Habilitationsschrift

Interactive Visual Analysis of Multi-Parameter Scientific Data

ausgeführt in den Jahren 2005–2014
zum Zwecke der Erlangung der *venia docendi* (Lehrbefugnis)
im Habilitationsfach „Informatik"

und eingereicht im Mai 2015
an der Technischen Universität Wien, Österreich,
Fakultät für Informatik,

von Dipl.-Ing. Dr. techn. Krešimir Matković,
Rötzergasse 20/5, A-1170 Wien, Österreich,
geboren am 19. August 1968 in Zagreb, Kroatien.
`Matkovic@VRVis.at`

Wien, im April 2015

Abstract

Increasing complexity and a large number of control parameters make the design and understanding of modern engineering systems impossible without simulation. Advances in simulation technology and the ability to run multiple simulations with different sets of parameters pose new challenges for analysis techniques. The resulting data is often heterogeneous. A single data point does not contain scalars or vectors only, as usual. Instead, a single data point contains scalars, time series, and other types of mappings. Such a data model is common in many domains. Interactive visual analysis utilizes a tight feedback loop of computation/visualization and user interaction to facilitate knowledge discovery in complex datasets.

Our research extends the visual analysis technology to challenging heterogeneous data, in particular to a combination of multivariate data and more complex data types, such as functions, for example. Furthermore, we focus on developing a structured model for interactive visual analysis which supports a synergetic combination of user interaction and computational analysis.

The concept of height surfaces and function graphs is a proven and well developed mechanism for the analysis of a single mapping. The state of the art when a set of such mappings is analyzed suggested a use of different descriptors or aggregates in the analysis. Our research makes it possible to analyze a whole set of mappings (function graphs, or height surfaces, for example) while keeping the original data. We advance the interactive visual analysis to cope with complex scientific data.

Most of the analysis techniques consider the data as a static source. Such an approach often hinders the analysis. We introduce a concept of interactive visual steering for simulation ensembles. We link the data generation and data exploration and analysis tasks in a single workflow. This makes it possible to tune and optimize complex systems having high dimensional parameter space and complex outputs.

Contents

Preface v

1 Introduction **1**
- 1.1 Simulation Data . . . 2
- 1.2 Interactive Visual Analysis of Complex Data . . . 3
- 1.3 Tasks Abstraction for IVA of Simulation Ensembles . . . 5
- 1.4 A Real-World Simulation Ensemble as an Illustrative Example . . . 6
- 1.5 Thesis Overview . . . 11
- 1.6 List of Selected Papers . . . 12
- 1.7 Overview of the Selected Papers and Contributions of the Author . . . 13
- 1.8 Summary . . . 20

2 Interactive Visual Analysis of Multi-Parameter Families of Function Graphs **23**
KREŠIMIR MATKOVIĆ, JOSIP JURIĆ, ZOLTÁN KONYHA, JÜRGEN KRASSER, AND HELWIG HAUSER
- 2.1 Introduction . . . 25
- 2.2 Motivation . . . 27
- 2.3 Technology Content . . . 28
- 2.4 Application . . . 35
- 2.5 Summary and conclusion . . . 37
- 2.6 Acknowledgements . . . 39

3 Interactive Visual Analysis of Families of Function Graphs **41**
ZOLTÁN KONYHA, KREŠIMIR MATKOVIĆ, DENIS GRAČANIN, MARIO JELOVIĆ AND HELWIG HAUSER
- 3.1 Introduction . . . 43
- 3.2 Related Work . . . 45
- 3.3 Data Model . . . 46
- 3.4 Tools for Analysis of Families of Function Graphs . . . 48
- 3.5 Analysis Procedures . . . 52
- 3.6 Engineering Application: Fuel Injection System Simulation . . . 56
- 3.7 Conclusion . . . 66

4 Interactive Visual Steering – Rapid Visual Prototyping of a Common Rail Injection System 69
Krešimir Matković, Denis Gračanin, Mario Jelović, and Helwig Hauser

4.1 Introduction and Related Work 71
4.2 Computational/Interactive/Simulation Steering 74
4.3 Rapid Visual Prototyping and the Design of a Common Rail Injection System 77
4.4 Conclusion 88
4.5 Acknowledgements 89

5 Visual Analytics for Complex Engineering Systems: Hybrid Visual Steering of Simulation Ensembles 91
Krešimir Matković, Denis Gračanin, Rainer Splechtna, Mario Jelović, Benedikt Stehno, Helwig Hauser, and Werner Purgathofer

5.1 Introduction 93
5.2 Related Work 97
5.3 Simulation and Visualization 99
5.4 Integrated Interactive Steering Workflow 101
5.5 Tasks and Steering Design 102
5.6 Case Study 106
5.7 Discussion and Conclusions 113

6 Interactive Visual Analysis of Multiple Simulation Runs Using the Simulation Model View: Understanding and Tuning of an Electronic Unit Injector 117
Krešimir Matković, Denis Gračanin, Mario Jelović, Andreas Ammer, Alan Lež, and Helwig Hauser

6.1 Introduction 119
6.2 Related Work 121
6.3 The Electronic Unit Injector 123
6.4 The Interactive Model View 125
6.5 Interactive Visual Analysis of an Electronic Unit Injector 129
6.6 Conclusion 141

7 Interactive Visual Analysis of Complex Scientific Data as Families of Data Surfaces 143
Krešimir Matković, Denis Gračanin, Borislav Klarin, and Helwig Hauser

7.1 Introduction 145
7.2 Related Work 147
7.3 Illustrative Example and Proposed Methodology 148
7.4 Data Model 156
7.5 Case Study — Analysis of a Slider Bearing 157
7.6 Conclusion 163
7.7 Acknowledgements 164

8 Path Line Attributes - an Information Visualization Approach to Analyzing the Dynamic Behavior of 3D Time-Dependent Flow Fields 167
KUANGYU SHI, HOLGER THEISEL, HELWIG HAUSER, TINO WEINKAUF, KREŠIMIR MATKOVIĆ, HANS-CHRISTIAN HEGE, AND HANS-PETER SEIDEL

8.1 Introduction 169
8.2 Related Work 171
8.3 Path Line Attributes 171
8.4 System overview 173
8.5 Applications 175
8.6 Conclusions 178
8.7 Acknowledgments 179

9 A Statistics-based Dimension Reduction of the Space of Path Line Attributes for Interactive Visual Flow Analysis 181
ARMIN POBITZER, ALAN LEŽ, KREŠIMIR MATKOVIĆ, AND HELWIG HAUSER

9.1 Introduction 183
9.2 Related Work 185
9.3 Statistical Analysis 186
9.4 Demonstration 192
9.5 Conclusions 199
9.6 Acknowledgments 200

Bibliography 201

Preface

This thesis presents selected parts of my research carried out at the VRVis Research Center in Vienna, Austria, in the years from 2005 to 2014.

First, I want to thank Helwig Hauser, a great scientist and a friend. Our numerous discussions on research challenges and research itself, as well as on many other general questions, encouraged and supported my research career in every imaginable respect. Helwig's devotion to science, hard work, and, at the same time, his view on life, have made him a role model I am constantly trying to follow.

I also thank Denis Gračanin, another great researcher and a friend. In spite of a great distance between Virginia and Austria, since we have first met, more than ten years ago, our collaboration only intensifies. Denis always supported my research ideas, our long discussions and time spent together resulted in numerous novel concepts and ideas. I enjoy the time we spend together discussing non-research oriented topics, as well.

My research would not have been possible without support of Georg Stonawski, the CEO of the VRVis, Werner Purgathofer, our scientific director, Katja Bühler, the leader of the visualization area at the VRVis, and M. Eduard Gröller, the key researcher of our research area. All four of them made the VRVis a great place to work, and made it possible for me to pursue research and follow my ideas.

Visualization research is often a collaborative research with experts from other domains. The research described in this thesis has been done mostly with experts from automotive industry. I would like to particularly thank Mario Jelović from AVL in Zagreb, Croatia, for numerous discussions on simulation and visualization. During the last ten years he became a friend as well, which I appreciate very much.

My gratitude also goes to all my former and present colleagues and collaborators for all the time we spent collaborating on interesting problems.

Finally, I want to express my deepest gratitude to my family – beloved wife Neda, our daughter Lovorka, and our son Mihovil. They have enriched my life in every possible aspect. Their patience, support, and love were essential in accomplishing this thesis. Thank you!

Vienna, Austria, April 2015 *Krešimir Matković*

1

Introduction

This thesis presents a subset of my research in the field of interactive visual analysis of multi-parameter scientific data. The first chapter serves as an introduction to the field, it provides a brief description of the simulation data and related analysis tasks, and provides the context for the papers presented in the following chapters. The papers making up the body of the thesis are listed and briefly described, with a special emphasis on the contribution of the author made to each individual work.

Simulation is an omnipresent support methodology for engineers and scientists across many domains. Many physical phenomena in nature, science and technology can be explored through simulation results. These phenomena include the weather and climate, blood flow, or the combustion processes in car engines, for example. The rapid improvements of modern computation technologies make it possible to simulate systems of increasing complexity having a large number of control parameters.

The development of new simulation technologies has always been accompanied by the necessary development of new and matching analysis methods. The visual analysis of simulation results has established itself as a successful method, because it makes use of the unique capabilities of the human perception and cognitive system – in addition to the enormous power of computers – to provide a deep understanding of large and complex data.

Enabled by the dramatic improvement of computing power and simulation technology, we now observe a transition towards a new kind of simulation: a physical phenomenon is not only considered by a single simulation, but by a multitude of simulations with a number of changing parameters. Such a methodology is referred to as ensemble simulation or multiple-runs simulation. Ensemble simulation has the great potential to provide a much deeper understanding of the investigated phenomenon, to study the variability and sensitivity of simulation models, and to enable statements about uncertainties that are always associated with such studies.

In order to fully exploit the opportunities that emerge from advanced ensemble simulation, effective and efficient analysis methods are needed. The exploration and analysis methods have to secure the successful exploitation of the great wealth of information that is contained in the simulation results.

Interactive visualization is a proven technology for the analysis of ensemble simulation data and for studying systems by varying a set of controllable inputs and observing simulated responses (outputs). The research described in this thesis deals with the interactive visual analysis of simulation data. Most of the data come from ensemble simulations. Two papers describe the analysis of data representing path lines' attributes. Although these are not ensemble simulation data, they are consistent with the same data model.

1.1 Simulation Data

The simulation can be modeled as a function that maps control parameters to output values. Simulation solvers generate different kinds of data, ranging from simple scalars, to curves, surfaces, volume data, or some more complex data types. There are numerous visualization methods to cope with complex data from a single simulation run. In the case of multiple simulation runs, i.e., simulation ensembles, the users typically extract interesting, scalar features from the data in order to ease the analysis. Traditional data analysis approaches, such as statistics or OLAP techniques [Tho97], usually consider the outputs to be scalar values only, and use data tables to capture the relations among control parameter values for the same simulation run [CMS99, Sam06].

For a simulation ensemble, the simulation process can be modeled as a function S that maps the control parameters $\mathbf{x} = (x_1, \ldots, x_m)$ (a control data point in $\mathbb{R}^m$) to the output values $\mathbf{y} = (y_1, \ldots, y_n)$ (an output data point in $\mathbb{R}^n$) where m is the number of control parameters and n is the number of outputs.

$$\mathbf{y} = S(\mathbf{x}) \tag{1.1}$$

Due to various physical constraints, the set of feasible control data points C is a subset of $\mathbb{R}^m$ and the set of feasible output data points O is a subset of $\mathbb{R}^n$. Therefore, a simulation ensemble E is a set of pairs of data points $(\mathbf{x}, \mathbf{y})$, $\mathbf{x} \in C$ and $\mathbf{y} \in O$.

As stated above, a single simulation run can produce outputs that are not scalars only. If there are, e.g., time dependent outputs, the above equation does not hold any more. One solution is to replace time dependent output with a series of features. If, for example, the simulation computes a force on a crankshaft as a function of time, the maximum force might be sufficient for certain analysis tasks. If we strive for a deeper insight, scalar features are not sufficient any more. We do need curves themselves in the analysis. Accordingly, an adequate simulation ensemble data model is needed to deal with more complex data. Our work deals with such

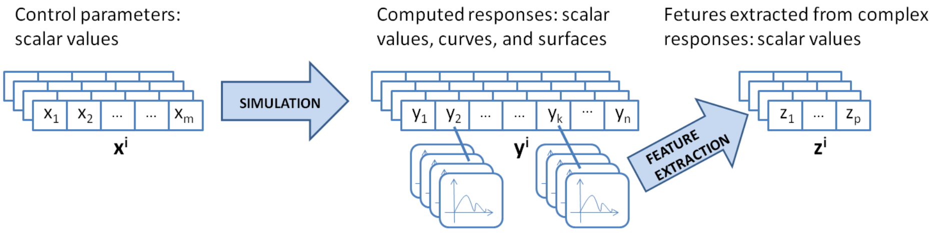


Figure 1.1: Simulation ensemble data model: control data points – control parameters, output data points – computed responses, and extracted features. For each output data point $\mathbf{y}^i$, there can be an output value y^i_j that is a complex data item – a curve or a surface. In the figure, y^i_2 and y^i_k are time dependent outputs, i.e., curves. Complex data items are replaced by one or more scalar values (features z) in the feature space. The original output space and the feature space are used in the analysis.

data. We have extended the data model to include complex data as atomic attributes. Our simulation ensemble data model uses a two-level data hierarchy for the output data points (Figure 1.1). The output values can be either scalars, 1D function graphs (usually time series data), or 2D function graphs or data surfaces. In our previous work [FMH08], which is not a part of this thesis, we also represent set-type data as an atomic dimension.

For each output data point $\mathbf{y}^i$, there are some dimensions y^i_k that are data series, we have a separate set of "sub-points" with its own length and number of dimensions. In this work we focus on curves and surfaces only. In addition to the complex data item (a curve or a surface), a set of scalar values – features z (such as, the maximum or the minimum value, or the area under the curve, for example) is also computed, to create a feature point $\mathbf{z}^i$ in the feature space F, a subset of $\mathbb{R}^p$, where p equals the total number of features computed for the data point $\mathbf{y}^i$. We use elements of all three spaces: the $\mathbf{x}^i$ (scalar control parameters), the $\mathbf{y}^i$ (scalar and complex responses computed), and the $\mathbf{z}^i$ (scalar features of complex responses), simultaneously in the analysis.

1.2 Interactive Visual Analysis of Complex Data

The complex data as described in the previous section pose new challenges for analysis techniques. A single data point in the resulting heterogeneous data does not contain scalars only, as usual. Instead, a single data point contains scalars, time series, surfaces, and other types of mappings. Although, in this thesis, we deal mostly with engineering data, such a data model is common in many domains. In our research (not only described in this thesis) we have identified the same data model for: intensive care unit data [MGA$^+$12, GMA$^+$11], animal trajectories [MWSB12], bio signals (ECG) [TMS$^+$08], image collections [MGF$^+$09], or meteorological data [MGKH07], for example.

Conventional analysis techniques are insufficient to cope with such complex data. Interactive visual analysis (IVA) facilitates knowledge discovery in complex datasets by utilizing a tight feedback loop of computation, visualization and user interaction [KAF$^+$08, KKEM10, TC06]. IVA provides an interactive and iterative exploration and analysis framework, where the user guides the analysis [Shn02], supported by a variety of computational analysis tools. This helps the user to explore and analyze the data, and to understand complex and often hidden relationships between certain data aspects. The Visual Information Seeking Mantra – overview first, zoom and filter, then details-on-demand – as identified by Shneiderman [Shn96], summarizes the most typical pattern in IVA. Interactive visual analysis is much more than the presentation of data; it supports the user in the analysis of complex and heterogeneous datasets [KH13, KMDH11, YRW07]. IVA has been successfully employed in many domains [KAF$^+$08, TMS$^+$08, DGH03].

Coordinated multiple views [Rob07] are often used as a proven concept in IVA. The main idea is to depict various dimensions using multiple views and to allow the user to interactively select (brush) subsets of the data in a view. Then all the corresponding data items in all linked views will be consistently highlighted [DGH03, MW95]. One of the first examples of linking and brushing of different visualization approaches in different views is a system called WEAVE [GRW$^+$00]. It was used to interactively analyze and visualize simulated data of a human heart.

The concept of height surfaces and function graphs is a proven and well developed mechanism for the analysis of a single mapping. The state of the art when a set of such mappings is analyzed suggested the use of different descriptors or aggregates in the analysis. Our research makes it possible to analyze a whole set of mappings (function graphs, or height surfaces, for example) while keeping the original data.

Furthermore, we also consider on-the-fly data generation through ensemble steering. If the dimensionality of the parameter space grows, it is prohibitively expensive to sample it densely. In order to make the process more efficient we couple the data generation, data exploration, and analysis tasks in a single workflow. Additionally, we incorporate automatic optimization methods based on regression models, to further increase the scalability of ensemble steering.

Finally, we show how IVA can be used to better understand time dependent flow data. There are numerous methods how to visualize and analyze such data, and we have extended the state of the art by a novel approach to studying different path lines' attributes, using the same data model, again.

The interactive visual analysis is by no means intended to replace conventional, automatic analysis methods, based on statistics, machine learning, or data mining, for example. The interactive visual analysis represents an efficient additional methodology which helps us to gain insights which are otherwise impossible. Only a clever combination of automatic and interactive techniques can solve really complex problems.

1.3 Tasks Abstraction for IVA of Simulation Ensembles

During several collaboration projects with domain experts we have identified four main tasks common to practically all examples of ensemble data exploration and analysis.

Two main tasks, which an analyst has during ensemble exploration and analysis, can be summarized as:

- **Model Reconstruction:** How do the dependent variables (outputs) change as the values of the independent variables (control parameters) are changed within a single ensemble? If the simulation is fast, this can be answered without ensemble data by simply running a simulation for the desired control parameters. In an interactive visual analysis environment we vary independent variables while studying the corresponding dependent variables deploying a coordinated multiple views techniques.
- **Output Analysis:** The desired/expected shape of the output (curve) is (approximately) known. The goal is to find combinations of control parameters that produce this shape within a single ensemble. The combinations producing undesirable/invalid output can be excluded based on how the deviations from the desired output depend on the control parameters. The correlation between various outputs is considered. Without ensemble simulation this is possible only if the inverse of the simulation mapping is known. As most of the modern solvers use numerical methods, finding an analytical inverse function is practically impossible in most of the cases.

In case of ensemble steering, where the user can request additional runs of the same model, or even request a model refinement, two additional tasks have to be supported:

- **Ensemble Growing:** Add simulation runs with different control parameter settings to the currently analyzed ensemble (based on previous findings). The control parameters can be sampled differently (resolution/distribution/range) to set up new simulation runs.
- **Model Refinement:** Create an ensemble for a (partially) refined simulation model. A new ensemble with a different parameter space is created. After applying this task all other tasks can be arbitrarily applied.

