created at the National Center for Atmospheric Research (NCAR) in the USA. It contains only data for one specific time step, but over 3 millions of cells (which was generated to test our 3D rendering capabilities). A time-dependent version, including less cells per time step was also used (see below).

(10) UNO: a data set resulting from a wind simulation around the UN headquarter in Vienna, Austria (see also figure 8.5, right).

(11) Flooding: resulting data from a simulation of the impact of a flooding after the break of a dam on a nearby obstacle (see figure 8.6, left). The (broken) dam is on the right side, the obstacle in the middle, and the flow direction of the sudden flooding from right to the left.

(12) Hurricane 1: this is the time-dependent down-sampled version of the same simulated hurricane data set as in data set (9). A case study analyzing the behavior of this simulated hurricane in detail with many video and graphics results is available at *http://www.VRVis.at/SimVis/Isabel.*

(13) Diesel Exhaust System: the diesel exhaust system data set as used in the first case study presented in detail in chapter 7 of this thesis.

(14) SkiJump: this data set resulted from a recent simulation of a ski jumper, which currently is developed at the Technical University of Graz, Austria, together with our industrial collaboration partner, AVL List GmbH (see also figure 8.6, right). The grid used for the CFD simulation contains over 700.000 cells (per time step), which is already considered to be rather large at the current state of daily use simulations.

(15) Secret Data Example: this example unfortunately cannot be published, as it is property of an industrial partner, which does not allow publishing (at least so far). Nevertheless it is included in the table of analyzed data sets, to show, that analysis of larger data sets can be accomplished already with the current SimVis setup. Especially the large amount of time steps (and the resulting very large overall data size) is to be mentioned for this case.

(16) Engine: this data set is a test data set for a simulation of combustion in a diesel engine based on time-varying grids (see figure A.2, right).

(17) Diesel Engine: this data set is the one, that was used in the second case study of analyzing data coming from a simulation of the combustion process in a diesel engine, which was also presented in detail in chapter 7.

Appendix B

CFD – Applications and Data Characteristics

In addition to the information on CFD and resulting data sets given in the "Introduction" section of this thesis, further information on typical application fields of CFD as well as on different grid types is given below.

Applications of CFD

As already mentioned above, CFD is employed in a wide range of application fields in research as well as in product development. The following list of common application fields is intended as a *rough overview.* It has been put together as result of an intensive internet research, including the study of webpages of commercial CFD simulation and post processing companies, as well as academic and CFD repository sites [220, 167, 23, 24, and many more]. Common applications, where CFD is often used, include:

Automotive industry: a large variety of different applications are of common interest, including for example:

- *Aerodynamics of the flow around a car:* questions of interest are the calculation of pressure and resistance distributions at vehicle components. An example is the design of outside mirrors for passenger cars, such that rain and dust particles are diverted away from the glass, and thus the mirrors remain clean.
- *Simulation of noise generation:* car manufacturers are also concerned with reducing noise levels, both inside the car and also externally, to make cars as silent as possible.
- *Ventilation and air conditioning systems:* these applications include flow calculations inside the passenger cabin, heating and cooling capabilities of air conditioning systems, deicing of windscreens, but also cooling ventilation for engine parts, for example.
- *Filters:* several kinds of filters are used in vehicles such as fuel filters, oil filters, and air filters. Usually the main aim is to increase the efficiency of filtration and the reduction of pressure loss, for example.
- *Fuel injection and combustion:* combustion problems generally involve complicated geometries, complex physics, simulation of heat transfer, chemical reactions and fluid flow. The main goals in this application field are the reduction of fuel consumption, achieving

complete combustion, such that no unburnt fuel pollutes the air, reducing noise emission due to knocking behavior of the pistons, etc. In such challenging applications, the benefits from CFD are extremely valuable. One case study in the context of a combustion application in a diesel engine can be found in chapter 7 of this thesis.

- *Exhaust systems:* to account for decreasing legal limits of pollution levels, car manufacturers are obliged to optimize exhaust systems. This includes catalytic converters and recently also diesel particulate filters used to filter soot particulates from the exhaust gases produced by diesel engines (see also chapter 7 for a detailed description of such an application case). In these applications, not only the calculation of flows, heat transfers, pressure distributions, etc., but additionally also the simulation of chemical reactions are of special interest.

Other aerodynamics applications: apart from the automotive industry, also a couple of other fields take advantage of CFD technologies for aerodynamics applications on a more or less regularly basis:

- *Aerodynamics of aircrafts:* The construction of aircrafts, both for passenger transport but also for military reasons is usually started by simulating new prototypes (or even parts of them) via CFD. Thus, properties such as speed of the planes, noise during flight operation, fuel consumption, etc., can be optimized. For military applications, storage and launching of attached missiles require thorough simulations, too.
- *Space applications:* launching and operation of spacecrafts, landing maneuvers, simulations of different object movement behaviors in weightlessness; all these applications are often simulated by CFD technologies, to study and teach future missions.
- *Sports industry:* Aerodynamics play a major role in most sportive speed events or competitions. Prominent examples include racing sport divisions (Formula 1, bike racing, skiing, rally sports, etc.), and also other areas like ski jumping, for example.

Hydrodynamics and environmental applications: another very wide field of applications for CFD simulations includes all different forms of hydrodynamics problems and cases as well as environmental applications. Some of the most researched problems in these fields include

- *Floodings:* Floodings can occur due to different reasons. Possible sources for floodings are either extreme weather conditions (too much rain for longer periods, extremely fast snow melting, etc.) or the breaking of dams and similar man-made obstacles for water regulations.
- *Wind and fire:* Especially the visualization of spread and directions of large fires, e.g.; forest fires, or the track of large and destructive hurricanes are examples for current research supported by the possibilities of latest CFD results.
- *Pollution scenarios:* In this application area for CFD simulations, pollution of air and water (rivers, lakes, seas) through industry, traffic and energy production plants are examples for the main current research activities. Again catalytic converters for diesel cars, but also general filter applications for exhaust gases of industrial plants or flows of dirty water through clearwells are typical applications in this field.

- *Maritime applications:* In maritime applications CFD can be employed for a very wide spectrum of applications, including, for example, flows around rudders, hydrofoils, or propellers of ships, calculation of ship resistance measurements, outboard motor operation and effectiveness, fluid sloshing in large tankers, ...
- *Mixing of fluids:* Mixing behaviors of different fluids (two or more) open also on interesting filed of study and research, which is actively supported by CFD. In this scope applications of multi-phase flows are of special importance.

Medical and biomedical applications: these applications are usually interdisciplinary problems involving different phenomena like the interaction of moving or deforming walls with the flow field and biochemical reactions. A few examples in this field include

- *Drug delivery*: Investigating and optimizing drug delivery via inhalers pose a typical application field for employing CFD to simulate the delivery process. Of special interest is the minimization of deposition, which is the main cause for loss of large parts of the dispensed drug, as well as the simulation of the aerosol transport in human airways.
- *Blood flow*: There are many different reasons, why doctors want to investigate the flow of blood in human arteries. The detection of aneurysms is one such example. Also the design of catheters or stents are often based on results from CFD simulations. Another example is the simulation of tumor dynamics and tumor-induced angiogenesis and related drug delivery through blood vessels to the tumor cells.

Various other industrial and consumer product applications: Here a couple of interesting applications from completely different fields are listed, to show how diverse the various application fields of CFD simulations can be:

- *Molding and casting:* The simulation of the injection or cooling processes for plastic molding and metal casting applications are becoming a standard tool for faster development of moulds and optimal filling procedures.
- *Turbomachinery design:* For turbomachinery design CFD can help with simulating the flows, thermal and pressure stresses, heat and turbulence distributions, etc. A couple of different turbomachinery applications (compressible or incompressible flows) include, for example, automotive torque converters, different types of fans and centrifugal compressors, different types of pumps, and rotor dynamics analysis.
- *HVAC:* HVAC stands for **H**eating, **V**entilating, and **A**ir **C**onditioning. These three application fields are very closely related to each other, and therefore build a branch of industry on its own. Besides heating and cooling applications in many places (e.g. rooms, cars, planes, etc.), ventilation systems play a major role for safety regulations in cases of fire, for example. Simulation can in all these applications help to simplify and optimize the design and planing phase.
- *Daily used consumer products:* Other application fields are very wide spread. One field is consumer products for daily use. Simulations have been used for design and understanding of toilet flush systems, for example.