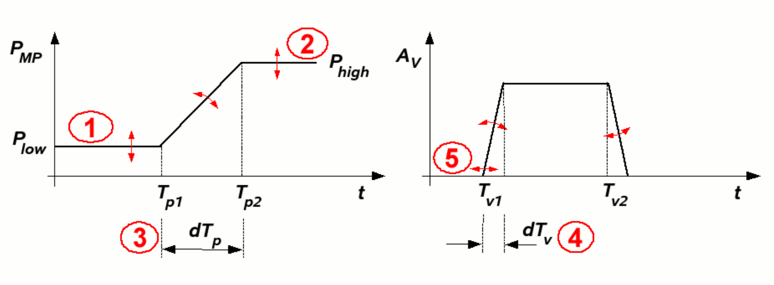


Figure 1.2: Three scalar control parameters describing the pressure at the injectors inlet: P_{low}, P_{high}, and dT_p. These parameters are marked with red numbers one to three. Two scalar features describing the actuator time signal, T_{v1} and dT_v are marked with numbers four and five.

These are only high-level tasks, and we describe the tasks on a much finer level later in the thesis. In order to support these analysis tasks we have to provide efficient means of forming a mental model of data and parameter space to understand the observed phenomena. Interactive visualization represents a premium choice here as it amplifies our cognition through exploration of our high-throughput visual channel and, by doing so, supports the mental model building.

1.4 A Real-World Simulation Ensemble as an Illustrative Example

In this section we exemplify the above described data model and the analysis tasks on an example of ensemble simulation of injection systems [KMG$^+$06], [MGJH08], [MGS$^+$14]. Besides helping us to answer the two high level questions in the analysis, the interactive visual analysis additionally supports the knowledge gaining process of the engineer

The common rail injection represents the standard injection system for modern Diesel car engines [BDHK05], [BH97]. It operates at a very high pressure level, with an electronic control unit which determines the fuel delivery, injection timing, injection pressure, and rate of injection, for multiple injection strategies. The modern common rail makes it possible for Diesel cars to achieve a level of performance and driving comfort similar to those of gasoline powered models with less fuel consumption and low exhaust emissions. The common rail system uses a high-pressure rail, the same for to all cylinders. The high pressure in the rail is used to precisely inject fuel into the cylinders. Due to high pressures and quick changes in the system, a modern common rail injection system operates in a condition which cannot be described sufficiently precise using classical fluid mechanics. The Diesel fuel cannot be considered incompressible when exposed to high pressure changes in

Control Parameter	Symbol	Number of variations
Low pressure on the injector inlet	P_{low}	5
High pressure on the injector inlet	P_{high}	5
Time interval of modulated pressure increase on the injector's inlet	dT_p	5
Time interval of the injector valve opening and closing	dT_v	5
Injector valve opening time	T_{v1}	7

Table 1.1: Five control parameters varied

short time periods. At high pressure, the fluid density, the module of elasticity, and the speed of sound are significantly altered. Furthermore, in a common rail system each cylinder and injector is influenced by the others through the rail. This requires a careful rethinking of traditional system design.

There are three main factors that influence the injection shape: the nozzle geometry, the injection pressure, and timings of valve opening and closing procedures. The nozzle geometry cannot be changed at run-time, and it is usually investigated by injector manufacturers. We focus on the remaining two factors in the analysis.

The independent variables in our ensemble are related to injection pressure and injector valve timings only. The injection pressure is controlled by the injection pressure modulation device. The pressure on the injector's inlet is described by three independent variables (Figure 1.2). The injection timing is controlled by the injector valve actuator. We model the timing using opening and closing times and velocities (Figure 1.2). Such a model results in five independent variables. Table 1.1 shows the varied control parameters and number of variations for each of them. The total number of variations, i.e., the number of simulation runs we have computed, equals $5^4 \times 7$, or 4375 runs. According to the notion introduced in Section 1.1, the $m = 5$ and the cardinality of C equals to 4375. For each simulation run the following time dependent and scalar response are computed:

- injection rate – $Q_{inj}(t)$,
- injection pressure – $P_{inj}(t)$,
- needle lift – $X_{needle}(t)$,
- amount of fuel flowing back to the fuel tank Q_{vo}, and
- spray penetration depth – L_p.

The $\mathbf{y}$ dimensionality equals five, so $n = 5$, and three out of five output values are time dependent. Additionally, we also compute the following scalar features, as they will be needed in the analysis:

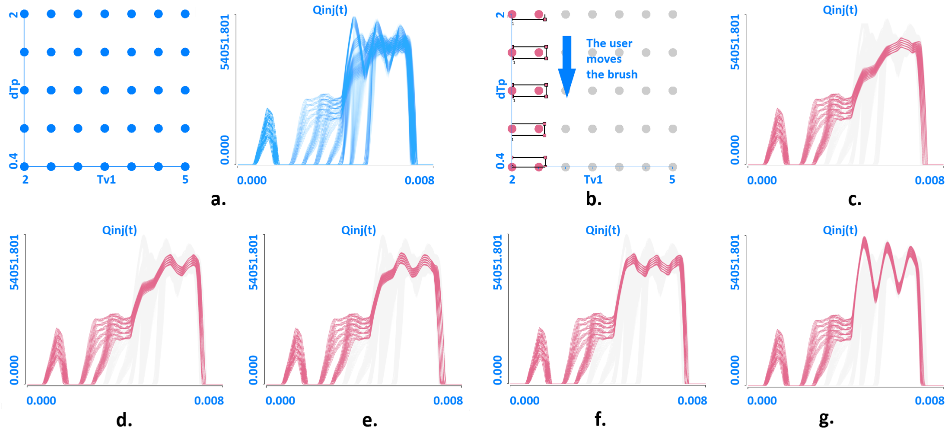


Figure 1.3: Exploring changes in responses as the control parameters change. **a.** A Minimum configuration for the IVA. The scatterplot shows two control parameters, and the curve view shows one of the time dependent response variables. **b.** The user brushes the upper left corner in the scatterplot and moves the brush downwards. **c.** to **g.** As the brush moves the corresponding curves are highlighted. The user knows that dT_p is decreasing, while T_{v1} remains low, and can focus on the responses only.

- amount of fuel injected during pilot injection – Q_p, computed as an integral of $Q_{inj}(t)$, $Q_p = \int_{t_1}^{t_2} Q_{inj}(t)$, where t_1 and t_2 represent starting and ending time of the pilot injection,
- amount of fuel injected during main injection – Q_m, computed as an integral of $Q_{inj}(t)$, $Q_m = \int_{t_3}^{t_4} Q_{inj}(t)$, where t_3 and t_4 represent starting and ending time of the main injection,
- needle opening velocity V_{open}, computed using the slope of $X_{needle}(t)$,
- needle closing velocity – V_{close}, computed using the slope of $X_{needle}(t)$, and
- average injection power – P_{ia}, computed from $Q_{inj}(t)$ and $X_{needle}(t)$.

These five scalar values represent scalar features z, in the feature space F with a p of five, according to the notation introduced in Section 1.1.

The first step is the ensemble computation. The simulation is run for each of the 4375 combinations of control parameters. Responses, as well as scalar features, as described above, are computed. The analysis can start. The user explores the simulation ensemble using a coordinated multiple views system. The coordinated multiple views include at least two views depicting different control parameters and output values. The most often used views include a scatterplot, a histogram, parallel coordinates, or a curve view. The views support linking and brushing – the user interactively selects some items (simulation runs in our case) in one view, and the

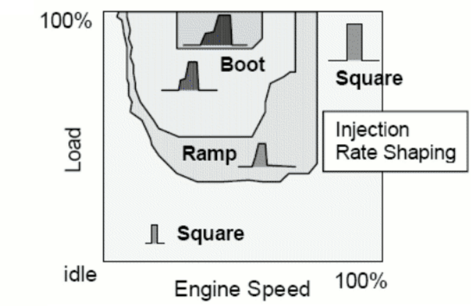


Figure 1.4: Characteristic shapes of the main injection curves for different combinations of engine speed and load. The injection parameters change at run-time in order to ensure desired injection characteristics.

corresponding items in all other views are highlighted. The user can now move the brush and observe what happens with the corresponding items. The selection can be done either in the control parameters space, or on the response values.

We first explore how changes in control parameters influence output values. We depict two of the control parameters using a scatterplot, and response variables under consideration using the curve view, as shown in Figure 1.3 a. We brush the upper left corner in the scatterplot (high values of dT_p and low values of T_{v1}), and move the brush towards the lower left corner. We observe changes in the other views as we move the brush. Figure 1.3 b. shows the brush positions (brushes are superimposed in this figure to save space), and Figures 1.3 c. to g. show what happens with one time dependent response variable – $Q_{inj}(t)$. We examine several responses in the same way in order to comprehend the system behavior.

The next step is the model reconstruction. We would like to identify control parameters resulting in a desired model behavior. The optimum curve shape depends on the engine operating point. Depending on engine speed and load, the desired shape varies. Figure 1.4 shows desired curve shapes for different operating points. We depict injection curves in a curve view and use several line brushes to "specify" the desired curve shape. The line brush selects all curves that cross a user-drawn line. By combining several lines using Boolean operators we can specify an arbitrary shape very fast. Figure 1.5 shows an example. We used five line brushes in order to specify a desired boot shape. The control parameters are shown in two scatterplots and one histogram. We can see that, in order to achieve the boot shape, we have to set T_{v1} to the lowest value, dT_p and P_{low} to the upper half of the available range, P_{high} to the lower half, and, as it can be seen in the histogram, dT_v can have any value.

After the analysis, we have gained valuable insight into the simulation ensemble and into the fuel injection process itself. The most important findings can be summarized as:

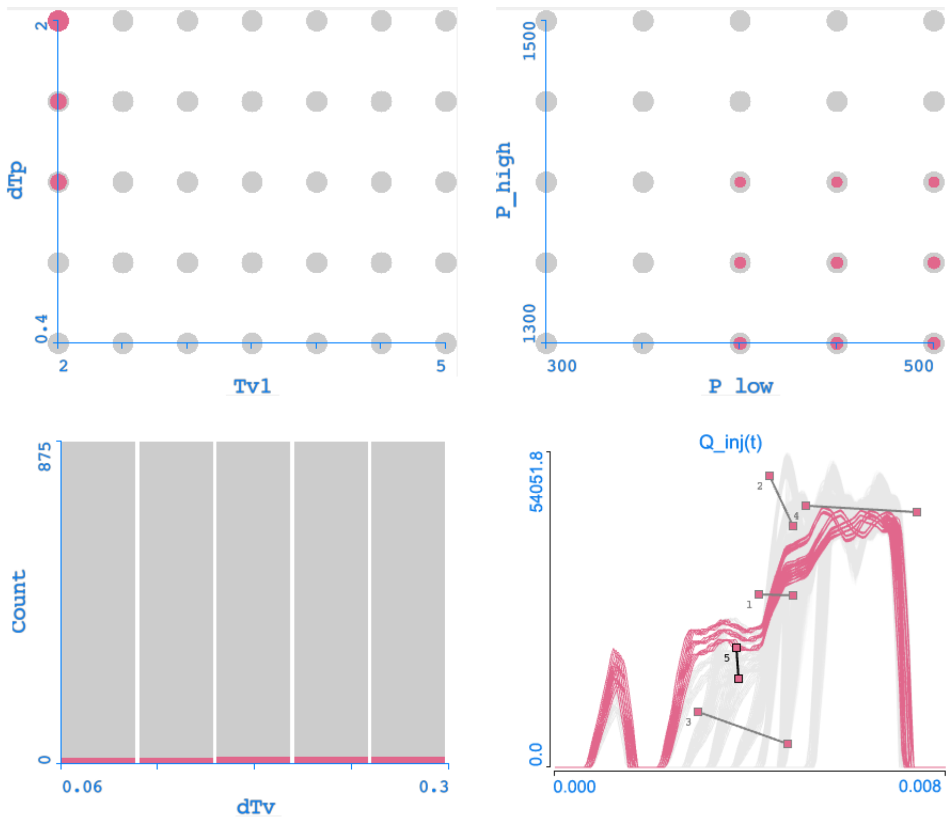


Figure 1.5: The model reconstruction - the user identifies control parameters for a desired injection shape. The shape is selected by means of several line brushes in the curve view. The scatterplots and the histogram show the corresponding control parameters.

- The amount of injected fuel in the main injection stage can be controlled by adjusting P_{high}.
- The amount of pilot injection is controlled mostly by P_{low}, but dT_v also influences it.
- A right choice of dT_p and T_{v1} is the key to achieving the desired injection shapes for various engine operating conditions.
- If pressure increases too fast on the injector's inlet, the resulting wave can be reflected into the fuel line, which impairs our control over the injection's shape.

Such findings would be practically impossible without interactive visual analysis and ensemble computation. An additional advantage of the here described approach is the better understanding of the complex parameters interplay. Engineers get a much better insight into system operation and influence of parameter changes. This illustrative example shows the basic idea only. More comprehensive examples are provided in the next chapters of the thesis.

1.5 Thesis Overview

The papers that constitute this thesis describe our research on the visual analysis of complex data. The first paper (Chapter 2) describes an early work on a specific case where the parameter space is two-dimensional only. We have analyzed a real-world data set – an engine characteristic diagram with corresponding cylinder pressure graphs. Chapter 3 describes then the next step in our research. The parameter space is now multidimensional, and there are several curves dimensions. We introduced the term *A Family of Function Graphs* here, and described the identified analysis procedures. As the parameter-space dimensionality grows further, computing the whole ensemble in advance becomes prohibitively expensive. A possible solution is interactive visual ensemble steering (Chapters 4 and 5). The analysis starts with an initial ensemble, which is iteratively extended during the exploration and analysis process. The user interactively selects areas of the parameter space, which have to be re-sampled. In order to support the user, we also include a regression-model based optimization, which suggests areas of potential interest. Chapter 6 describes how we included the simulation-model view in order to amplify the mental model of the user. Simulation experts are familiar with the simulation model view, and we used this well-known paradigm as an interface to access abstract views employed in the analysis. As described above, we do not limit the data-model extension to curves only. Chapter 7 describes our research on families of surfaces and the corresponding analysis methodology. In contrast to curves, where it is very efficient to depict many curves (employing, e.g., density mapping) in order to get insight into a family of curves characteristics, depicting many surfaces at once does not solve the problem. We designed an analysis approach based on projections, which allows us to increase the number of visualized surfaces by abstraction. The approach is illustrated using a climatological research example, as well as an example from automotive simulation. Finally, the last two chapters (Chapter 8 and 9) describe how the new data model can be applied to non-ensemble data. In these chapters we analyze time dependent 3D flow based on path-lines attributes. For each path-line we compute a set of attributes, some of them are scalar, and some are time dependent. We analyze the path-lines attributes data then. We also explored which attributes are the most representative ones using several representative data sets of various complexity.

1.6 List of Selected Papers

This thesis contains the following papers [MJK$^+$05, KMG$^+$06, MGJH08, MGS$^+$14, MGJ$^+$10, MGKH09, STH$^+$09, PLMH12]:

1. K. Matković, J. Jurić, Z. Konyha, J, Krassser, H. Hauser: Interactive Visual Analysis of Multi-Parameter Families of Function Graphs, *in Proceedings of 3rd International Conference on Coordinated & Multiple Views in Exploratory Visualization 2005*, London UK, 2005
2. Z. Konyha, K. Matković, D. Gračanin, M. Jelović, H. Hauser: Interactive Visual Analysis of Families of Function Graphs, *in IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 6, 2006
3. K. Matković, D. Gračanin, M. Jelović, H. Hauser: Interactive Visual Steering - Rapid Visual Prototyping of a Common Rail Injection System, *in IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, 2008
4. K. Matković, D. Gračanin, R. Splechtna, M. Jelović, B. Stehno, H. Hauser, W.Purgathofer: Visual Analytics for Complex Engineering Systems: Hybrid Visual Steering of Simulation Ensembles, *in IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, 2014.
5. K. Matković, D. Gračanin, M. Jelović, A. Ammer, A. Lež, H. Hauser: Interactive Visual Analysis of Multiple Simulation Runs Using the Simulation Model View: Understanding and Tuning of an Electronic Unit Injector, *in IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, 2010
6. K. Matković, D. Gračanin, B. Klarin, H. Hauser: Interactive Visual Analysis of Complex Scientific Data as Families of Data Surfaces, *in IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, 2009
7. K. Shi, H. Theisel, H. Hauser, T. Weinkauf, K. Matković, H.-C. Hege, H.-P. Seidel: Path Line Attributes - an Information Visualization Approach to Analyzing the Dynamic Behavior of 3D Time-Dependent Flow Fields, *in Topology-Based Methods in Visualization II*, Springer, 2009
8. A. Pobitzer, A. Lež, K. Matković, H. Hauser: A Statistics-based Dimension Reduction of the Space of Path Line Attributes for Interactive Visual Flow Analysis, *in Proceedings of IEEE Pacific Visualization Symposium (PacificVis) 2012*, Songdo, Korea, 2012

The papers included in this thesis appear unmodified in their original, published form, except for the typesetting, which has been adapted to conform to the style of this thesis. No textual changes were performed. The bibliography sections were joined into a single bibliography at the end of this thesis.

1.7 Overview of the Selected Papers and Contributions of the Author

This section briefly describes each of the publications contained in this thesis. The papers represent a sample of the research work which the author carried out in the years from 2005 to 2014. In total, the author has co-authored 79 peer-reviewed papers at the time of writing this thesis.

The publications selected for inclusion in this thesis focus on the author's work on exploration, analysis, and developing insight into complex, multi-parameter scientific data. The data represent results from multiple simulation runs or from complex, 3D computational fluid dynamics simulations. In addition to publishing articles at the world's leading forums for visualization research, the author has also co-authored a book on interactive 3D graphics in Croatian. The book has been approved as a university text book by the University of Zagreb.

Visualization research is a highly collaborative discipline, where researchers from different domains collaborate in order to solve complex problems. As a consequence, none of the papers in this thesis is a single-author paper. Single-author papers are the exception, not the rule, in the field of visualization. The author substantially contributed to each of the included publications. The papers where the thesis author is not the first author, were written in the scope of the PhD studies of the first authors. The following sections explain the thesis author's contributions to the individual papers in addition to summarizing them.

1.7.1 Chapter 2: Interactive Visual Analysis of Multi-Parameter Families of Function Graphs

In this chapter, we describe a method which can be used to analyze and explore data sets containing families of function graphs, where each of them corresponds to a point in a 2D domain. One easily comprehensible example of such a data set is a diagram of temperature over time measured on a number of places within a certain geographical area. The geographical map represents the 2D domain, and the temperature vs. time data at specific points on the map represent additional two dimensions of the data set. In this case the data is available for some points only (positions of measuring devices) and can be interpolated for other points. When interpolating, special attention has to be paid to what is physically plausible; simple linear interpolation can yield wrong results in many cases.

Data exploration is usually the first step in the analysis. Analysts try to find locations in the attribute space that generate function graphs which have some interesting or unusual property, or conversely, ones that exhibit some regularity or follow a certain pattern. This calls for a visualization method that allows the user a quick and easy, yet accurate exploration of the data. Two simultaneous views of the data should be provided: (1) a good overall view of all function graphs to support navigation in the data set as well as (2) a detailed view which offers more information about specific graphs. Picking graphs from the attribute space must be made as effortless as possible and should require a minimum user interaction.

Our solution includes two linked views: a map view (or attribute space view), where all function graphs are represented as a point or an icon on the map, and a linked function graph view. The map view provides additional visualization possibilities and allows for user interaction. Very positive feedback from the domain experts, who helped us in this research, encouraged us to proceed with the exciting research, described in the following chapters.

The author has been involved in the idea development from the very beginning, he proposed the two-way linking and brushing approach and implemented the idea. He wrote most of the paper, using feedback from all co-authors. The paper was presented at the 3rd International Conference on Coordinated & Multiple Views in Exploratory Visualization, which was held in 2005, in London, UK, and it was published in *Proceedings of 3rd International Conference on Coordinated & Multiple Views in Exploratory Visualization 2005.*

1.7.2 Chapter 3: Interactive Visual Analysis of Families of Function Graphs

In this chapter, we describe an approach for the interactive visual analysis of an especially challenging set of problems that includes several (usually large) families of function graphs $f_i(\mathbf{x},t)$. The research presented in this chapter is a result of a close collaboration with domain experts. Only such a close collaboration could help us to identify analysis tasks and, consequently, to develop a corresponding support methodology. In contrast to the previous chapter, the parameter space is multidimensional in this case.

We present a new approach for the interactive visual exploration and analysis of measurement and simulation data. This approach is general enough for a number of application scenarios that share the same characteristics, including multi-parameter tuning problems. In order to visually relate the multivariate dependent variables to their multidimensional reference parameters we suggest a combination of different kinds of views with specific brushing interactions. All are adapted to work well for the families of function graphs in order to facilitate the interactive visual exploration and analysis of such data sets.

The usability of the ideas has been evaluated in two very different settings: the analysis of road traffic data and the optimization of a fuel injection system. We describe analysis procedures as well as practical aspects of the interactive visual analysis, which are specific to this special kind of data. We adopted the well-proven setup of multiple, linked views with advanced interactive brushing to assess the data.

Standard views such as histograms, scatterplots, and parallel coordinates are used to jointly visualize parameters as well as dependent data.

The author has been involved in the idea development from the very beginning, he proposed the density mapping for the curve view, and implemented the idea. He also contributed to the writing of the paper. The paper was published in *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 6, 2006.

1.7.3 Chapter 4: Interactive Visual Steering - Rapid Visual Prototyping of a Common Rail Injection System

Interactive visual steering has been a common goal of the visualization research community for twenty years, but, interestingly, it is rarely realized in practice. In this chapter we describe a successful realization of a tightly coupled visual steering loop, integrating simulation and interactive visual analysis in a prototyping environment for automotive-industry system design.

In order to realize an interactive ensemble steering framework, a fast simulation engine is needed. In contrast to the usual, very time consuming 3D CFD simulation, 1D CFD, which is alternatively used in injection system simulation, can be computed very fast. It is possible to run tens of thousands of simulations. However, with a high-dimensional parameter space (tens or hundreds of parameters) a brute force approach, where a simulation is executed for all possible parameter combinations, is often far from feasible. Interactive visual simulation steering represents a possible solution though.

The model complexity did not allow us to run all possible simulations at the beginning and to analyze the results then. Such an approach would also result in numerous unnecessary simulation runs and would waste time and computational resources. Furthermore, we did not have a complete model at the beginning. It was gradually built as we gained insight during the design process.

Our research makes it possible to define new simulations using the visualization tool. The visualization tool is used for the analysis and steering of the simulation. This makes it easy for the domain expert to generate new simulations and to refine or to filter the simulation dataset. We provide four basic operations: refining or coarsening some control parameters; narrowing down the control parameter interval; adding new control parameters; and removing some existing control parameters. If we represent data in tabular form, the basic operations correspond to adding and removing rows (refinement and filtering parameters) or adding or removing columns (adding or removing parameters). The domain expert estimates the coarse boundaries of the parameters, runs a sufficient number of simulations and sees what parameter values make sense and what values are not allowed based on the output values.

Our approach, the use of interactive visualization and coordinated multiple views as a steering mechanism for simulation, proved to be very efficient. In this chapter we show how the tight integration of visualization and simulation can significantly improve an engineer's workflow as compared to decoupled systems. The results, which we achieved for the common rail injection system, strongly suggest that our approach has a very good potential of being generalized to other, similar scenarios.

The author has been involved in the idea development from the very beginning, he carried out numerous interviews with domain experts from the automotive industry, and participated in all discussions, from the first concept to the final solution. He also implemented the idea and significantly contributed to the writing of the paper, taking all co-authors' feedback into account. The paper was presented at the IEEE Visualization 2008 conference in Columbus, Ohio, US, and it was published in *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, 2008

1.7.4 Chapter 5: Visual Analytics for Complex Engineering Systems: Hybrid Visual Steering of Simulation Ensembles

As the number of parameters increases pure interactive navigation in the parameter space becomes tedious. In order to support the user in identifying relevant areas in the multidimensional parameter space that need a refinement, we extended the steering framework by automatic optimization. The new approach is called hybrid steering, as it integrates interactive and automatic optimization. Hybrid steering allows a domain expert to interactively select data points (in a visualization) in an iterative manner, approximate the values in a continuous region of the simulation space (by regression) and automatically find the "best" points in this continuous region based on the specified constraints and objectives (by optimization). We argue that with the full spectrum of optimization options, the steering process can be improved substantially.

The target users of the proposed solution are designers of complex systems that are based on simulation ensembles. The research described in this chapter is a result of a long-term collaboration between visualization and simulation experts. In our opinion, neither group alone could come to such a solution. The newly proposed approach is inspired by an actual application in the automotive industry.

The main contributions of this research can be summarized as: (1) A case study demonstrating Hybrid Visual Steering, a novel simulation ensembles steering and exploration approach. This approach combines interactive exploration and analysis with automatic optimization based on regression models. (2) The task abstractions and the supporting visualization system, including two improved views, the Parameters Exploration View and the Regression Exploration View. (3) The tight integration of all relevant components in an interactive workflow. (4) An evaluation of the proposed approach based on a case study from the automotive industry including user feedback.

The author initiated the main idea on hybrid steering. He contributed to the finally developed concept, was responsible for the communication with the domain experts and the case study realization. He also implemented the idea and significantly contributed to the writing of the paper, taking all co-authors' feedback into account. The paper was presented at the IEEE VIS 2014 conference in Paris, France, and it was published in *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, 2014.

1.7.5 Chapter 6: Interactive Visual Analysis of Multiple Simulation Runs Using the Simulation Model View: Understanding and Tuning of an Electronic Unit Injector

Modeling is the first step in the simulation. The common characteristics shared by most of the models, simple or complex, is that they are created from basic building blocks. There is a set of control parameters for each block, and based on them, the state parameters are computed. The state parameters in such a model show how a block behaves given the values of the control parameters. The connections and dependencies among the blocks and the overall structure of the model determine the model's behavior. A visual representation of the model, usually in the form of a 2D graph, captures these dependencies and allows an engineer to have a good understanding of the model.

A simulation produces results that capture the behavior of the model, depending on the control parameters. However, the connection between the model (and its visual representation) on the one side and the numerical data (as produced by the simulation) on the other side is often missing from the analysis. The engineer can use various visualization techniques and data views to get an insight into the simulation results. However, there is still a gap, both cognitive and visual, that needs to be closed (in the context of interactive visual analysis). The model view is decoupled from the visualization of the results. To mitigate this problem, it is possible to integrate some of the simulation results of a single simulation run within the display of the simulation model.

If dealing with multiple simulation runs, the same model is used with varying values of the control parameters. Closing the gap in such a scenario presents an even greater visualization challenge and we propose a solution in this chapter. As there can be thousands or tens of thousands of runs that generate a huge amount of complex data, a visualization and analysis solution than can cope with this challenge and bridge the gap between the model and the simulation results is needed.

We propose a new view, the simulation model view, that provides an additional context for the simulation results to close the gap between the model and data for multiple simulation runs. In the case of multiple simulation runs, the simulation model view blocks should show multiple values of the parameters. Moreover, the descriptive parameters are often time-dependent making the problem even more complex. The view provides a 2D graph where each node represents a building block of the simulation model. The values of both the control and state parameters are displayed directly within the node in the simulation model view. As the available screen space is very limited, we propose a three levels of detail approach to achieve a compromise between the amount of displayed information and the available space for each block. The simulation model view is integrated into a coordinated multiple views (CMV) system. The benefits of multiple linked views and composite brushing facilitate the use of the simulation model view, especially when dealing with multiple simulation runs.

While observing simulation experts, the author initiated the main concept. He contributed to the main idea, was responsible for the communication with the domain experts and for the case study realization. He supervised the model view implementation by Andreas Ammer, significantly contributed to the writing of the paper, and took all co-authors' feedback into account. The paper was presented at the IEEE Visualization 2010 conference in Salt Lake City, Utah, US, and it was published in *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, 2010.

1.7.6 Chapter 7: Interactive Visual Analysis of Complex Scientific Data as Families of Data Surfaces

In this chapter we introduce families of surfaces. The data model supports surfaces as atomic units. While it is almost trivial to visualize one such data surface, the visual exploration and analysis of many data surfaces is a grand challenge, stressing the users' perception and cognition.

We propose an approach that carefully integrates different projections and aggregations of the data surfaces at three different levels: one scalar aggregate per surface, a 1D profile per surface, or the surface as such. The simplest level captures one surface as one aggregation scalar, such as the surface's maximum, minimum, median, or mean, for example. These scalars can be easily visualized using standard views, such as, e.g., parallel coordinates. In the next step, experts want to investigate the surfaces $f(x,y)$ with respect to each of the two axes of the surfaces. As the view onto the surface along one axis can be considered as a collection of curves, we use the curve view to depict the surfaces at this level. Finally, analysts want to see the surface itself at some point of the analysis. Although we cannot efficiently visualize whole families of surfaces at once, we can visualize one surface (or a few of them using a real 3D view, or a 2D height map).

In this chapter we demonstrate the necessity for a flexible visual analysis system that integrates many different (linked) views for making sense of this highly complex data. To demonstrate the usefulness of the new approach, we exemplify it in the context of a meteorological multi-run simulation data case and in the context of an engineering domain, where our collaborators are working with the simulation of elastohydrodynamic (EHD) lubrication bearing in the automotive industry.

The author initiated the main concept and refined it in numerous discussions with Helwig Hauser and Denis Gračanin. He implemented the main idea, significantly contributed to the writing of the paper, and was also responsible for the communication with the domain experts. The paper was presented at the IEEE Visualization 2009 conference in Atlantic City, New Jersey, US, and it was published in *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, 2009.

1.7.7 Chapter 8: Path Line Attributes - an Information Visualization Approach to Analyzing the Dynamic Behavior of 3D Time-Dependent Flow Fields

In this chapter we are not dealing with ensemble simulation data. Instead, we have data from a single simulation run, describing a 3D time dependent vector field. Such data is complex in itself, and an efficient analysis of a time-dependent field is still a challenge in industry and research. We describe an alternative approach to the vector field analysis.

Our approach starts with the extraction of a number of features, both scalar values and time series, at each point of a regular sampling of the 4D space-time domain. We have focused on properties describing the (local or global) behavior of path lines, being either classical and well-established values in vector algebra, or properties newly suggested in this paper. The result of this step is a *path line attribute data set*: a four-dimensional multivariate data set collecting all computed path line properties. Although this is not an ensemble simulation data, it perfectly fits to the newly introduced data model.

We interactively drill down using the path lines attributes and then show the selected path lines and the interesting properties facilitating a focus+context visualization. In addition to standard 2D views, a 3D view showing the path lines themselves is also deployed. This way the user is able to do a simultaneous exploration in the 4D space-time domain of the flow and in the abstract path line attribute space. This results in new insights into characteristic substructures of the flow which leads to a better understanding of time-dependent vector fields.

The author contributed to the main idea development, especially with respect of the use of interactive visual analysis. He implemented the necessary extensions to the ComVis tool. He also contributed to the writing of the paper, in particular to the parts related to ComVis and to the interactive visual analysis. The paper was presented at the TopoInVis Workshop 2007 in Leipzig, Germany, and it was published as a chapter in: *Topology-Based Methods in Visualization II*, Springer, 2009

1.7.8 Chapter 9: A Statistics-based Dimension Reduction of the Space of Path Line Attributes for Interactive Visual Flow Analysis

This chapter represents an extension of the research described in the previous chapter. There are many path line attributes we can compute, and it is not clear which to select. As the attributes computation can be computationally demanding, a clever strategy for the choice of parameters is needed. Furthermore, it can be expected that a brute force approach which would generate many parameters produces a considerable information overhead, since many of the attributes are computed from the same velocity field. In general, different feature detectors may systematically correlate to each other because they either describe the same aspect of the flow or are related to each other by physical principles (e.g., velocity and vorticity by the vorticity equation). From the practical side, a systematic analysis of a data set becomes increasingly challenging the more dimensions it contains. Hence, a canonical question in this context is: Is there a common subset of the path line attributes that captures "all" complexity of the data set? Or, in other words, the question we investigated is: What is the intrinsic dimensionality of the path line attribute space?

In this chapter we investigate several CFD data sets with exploratory factor analysis, with the goal to find a common set of variables that can be used as a starting point for a deeper analysis of CFD data. This variable set should capture the underlying physical processes in fluids with as little as possible redundancy. The analyzed data sets span different geometries, constant/non-constant inflows as well as different simulation methods to prevent the variable set from being specific for one type of simulation/geometry/application. In total, we analyzed five different data sets with different geometries and simulation methods and one analytic data set. For greatest possible generality, we use only the velocity fields to calculate the path lines and their attributes. This means that similar factor patterns across the data sets are due to the common underlying principles of fluid dynamics and not due to similarity in the data sets. We identified six representative path line attributes.

The author contributed to the main concept and idea development. He especially contributed to the parts related to the exhaust manifold. He participated in numerous discussions and supervised Alan Lež, one of the co-authors, during his research. The paper was presented at the IEEE Pacific Visualization Symposium 2012 in Songdo, Korea, and it was published in *Proceedings of IEEE Pacific Visualization Symposium (PacificVis) 2012*.

1.8 Summary

As the systems' complexity grows, conventional analysis methods are not sufficient an more. This thesis describes a part of our research on the analysis of complex simulation examples. The main challenge is not the data size (although, ensembles can be considered large), but the data complexity. Complex data and multi-dimensional parameter spaces still represent a challenge for analysis. The methods described in this thesis have been evaluated in numerous sessions with domain experts. We have often tried to quantify the speed-up compared to the conventional workflow. In many cases the engineers could not explicitly say how much faster they were. They often estimated a speed-up of at least an order of magnitude. Besides speed-up, they often pointed out the additional insight they gained through interactive visual analysis. They often said, that many analysis tasks are simply impossible using a conventional approach. Probably, the biggest gain is exactly the insight they gain. Better understanding of the system makes them more efficient in future designs as well. Based on our research, the AVL company [AVL15] incorporated many of the newly proposed approaches in their commercially available software suit, which proves the usefulness of the here proposed methodology.

The research on the parameter space exploration and ensembles analysis is far from complete. Interestingly, in 2005, when we started with ensemble simulation analysis, there were just a few papers dealing with multiple simulation runs. The trend has changed since then, and in the recent years many researchers deal with the same approach. We are currently researching hierarchical simulation models. Hierarchical simulation models have certain parts modeled using different levels of detail. This adds complexity and requires a careful rethinking of the available analysis methods. The really large ensembles resulting from parameter spaces having thousands of parameters (which is not unrealistic) pose a great challenge, and will probably be an active area of research in the near future, as well.

2

Interactive Visual Analysis of Multi-Parameter Families of Function Graphs

Published as:

Krešimir Matković and Josip Jurić and Jürgen Krasser and Helwig Hauser:
Interactive Visual Analysis of Multi-parameter Families of Function Graphs
Proceedings of Third International Conference on Coordinated and Multiple Views in Exploratory Visualization, 2005. (CMV 2005)

Interactive Visual Analysis of Multi-Parameter Families of Function Graphs

KREŠIMIR MATKOVIĆ, JOSIP JURIĆ, ZOLTÁN KONYHA,
JÜRGEN KRASSER, AND HELWIG HAUSER

Abstract

The paper describes a method developed for interactive data visualization and exploration with applications in the automotive industry. The input data set contains a large number of function graphs. Each of the graphs is characterized by a set of basic attributes. The technique that is used for visualization includes two linked views: a map view (or attribute space view), where all function graphs are represented as a point or an icon on the map, and a linked function graph view. The map view provides additional visualization possibilities and allows user interaction. Additional features like brushing in both views, graph management, and related issues like interpolation of the graphs are described.

2.1 Introduction

The amount of information we are being confronted with nowadays increases constantly. Effectively exploring and analyzing huge amounts of data is a challenging problem, perhaps even more than collecting or processing this data.

Plain data tables or simple 2D charts are not always very useful when it comes to large data sets. Furthermore, people often look for various correlations and want to compare specific data. Information visualization tries to make it easier for the user to analyze and explore large data sets by cleverly and interactively displaying information [CMS03, CMS99, Tuf90, Tuf01].

Information to be explored can come from various disciplines and application areas and there is no universal solution that can fit all of those areas. In this paper we describe a method which can be used to analyze and explore data sets containing families of function graphs, where each of them corresponds to a point in a 2D domain. The data set looks like this: we have a 2D domain and for each point in this domain (or for some of the points) there is a corresponding graph describing an attribute in two other dimensions.

One easily comprehendable example of such a data set is a diagram of temperature over time measured on a number of places within a certain geographical area. The geographical map represents the 2D domain, and the temperature vs time data

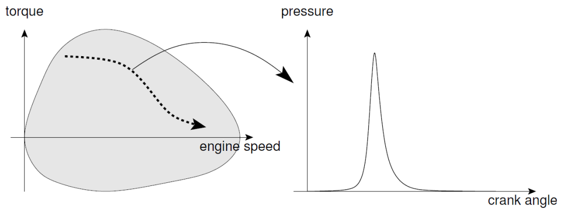


Figure 2.1: Left: an illustrative example of an engine characteristic diagram that shows torque versus engine speed. The engine's working point is always within the area shown in this diagram. The dotted line indicates the motion of the engine's working point as the car accelerates. Right: gas pressure in one of the cylinders versus crankshaft angle at one specific point in the engine characteristic diagram.

at specific points on the map represent additional two dimensions of the data set. Figure 2.2 illustrates the example.