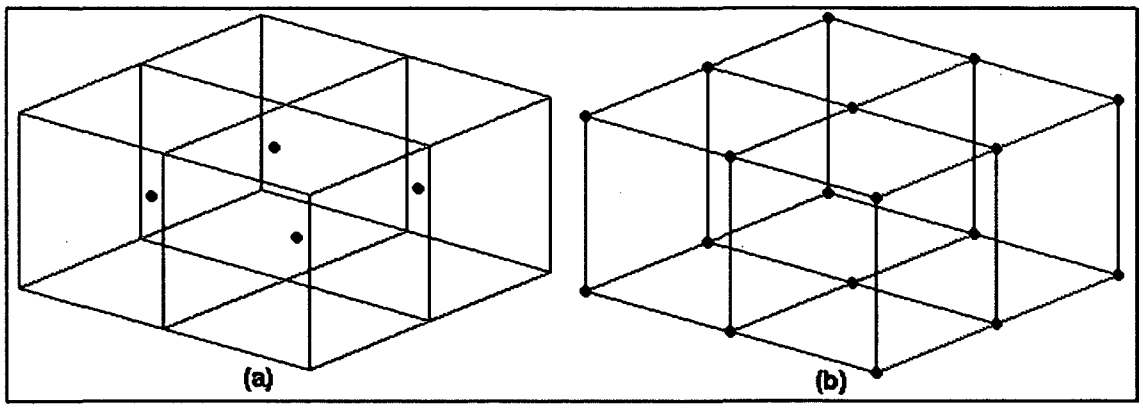


Figure B.1: Comparison of a *cell-centered grid* scheme (a) with a *mesh-centered grid* scheme (b). The large dots represent the spatial position of the points of computations [173].

Different Grid Types for CFD Data

After the discussion of different application fields, different grid types which are often used in CFD simulation, are discussed below. As already mentioned in chapter 1, the basic structures, that are used to build up grids during the discretization step are **nodes** (or **vertices**, or **points**) and **cells** (or **elements**, or **volumes**) [177]. In the following we discuss two different ways of how to categorize different types of grids, namely according to the location of the simulated data, and according to the cell types.

Grid types according to location of the data:

The first differentiation concerns the location of the simulated data. The grid can be either **mesh-centered** or **cell-centered** [173]. Mesh-centered grids are sometimes also called **node-centered**, whereas cell-centered grids are also known as **element-centered**.

When computations are performed on a mesh-centered grid, the computation points are the grid nodes or the vertices of the grid cells. With a cell-centered grid, computations are performed at the cell centers (see also figure B.1).

When a data set is based on a cell-centered grid, there is one value in the data set for each cell (for each data attribute). These data values are constant (or simple) over one complete cell. The data sets used during the work on this thesis have all been based on cell-centered grids. For other applications, with data values at the vertices of the cells, interpolation is required.

Cell-centered grids are mainly used in fluid dynamics simulations and for most problems based on PDEs. Mesh-centered grids are mostly used in structural dynamics simulations (FEM) [116].

Grid types according to cell types:

A grid can be either composed of cells of only one cell type, or it can include cells of multiple different types. In both cases, the types of cells furthermore differentiate grids into two more (very prominent) cases: **structured grids** and **unstructured grids**.

- In a structured grid, each (internal) cell of the mesh is surrounded by exactly the same

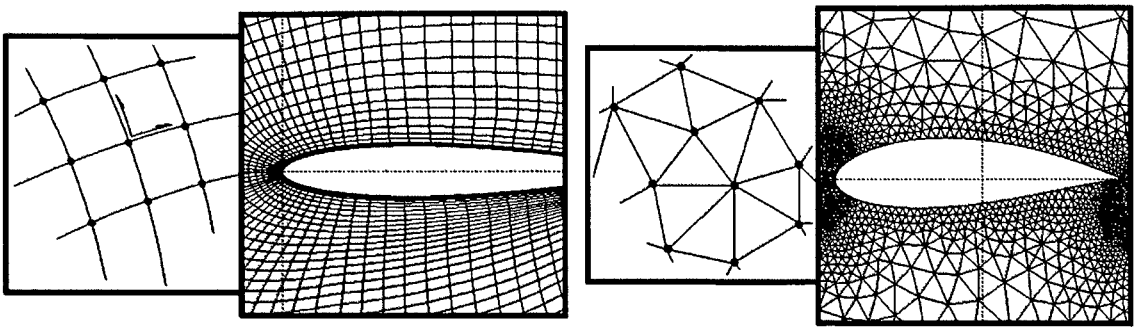


Figure B.2: Comparison of *structured grids* (left side) with an *unstructured grid* scheme (right side): for both variants an illustration and one example in 2D are presented [175].

equal number of adjacent cells. Additionally three directions within the mesh (in the 3D case) can be identified, by associating a coordinate system with the mesh faces (see also figure B.2, left side, for an illustration and one example of a structured grid in 2D).

- In unstructured grids, the number of cells surrounding one (internal) cell is not necessarily constant. Thus, in an unstructured grid an arbitrary number of cells can meet at a single node. Connections and neighboring information have to be stored explicitly for each cell. Most common representatives of unstructured grids (or meshes) are tetrahedral meshes, but also hexahedral meshes can be unstructured in certain cases (see also figure B.2, right side, for an illustration and one example of an unstructured grid in 2D).

Structured grids: grids belonging to the class of structured grids can be further subdivided into several subclasses, according to the types of cells which they are based on. In **rectilinear** grids the cells lie aligned to axes in space. If the axes are orthogonal to each other, we call the grid a **rectangular grid**. A special case of rectangular grids are **regular** grids, where the cell-length along each of the axes is constant. If the spacing between nodes along all axes is the same, then these regular grids are called **Cartesian grids**. In contrast to the subclass of rectilinear grids, **curvilinear grids** are built up along directions curving through space. Curvilinear grids are more commonly used, as modeling of curved, more complex geometries is easier compared to using only rectilinear grids. Examples of curvilinear grids include the special cases of *near-orthogonal grids*, *expansion grids*, *boundary-fitted grids*, and *circular* (or *elliptic*) *grids* (see also figure B.3 for an illustration of examples).

A special case of structured grids are so-called **multi-block structured grids**, which are consisting of multiple blocks of structured grids, which are connected to each other. Thereby, the flexibility of curvilinear grids and the simpler computational effort of rectilinear grids can be easily combined where needed (see figure B.4, left side, for a very simple illustration of an example).

Unstructured grids: when the geometry cannot be modeled using structured or multi-block structured grids, then usually unstructured grids are used. In unstructured grids, basically all cell types are possible (see also above).

Hybrid grids: Similarly to the concept of multi-block structured grids, where multiple structured grids are combined to model the simulation space, also structured and unstructured