Another example of a similar data set comes from automotive engine design. An engine characteristic diagram is a 2D diagram that shows engine speed (in rpm) on the X axis and torque at the crankshaft on the Y axis. At every engine speed there is a maximum torque that the engine can produce when the the accelerator is fully depressed. There is also a minimum torque (with the accelerator not depressed at all) which is negative and can be used to decelerate the car. While driving a car the working point of the engine follows a path within the engine characteristic diagram.

An example of the engine characteristic diagram is shown in Figure 2.1. For each point of this diagram there are several interesting properties to be investigated, for instance gas pressure in cylinders or main bearing forces as functions of crank angle.

Time-series are a special kind of function graph. This special case has been dealt with before. *Time Searcher* by Hochheiser et al. [HBMS03] is used to interactively explore time-series data. Time-boxes are used to specify queries. Time-boxes can be manipulated to specify various times.

Van Wijk and van Sellow [WS99] have used calendar view together with time-series data. The main scope was again various time series data. Since data was gathered for various days, the calendar was used to visualize time-series data origins. The authors also showed how clusters can help in finding similar time-series in the data set

Li Zhang at al. [ZZR03] used higher Fourier harmonics to enhance the visualization of time series data.

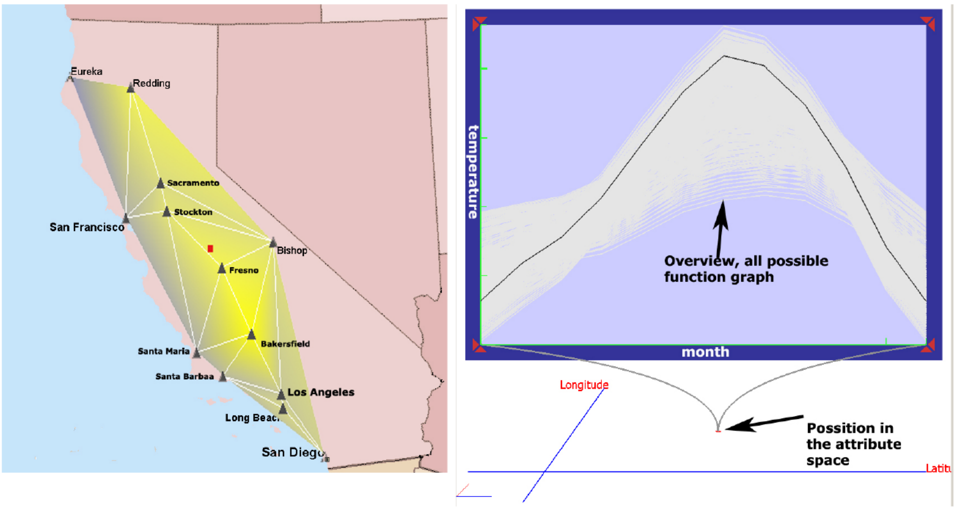


Figure 2.2: Left: attribute space view and right: function graph view.

2.2 Motivation

When users work with large sets of function graphs associated with points in the attribute space they often want to explore the data first. They try to find locations in attribute space that generate function graphs which have some interesting or unusual property, or conversely, ones that exhibit some regularity or follow a pattern. They often have little a priori knowledge about the data set. They do not exactly know what pattern or property they are looking for, or where the feature is likely to appear in the attribute space.

This calls for a visualization method that allows quick and easy, yet accurate exploration of the data. Two simultaneous views of the data set should be provided: (1) a good overall view of all function graphs to support navigation in the data set as well as (2) a detailed view which offers more information about specific graphs. Picking graphs from the attribute space must be made as effortless as possible and should require minimum user interface action.

After some interesting locations have been identified in the attribute space, another typical task is the detailed comparison of graphs associated with these areas. In order to perform this efficiently one must be able to observe the specific graphs at the same time and still not loose the context where the graphs belong [BMMS91].

Working with static 2D charts is obviously not practical if the attribute space is large. For example, numerical simulations where a large number of parameterized cases are calculated produce large sets of result graphs. These graphs must be examined in search of model parameters satisfying a certain criteria. It is clearly impossible to explore a set of thousand function graphs without the possibility of navigating the data set interactively. More complex queries like "show me points in the attribute space whose function graphs fulfill this-and-that criteria" are obviously even less feasible without interactive visualization methods.

Although displaying the data set as a 3D animated sequence where the time represents one of the coordinates is an obvious approach, it can only be useful in some cases. It is suitable to offer an overview of the data set but it is less practical for interactive navigation. It is even less suitable to show detailed and accurate information about graphs of specific points in the attribute space. In cases when the spatial distribution of the data at a specific position in time is less important than the temporal behaviour of a specific point in the attribute space, this way of displaying the data may even be inappropriate.

Therefore we suggest a different approach. Naturally, one must note that there is no perfect solution which could aid all visualization tasks even for one specific type of data. Existing methods that show animated views for a better overview in time domain lack numeric accuracy. Sheets of numeric data and charts are accurate but lack overview. We assume that the method suggested in this paper could complement these existing methods and offer an overview and detailed information simultaneously.

This is a way to explore the data, in particular when many variants have to be examined, which is a typical scenario in optimization, for example. Interaction opens new ways of data exploration.

2.3 Technology Content

The main idea is to have two linked views: the map view and the curve view. The map view displays the domain (map, engine characteristic,...) while the curve view displays the data available for each point from the domain (temperature over time, cylinder pressure over crank angle,...) The map view can contain additional information such as glyphs for points where data is available, or a color-map of some interesting data characteristic (e.g. the maximum, the average, or mean value of the data) The system is interactive and allows the user to navigate and select points both in the map view and in the curve view. The curve view can display more curves simultaneously in 2D or, optionally, in 3D.

Furthermore, the function graph view contains an overview part (showing the actual position of the selected point in the attribute space) and the graph corresponding to that point. Besides multiple selection in the attribute view, a collecting principle is described as well. The user can collect various graphs during the exploration, and compare them afterwards.

Finally, we describe an application which is implemented based on above mentioned principles. The application is used to visualize engine characteristic diagram.

It has been developed together with AVL [AVL15], one of the leading companies in the field of powertrain simulation, measurement and design.

2.3.1 Basic Idea

We describe a general framework for interactive analysis and exploration of data sets containing a family of function graphs each of them belonging to a point in 2D domain. The main idea is to have an interactive visualization method, which will make it possible to easily select a point in the attribute space and get the corresponding graph. Furthermore, the inverse process, selecting a function graph with a certain characteristic in order to locate similar graphs in the domain should also be possible.

2.3.2 Function graphs and interpolation

The data set we are concentrating on consists of a number of function graphs:

$$f_P(x)$$

Each of the function graphs is characterized by a set of parameters (or attributes) $P = \{p_1, p_2, ..., p_N\}$. In a general case this gives N-dimensional attribute space. The function graphs from the input data set are represented as points in this N-dimensional space and they are scattered non-uniformly across this space. The complete input data set can be formalized as:

$$\mathcal{F} = \{f_{Pi}\}$$

In this paper we concentrate on a case where 2 out of N parameters are visualized in the attribute view. This gives us a 2D space which can be thought of as a map. The other linked view displays interactively selected function graphs.

In most cases the function graphs are not defined for each point in the map view. If the defined points are dense enough, it might be sufficient to choose the nearest defined point. However, there might be some applications where only few points in the 2D domain are available. These points can be scattered in the parameter domain on an unstructured grid and just picking the nearest defined graph when performing a selection in the map view is not always satisfactory.

The natural solution is to provide interpolated curves at all points within the map view for which the data is not available [She68]. Of course, the interpolation can only be done if the nature of the data allows it. The first requirement is that the data set is continuous.

Let us introduce the interpolation operator Ω :

$$\begin{aligned}\hat{f}_P(x) &= \Omega(\mathcal{F},P)\\ &= \Omega_P\mathcal{F}\end{aligned}$$

The unknown function graph $\hat{f}_P(x)$ can be interpolated from the graphs of the neighboring data points with a method that best fits the data.

To enable efficient interpolation, the Delaunay triangulation is performed as a first step on all the defined points within the 2D map .

If a graph $\hat{f}_P$ at point $\hat{P}$ needs to be interpolated the following steps are performed:

1. The triangle $\triangle(P_1,P_2,P_3)$ containing the point $\hat{P}$ is located.
2. The homogeneous barycentric coordinates (t_1,t_2,t_3) of the point $\hat{P}$ are computed. Homogeneous barycentric coordinates are barycentric coordinates normalized such that they become the actual areas of the subtriangles $\triangle(\hat{P},P_2,P_3)$, $\triangle(P_1,\hat{P},P_3)$, $\triangle(P_1,P_2,\hat{P})$ normalized by the area of the original triangle $\triangle(P_1,P_2,P_3)$. These coordinates are also called *areal coordinates* [Cox69].
3. The resulting function graph is computed as the weighted sum of the three related function graphs using homogeneous barycentric coordinates as weighting factors.
$$\hat{f}_P(x) = t_1 f_{P1}(x) + t_2 f_{P2}(x) + t_3 f_{P3}(x)$$

Using linear interpolation for interpolating temperature data on a map is likely to be a valid approach, assuming the data is not sampled too sparsely. In this case trilinear interpolation of the graphs of the three nearest points on the map is an acceptable way to fill in the missing data.

Interpolating gas pressure curves in an engine characteristic diagram is an interesting example where a more sophisticated interpolation method is required. Using a simple linear interpolation may lead to incorrect results due to the specific shape of the curve (Figure 2.3), and possible phase shifts between the cylinder pressure curves at various engine speeds.

One possible solution to the problem is to align all involved cylinder pressures by cyclic shifting so that the x - coordinate of the peak value for all involved curves matches. The phase shift of the resulting graph is interpolated between the phase shifts that were used for the initial alignment using the same weighting factors (t_1,t_2,t_3) (Figure 2.4)

Another solution to this problem is to use interpolation in frequency domain instead of direct interpolation. The function graphs are transformed to the frequency

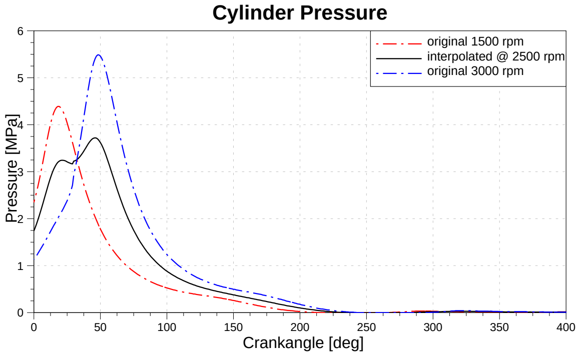


Figure 2.3: An erroneous interpolation of cylinder pressure. There is a phase shift between the pressure graphs (dashed lines) of different crankshaft angles, thus peaks are at different angles. Simple linear interpolation produces a graph (plain line) with two local maxima.completely unrealistic of cylinder pressure.

domain first, then the weighted sum using homogeneous barycentric coordinates is calculated for phase and amplitude data before converting them back to the original domain. Both methods are specific and usable for cylinder pressure diagrams because the nature of the diagram allows and requires such interpolation.

2.3.3 Visual Metaphor

Since there is an attribute space and a family of curves we propose to use two views for visualization[WS99].

The attribute space view is used to depict the domain. Points where function graphs exist are of main interest in the attribute space. If there are just a few points, all of them will be depicted. In case of large number of points (maybe even more points per pixel), only a subset will be shown. Furthermore, one characteristic of the corresponding function graph, such as maximum value, minimum value, average, etc. can be shown in the attribute space as well. Trilinear color interpolation will be used to show this characteristic. If the points are too sparse, additional points have to be inserted into the triangulation in order to avoid the visual artifacts of the trilinear interpolation. To create these additional nodes the interpolation described in Section 2.3.2 is used - same as when interpolating the data for the function graph view.

The second view is used to show function graphs. It shows function graphs in a plane. Besides the current function graph, all function graphs are shown in light gray in the same plane as an overview. Furthermore the 2D attribute space is depicted perpendicular to the plane, and the origin point of the highlighted graph is shown as well. This additional overview information helps the user in seeing a particular

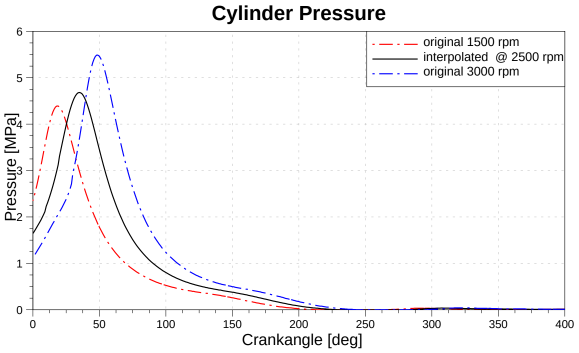


Figure 2.4: The correct interpolation of cylinder pressure. The peak values of the pressure graphs are aligned before linear interpolation. The resulting interpolated pressure graph has a very natural shape. Compare this to Figure 2.3

graph in the overall context. Note this graph's source is highlighted in attribute space itself as well. This dual highlighting helps the user to keep concentrated on the graph view, and still have an overview of the graph's location in the attribute space. Without the overview feature, the user would have to repeatedly change focus from graph view to the attribute view. Figure 2.2 shows the attribute view, with the maximum value depicted using a color scale, and the function graph view with overview and one curve highlighted. A real map image has been used as background for the attribute view.

2.3.4 Workflow and Interaction

The usual workflow while exploring a data set consisting of family of function graphs involves two main actions:

- attribute based curve exploration, and
- locating graphs in attribute space based on their properties.

This means that user usually browses through the attribute space and observes the function graphs, and later selects some characteristics in the function graphs and finds corresponding positions in the attribute space. The straightforward interaction for the first action is "mouse-over", which depicts a function graph corresponding to the point that is currently under the mouse pointer. Depending on the mouse pointer position, this can be either an existing function graph or an interpolated one. This information should also be displayed to the user.

Seeing only one graph at a time makes it difficult to compare the curves. A method for selection in attribute space is required. There are numerous possibilities

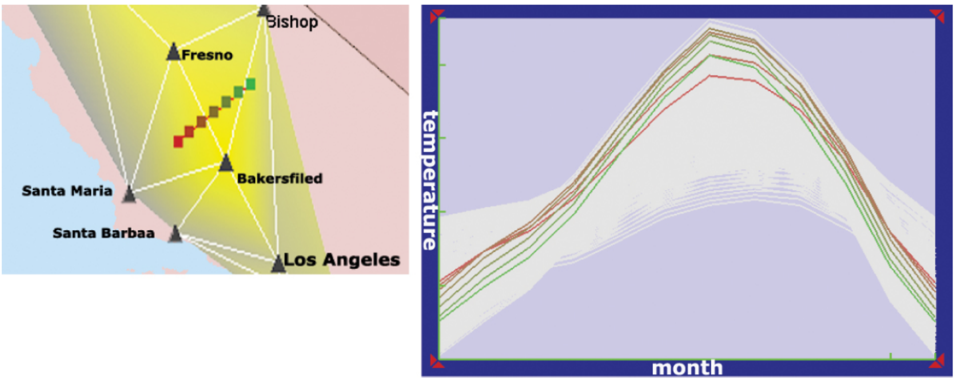


Figure 2.5: A 2D display of function graphs that belong to a line in the attribute space. Note that the overview is depicted as context and the colors of the seven function graphs match those of the points in the line they are associated with.

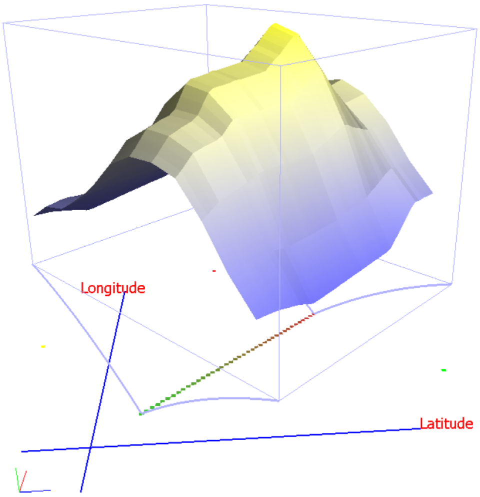


Figure 2.6: A 3D display of function graphs that belong to a line in the attribute space.

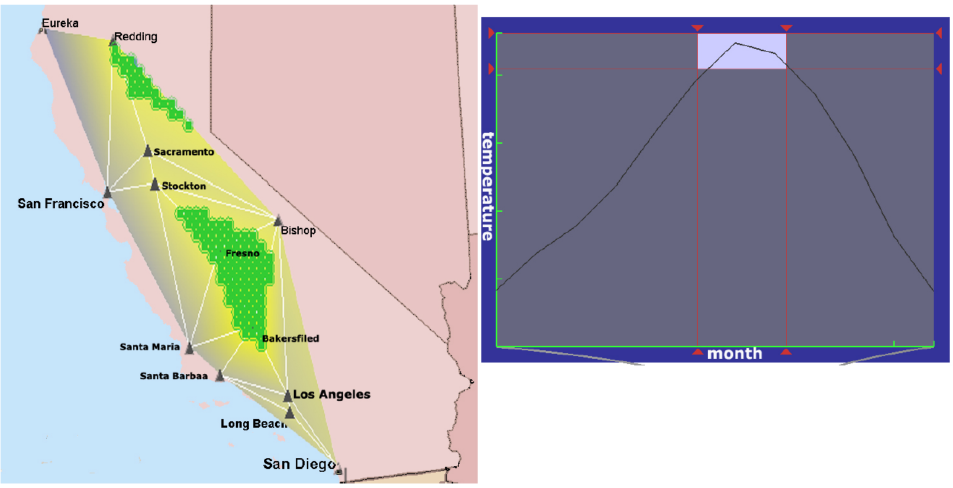
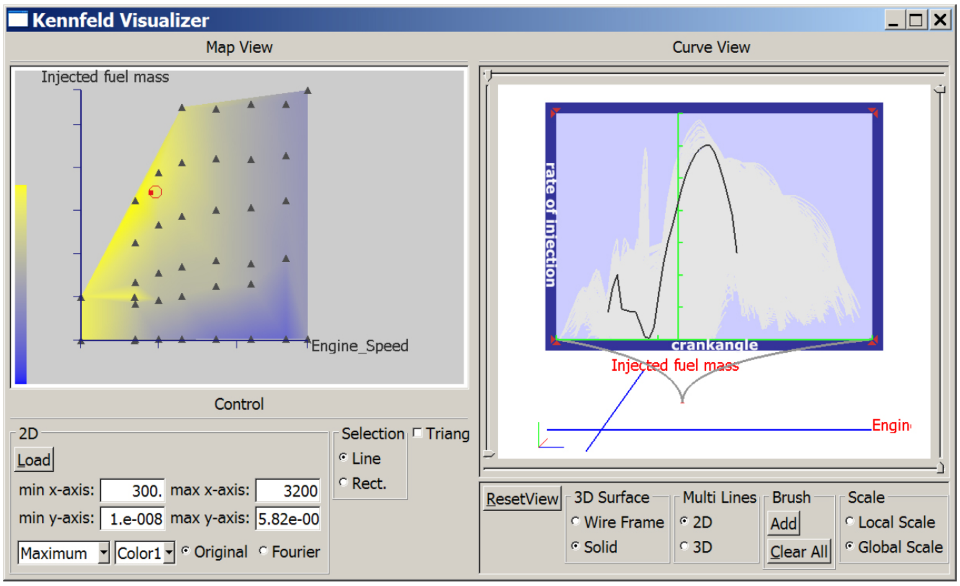


Figure 2.7: A brushing example. In the function graph view the user has highlighted a rectangle to select very high temperature in July. All points whose temperature graphs pass through the highlighted rectangle are marked in the attribute view to display areas of high July temperature.

to select multiple points in attribute space. A line selection method which allows the user to select a line in the attribute space is the simplest. If we want to have separable, independent axes in the attribute selection, a rectangle selection with the axes aligned with the attribute space is needed. With both line and rectangle selection, the selection can be colored using a color gradient. The color gradient provides additional information about the sources of the function graphs. The color range for this gradient is chosen to be distinct from the color coding used in the attribute view.

When the user selects multiple points in the attribute view, more function graphs are displayed simultaneously. Each graph is drawn in the color of its selection point. Figure 2.5 illustrates a simple line selection, using the temperature data set. Note that the colors of the function graphs are the same as that of the corresponding points in the selection in attribute space. Having multiple function graphs corresponding to a line in the attribute space, the resulting curves can be extruded to create a 3D surface. 3D surfaces can offer new insights into the data. Figure 2.6 shows another line selection depicted as 3D surface. Note that extrusion to 3D is possible only for line selections in the attribute view.

It is possible to combine more selections using Boolean algebra, which adds additional possibilities to the data exploration and analysis. Selections are very powerful for exploration, but they have a limited lifetime. When the user makes a new selection the old one is lost. That is why a collection principle is introduced. Collections are actually containers that hold a number of graphs. As the user browses



min x-axis: 300. max x-axis: 3200
min y-axis: 1.e-008 max y-axis: 5.82e-00
Maximum Color1 ● Original ○ Fourier
Selection ● Line ○ Rect. ☐ Triang
3D Surface ○ Wire Frame ● Solid
Multi Lines ● 2D ○ 3D
Scale ○ Local Scale ● Global Scale

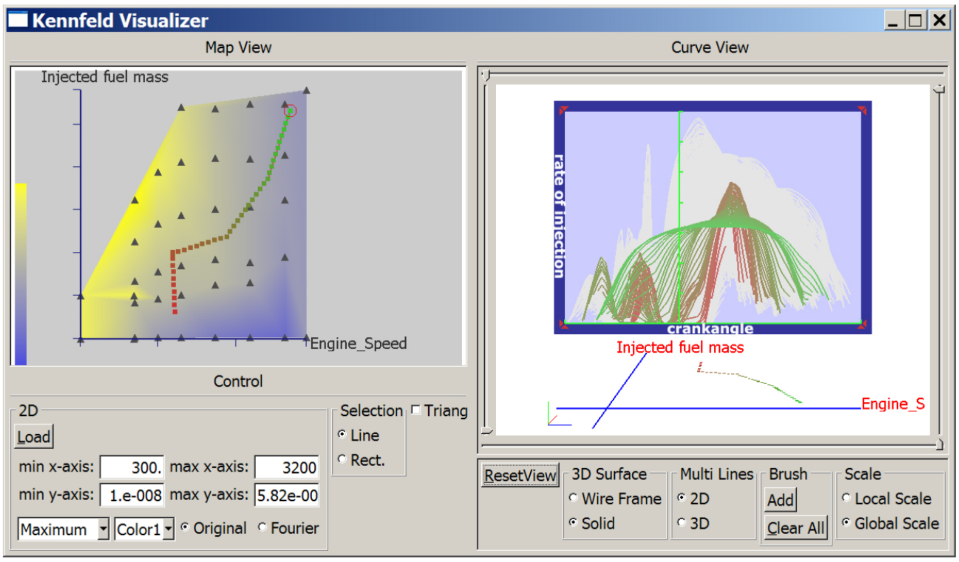
Figure 2.8: The typical shape of an injection curve. Two peaks, the smaller pre-injection and the main injection, are clearly visible.

the attribute space and finds a graph of interest he can add it to a collection and continue browsing. Graphs can be added to existing collections at any later point. Several collections may be defined and used simultaneously, much like having multiple clipboards in a text editor. It is also possible to create new collections via Boolean operations on other collections. After the user has created collections he can simply display and explore them.

Once a selection is made or a collection is displayed, the user can start working with the function graph view. The user often wants to locate the graphs in the attribute space having certain characteristics. The use of rectangular brush [AS94, FS95] for selection of the interesting areas within the function graph view provides an excellent way to do this kind of analysis. The points corresponding to the graphs passing through the selected rectangle are highlighted in the attribute space. For instance, by use of brushing it is easy to find which areas on the map had average temperatures in July over some threshold. Figure 2.7 shows such an example.

2.4 Application

We have applied the above described principles in the *KennfeldView* tool developed together with AVL. The application is implemented in C++ using OpenGL [Ope] and FOX Toolkit [FOX]. The tool is available on several platforms, including MS



min x-axis: 300. max x-axis: 3200
min y-axis: 1.e-008 max y-axis: 5.82e-00
Maximum Color1 ● Original ○ Fourier
Selection: ● Line ○ Rect.
3D Surface: ○ Wire Frame ● Solid
Multi Lines: ● 2D ○ 3D
Scale: ○ Local Scale ● Global Scale

Figure 2.9: Polyline selection in the attribute space. The data points of the line are drawn in colors ranging from red to green. Each function graph is drawn in the same color as the point it is associated with to help the user identify which point the graph belongs to.

Windows and Linux. Primary application of the tool is the exploration and analysis of engine characteristic diagrams. After the data is gathered either by simulation or measurement on real engines, the user loads the data, and starts exploration. The attribute view can display maximum, minimum, average and mean value of the corresponding graphs as underlying color map. The *KennfeldView* supports three types of interpolation: linear, modified linear and interpolation in Fourier space. A proper interpolation has to be selected based on the input data. In order to perform the interpolation the input points in the attribute view are tessellated. The user can choose to display the tessellation of the attribute view, or the icons on the points where graph data is available, or both.

Once the data is loaded and interpolation type, color scale and underlying map are chosen, the user can start data exploration. While moving the mouse pointer over the attribute view the corresponding graphs are instantly displayed in the graph view. Single or multiple selection, as described in Section 2.3.4 are supported. If the user chooses to display multiple selections as a 3D surface (possible only for line selection) the surface can be displayed either as a surface or a wire-frame. The camera can be freely moved in this mode. 2D display of multiple selections does not allow camera movement, but allows brushing in the graph view, which is a very important feature for engineers.

Let us show how an actual data exploration might look like: we have two data sets, the first describing the rate of injection as a function of a crankangle. Rate of injection indicates how fuel mixture is injected into the cylinder. Figure 2.8 shows a characteristic injection curve. Two peaks are clearly visible, the first one, so called *pre-injection*, and the second one, the *main injection*. Note that this form of injection curve is desired and engine designers try to achieve such engine characteristics. On the left hand side in the figure 2.8 the corresponding attribute space is shown, which depicts injected fuel mass and engine speed in rpm. The depicted attribute space represents the working range of the engine. The working point of the engine can be only be located within the depicted area in the attribute view. The engineers can use the tool to explore injection curves at various working points and to quickly see if there are some areas where the curve has only one peak (undesired curve shape), or has too low pre-injection ratio. Figure 2.9 shows an example of a polyline selection in attribute space. Note how the color changing from red to green aids the user in identifying areas that have an undesired (single) peak curve shape in the selection.

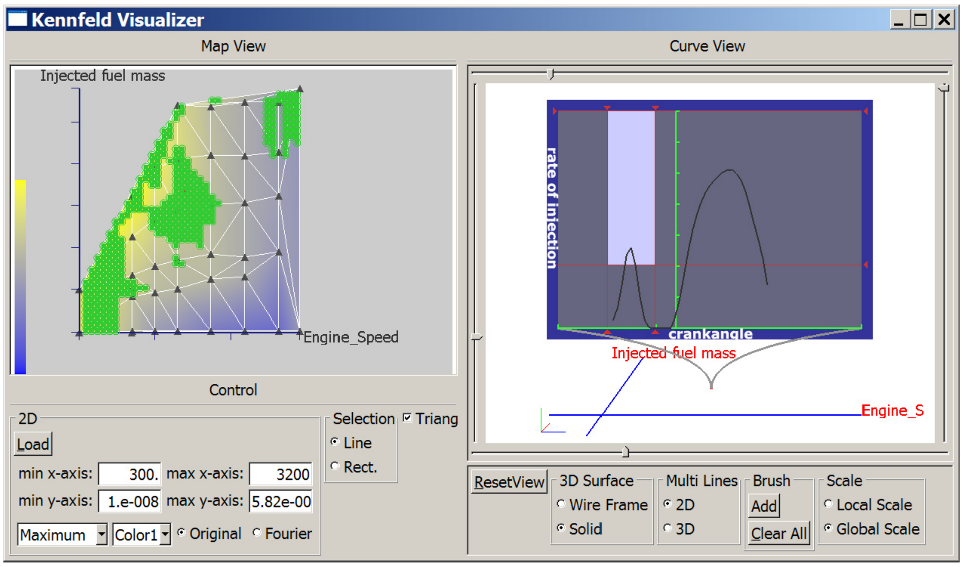
An inverse kind of exploration is shown in figure 2.10. In this case we have selected the area where the pre-injection amount exceeds certain value. We can clearly see all the areas in the attribute space where function graphs having this property are located.

Finally, figure 2.11 shows the cylinder pressure in the function graph view and the working area of the engine in the attribute view described by the load signal and engine speed. The attribute space determines the domain where the operating point of the engine may be while the engine is running. The load signal is directly proportional to the pressure on the accelerator. A rectangle selection with gradient brush is shown, which is a good way to explore tendencies with this kind of data.

The *KennfeldView* tool has been recently demonstrated to a group of mechanical engineers at AVL List GmbH. They had good initial impression about the application and are interested in integrating it into one of AVL's software products.

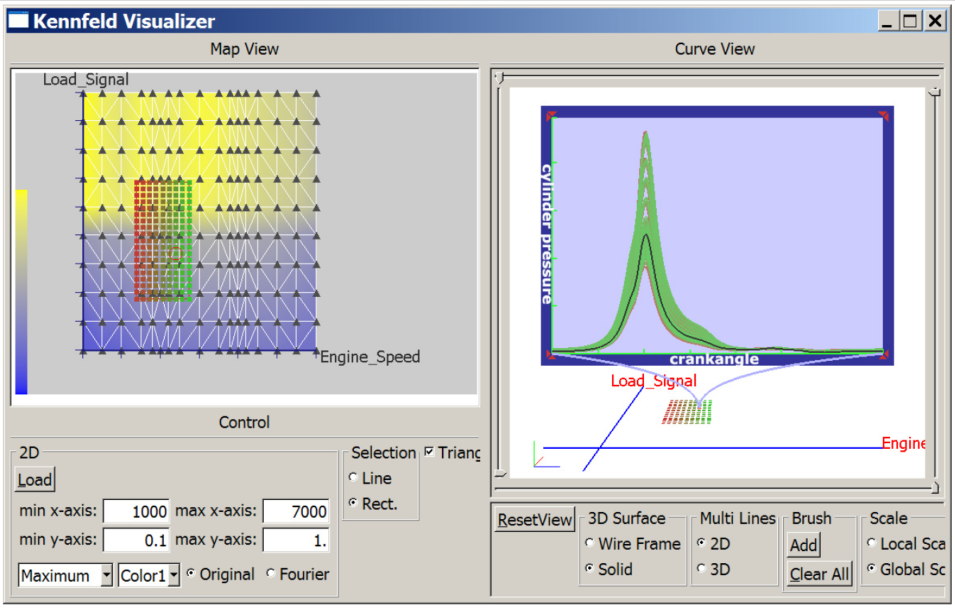
2.5 Summary and conclusion

We have presented a method to visualize a family of function graphs where each of them is associated with point in a 2D domain. Such data sets are ubiquitous, from all kinds of data bound to the map of an area (meteorological data, population, pollution, etc.), to special cases in engineering, like the working domain of the car engine linked to various crank angle dependent attributes at each operating point. We have described an interactive, dual linked view solution consisting of an attribute view and a graph view. Users can browse through the data by simply moving the mouse pointer over the attribute view and explore related function graphs in the graph view. Brushing in the graph view adds additional dimension to the exploration by highlighting the locations of graphs with certain characteristics on the 2D map. The engineers at AVL dealing with engine design are pleased with the



min x-axis: 300. max x-axis: 3200
min y-axis: 1.e-008 max y-axis: 5.82e-00
Maximum Color1 ● Original ○ Fourier
Selection ☑ Triang
● Line
○ Rect.
3D Surface: ○ Wire Frame ● Solid
Multi Lines: ● 2D ○ 3D
Scale: ○ Local Scale ● Global Scale

Figure 2.10: A brushing example. The user has highlighted areas where pre-injection ratio is above a certain threshold.



min x-axis: 1000 max x-axis: 7000
min y-axis: 0.1 max y-axis: 1.
Maximum Color1 ● Original ○ Fourier
Selection ☑ Triang
○ Line
● Rect.
3D Surface: ○ Wire Frame ● Solid
Multi Lines: ● 2D ○ 3D
Scale: ○ Local Sca ● Global Sc

Figure 2.11: Rectangle selection in attribute space. The curves in this example depict pressure in cylinders versus crankshaft rotation angle. The attribute space is defined with load signal (which is proportional to the pressure on the accelerator) and engine speed in rpm.

new interactive solution. It has many advantages over the previously used static 2D charts. *KennfeldView* may evolve to become a component of AVL's software suite.

2.6 Acknowledgements

We express special thanks to Andreas Münzer of AVL [AVL15] for supplying the engine characteristic diagram data set. We also thank Robert S. Laramee of VRVis for his valuable feedback. Parts of this work have been carried out in the scope of applied and basic research at the VRVis Research Center which is funded by an Austrian governmental research program called K plus (`www.kplus.at`).

3

Interactive Visual Analysis of Families of Function Graphs

Published as:

Zoltán Konyha, Krešimir Matković, Denis Gračanin, Mario Jelović,
and Helwig Hauser:
Interactive Visual Analysis of Families of Function Graphs
IEEE Transactions on Visualization and Computer Graphics, 12(6):1373–1385, 2006.

Interactive Visual Analysis of Families of Function Graphs

ZOLTÁN KONYHA, KREŠIMIR MATKOVIĆ, DENIS GRAČANIN,
MARIO JELOVIĆ AND HELWIG HAUSER

Abstract

The analysis and exploration of multi-dimensional and multi-variate data is still one of the most challenging areas in the field of visualization. In this paper, we describe an approach to visual analysis of an especially challenging set of problems that exhibit a complex internal data structure. We describe the interactive visual exploration and analysis of data that includes several (usually large) families of function graphs $f_i(\mathbf{x},t)$. We describe analysis procedures as well as practical aspects of the interactive visual analysis which are specific to this special kind of data (especially targeting the function graph characteristic of the data). We adopted the well-proven setup of multiple, linked views with advanced interactive brushing to assess the data. Standard views such as histograms, scatterplots, and parallel coordinates are used to jointly visualize parameters as well as dependent data. To support an iterative visual analysis, we provide means to build up complex composite brushes that span multiple views and that are constructed using different combination schemes. We demonstrate that engineering applications represent a challenging area for visual analytics. As a case study, we describe how technology is used in the optimization of a fuel injection systems in Diesel engines of passenger cars.

3.1 Introduction

The development of effective visualization and interaction techniques requires the understanding of the properties of the data and the typical tasks the users want to perform [TC05]. Unfortunately, this requirement is not always met, often because of insufficient collaboration and communication between visualization experts and the users. The users' ultimate goal is always to find expected phenomena to support (or reject) their hypotheses or to discover unexpected results that question their assumptions or the validity of the data acquisition process. This can lead to the generation of new hypotheses.

The challenges of data analysis and exploration are associated with very large data sets, increased dimensionality and the consideration of data semantics, including features, focus and context [DGH03]. Therefore, a visualization tool should be designed in close collaboration with potential users. Tool developers must be aware of the users' actual requirements, the usual tasks they need to solve, the shortcomings of their previously used tools, and their feedback on new ideas. A part of that process is a development of intuitive and effective visualization and interaction techniques based on a common data model. If designed well, the same principles can be used across several application domains, from real-time data monitoring to engineering design applications, including simulations.

Modern simulation software can generate massive amounts of data that require suitable analysis techniques to get an insight into the practical implications of the results. Simulation is increasingly used to assess the quality and potential of new designs early in the process (e.g. aircrafts and cars). Building real prototypes is time-consuming and expensive. Even though measurements on test bed systems are likely to remain an important way to verify designs in the future, the use of computational simulation in the design and production process can help to minimize the costs of the development and shorten the time-to-market for new products.

For example, in the automotive industry many different aspects of new designs are checked using simulation long before a new car is manufactured. Examples include mixture formation and combustion, engine cooling and filter re-generation, air conditioning in the passenger cabin and front shield deicing, as well as many others. Additionally, the increasing complexity of automotive subsystems, e.g., the power train, the intake and exhaust system, or the fuel injection subsystem, also requires simulation for optimization. The tuning of the injection system of modern cars, for example, is a multi-parameter optimization process. The operation of the injection system depends on several parameters in a very indirect way, thus optimization by experience and/or intuition is usually not possible.

In this paper, we present a new approach to the interactive visual exploration and analysis of measurement and simulation data. This approach is general enough for a number of application scenarios that share the same characteristics, including multi-parameter tuning problems. A major challenge (in general) is how to visually relate the multi-variate dependent variables to their multi-dimensional reference parameters (independent variables). We suggest a combination of different kinds of views with specific brushing interactions, all adapted to work well for the families of function graphs in order to facilitate the interactive visual exploration and analysis of such data sets. We have investigated the usability of our ideas in two very different settings: the analysis of road traffic data and the optimization of a fuel injection system. The road traffic data set serves as an illustrative example for the introduced technological concepts while the fuel injection system data set provides a case study, described in detail in Section 3.6.

The remainder of the paper is organized as follows. Section 3.2 provides an overview of related work. Section 3.3 gives a brief description of the data model used and the exploration procedures. Section 3.4 describes our proposed tools and methods for supporting these tasks. Section 3.5 introduces the typical tasks in the analysis of such data sets. Section 3.6 describes the use of the developed approach for a real world automotive engine design task. Section 3.7 provides closing remarks and directions for future work.