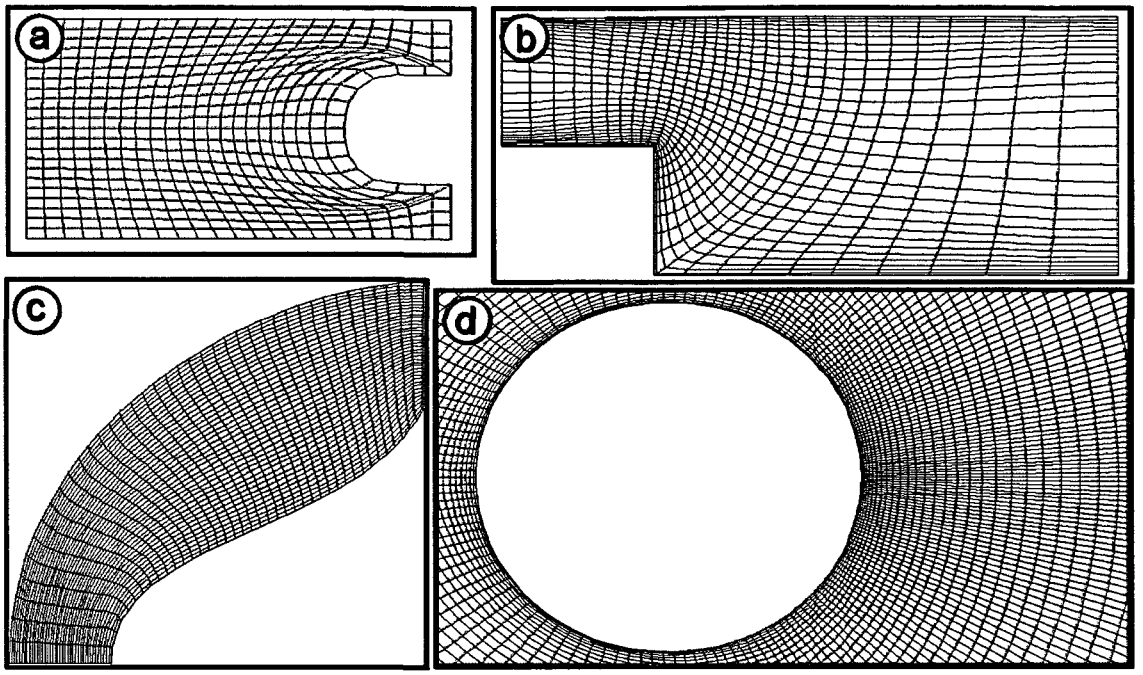


Figure B.3: Four examples of curvilinear grids: a *near-orthogonal grid* (a), an *expansion grid* (b), a *boundary-fitted grid* (c), and a *circular* (or *elliptic*) *grid* (d) (all from [62]).

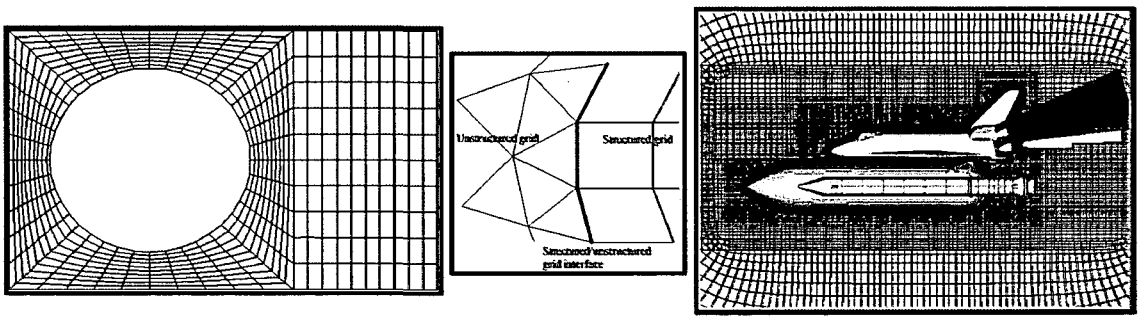


Figure B.4: Different ways of combining multiple grids: combining multiple structured grids to a *multi-block structured grid* (left); combing structured with unstructured grids in a *hybrid grid* (illustration in the middle); using multiple *overlapping grids* to represent different parts of the geometry or spatial domain (right) (all from [62]).

grids can be combined to build up hybrid grids. For an example illustration of how the connection of two such grids can be realized, see figure B.4, middle.

Overlapping grids: In contrast to using one grid to represent the full spatial simulation space, or multi-block structured and/or hybrid grids, sometimes also the combination of (spatially distinct or) overlapping grids can be required (see figure B.4, right side, for an application example).

Acknowledgments

First of all, I would like to thank my advisors *Helwig Hauser* and *Prof. Eduard M. Gröller*, whose input and help during supervision of my work provided the main basis for this work. Especially the large number of fruitful discussions with Helwig, who supervised many of my works during the last 9 years of my studies and research, provided a major contribution to the development of the SimVis framework.

A very special thank-you also goes to *Prof. Heidrun Schumann* from the University of Rostock, Germany, for showing great interest in my works during the last years, and finally accepting the position of reviewing my PhD work.

Furthermore, there have been many people, who contributed in the one or other way to the development of the SimVis technology. Especially *Martin Gasser*, who first worked as a student on two computer science projects in the field of our SimVis research, and then became a colleague at the VRVis Research Center, helped in developing parts of the SimVis prototype system. Other students which contributed to the current state of the SimVis approach in the course of their studies are (in alphabetical order): *Fabian Bendix*, *Florian Ledermann*, *Matej Mlejnek*, *Philipp Muigg*, and *Matej Novotny*. There are also a couple of colleagues at the VRVis Center, who contributed to the work on the SimVis system. I want to especially thank *Markus Hadwiger*, who helped with parts of the underlying mesh-library system, *Harald Piringer*, who developed the combined 2D/3D scatterplot view for our framework, and *Lukas Mroz*, who took part in many inspiring discussions during the design phase of the first framework prototype.

Special gratitude goes also to my colleague *Michael Mayer* at the VRVis office in Graz, Austria, who took part in intense cooperations during the two case studies presented in this thesis. Other coauthors of the publications related to this thesis, who thus contributed to the works presented include (in alphabetical order): *Robert Kosara* (VRVis), *Robert S. Laramee* (VRVis), *Frits H. Post* (TU Delft), *Benjamin Vrolijk* (TU Delft), and *Daniel Weiskopf* (University of Stuttgart).

Most of this work has been done in the VRVis Research Center, which is funded in part by the Kplus program of the Austrian government. The CFD data sets from the automotive industry shown throughout this thesis are courtesy of AVL List GmbH, Graz, Austria. The hurricane data set is courtesy of the National Center for Atmospheric Research in the United States. The stock market data set shown in the summary chapter is courtesy of the Wiener Börse AG.

Last, but not least I would like to thank especially *my family* for their great support during the last years, while I was working on this thesis. And a very special thank-you goes to my girl-friend *Bianca Balog* for tolerating the long nights writing up this thesis and for making me happy, whenever I needed her support.

Curriculum Vitae

About Helmut Doleisch:

Dipl.-Ing. Helmut Doleisch, born on 5^{th} March 1975, in Vienna, Austria, as the first son of Dr. Mag. Helga Doleisch (maiden name Weihsberger) and Dipl.-Ing. Manfred Doleisch

Contact information:

Aspettenstrasse 34/26/4, A-2380 Perchtoldsdorf, email: Doleisch@VRVis.at

Professional Activities:

- From Oct. 1996 to Jan. 1999: **Studienassistent (assistent)** at the Institute of Computer Graphics, Vienna University of Technology, Austria
- From March 1999 to Sept. 1999: **Forschungsassistent (research assistent)** at the Institute of Computer Graphics, Vienna University of Technology, Austria
- From Sept. 1999 to Aug. 2000: **Universitätsassistent (university assistent)** at the Institute of Computer Graphics, Vienna University of Technology, Austria
- Since Sept. 2000: **Junior researcher** (department: "Interactive Visualization") in the VRVis Research Center in Vienna, Austria

Activities related to scientific work:

- **2 computer science projects** during my studies of computer science at the Institute of Computer Graphics, Vienna University of Technology, about *"Visualizing Dynamical Systems and their Topology"* (March 1996 – Nov. 1997)
- My **diploma thesis project** *"Multimodal Visualization of Anatomical and Functional Volume Data of the Human Brain"* at the Institute of Computer Graphics, Vienna University of Technology (Feb. 1998 – Dec. 1998)
- **Best Paper Award** at the Spring Conference on Computer Graphics and its Applications 1999 (SCCG'99), Budmerice, Slovak Republic, April 1999
- **Session Chair** (leading presentation and discussion sessions) at the WSCG Conference in Feb. 2002, Plzen, Czech Republic
- **Reviewer for the journal** *Computers and Graphics* by Elsevier (2000)
- **Reviewer for the conferences** *IEEE Visualization* (2000–2004), the *Joint EG - IEEE TCVG Symposium on Visualization* (2002–2004), *IEEE Symposium on Information Visualization* (2004), *WSCG* (1999, 2003&2004)

Activities related to teaching:

- Coach of many **student projects** and **seminar works** in the studies of computer science, including those of M. Gasser, P. Muigg, F. Ledermann, and others
- **Organizing and lecturing** of two **advanced courses on computer graphics** in 1999 at the Vienna University of Technology, Austria
- **Teaching assistent (Tutor)** at the Institute of Computer Graphics, Vienna University of Technology in 1996 for the 2^{nd} year lab about algorithms and data structures
- **Assistent (Studienassistent)** at the Institute of Computer Graphics, Vienna University of Technology from Oct. 1996 to Jan. 1999, mainly administrating and organizing the 1^{st} and 2^{nd} year labs about algorithms and data structures (hundreds of students)