3.2 Related Work

Interactive visual exploration and analysis can benefit from previous results in many areas. We first address visual analytics and then provide an overview of visualization techniques for high dimensional data and the linked views principle.

Thomas and Cook [TC05] define visual analytics as "*the science of analytical reasoning facilitated by interactive visual interfaces*." It is a wide-ranging field of science that involves visualization and interaction methods combined with analytical reasoning, data representation and transformation as well as production and presentation of the results. Therefore it is difficult to find previously published work that encompasses all aspects. We are able to collect and generate data at an increasingly fast rate, but capability of analyzing the collected data lags behind [TC06]. Here we focus on work that describes how users gain insight into data, find expected and unexpected features and make decisions using visual tools.

Trafton et al. [TKT$^+$00] present a study of how experienced weather forecasters interpret complex visualizations to build their own qualitative mental models of weather conditions which in turn are the basis of the weather report they produce. In their experiments they show how the users perform *convergent thinking* (assembling evidence to support a hypothesis) and *divergent thinking* (thinking creatively to identify alternatives) at various stages of their work. Saraiya et al. [SND04] evaluated five visualization tools to determine which one provides the best insight into the specific data. González et al. [GK03] describe how an information visualization system was used by administrative data analysts. Ahlberg et al. [AS94] introduce a visual information seeking technique which focuses on rapid filtering and progressive refinement of search parameters.

There are numerous publications of having scientific visualization (SciVis) techniques applied to the visualization of simulation data with a perspective of engineering. Laramee et al. [LGD$^+$05] exemplify a thorough visual analysis of the coolant flow through the cooling jacket of a car engine by using various different flow visualization methods to reveal different aspects of the simulation data. Konyha et al. [KMH03] propose 3D icons for the analysis of simulation data of chain and belt drives.

There are simulation data types that are more effectively explored by even more abstract visualization methods. In these cases the user may be able to gain more insight if information visualization (InfoVis) techniques are used instead of or together with SciVis methods. Matković et al. [MJJ$^+$05] describe a method for the analysis of a fuel injection system. The time series data from the simulation is reduced and described by a set of scalars which results in a highly abstract view of the injection system. The case study section of this paper describes in detail how engineers can use the interactive, linked InfoVis views to explore and analyze simulation data of a fuel injection system.

The body of literature about the visualization of high dimensional data is vast. Following the terminology of Wong et al. [WB97] we focus on the visualization of *multidimensional* data. Keim [Kei00] classifies methods for visualization of high dimensional data into four groups: geometric projections, hierarchical methods, iconic methods and pixel-based techniques. In our work we have considered only geometric projections so far. Geometric projections include two of the most popular information visualization techniques: scatterplots [Tuf01], [Cle85] and parallel coordinates [ID90], [The00], [FWR99], [Kan01], [HGP99].

Using multiple, interactively linked views of the same data set allows the user to productively combine the information he or she gathers from the different views. The Attribute Explorer [TSWB94], [ST98] uses linked histograms to simultaneously represent the interaction between attributes and allow the user to narrow the focus by defining limits on certain attributes. The Influence Explorer [TSDS96] was developed for the exploration of data computed from a model given sets of parameter values as input. The user can select a set of points in either the parameter or the result spaces and see how this set corresponds to points in other dimensions in both spaces. Gresh et al. [GRW$^+$00] present an approach that links 3D visualizations to statistical representations to facilitate effective exploration of medical data. Doleisch et al. [DMG$^+$05] have used multiple linked views, including adapted information visualization views in the analysis of CFD simulation data. Piringer et al. [PKH04] interlink 2D/3D scatterplots and histograms with smooth brushing. Schafhitzel et al. [SWE05] link several texture-advected flow visualizations on slices with the 3D view of the vector field in an attempt to overcome occlusion problems. Matković et al. [MJK$^+$05] use Timebox-like [HS04] brushing to link the graph of a function to a scatterplot display of its parameter space. More advanced multiple view visualization systems can be configured freely to suit various data sets [NS00] and allow flexible coordination of views [Wea04]. As the number of linked views and the amount of coordination increases it may become necessary to visualize the visualization's structure and operation [Wea05].

3.3 Data Model

Generally speaking, a data model consists of a data definition and a manipulation language (structuring and operational definitions) [Tho97]. Data definitions that result from an engineering simulation, a real-world sensor data set, or intelligence data may be very similar. Consequently, the data sets under consideration share some common characteristics. The data sets contain values for m independent variables and n dependent variables.

The independent variables $\mathbf{x} = [x_1, \ldots, x_m]$ and their values define a subset I of the data set. A member of $I \subseteq \Re^m$ represents a specific set of values $\mathbf{x}_i$ of independent variables. For each $\mathbf{x}_i$, the corresponding set of values of dependent variables is provided. There are two types of dependent variables, regular and function graphs. While regular variables have a singular value for each $\mathbf{x}_i$, function graph variables use time as an additional independent variable to provide a set of values for each $\mathbf{x}_i$. A function graph can be visualized as 2D plot that shows how the value of a dependent variable changes over time. In other words, the regular variables $\mathbf{r} = [r_1, \ldots, r_{n_r}]$ depend only on $\mathbf{x}$ while the function graph variables $\mathbf{f} = [f_1, \ldots, f_{n_f}]$ depend on $\mathbf{x}$ and time $t \in \Re$. For a specific set of values $\mathbf{x}_i$ of independent variables and fixed time t_j we can define the set of values of dependent variables as $\mathbf{d} = [r_1(\mathbf{x}_i), \ldots, r_{n_r}(\mathbf{x}_i), f_1(\mathbf{x}_i, t_j), \ldots, f_{n_f}(\mathbf{x}_i, t_j)]$, $n_r + n_f = n$. The dependent variables and their values (possibly, over time) define a subset D of the data set. For a given function graph variable, $f_j(\mathbf{x}, t)$, we define a family of function graphs as a set of function graphs for each possible value of $\mathbf{x}$, $\{f_j(\mathbf{x}_i, t) | \forall \mathbf{x}_i \in I\}$.

Once the data set is defined, the question is how to analyze the data. In our data model, the manipulation language is an exploration language that enables search and pattern discovery without modifying the data set. From the visual analytics point of view, the goal is to discover, in an iterative manner, trends, tendencies and outliers in the data and to see how patterns in D map to the corresponding subsets in I and vice-versa. In order to achieve that, data exploration techniques must be conceptually simple, easily combined and visually intuitive.

The visualization framework is based on the described data model and a set of visual operators (brushing techniques) and views (histograms, scatterplots, parallel coordinates, etc.) that are linked together. The design of interactive visual analysis within this framework is based on the following principles. The analyst can select a varying number of views. Within each view, the variables of interest can be selected and the corresponding values displayed. The visual operators are used to select a subset of "interesting" values for the specific variables in the view. The selection is immediately displayed in all other views. Families of function graphs are of special importance in providing a visual space for patterns. Within a family of function graphs, we would like to select function graphs based on their shapes. It is possible to use a combination of function graph values to specify the desired shape of a function graph, i.e. the pattern.

We will use a real-world road traffic measurements data set to illustrate the concepts described in Sections 3.4 and 3.5. The data set is provided by the Traffic Management Center of Minnesota Department of Transportation [twi] that maintains an archive database of road traffic measurements from the freeway system in the Twin Cities metropolitan area. The data set contains 28 days of measurements from approximately 4,000 sensors grouped into about 1,000 stations covering ten main roads. Opposite directions on a road (e.g. north-bound vs. south-bound) are treated separately, thus effectively creating 20 one-way roads. I consists of the posi-

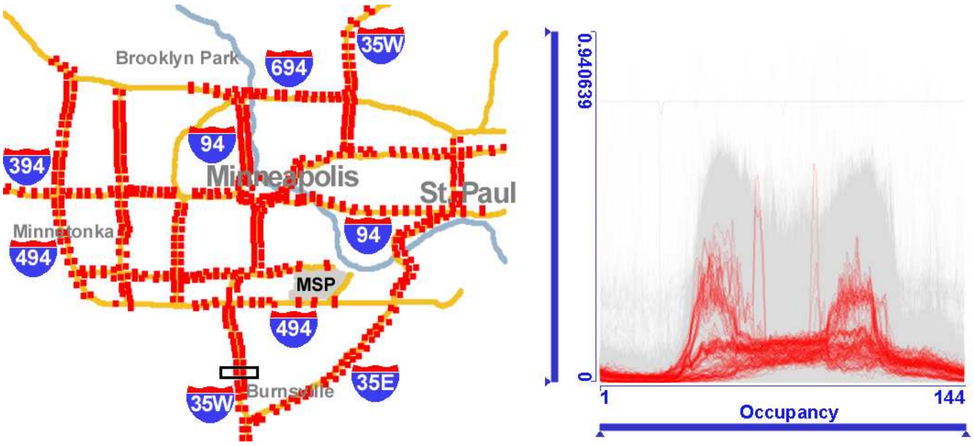


Figure 3.1: Left: map of all sensor locations in the Minneapolis freeway system traffic data. A schematic map is underlayed to provide context information. Each red dot marks the location of one *station* (which usually encompasses several sensors, *detectors*, one per lane). Right: road traffic occupancy graphs. Data from some sensors (marked with the black rectangle on road 35W in the map) is highlighted in red. Occupancy is defined as a percentage of time a detector detects vehicles. It is measured in ten minute intervals. An occupancy value of 0.7 means that for seven out of ten minutes a sensor detected vehicles.

tions of the sensors, road numbers and weekdays. The sensors report traffic volume and occupancy, thus D consists of two families of function graphs in this data set.

3.4 Tools for Analysis of Families of Function Graphs

We have developed a tool based on premises described in Section 3.3. The combination of basic, highly interactive views is sufficient to carry out a wide range of sophisticated analysis tasks. Interactivity plays a crucial role in analysis. Important and novel aspects that support interactive procedures are described in the following.

We currently offer up to six linked views including histograms, scatterplots, parallel coordinates and function graphs. We do not make any assumptions about independent and dependent variables in the sense that we would restrict any of the basic view to display either of them. The inputs of the views can be mapped to any attribute of the data set, both independent and dependent variables. The user can arrange the views as desired, can have more than one instance of the same view type showing the same or different attribute sets. It is possible to temporarily maximize one view for more detailed examinations. Histograms, parallel coordinates and scatterplots are standard, well known views [CMS99], thus we do not describe them here in detail. However, it is worth mentioning that the point size in scatterplot views can optionally be proportional to the number of data items represented

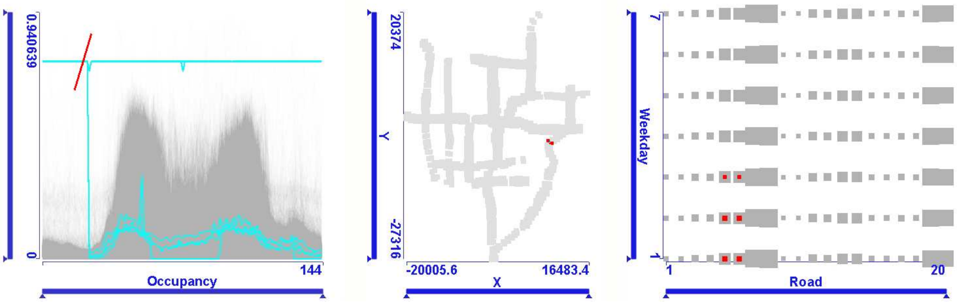


Figure 3.2: Several occupancy graphs of atypical shape have been selected by the red line brush. We conclude from very high occupancy values that these graphs indicate malfunctioning sensors. In the linked map view (scatterplot view of sensor coordinates) we can see that there are two malfunctioning sensors next to each other. In another linked scatterplot view weekdays and road numbers are displayed. Each column represents one direction (for instance, south-bound) of a road. We can see that these sensor are on road 35E and they did not work for three days.

by a single point. The more items a point represents the larger it is. An example is shown in Fig. 3.2: larger points indicate more sensors on the road. Similarly, the sizes of points highlighted in the focus set are also proportional to the number of items brushed (Fig. 3.11). Thus the ratio of brushed items versus context represented by a point in the scatterplot is indicated by point sizes.

The function graph view displays a family of function graphs at once. If the number of function graphs in the family is large then the display can become visually cluttered and non-informative. In order to represent the characteristics of the family better, we can (optionally) render the pixels through which more graphs pass through with higher intensity.

In the following we will describe generic interaction principles and elaborate on the specific requirements of brushing and linking in various views.

3.4.1 Generic Interaction Features

If the basic views listed above are independent then they provide limited insight into the data set. However, if areas of focus can be highlighted with applicable brushing techniques and this focus area is linked to other views then correlations and dependencies in the data can be revealed. Our system supports interactive brushing and linking and the number of currently brushed data items versus total is always indicated. The user can perform brushing in any of the views and all other views will also highlight the brushed items while the context is shown in a different, less saturated color. Whenever applicable, the view can be zoomed to show the brushed region only. The brushes can be resized and dragged to new locations which helps

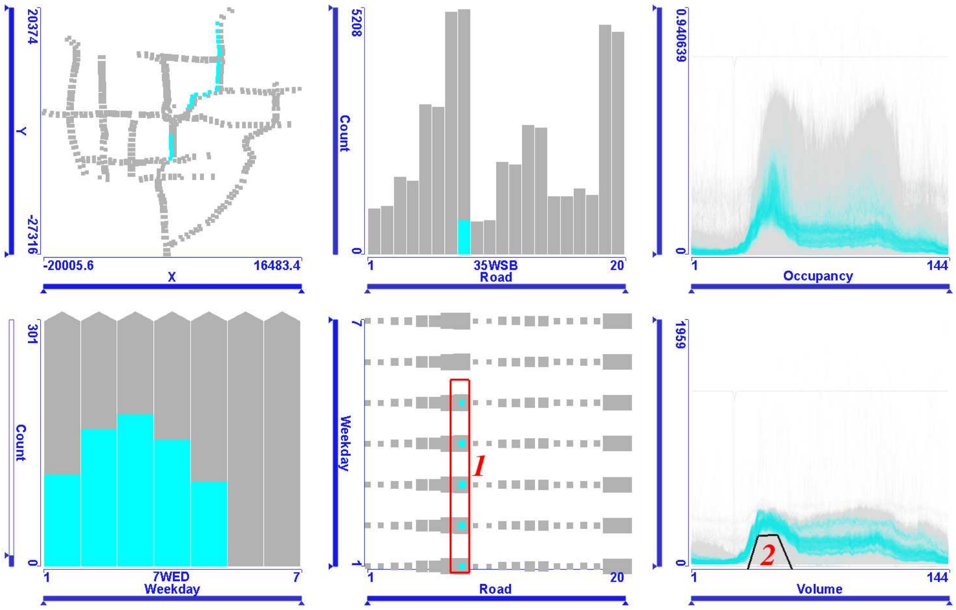


Figure 3.3: Snapshot from an interactive visual analysis session of traffic data in the Minneapolis metropolitan area. The creation order of brushes is indicated by red numbers. Here we look for locations of high volume morning traffic on a given road on weekdays. The user has selected the road and weekdays in the scatterplot (brush 1) and then removed low traffic volume graphs intersecting brush 2 using SUB operation. The locations are highlighted in the linked map. The road number (35WSB, i.e. 35W south-bound) is shown on mouse-over. We can see that heavy morning traffic on south-bound 35W mainly occurs mainly north of the downtown, i.e., towards it.

in the interactive data exploration. A tabular display of the currently brushed items can be opened when the user needs detailed numeric information.

With simple brushing and linking it is usually a problem to locate the matching brushed data items in different views. If more data items are brushed in a view then all corresponding items are highlighted in other views, but we cannot visually identify the matching items in the different views. We have applied an optional *c*olor gradient along the brush and used this color gradient in the linked views to establish visual identification of the correlated data items. This aids the user in discovering tendencies in the data set. See Fig. 3.9 for an illustration of the gradient brush.

Another improvement is *c*omposite brushing, a query tool which is a result of logical operations performed on brushes. Composite brushing makes it possible to build queries that specify several overlapping or intersecting ranges of criteria in the same or different views. We could have chosen to offer AND, OR and NOT operations to composite brushes and add a formula editor to allow controlling the order of operations by bracketing. In contrast, we use composite brushing similar as in the SimVis system [DGH03] by offering AND, OR and SUB operations where the first operand is always the result of the latest composition. This allows a simplified, intuitive, and more iterative workflow compared to working with a formula editor. The user defines the first brush, then (optionally) selects a Boolean operation, adjusts the composition setting (to either AND, OR, or SUB) and then defines the next brush to adjust the current selection. Following brushes and operations will be applied to the result of prior brushes only. Iteratively, every new brush alters the current selection status according to the composition rule in use. The process continues in this way: new brushes and operations are applied to the latest state only. Each new brushing operation provides immediate visual feedback and the user can interactively refine (using AND and SUB) or broaden (using OR) the current selection and steer the information drill down. The user can also resize or move any existing brush in the chain to gain even more flexibility.

Brushing and linking is a powerful feature in understanding how outputs depend on inputs and finding input parameter sets when desired properties of outputs are known. Because we treat all parameters in the same manner, one can brush in views of independent parameters and study how dependent parameters change in other views, or perform the inverse kind of investigation to find suitable inputs for specified results by brushing in the views showing output parameters.

Brushing conventional views is quite straightforward. The user can select histogram bins, rectangular areas in scatterplot views or ranges of a parallel coordinates axes. We have introduced a novel brushing tool in the function graphs view and it will be described in more detail.

3.4.2 Brushing Function Graphs

We suggest two brushing methods to meet the specific requirements of queries on families of function graphs.

A *line brush* is a simple line segment drawn in the function graph view. It selects all function graphs that intersect the line Fig. 3.2 shows an example of selecting several graphs that have high and constant occupancy value indicating malfunctioning sensors. Linking them to the corresponding points in I in the map, we identify those sensors. It is very easy to exclude outliers in a family of function graphs or to isolate curves with desired characteristics with just a few line brushes (brush 2 in Fig. 3.3). Additionally, it is very useful to also provide a polyline brushing opportunity, i.e., a brush in the form of a polyline which selects all function graphs which intersect any of the polyline segments. The line brush, together with the above-mentioned composition functionality, assists the user when he or she is looking for graphs whose approximate shape is known. The logical operation can be defined individually for each line brush which supports very complex queries. An example of composite brushing is provided in Fig. 3.5. A complex combination of line brushes is used to include and remove various graph shapes in the focus set. We have found compositions of line brushes very intuitive and effective in brushing function graphs.