Education:

- **Volksschule** (primary school) in Perchtoldsdorf, Austria (Sept. 1981 – June 1985)
- **Bundesrealgymnasium** (combined secondary school and high school) in Perchtoldsdorf, Austria (Sept. 1985 – June 1993). **Graduation** (Matura) **with highest distinction** on June 14^{th}, 1993
- **Studies of Informatik (computer science)** at the Vienna University of Technology, Austria (Oct. 1993 – Jan. 1999), special focus on **computer graphics** from Nov. 1995 on, including more than ten special lectures and labs, five special research seminars and two computer science projects in the field of computer graphics
- Member of the **Computer Graphics Student Club** at the Institute of Computer Graphics, Vienna University of Technology (March 1996 – May 1998)
- Member of the **Visualization Group** at the Institute of Computer Graphics, Vienna University of Technology (Oct. 1996 – Aug. 2000)
- Member of the **scientific staff** at the Institute of Computer Graphics, Vienna University of Technology (March 1997 – Aug. 2000)
- Finishing of **diploma thesis** *Multimodal Visualization of Anatomical and Functional Volume Data of the Human Brain"* at the Institute of Computer Graphics, Vienna University of Technology, in Dec. 1998. **Graduation (with highest distinction) to "Diplom-Ingenieur der Informatik" (computer science)** in Jan. 1999
- Since Sept. 2000 further studies of Informatik (computer science) at the Vienna University of Technology, resulting in work on this **PhD thesis** at the VRVis Research Center

Publications and Talks:

Please see the **Related Publications** and **Bibliography** sections of this thesis for the relevant publications related to the contents of this thesis.
For a full, more up-to-date list of all my publications and talks, please visit *http://www.VRVis.at/vis/staff/doleisch/*.

Bibliography

[1] C. Ahlberg. Spotfire: an information exploration environment. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 25(4):25–29, 1996.

[2] C. Ahlberg and E. Wistrand. IVEE: an information visualization and exploration environment. In *Proc. IEEE Symposium on Information Visualization 1995 (InfoVis '95)*, pages 66–73, 142–143, 1995.

[3] J. Anderson. *Computational Fluid Dynamics: The Basics with Applications.* McGraw Hill, 1995.

[4] M. Ankerst, D. Keim, and H.-P. Kriegel. Circle Segments: A technique for visually exploring large multidimensional data sets. In *Proceedings IEEE Visualization '96*, volume Late Breaking Hot Topics, 1996.

[5] ATI web page. See URL: `http://www.ati.com/`.

[6] AVL List GmbH. See URL: `http://www.avl.com/`.

[7] AWT Native Interface. See URL: `http://java.sun.com/j2se/1.3/docs/guide/awt/AWT_Native_Interface.html`.

[8] C. Bajaj, V. Pascucci, and G. Rabbiolo. Hypervolume visualization: A challenge in simplicity. In *Proc. IEEE Symposium on Volume Visualization '98 (VolVis '98)*, pages 95–102, 1998.

[9] R. Baker. *Flow Measurement Handbook.* Cambridge Univ. Press, 2000.

[10] M. Baldonado, A. Woodruff, and A. Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, pages 110–119, 2000.

[11] D. Banks and B. Singer. Vortex tubes in turbulent flows: Identification, representation, reconstruction. In *Proceedings IEEE Visualization '94*, pages 132–139, 1994.

[12] D. Banks and B. Singer. A predictor-corrector technique for visualizing unsteady flow. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):151–163, 1995.

[13] R. Batra and L. Hesselink. Feature comparisons of 3D vector fields using earth mover's distance. In *Proceedings IEEE Visualization '99*, pages 105–114, 1999.

[14] D. Bauer and R. Peikert. Vortex Tracking in Scale-Space. In *Proc. of the 4th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2002)*, pages 233–240. Springer-Verlag, 2002.

[15] B. Becker. Volume rendering for relational data. In *Proc. IEEE Symposium on Information Visualization 1997 (InfoVis '97)*, pages 87–91, 1997.

[16] R. Becker and W. Cleveland. Brushing scatterplots. *Technometrics*, 29(2):127–142, 1987.

[17] E. Boring and A. Pang. Directional flow visualization of vector fields. In *Proceedings IEEE Visualization '96*, pages 389–392, 1996.

[18] S. Bryson, D. Kenwright, and M. Gerald-Yamasaki. FEL: The field encapsulation library. In *Proceedings IEEE Visualization '96*, pages 241–248, 1996.

[19] S. Bryson and C. Levit. The Virtual Wind Tunnel. *IEEE Computer Graphics and Applications*, 12(4):25–34, 1992.

[20] A. Buja, J. McDonald, J. Michalak, and W. Stuetzle. Interactive data visualization using focusing and linking. In *Proceedings IEEE Visualization '91*, pages 156–163, 1991.

[21] B. Cabral and L. Leedom. Imaging Vector Fields Using Line Integral Convolution. In *Poceedings of ACM SIGGRAPH 1993*, Annual Conference Series, pages 263–272. ACM Press / ACM SIGGRAPH, 1993.

[22] S. Card, J. MacKinlay, and B. Shneiderman. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers, 1998.

[23] CFD Online: An online repository for CFD resources. See URL: `http://www.cfd-online.com`.

[24] CFD Review: Serving the CFD Community with News, Articles, and Discussions. See URL: `http://www.cfdreview.com`.

[25] H. Chen. Compound brushing. In *Proc. IEEE Symposium on Information Visualization 2003 (InfoVis 2003)*, pages 181–188, 2003.

[26] H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 68:361–368, 1973.

[27] W. Cleveland. *The Elements of Graphing Data*. Wadsworth Inc, 1985.

[28] J. Clyne and J. Dennis. Interactive direct volume rendering of time-varying data. In *Proc. of the 1st Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym '99)*, pages 109–120. Springer-Verlag, 1999.

[29] P. Craig and J. Kennedy. Coordinated graph and scatter-plot views for the visual exploration of microarray time-series data. In *Proc. IEEE Symposium on Information Visualization 2003 (InfoVis 2003)*, pages 173–180, 2003.

[30] R. Crawfis, N. Max, B. Becker, and B. Cabral. Volume rendering of 3D scalar and vector fields at LLNL. In *Proceedings, Supercomputing '93: Portland, Oregon, November 15–19, 1993*, pages 570–576. IEEE Computer Society, 1993.

[31] B. Csébfalvi, L. Mroz, H. Hauser, A. König, and M. Gröller. Fast visualization of object contours by non-photorealistic volume rendering. *Computer Graphics Forum*, 20(3):452–460, 2001.

[32] Online information service on clean diesel engines and diesel emissions. See URL `http://www.dieselnet.com/`.

[33] H. Doleisch, M. Gasser, and H. Hauser. Interactive feature specification for focus+context visualization of complex simulation data. In *Proc. of the 5th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2003)*, pages 239–248. Springer-Verlag, 2003.

[34] H. Doleisch and H. Hauser. State of the art report 2000 on feature-based flow visualization. Technical Report TR-VRVis-2000-004, VRVis Research Center, 2000.

[35] H. Doleisch and H. Hauser. Smooth Brushing for Focus+Context Visualization of Simulation Data in 3D. In *Journal of WSCG*, volume 10, pages 147–154, Plzen, 2002.

[36] H. Doleisch, H. Hauser, M. Gasser, and R. Kosara. Interactive focu+context analysis of large, time-dependent flow simulation data. Technical Report TR-VRVis-2004-024, currently in submission, VRVis Research Center, 2004.

[37] H. Doleisch, M. Mayer, M. Gasser, P. Priesching, and H. Hauser. Interactive feature specification for the visual analysis of a diesel engine. Technical Report TR-VRVis-2004-011, currently in submission, VRVis Research Center, 2004.

[38] H. Doleisch, M. Mayer, M. Gasser, R. Wanker, and H. Hauser. Case Study: Visual Analysis of Complex, Time-Dependent Simulation Results of a Diesel Exhaust System. In *Proc. of the 6th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2004)*, pages 91–96. Springer-Verlag, 2004.

[39] H. Doleisch, P. Muigg, and H. Hauser. Interactive Visual Analysis of Hurricane Isabel with SimVis. See URL: `http://www.VRVis.at/SimVis/Isabel/`, 2004.

[40] D. Ebert and P. Rheingans. Volume illustration: Nonphotorealistic rendering of volume models. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):253–264, 2001.

[41] D. Ebert, R. Rohrer, C. Shaw, P. Panda, J. Kukla, and D. Roberts. Procedural shape generation for multi-dimensional data visualization. *Computers and Graphics*, 24(3):375–384, 2000.

[42] D. Ebert, R. Yagel, J. Scott, and Y. Kurzion. Volume rendering methods for computational fluid dynamics visualization. In *Proceedings IEEE Visualization '94*, pages 232–239, 1994.

[43] Webpage about FAME at AVL. See URL: `http://www.avl.com/fame`.

[44] A. Fenlon and T. David. An integrated visualization and design toolkit for flexible prosthetic heart valves. In *Proceedings IEEE Visualization 2000*, pages 453–456, 2000.

[45] A. Filippone. Advanced Topics in Aerodynamics: The Wind Tunnel. See URL: `http://aerodyn.org/WindTunnel/wind_tunnel.html`, 2004.

[46] Webpage about FIRE at AVL. See URL: `http://www.avl.com/fire`.

[47] T. Frühauf. Raycasting vector fields. In *Proceedings IEEE Visualization '96*, pages 115–120, 1996.

[48] Y. Fua, M. Ward, and E. 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.

[49] Y.-H. Fua, M. Ward, and E. Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. In *Proceedings IEEE Visualization '99*, pages 43–50, 1999.

[50] G. 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.

[51] G. Furnas. Generalized fisheye views. In *Proc. of the ACM CHI '86 Conf. on Human Factors in Computing Systems*, pages 16–23, 1986.

[52] G. Furnas and A. Buja. Prosection views: dimensional inference through sections and projections. with a discussion by John F. Elder IV, Shingo Oue and Daniel B. Carr and a rejoinder by the authors. *Journal of Computational and Graphical Statistics*, 3(4):323–385, 1994.

[53] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley, 1995.

[54] H. Garcke, T. Preußer, M. Rumpf, A. Telea, U. Weikard, and J. van Wijk. A continuous clustering method for vector fields. In *Proceedings IEEE Visualization 2000*, pages 351–358, 2000.

[55] H. Garcke, T. Preußer, M. Rumpf, A. Telea, U. Weikard, and J. van Wijk. A phase field model for continuous clustering on vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):230–241, 2001.

[56] M. Gasser. Fast focus+context visualization of large scientific data. Technical Report TR-VRVis-2004-005, VRVis Research Center, 2004.

[57] GL4Java API. See URL: `http://www.jausoft.com/gl4java.html`.

[58] T. Glau. Exploring instationary fluid flows by interactive volume movies. In *Proc. of the 1st Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym '99)*, pages 277–283. Springer-Verlag, 1999.

[59] A. Globus, C. Levit, and T. Lasinski. A tool for visualizing the topology of three-dimensional vector fields. In *Proceedings IEEE Visualization '91*, pages 33–40, 1991.

[60] D. Gresh, D. Rabenhorst, A. Shabo, and S. Slavin. PRIMA: A case study of using information visualization techniques for patient record analysis. In *Proceedings IEEE Visualization 2002*, pages 509–512, 2002.

[61] D. Gresh, B. Rogowitz, R. Winslow, D. Scollan, and C. Yung. WEAVE: A system for visually linking 3-D and statistical visualizations, applied to cardiac simulation and measurement data. In *Proceedings IEEE Visualization 2000*, pages 489–492, 2000.

[62] Introduction to Grid Generation. See URL: `http://www.sintef.no/static/am/comp/software/griddler/course/intro_gridgen.ppt`. published by SINTEF, Norway.

[63] M. in den Haak, H. Spoelder, and F. Groen. Matching of images by using automatically selected regions of interest. In *Computing Science in the Netherlands '92*, pages 27–40, 1992.

[64] R. Haimes. Using residence time for the extraction of recirculation regions. Technical Report AIAA-99-3291, Department of Aeronautics and Astronautics, Massachusettes Institute of Technology, 1999.

[65] R. Haimes and D. Darmofal. Visualization in computational fluid dynamics: A case study. In *Proceedings IEEE Visualization '91*, pages 392–397, 1991.

[66] H. Hauser. Generalizing focus+context visualization. In *Dagstuhl Seminar 03231: Scientific Visualization: Extracting Information and Knowledge from Scientific Data Sets, also available as VRVis Technical Report TR-VRVis-2003-037*, 2003.

[67] H. Hauser, R. Laramee, and H. Doleisch. State-of-the-Art Report 2002 on Flow Visualization. Technical Report TR-VRVis-2002-003, VRVis Research Center, 2002.

[68] H. Hauser, F. Ledermann, and H. Doleisch. Angular brushing of extended parallel coordinates. In *Proc. IEEE Symposium on Information Visualization 2002 (InfoVis 2002)*, pages 127–130, 2002.

[69] H. Hauser and M. Mlejnek. Interactive volume visualization of complex flow semantics. In *Proc. of the 8th Fall Workshop on Vision, Modeling and Visualization (VMV 2003)*, pages 191–198, 2003.

[70] H. Hauser, L. Mroz, G. Bischi, and E. Gröller. Two-level volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):242–252, 2001.

[71] H. Hauser, L. Mroz, G.-I. Bischi, and M. E. Gröller. Two-level volume rendering - fusing MIP and DVR. In *Proceedings IEEE Visualization 2000*, pages 211–218, 2000.

[72] B. Heckel, G. Weber, B. Hamann, and K. Joy. Construction of vector field hierarchies. In *Proceedings IEEE Visualization '99*, pages 19–26, 1999.

[73] J. Helman and L. Hesselink. Representation and Display of Vector Field Topology in Fluid Flow Data Sets. *IEEE Computer*, 22(8):27–36, August 1989.

[74] J. Helman and L. Hesselink. Surface representations of two- and three-dimensional fluid flow topology. In *Proceedings IEEE Visualization '90*, pages 6–13, 1990.

[75] J. Helman and L. Hesselink. Visualizing Vector Field Topology in Fluid Flows. *IEEE Computer Graphics and Applications*, 11(3):36–46, May 1991.

[76] C. Henze. Feature detection in linked derived spaces. In *Proceedings IEEE Visualization '98*, pages 87–94, 1998.

[77] W. Hibbard and D. Santek. Interactivity is the Key. In *Proceedings of the Chapel Hill Workshop on Volume Visualization*, pages 39–43, 1989.

[78] Hyper-Threading Technology Architecture and Microarchitecture. See URL: `http://www.intel.com/technology/itj/2002/volume06issue01/vol6iss1_hyper_threading_technology.pdf`.

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

[80] A. Inselberg and B. Dimsdale. Parallel coordinates for visualizing multi-dimensional geometry. In *Computer Graphics 1987 (Proceedings of CG International '87)*, pages 25–44, 1987.

[81] A. Inselberg and B. Dimsdale. Parallel coordinates: a tool for visualizing multidimensional geometry. In *Proceedings IEEE Visualization '90*, pages 361–378, 1990.

[82] J. Jeong and F. Hussain. On the identification of a vortex. *Journal of Fluid Mechanics*, 285:69–84, 1995.

[83] M. Jiang, R. Machiraju, and D. Thompson. A Novel Approach to Vertex Core Region Detection. In *Proc. of the 4th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2002)*, pages 217–225, 2002.

[84] Java Native Interface. See URL: `http://java.sun.com/j2se/1.3/docs/guide/jni/`.

[85] B. Jobard and W. Lefer. Creating Evenly–Spaced Streamlines of Arbitrary Density. In *Proc. of the 8th Eurographics Workshop on Visualization in Scientific Computing '97*, pages 43–56. Springer-Verlag, 1997.

[86] B. Jobard and W. Lefer. Unsteady flow visualization by animating evenly-spaced streamlines. *Computer Graphics Forum (Eurographics 2000)*, 19(3):31–39, 2000.