A *rectangular brush* selects all curves which pass through the rectangle. The timebox widget [HS04] is an analogy to rectangular brushing. We have enhanced the original idea by allowing the user to optionally limit the brushing to function graphs that enter and leave the rectangular brush at given edges. Probably the most useful ones are those where the function graph is required to enter and leave at the bottom or on the top edge of the rectangle. These function graphs have a local maximum or minimum inside the rectangle, which is often a criterion in the analysis of time series data. This is especially useful if the display of a family of function graphs is dense and areas of maxima and minima are overlapped by other function graphs. The rectangular brush can be represented as a composition of line brushes.

3.5 Analysis Procedures

The shapes of the graphs depend on the independent variables $\mathbf{x}$ and in practical cases the shapes usually exhibit similarities for slight variations in the variable values, albeit this correlation may be quite indirect. For example, complex physical systems can be considered “black boxes” that return output for an input parameter set, but their exact dependencies on the inputs are unknown. This can also happen if a system is simulated using a computer: the boundary conditions can have so diverse effects on the results that in the analysis of such systems it is more feasible to reconstruct the black box by exploration rather than by trying to deduce its internals from the simulation process. Analysis and exploration of this class of data involves several types of procedures, including discovering trends and tendencies or finding outliers in D. For certain data sets, similar analysis of I can also be of interest. However, in this section we focus on finding patterns and dependencies in the union of I and D.

3.5.1 Black Box Reconstruction

We call the process of understanding the influence of independent variables on dependent function graph variables *black box reconstruction*. To accomplish this, we usually need to have an overview of the entire data set, following the principles of Schneiderman’s Visual Information Seeking Mantra: overview first, zoom and filter, then details-on-demand [Shn96]. We are interested in how function graphs in D change as values of independent variables are changed. We want to fix some input parameters to reduce the focus area and vary others while studying the corresponding output graphs. This is an interactive and iterative data exploration process: brushes are created and moved to areas of interest. When we have built up an overview of the dependencies we want to zoom in on details in both I and D in order to discover more subtle correlations in the data. In case of function graphs it

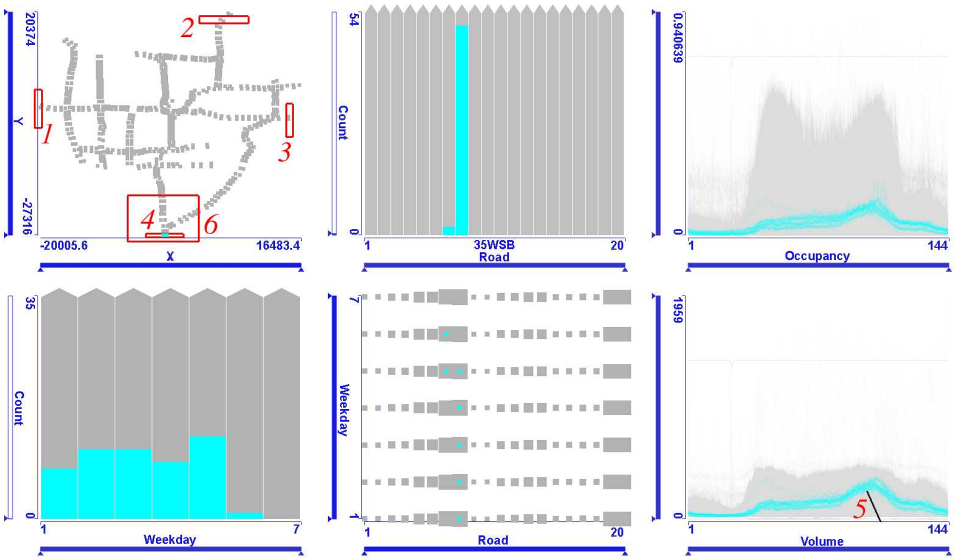


Figure 3.4: Another snapshot of an iterative, interactive visual analysis of the Minneapolis traffic data. The creation order of brushes is indicated by red numbers. First, entries into the freeway system are selected using the union (OR) of four brushes (labeled 1 to 4) in the top left map. Next, low volume evening traffic is excluded from the focus by SUBtracting graphs which intersect brush 5. Using logical AND with brush 6 in the map view we can restrict the investigation to one specific entry. By dragging this brush to other entries we can quickly change the focus to one of the four entries. Thereby a comparison of traffic patterns with respect to the four entries is possible. In this snapshot evening traffic on the south entry is shown. The highlighted points in the lower middle scatterplot and the linked histograms reveal that heavy traffic direction is south-bound from Mondays to Thursdays, but interestingly, shifts to north-bound on Fridays and Saturdays. (road number and direction are displayed on mouse-over).

is especially important to provide context information so that changes in shape are more obvious as various ranges of values are brushed.

An example of black box reconstruction is shown in Fig. 3.4. We are interested in evening traffic characteristics entering the Minneapolis area freeway system. The freeway system has entrances from North, South, East and West. We first brush these entry points in the map view with a logical OR combination of four brushes labeled 1 through 4. Then low traffic volume in the evening is excluded from the focus set using a line brush with SUB operation (5). Finally, we create a sixth, larger brush in the map using the AND operation to restrict the focus set to one of the entries. By dragging this last brush to the other three entries we can quickly change the focus and compare traffic patterns of the four entry points, while still being accurate with respect to brushes 1 to 4. After each interaction step all views are immediately updated in order to support iterative analysis.

3.5.2 Analysis of Families of Function Graphs

In this kind of analysis we (approximately) know the desired or expected shape of the function graphs and our goal is to find combinations of independent variables that produce these shapes. We also want to exclude combinations that produce undesirable or invalid graphs and we want to find out how the deviations from the desired shape depend on the input parameters. This could be considered an inverse investigation to the one in the previous section. However, this kind of analysis requires that the we have an idea of the operation of the black box so that we avoid erroneously identifying dependencies that are mere coincidence in reality.

The procedure requires focus+context views of the graphs where criteria can be defined to select graphs of specific shapes. The desired shapes of graphs can be characterized by brushing. The typical workflow is locating invalid or undesired graphs first, as illustrated in Fig. 3.2. We brush them in the function graph view and find the related values of independent variables, in this case locations of the malfunctioning sensors. We will exclude these items from further analysis.

The desired properties of a function graph can be defined by line brushes, as shown in Fig. 3.3. Here we look for locations of high volume morning traffic on a given road on weekdays. This can be accomplished by selecting the road and weekdays in the scatterplot (brush 1) and then removing low traffic volume function graphs using SUB operation (brush 2). The locations are highlighted in the linked map.

3.5.3 Multidimensional Relations

Another interesting aspect of the analysis is the correlation between various families of function graphs. We want to investigate features of one family of function graphs depending on the properties of a set of function graphs in another family, for example, relationships between traffic volume and occupancy. This analysis within multidimensional time series data requires that graph families are displayed simultaneously and the user can interactively brush specific groups of graphs in one family and study the corresponding ones in the other families. We may also want to narrow the search by specifying filters on the graph's input parameters $\mathbf{x}$. Furthermore, we want to be able to define criteria on various families of graphs.

As an illustrative example let us consider the following query on the traffic data: we look for areas where traffic is strong, but still moving both in the morning and in the afternoon. These are heavily used roads without traffic jams. If cars drive at higher speeds then many cars pass over the sensors but with relatively large gaps. This is indicated by high volume and relatively low occupancy values. In the analysis tool this is expressed as a combination of brushes in both families of function

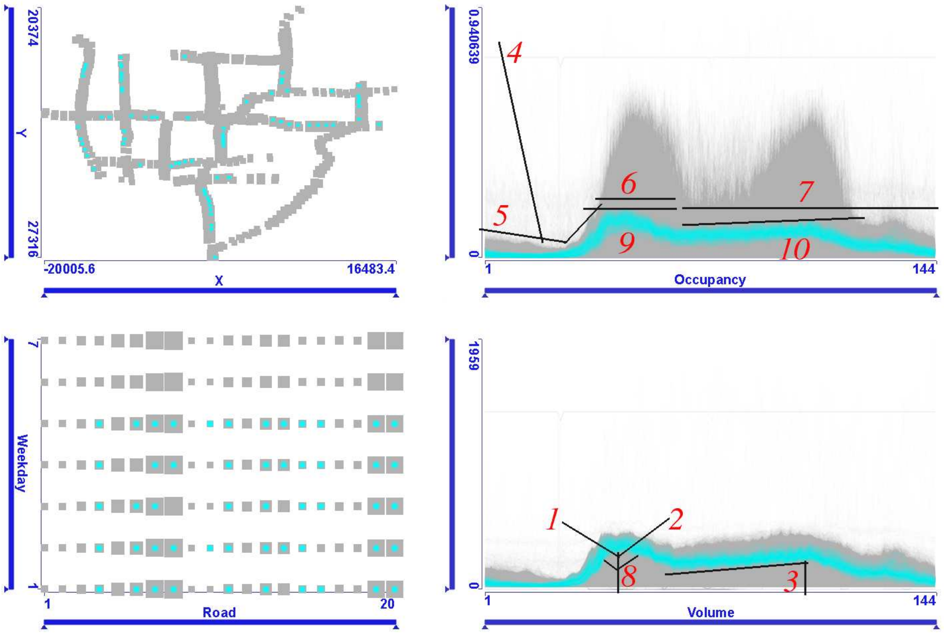


Figure 3.5: We look for roads where traffic is high both in the morning and in the afternoon. First we select heavy morning traffic (1 AND 2), then remove low evening traffic volume (SUB 3). Data from malfunctioning sensors is removed (SUB 4). Next we remove high occupancy graphs by subtracting brushes 5, 6 and 7. The desired graph shapes are further refined by removing graphs which intersect brushes 8, 9 and 10. The combined criteria on the two graphs reveals the roads in question on the map. We can also see in the lower scatterplot view that interestingly, on specific roads, no points are highlighted on Wednesdays. This means traffic on these roads does not follow this pattern.

graphs. There is no direct correlation between the two graphs. The investigation is demonstrated in Fig. 3.5. We brush in the volume and occupancy function graph views and study the linked map. First large morning traffic volume is brushed using two line brushes (1 AND 2). Then we remove function graphs with low traffic volume in the evening (SUB 3). Now we narrow down the search in the occupancy function graph view. Data from malfunctioning sensors is excluded (SUB 4). Then we limit the occupancy by removing function graphs which intersect brushes 5, 6 and 7. Now we have a view of the areas where traffic is strong but moving in the morning and in the evening. We can further refine the desired traffic volume shape in the morning by removing the function graphs which intersect the three line brushes (SUB 8). Finally we limit the occupancy to even lower ranges by removing function graphs that intersect brushes 9 and 10.

3.5.4 Hypothesis Generations via Visual Analysis

There is a particularly strong need in engineering applications to perform automatic optimization of designs using several simulation iterations with suitably varied boundary conditions. The automatic optimization process must have an approximate model of the simulation so to know how boundary conditions should be adjusted in search for an optimum. Visual analysis can be used to create hypotheses and rules that the automatic optimization can use in its simplified model and also to find out if the optimization misses some families of function graphs while searching for an optimum.

Gaining insight into the current design and setting up hypotheses about its operation via visual analysis has a very important additional advantage over pure numeric optimization. When designing a new component, engineers almost never start from scratch, but the new design evolves from an old one. Because of this iterative nature of design in engineering, the insight gained from analysis of previous designs can be useful in improving future ones. This also means that simulation models of new design are not radically different from that of the old ones and their results are comparable to some extent. By analyzing the relationships between the two, tendencies can be found and extrapolated to improve future design.

3.6 Engineering Application: Fuel Injection System Simulation

The science of visual analytics is also very applicable in engineering applications. Simulation and measurement data sets are vast, optimization goals are often conflicting, the tendencies and dependencies in the data can be indirect and engineers need to find the best compromise. Designers must make defensible and responsible decisions because design mistakes can have very expensive consequences if shortcomings are discovered during production. Time-to-market for new designs will be short, so they must work under time pressure and communicate their findings to collaborating teams. In this section we demonstrate the applicability of our approach to the analysis of Diesel injection system simulation data.

3.6.1 Diesel Common Rail Injection Systems

There are many (often conflicting) goals of Diesel engine design including high power, good fuel efficiency, meeting emission regulations, low noise levels and driveability (steady and reliable torque at various engine speeds). The fuel injection system is the key Diesel engine component to achieve those goals. The following properties are considered important in the fuel injection procedure:

- high injection pressure for good atomization and combustion,

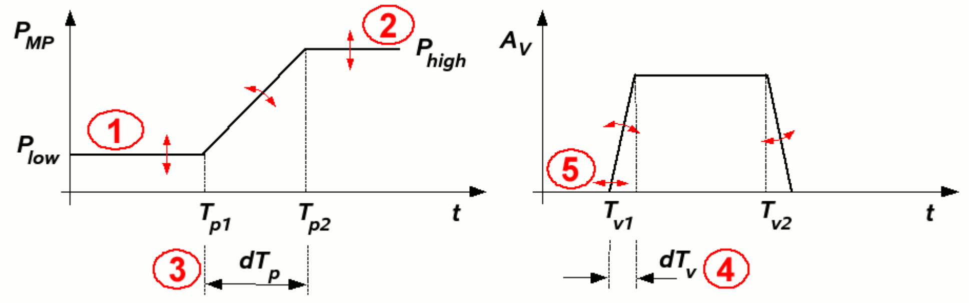


Figure 3.6: Control parameters for the simulation. Left: inlet pressure characteristics are pressure levels P_{low}, P_{high} and the time interval of pressure increase dT_p. Right: injector valve opening/closing properties are the time T_{v1} when the valve starts to open and the opening/closing time interval dT_v.

- flexible timing of the injection,
- short pre-injection before the main burst to reduce combustion noise,
- accurate control of injected fuel quantity,
- ability to inject small amounts of fuel to achieve economical operation and good emission properties.

A specific type of injection systems, the common rail injection system can be controlled in a very flexible way, thus it is seen as a very popular option by many manufacturers. Common rail injection system have several attractive characteristics: injection pressure and quantity can be controlled with a high degree of flexibility, multiple fuel injections are possible within one injection cycle and the time and duration of the injections can be controlled precisely by the engine control unit based on the engine speed and load. These properties are key factors in meeting current and future very stringent emission regulations. In our case study we use the simulation results of a conventional series common rail Diesel fuel injection system [MJJ$^+$05, BH97].

3.6.2 Fuel Injection Simulation

The fuel injection simulation data is from AVL-List GmbH. The simulation is based on the theory of 1D fluid dynamics and 2D vibrations of multi-body systems. In 1D fluid dynamics pressure is uniform on pipe slices perpendicular to the axis. The simulation was run for a number of *cases*. Each case is represented by its own set of simulation control parameter values. The software can automatically loop the parameters over a specified range and run a simulation variant for each resulting case. The engineers study the resulting output data and attempt to find ideal simulation

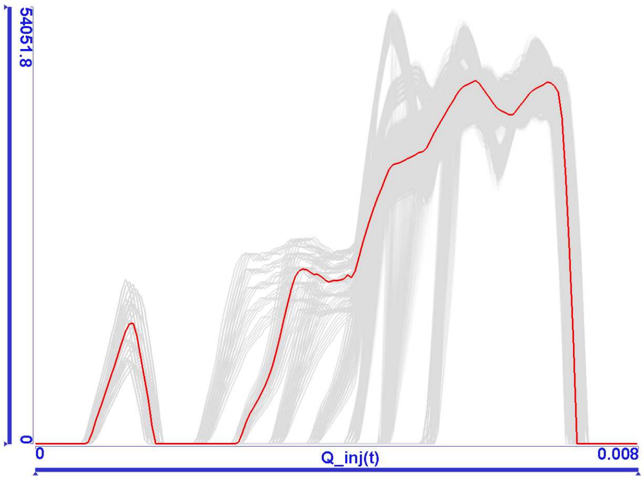


Figure 3.7: A typical shape of the fuel injection rate graph is highlighted in red. There is a short pilot injection first, followed by the main injection. The graphs resulting from other combinations of control parameters are gray.

input parameters for various engine operating situations. This data set follows the model introduced in Section 3.3: I consists of the simulation control parameters and D consists of the simulation output.

Simulation Control Parameters

The injection shape depends mainly on three factors: the nozzle geometry, injection pressure and timings for valve opening and closing procedures. The influence of control parameters related to nozzle geometry has already been investigated in our previous work [MJJ$^+$05]. Once a (nearly) optimal nozzle geometry is found and it goes into production it cannot be changed very often because this would be too expensive. The focus of fuel injection optimization afterwards is usually on varying the remaining two factors.

Therefore, the independent variables in our current investigation are related to injection pressure and injector valve timings only. The injection pressure is controlled by the injection pressure modulation device which is positioned between the rail and injector. In our investigations this device is not modeled in detail, but we take the modulated pressure as input. The characteristics of the pressure on the injector's inlet are described by 3 parameters (Fig. 3.6). The injector valve actuator that controls the injection timing is described by its opening/closing times and velocities. Although this is a simplified model it allows the simulation of various types of valve actuators including the popular solenoid type or the more recent piezoelectric ones. Consequently, we have I of 5 dimensions. In parenthesis we indicate the number of variations for each input parameter.

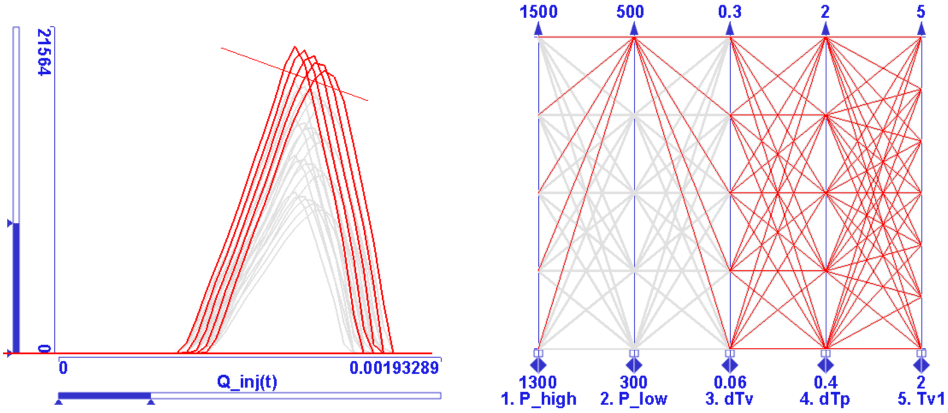


Figure 3.8: Pilot injections with high amount of fuel are brushed with a line brush. As seen in the parallel coordinates view of the independent variables, these all correspond to high P_{low} values.

1. P_{low}: low pressure on the injector inlet (5),
2. P_{high}: high pressure on the injector inlet (5),
3. dT_p: time interval of modulated pressure increase on the injector's inlet (5),
4. dT_v: time interval of the injector valve opening and closing (5),
5. T_{v1}: injector valve opening time (7).

The total number of variations of the independent variables is $5^4 \times 7$, which means 4375 different sets of simulation boundary conditions.