[87] D. Kalivas and A. Sawchuk. A region matching motion estimation algorithm. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 54(2):275–288, 1991.

[88] T. Keahey and E. Robertson. Techniques for non-linear magnification transformations. In *Proc. IEEE Symposium on Information Visualization 1996 (InfoVis '96)*, pages 38–45, 1996.

[89] D. Keim. Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):59–78, 2000.

[90] D. Keim, M. Hao, and U. Dayal. Hierarchical pixel bar charts. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):255–269, 2002.

[91] D. Keim, M. Hao, J. Ladisch, M. Hsu, and U. Dayal. Pixel bar charts: A new technique for visualizing large multi-attribute data sets without aggregation. In *Proceedings IEEE Visualization 2001*, pages 113–120, 2001.

[92] D. Keim and H. Kriegel. VisDB: Database exploration using multidimensional visualization. *IEEE Computer Graphics and Applications*, 14(5):40–49, 1994.

[93] D. Keim, H.-P. Kriegel, and M. Ankerst. Recursive pattern: A technique for visualizing very large amounts of data. In *Proceedings IEEE Visualization '95*, pages 297–286, 1995.

[94] D. Keim, W. Müller, and H. Schumann. Visual data mining. In *Eurographics 2002 State-of-the-Art Reports*, pages 49–68, 2002.

[95] D. Kenwright. Automatic detection of open and closed separation and attachment lines. In *Proceedings IEEE Visualization '98*, pages 151–158, 1998.

[96] D. Kenwright and R. Haimes. Vortex identification - applications in aerodynamics. In *Proceedings IEEE Visualization '97*, pages 413–416, 1997.

[97] D. Kenwright and R. Haimes. Automatic Vortex Core Detection. *IEEE Computer Graphics and Applications*, 18(4):70–74, 1998.

[98] D. Kenwright, C. Henze, and C. Levit. Feature extraction of separation and attachment lines. *IEEE Transactions on Visualization and Computer Graphics*, 5(2):135–144, 1999.

[99] R. Kirby, H. Marmanis, and D. Laidlaw. Visualizing Multivalued Data from 2D Incompressible Flows Using Concepts from Painting. In *Proceedings IEEE Visualization '99*, pages 333–340, 1999.

[100] R. Klassen and S. Harrington. Shadowed hedgehogs: A technique for visualizing 2D slices of 3D vector fields. In *Proceedings IEEE Visualization '91*, pages 148–153, 1991.

[101] E. Klement, R. Mesiar, and E. Pap. *Triangular Norms*, volume 8 of *Trends in Logic*. Kluwer Academic Publishers, 2000.

[102] J. Kniss, G. Kindlmann, and C. Hansen. Interactive volume rendering using multidimensional transfer functions and direct manipulation widgets. In *Proceedings IEEE Visualization 2001*, pages 255–262, 2001.

[103] R. Kosara, F. Bendix, and H. Hauser. TimeHistograms for large, time-dependent data. In *Proc. of the 6th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2004)*, pages 45–54, 2004.

[104] R. Kosara, H. Doleisch, M. Gasser, and H. Hauser. The SimVis system for interactive visual analysis of flow simulation data. In *Proceedings of the 2004 Conference "Virtual Product Development (VPD) in Automotive Engineering"*, 2004.

[105] R. Kosara, H. Hauser, and D. Gresh. An interaction view on information visualization. In *Eurographics 2003 State-of-the-Art Reports*, pages 123–137, 2003.

[106] R. Kosara, S. Miksch, and H. Hauser. Semantic depth of field. In *Proc. IEEE Symposium on Information Visualization 2001 (InfoVis 2001)*, 2001.

[107] R. Kosara, S. Miksch, and H. Hauser. Focus + context taken literally. *IEEE Computer Graphics and Applications*, 22(1):22–29, 2002.

[108] R. Kosara, G. Sahling, and H. Hauser. Linking scientific and information visualization with interactive 3D scatterplots. In *Proceedings of the 12th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*, pages 133–140, 2004.

[109] M. Kreuseler, N. López, and H. Schumann. A scalable framework for information visualization. In *Proc. IEEE Symposium on Information Visualization 2000 (InfoVis 2000)*, pages 27–38, 2000.

[110] J. Lamping and R. Rao. The hyperbolic browser: A focus+context technique for visualizing large hierarchies. *Journal of Visual Languages and Computing*, 7(1):33–55, 1996.

[111] J. Lamping, R. Rao, and P. Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proc. of the ACM CHI '95 Conf. on Human Factors in Computing Systems*, pages 401–408, 1995.

[112] R. Laramee and H. Hauser. Geometric flow visualization techniques for cfd simulation data. Technical Report TR-VRVis-2004-029, VRVis Research Center, 2004.

[113] R. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F. Post, and D. Weiskopf. The state of the art in flow visualization: Dense and texture-based techniques. *Computer Graphics Forum*, 23(2), 2004.

[114] R. Laramee, B. Jobard., and H. Hauser. Image Space Based Visualization of Unsteady Flow on Surfaces. In *Proceedings IEEE Visualization 2003*, pages 131–138, 2003.

[115] Y. Lavin, R. Batra, and L. Hesselink. Feature comparisons of vector fields using earth mover's distance. In *Proceedings IEEE Visualization '98*, pages 103–110, 1998.

[116] O. Lawlor. Data Structures for Scientific Computing. See URL: `http://charm.cs.uiuc.edu/presentations/charm_scientific_data2003.htm`, 2003. Presentation at the Center for Simulation of Advanced Rockets (CSAR), University of Illinois at Urbana-Champaign.

[117] B. van Leer. The computation of steady solutions to the euler equations: A perspective. Technical report, University of Michigan, 1986.

[118] W. de Leeuw and R. van Liere. Collapsing Flow Topology Using Area Metrics. In *Proceedings IEEE Visualization '99*, pages 349–354, 1999.

[119] W. de Leeuw and R. van Liere. Visualization of Global Flow Structures Using Multiple Levels of Topology. In *Proc. of the 1st Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym '99)*, pages 45–52. Springer-Verlag, 1999.

[120] W. de Leeuw and R. van Liere. Multi-level Topology for Flow Visualization. *Computers and Graphics*, 24(3):325–331, 2000.

[121] W. de Leeuw, H.-G. Pagendarm, F. Post, and B. Walter. Visual Simulation of Experimental Oil-Flow Visualization by Spot Noise from Numerical Flow Simulation. In *Proc. of the 6th Eurographics Workshop on Visualization in Scientific Computing '95*, pages 135–148. Springer-Verlag, 1995.

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

[123] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8(3):29–37, 1988.

[124] Y. Levy, D. Degani, and A. Seginer. Graphical visualization of vortical flows by means of helicity. *AIAA Journal*, 28(8):1347–1352, 1990.

[125] X. Li and H. Shen. Adaptive volume rendering using fuzzy logic control. In *Proc. of the 3rd Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2001)*, pages 253–262, 2001.

[126] H. Löffelmann. *Visualizing Local Properties and Characteristic Structures of Dynamical Systems.* PhD thesis, Vienna University of Technology, Austria, November 1998. See also `http://www.VRVis.at/vis/resources/diss-HL/`.

[127] H. Löffelmann, H. Doleisch, and E. Gröller. Visualizing dynamical systems near critical points. In *14th Spring Conference on Computer Graphics*, pages 175–184, 1998.

[128] H. Löffelmann and E. Gröller. Enhancing the visualization of characteristic structures in dynamical systems. In *Proc. of the 9th Eurographics Workshop on Visualization in Scientific Computing '98*, pages 59–68. Springer-Verlag, 1998.

[129] D. Lovely and R. Haimes. Shock detection from computational fluid dynamics results. In *Proceedings of the 14th AIAA Computational Fluid Dynamics Conference*, 1999.

[130] K.-L. Ma, J. van Rosendale, and W. Vermeer. 3D shock wave visualization on unstructured grids. In *Proc. IEEE Symposium on Volume Visualization '96 (VolVis '96)*, pages 87–94, 1996.

[131] J. Mackinlay, G. Robertson, and S. Card. The perspective wall: Detail and context smoothly integrated. In *Proceedings of ACM CHI'91 Conference on Human Factors in Computing Systems and Graphics Interface, ACM SIGCHI*, 1991.

[132] A. Martin and M. Ward. High dimensional brushing for interactive exploration of multivariate data. In *Proceedings IEEE Visualization '95*, pages 271–278, 1995.

[133] N. Max, R. Crawfis, and D. Williams. Visualizing Wind Velocities by Advecting Cloud Textures. In *Proceedings IEEE Visualization '92*, pages 179–184, 1992.

[134] P. Moin and J. Kim. The structure of the vorticity field in turbulent channel flow. part 1. analysis of instantanious fields and statistical correlations. *Journal of Fluid Mechanics*, 155:441–, 1985.

[135] K. Morton and M. Rudgyard. Shock recovery and the cell vertex scheme for the steady euler equations. In D. Dowyer, M. Hussaini, and R. Voigt, editors, *Lecture Notes in Physics*, pages 424–428. Springer-Verlag, 1989.

[136] L. Mroz and H. Hauser. RTVR - a flexible java library for interactive volume rendering. In *Proceedings IEEE Visualization 2001*, pages 279–286, 2001.

[137] T. Munzner. Exploring large graphs in 3D hyperbolic space. *IEEE Computer Graphics and Applications*, 18(4):18–23, 1998.

[138] D. Musser. Introspective sorting and selection algorithms. *Software Practice and Experience*, 27(8):983–993, 1997.

[139] NVIDIA web page. See URL: `http://www.nvidia.com/`.

[140] NVIDIA OpenGL Specs. See URL: `http://developer.nvidia.com/object/nvidia_opengl_specs.html`.

[141] K. Ono, H. Matsumoto, and R. Himeno. Visualization of thermal flows in an automotive cabin with volume rendering method. In *Proc. of the 3rd Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2001)*, pages 301–308. Springer-Verlag, 2001.

[142] H.-G. Pagendarm, B. Henne, and M. Rutten. Detecting vortical phenomena in vector data by medium-scale correlation. In *Proceedings IEEE Visualization '99*, pages 409–412, 1999.

[143] H.-G. Pagendarm and B. Seitz. An algorithm for detection and visualization of discontinuities in scientific data fields applied to flow data with shock waves. In *Scientific Visualization: Advanced Software Techniques*, pages 161–177. Ellis Horwood Limited, 1993.

[144] H.-G. Pagendarm and B. Seitz. Feature detection from vector quantities in a numerically simulated hypersonic flow field in combination with experimental flow visualization. In *Proceedings IEEE Visualization '94*, pages 117–123, 1994.

[145] R. Peikert and M. Roth. The Parallel Vectors Operator - A Vector Field Visualization Primitive. In *Proceedings IEEE Visualization '99*, pages 263–270, 1999.

[146] H.-P. Pfister, B. Lorensen, Ch. Bajaj, G. Kindlmann, W. Schroeder, L. Sobierajski-Avila, K. Martin, R. Machiraju, and J. Lee. Visualization viewpoints: The transfer function bake-off. *IEEE Computer Graphics and Applications*, 21(3):16–23, 2001.

[147] R. Pickett and G. Grinstein. Iconographic displays for visualizing multidimensional data. In *Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, pages 514–519, 1988.

[148] H. Piringer, R. Kosara, and H. Hauser. Interactive focus+context visualization with linked 2D/3D scatterplots. In *Proc. of the Intl. Conference on Coordinated & Multiple Views in Exploratory Visualization (CMV 2004)*, pages 49–60, 2004.

[149] L. Portela. *On the Identification and Classification of Vortices.* PhD thesis, Stanford University, School of Mechanical Engineering, 1997.

[150] F. Post, B. Vrolijk, H. Hauser, R. Laramee, and H. Doleisch. Feature extraction and visualization of flow fields. In *Eurographics 2002 State-of-the-Art Reports*, pages 69–100, Saarbrücken Germany, 2002.

[151] F. Post, B. Vrolijk, H. Hauser, R. Laramee, and H. Doleisch. The state of the art in flow visualization: Feature extraction and tracking. *Computer Graphics Forum*, 22(2):775–792, 2003.

[152] G. Reina and T. Ertl. Volume visualization and visual queries for large high-dimensional datasets. In *Proc. of the 6th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2004)*, pages 255–260, Konstanz, Germany, 2004.

[153] F. Reinders. *Feature-Based Visualization of Time-Dependent Data.* PhD thesis, Delft University of Technology, The Netherlands, March 2001.

[154] F. Reinders, M. Jacobson, and F. Post. Skeleton graph generation for feature shape description. In *Proc. of the 2th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2000)*, pages 73–82. Springer-Verlag, 2000.

[155] F. Reinders, F. Post, and H. Spoelder. Attribute-based feature tracking. In *Proc. of the 1st Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym '99)*, pages 63–72. Springer-Verlag, 1999.

[156] F. Reinders, F. Post, and H. Spoelder. Visualization of Time-Dependent Data with Feature Tracking and Event Detection. *The Visual Computer*, 17(1):55–71, 2001.

[157] F. Reinders, H. Spoelder, and F. Post. Experiments on the accuracy of feature extraction. In *Proc. of the 9th Eurographics Workshop on Visualization in Scientific Computing '98*, pages 49–58. Springer-Verlag, 1998.

[158] J. van Rosendale. Floating shock fitting via lagrangian adaptive meshes. Technical Report 94-89, Institute for Computer Applications in Science and Engineering, 1994.

[159] M. Roth. *Automatic Extraction of Vortex Core Lines and Other Line-Type Features for Scientific Visualization.* PhD thesis, Swiss Federal Institute of Technology, ETH Zürich, 2000.

[160] M. Roth and R. Peikert. Flow visualization for turbomachinery design. In *Proceedings IEEE Visualization '96*, pages 381–384, 1996.

[161] M. Roth and R. Peikert. A higher-order method for finding vortex core lines. In *Proceedings IEEE Visualization '98*, pages 143–150, 1998.

[162] I. Sadarjoen and F. Post. Deformable surface techniques for field visualization. *Proceedings of Eurographics '97*, 16(3):109–116, 1997.

[163] I. Sadarjoen and F. Post. Geometric methods for vortex extraction. In *Proc. of the 1st Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym '99)*, pages 53–62, 1999.

[164] I. Sadarjoen and F. Post. Detection, Quantification, and Tracking of Vortices using Streamline Geometry. *Computers and Graphics*, 24(3):333–341, 2000.

[165] I. Sadarjoen and F. Post. Techniques and applications of deformable surfaces. In *Scientific Visualization, Proceedings Dagstuhl '97*, pages 277–286. IEEE Computer Society, 2000.

[166] I. Sadarjoen, F. Post, B. Ma, D. Banks, and H. G. Pagendarm. Selective visualization of vortices in hydrodynamic flows. In *Vis98*, pages 419–422, 1998.

[167] A. Salih. Computational Fluid Dynamics – An Introduction. See URL: `http://www.sali.freeservers.com/engineering/cfd/`, 2004. Introduction and Link-Collection for CFD Resources, Computational Fluid Dynamics Researcher, Indian Institut of Technology, Bombay.

[168] R. Samtaney, D. Silver, N. Zabusky, and J. Cao. Visualizing features and tracking their evolution. *IEEE Computer*, 27(7):20–27, 1994.

[169] M. Sarkar and M. Brown. Graphical Fisheye Views. *Communications of the ACM*, 37(12):73–83, 1994.

[170] G. Scheuermann, H. Hagen, H. Krüger, M. Menzel, and A. Rockwood. Visualization of Higher Order Singularities in Vector Fields. In *Proceedings IEEE Visualization '97*, pages 67–74, 1997.

[171] G. Scheuermann, H. Krüger, M. Menzel, and A. P. Rockwood. Visualizing nonlinear vector field topology. *IEEE Transactions on Visualization and Computer Graphics*, 4(2):109–116, 1998.

[172] G. Scheuermann, X. Tricoche, and H. Hagen. C1-interpolation for vector field topology visualization. In *Proceedings IEEE Visualization '99*, pages 271–278, 1999.

[173] W. Schroeder, K. Martin, and W. Lorensen. *The Visualization Toolkit: An Object Oriented Approach to Computer Graphics.* Prentice-Hall, 2nd edition, 1998.

[174] M. Schulz, F. Reck, W. Bartelheimer, and T. Ertl. Interactive Visualization of Fluid Dynamics Simulations in Locally Refined Cartesian Grids. In *Proceedings IEEE Visualization '99*, pages 413–416, 1999.

[175] R. Selvam and S. Govindaswamy. Aeroelastic Analysis of Bridge Girder Section using Computer Modeling. See URL: `http://ntl.bts.gov/DOCS/ch5.html`, 2001. A Report for the Mack Blackwell Transportation Center, stored at the National Transportation Library.

[176] K. Setarehdan, S. Singh, and J. Suri. *Advanced Algorithmic Approaches to Medical Image Segmentation.* Springer-Verlag UK, 2002.

[177] C. Shaw. *Using Computational Fluid Dynamics.* Prentice Hall, 1992.

[178] B. Shneiderman. Dynamic queries for visual information seeking. *IEEE Software*, 11(6):70–77, 1994.

[179] C. Silva, J. Mitchell, and P. Williams. An exact interactive time visibility ordering algorithm for polyhedral cell complexes. In *Proc. IEEE Symposium on Volume Visualization '98 (VolVis '98)*, pages 87–94, 1998.

[180] D. Silver and X. Wang. Volume tracking. In *Proceedings IEEE Visualization '96*, pages 157–164, 1996.

[181] D. Silver and X. Wang. Tracking and Visualizing Turbulent 3D Features. *IEEE Transactions on Visualization and Computer Graphics*, 3(2):129–141, 1997.

[182] D. Silver and X. Wang. Tracking features in unstructured datasets. In *Proceedings IEEE Visualization '98*, pages 79–86, 1998.

[183] D. Silver and X. Wang. Visualizing evolving scalar phenomena. *Journal of Future Generations of Computer Systems*, 15(1):99–108, 1999.

[184] Project home page of presented work. See URL: `http://www.VRVis.at/vis/research/simvis-time/`.

[185] Intel Streaming SIMD Extensions. See URL: `http://x86.ddj.com/articles/sse_pt1/simd1.htm`.

[186] C. Stolte, D. Tang, and P. Hanrahan. Multiscale visualization using data cubes. In *Proc. IEEE Symposium on Information Visualization 2002 (InfoVis 2002)*, pages 7–14, 2002.

[187] C. Stolte, D. Tang, and P. Hanrahan. Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):52–65, 2002.

[188] D. Sujudi and R. Haimes. Identification of swirling flow in 3D vector fields. Technical Report AIAA-95-1715, American Institute of Aeronautics and Astronautics, 1995.

[189] J. Swan, M. Lanzagorta, D. Maxwell, E. Kou, J. Uhlmann, W. Anderson, H. Shyu, and W. Smith. A computational steering system for studying microwave interactions with missile bodies. In *Proceedings IEEE Visualization 2000*, pages 441–444, 2000.

[190] D. Swayne, D. Cook, and A. Buja. XGobi: interactive dynamic graphics in the X window system with a link to S. In *Proceedings of the ASA Section on Statistical Graphics*, pages 1–8. American Statistical Association, 1991.

[191] D. Swayne, D. Cook, and A. Buja. XGobi: Interactive dynamic data visualization in the X window system. *Journal of Computational and Graphical Statistics*, 7(1), 1998.

[192] Webpage about SWIFT at AVL. See URL: `http://www.avl.com/swift`.

[193] A. Telea and J. van Wijk. Simplified Representation of Vector Fields. In *Proceedings IEEE Visualization '99*, pages 35–42, 1999.

[194] A. Telea and J. van Wijk. 3D IBFV: Hardware-Accelerated 3D Flow Visualization. In *Proceedings IEEE Visualization 2003*, pages 233–240, 2003.

[195] H. Theisel. Higher order parallel coordinates. In *Proc. of the 5th Fall Workshop on Vision, Modeling and Visualization (VMV 2000)*, pages 415–420, 2000.

[196] L. Treinish. Multi-resolution visualization techniques for nested weather models. In *Proceedings IEEE Visualization 2000*, pages 513–516, 2000.

[197] X. Tricoche, G. Scheuermann, and H. Hagen. A topology simplification method for 2D vector fields. In *Proceedings IEEE Visualization 2000*, 2000.

[198] X. Tricoche, G. Scheuermann, H. Hagen, and S. Clauss. Vector and Tensor Field Topology Simplification on Irregular Grids. In *Proc. of the 3rd Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2001)*, pages 107–116. Springer-Verlag, 2001.

[199] X. Tricoche, T. Wischgoll, G. Scheuermann, and H. Hagen. Topology tracking for the visualization of time-dependent two-dimensional flows. *Computers and Graphics*, 26(2):249–257, 2002.

[200] G. Turk and D. Banks. Image-Guided Streamline Placement. In *SIGGRAPH 96 Conference Proceedings*, pages 453–460, 1996.

[201] Specification of ARB_vertex_buffer_object. See URL: `http://oss.sgi.com/projects/ogl-sample/registry/ARB/vertex_buffer_object.txt`.

[202] V. Verma, D. Kao, and A. Pang. A Flow-guided Streamline Seeding Strategy. In *Proceedings IEEE Visualization 2000*, pages 163–170, 2000.

[203] J. Villasenor and A. Vincent. An algorithm for space recognition and time tracking of vorticity tubes in turbulence. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 55(1):27–35, 1992.

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

[205] VRVis Research Center. See URL: `http://www.VRVis.at/`.

[206] T. van Walsum. *Selective Visualization on Curvilinear Grids.* PhD thesis, Delft University of Technology, Dec. 1995.

[207] T. van Walsum, F. H. Post, D. Silver, and F. J. Post. Feature Extraction and Iconic Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2(2):111–119, 1996.

[208] M. Ward. XmdvTool: Integrating multiple methods for visualizing multivariate data. In *Proceedings IEEE Visualization '94*, pages 326–336, 1994.

[209] C. Weigle and D. Banks. Extracting iso-valued features in 4-dimensional scalar fields. In *Proc. IEEE Symposium on Volume Visualization '98 (VolVis '98)*, pages 103–110, 1998.

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

[211] P. Wesseling. *Principles of Computational Fluid Dynamics.* Springer Verlag, 2000.

[212] R. Westermann. The rendering of unstructured grids revisited. In *Proc. of the 3rd Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2001)*, pages 65–74. Springer-Verlag, 2001.

[213] J. van Wijk. Spot noise-Texture Synthesis for Data Visualization. In *Computer Graphics (Proceedings of ACM SIGGRAPH 91)*, volume 25, pages 309–318, 1991.

[214] J. van Wijk. Image Based Flow Visualization. *ACM Transactions on Graphics*, 21(3):745–754, 2002.

[215] J. van Wijk. Image Based Flow Visualization for Curved Surfaces. In *Proceedings IEEE Visualization 2003*, pages 123–130, 2003.

[216] G. Wills. 524,288 ways to say "this is interesting". In *Proc. IEEE Symposium on Information Visualization 1996 (InfoVis '96)*, pages 54–61, 1996.

[217] P. Wong and R. Bergeron. Multiresolution multidimensional wavelet brushing. In *Proceedings IEEE Visualization '96*, pages 141–148, 1996.

[218] P. Wong, H. Foote, R. Leung, E. Jurrus, D. Adams, and J. Thomas. Vector fields simplification - a case study of visualizing climate modeling and simulation data sets. In *Proceedings IEEE Visualization 2000*, 2000.

[219] Webpage of the World Wide Web Consortium about XML. See URL `http://www.w3.org/XML/`.

[220] M. Xue. Computational Fluid Dynamics – An Introduction. See URL: `http://twister.caps.ou.edu/CFD2003/`, 2003. lecture notes on an introductory lecture about CFD, School of Meteorology, University of Oklahoma.

[221] L. Yates and G. Chapman. Streamlines, vorticity lines, and vortices. Technical Report AIAA-91-0731, American Institute of Aeronautics and Astronautics, 1991.

[222] L. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[223] M. Zöckler, D. Stalling, and H. Hege. Interactive visualization of 3D-vector fields using illuminated streamlines. In *Proceedings IEEE Visualization '96*, pages 107–113, 1996.