Simulation Output

For each combination of the independent variables the simulator computes three sets of time-dependent results: $Q_{inj}(t)$: injection rate, $P_{inj}(t)$: injection pressure and $A_n(t)$: needle lift. In other words, there are three families of function graphs in this data set. Furthermore, the following not time-independent results are computed: Q_p: amount of fuel injected during pilot injection, Q_m: amount of fuel injected during main injection, Q_{vo}: amount of fuel flowing back to the fuel tank, V_{open}: needle opening velocity, V_{close}: needle closing velocity, L_p: spray penetration depth, P_{ia}: average injection power.

3.6.3 Analysis of Fuel Injection Simulation Data

The typical shape of the injection curve is shown in Fig. 3.7. There are usually one or two small peaks called *pilot injection* in the first quarter of the injection

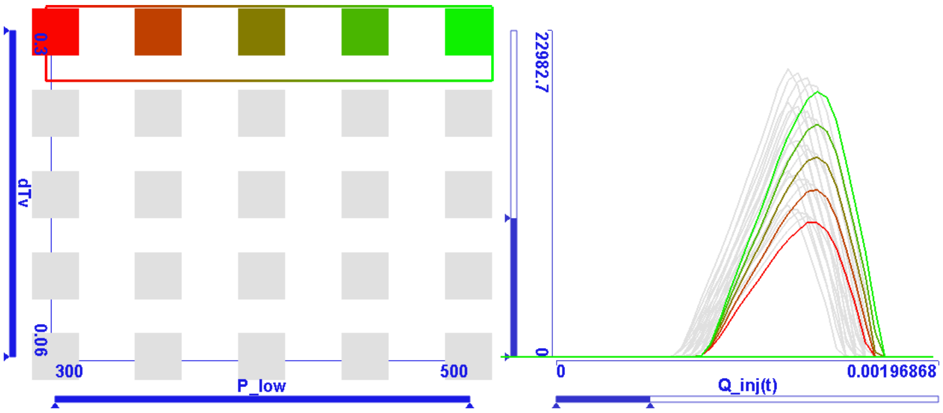


Figure 3.9: High P_{low} simulation parameters are brushed in the scatterplot diagram using a gradient brush. The correlation of P_{low} and the amount of pilot injection is revealed by the color gradient.

procedure in order to reduce combustion noise and NO_X emission in combination with the main injection. As Fig. 3.7 shows, there was one pilot injection in our case. Arbitrary shaped injection rate graphs cannot be produced in simulation because of the physical requirements of the combustion in the engine. The engine will not run properly if the injection rate function graph does not follow shapes similar to the ones in Fig. 3.7. The goal is to find combinations of simulation parameters that control the volume of the pilot injection and produce the desired shape of the main injection.

Analysis of Pilot Injection

We investigate how the amount of fuel and the timing of the pilot injection depend on the control parameters. We zoom in on the first peak of the $Q_{inj}(t)$ function graph and first select function graphs with high peaks using a line brush (Fig. 3.8). The corresponding items are highlighted in the parallel coordinate view of the input parameters It is evident strong pilot injections are linked with high P_{low} values. We suspect that there is a correlation between P_{low} and the amount of fuel injected during pilot injection. To support this hypothesis we can brush all P_{low} inputs with a gradient brush in a scatterplot view. The color gradient from red to green (Fig. 3.9) establishes visual links between the brushed items in the scatterplot view of the injection control parameters and the graph view of the injection rate.

Next, we try to find the parameters that determine the timing of the pilot injection. We brush the peaks of the graph with a line brush and examine the parallel coordinate view of the control parameters. We conclude that time of pilot injection's peak depends on dT_v. This hypothesis can be counter-checked in an interactive way. A brush is panned over the scatterplot diagram of P_{low} and dT_v and the highlighted

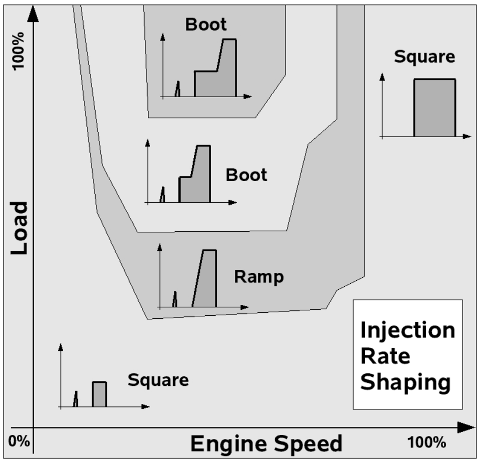


Figure 3.10: Ideal shape of the main injection for various engine operating points defined by engine speed and load. The shape can be classified into three types: square (injection rate steeply increases to a maximum level), ramp (the slope is more gentle) and boot (following a nearly horizontal segment injection rate rapidly increases to maximum level at the moment of ignition). This classification is somewhat arbitrary, since the shape changes from square to boot in a continuous manner as the control parameters vary.

injection rate function graphs are studied. We find that large dT_v causes the pilot injection to start later and also to have slightly lower volume. The fully covered axes of the three other control parameters in the parallel coordinates suggest that the pilot injection's shape does not depend on them.

Analysis of Main Injection

The optimal shape of the main injection is different for each particular engine operating point (Fig. 3.10). The engine control unit (ECU) measures engine speed and load to determine the current operating point. For each operating point the ECU contains a lookup table of injection control parameters used to control the injection system. The goal of the engineers is to find suitable sets of control parameters for some characteristic points in the diagram and understand how various properties of the injection rate graph can be controlled. In the following we investigate how suitable control parameters can be found for specific main injection shapes. For each case we also demonstrate some additional dependencies and tendencies in the data set.

Square Square main injection shape is desirable when load is very low or when the engine speed and load are both high. We used a combination of three line

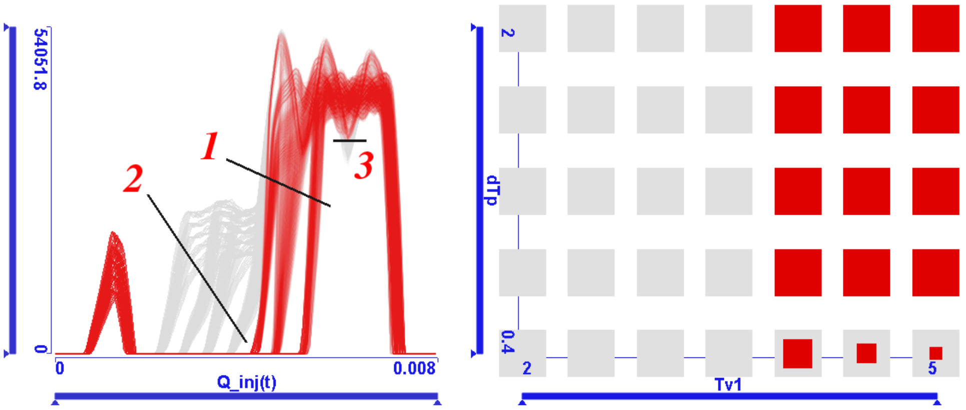


Figure 3.11: Ranges of control parameters that produce square shaped injections. First the user attempted to select square shaped graphs with brush 1. This brush selects several not square shaped graphs, too. These are removed by SUBtracting graphs that intersect brush 2. Brush 3 removes graphs that drop under a certain threshold in the main injection part. This property is a result of vibrations in the fuel line, which are to be avoided.

brushes to select square shaped injection rate graphs (Fig. 3.11). The aim of the rightmost one is to exclude undesired shock wave reflections.

In a similar process to the one used when investigating the pilot injection we discover that T_{v1} is high for the brushed graphs. That means the injector valve opens late when the pressure on its inlet is already very high. This leads to a square shape of the injection rate. As the rightmost brush is created we also observe in the linked scatterplot diagram that most of the items that have low dT_p (time interval of modulated pressure increase) are removed from the focus. This means dT_p must not be very low in order to avoid shock wave reflections.

We also study the desired needle opening and closing velocities and the correlations between the injection rate and the needle lift graphs for this case. In order to do so the high T_{v1} and dT_p region of the scatterplot diagram is brushed in Fig. 3.12. The highlighted points in the V_{close}/V_{open} scatterplot diagram show that fairly fast needle opening and closing is required for square shaped injections. The needle lift graph (bottom right) is also linked and the color gradient of the brush shows a strong correlation between the needle lift and the injection rate graphs.

Ramp Ramp-shaped main injection is desirable when the engine speed and load are in mid-range. In the previous case we have found correlation between T_{v1} and the shape of the injection rate graph. We also know that the time interval of the modulated pressure increase on injector's inlet should be fairly high to avoid reflections. Based on this we already start the investigation by brushing cases when the

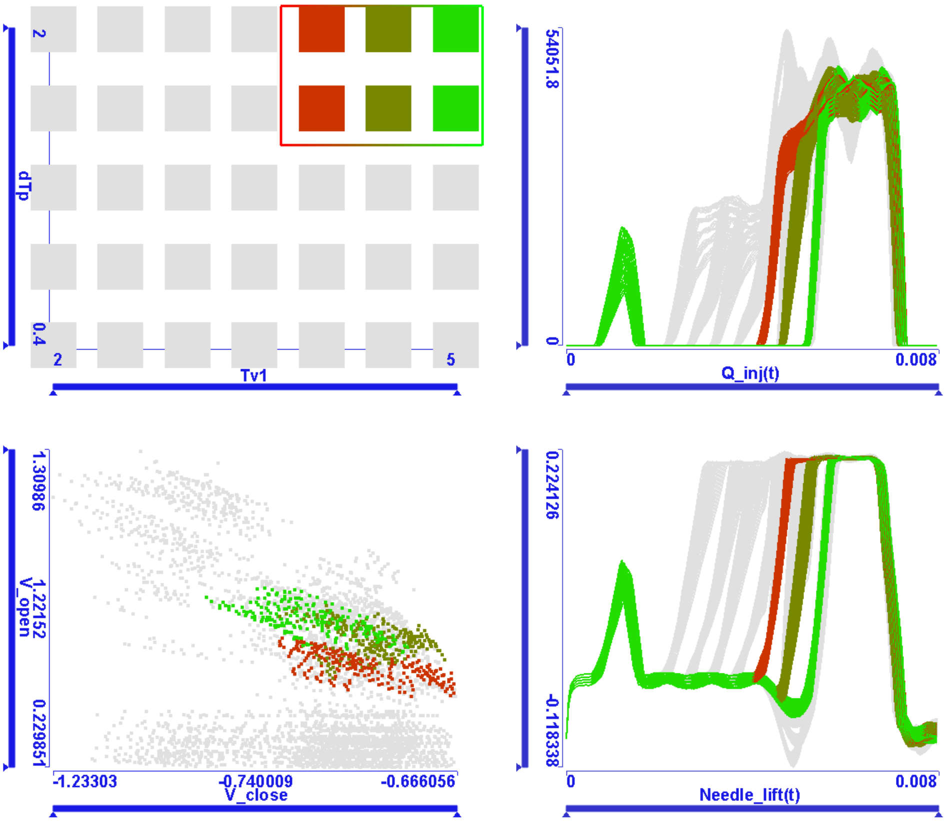


Figure 3.12: Required needle movement characteristics for square shaped main injections. Top left: control parameters that produce square shaped injections are brushed. Top right: red to green gradient shows that earlier valve opening times cause deviation from the ideal square shape. Bottom left: needle opening and closing velocities must be fairly high for this shape. Bottom right: the shape of the needle lift graph is closely correlated to that of the injection rate graph.

injector valve starts opening a little later and we exclude low dT_p ranges as shown in the top left scatterplot diagram in Fig. 3.13.

The histogram of P_{high} is also brushed and the intersection of the two brushes is studied in the injection rate and injection pressure graphs. We observe that the corresponding graphs of injection rate and pressure are similar in shape but differ in their maxima as P_{high} is varied. This is shown in Fig. 3.13

Boot Boot-shaped main injection is desirable for engine operating points of mid-range engine speeds and high load. From our previous experience we assume that the injector valve has to be opened very early to achieve this shape. This assumption is easily verified by brushing the corresponding region in the scatterplot diagram of T_{v1} (injector valve opening time) and dT_p as shown in Fig. 3.14.

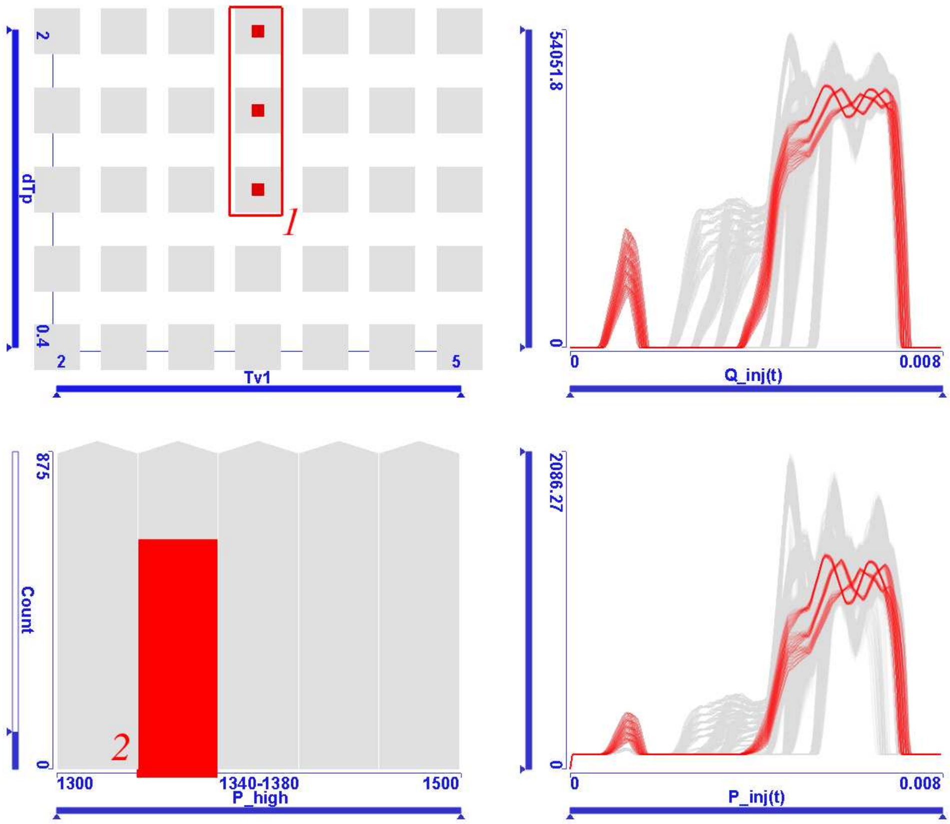


Figure 3.13: Top left: brush 1 selects control parameters for ramp shaped main injections. Bottom left: brush 2 is probed in the histogram of the high pressure on injector inlet using the AND operation. Top right: injection rate graphs of the brushed items. Bottom right: injection pressure graphs of the brushed items. By dragging the brush in the histogram view and studying the linked function graph view we observe similar shaped graphs with different maxima.

Now we investigate how desired amounts of fuel in the main injection and various injection penetration levels are achieved. The scatterplot diagram of these simulation outputs is brushed and the brushed items are observed in the linked views in Fig. 3.15. We observe that the brushed injection rate graphs are all boot shaped. In the parallel coordinates view it is obvious that boot shaped injection does not require fast injection rate increase, but fast needle closing and injection rate decrease are still preferred. We also discover that for deep fuel spray penetration and high injection powers (brushed in green in Fig. 3.15) fast needle closing velocities are required. The injected fuel mass (Q_m) and the amount of fuel returned to the fuel tank (Q_{vo}) are both fairly high. This matches our expectations, since we see in the parallel coordinate view that the P_{high} input was also quite high in these cases.

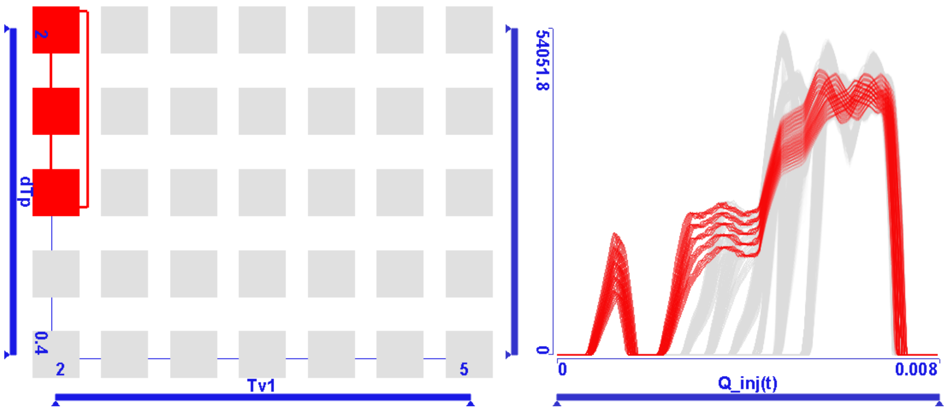


Figure 3.14: If the injector valve is opened very early than the injection rate quickly increases to the "boot" level. It reaches its maximum when the mixture is ignited in the combustion chamber.

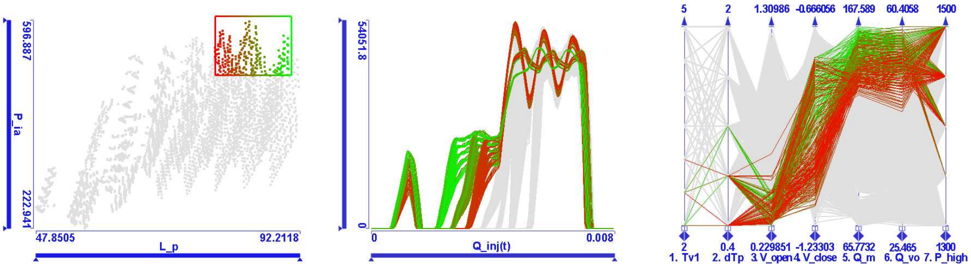


Figure 3.15: We investigate the conditions when fuel is injected deep in the combustion chamber and with high power. The corresponding items are brushed in the scatterplot diagram. The linked graphs show that this requires boot shaped main injections. The desired needle opening and closing velocities are highlighted in the parallel coordinate view.

Insight Gained from Analysis

In this example we have gained valuable insight into the fuel injection simulation data set and thereby into the fuel injection process itself, too.

We found that the amount of injected fuel in the main injection stage can be controlled by adjusting P_{high}. The amount of pilot injection is controlled mostly by P_{low}, but the time interval of the injector valve's opening also has some influence on it. We observed that choosing the inlet pressure and the time when the injector valve opens is the key to achieving the desired injection shapes for various engine operating conditions. When pressure increases too fast on the injector's inlet then the resulting wave can be reflected into the fuel line which impairs our control over the injection's shape.

By studying the needle lift function graph and the related V_{open} and V_{close} simulation outputs we are able define the desired needle characteristics for specific injection shapes and we also saw how tightly the injection rate and the needle lift are correlated.

Additional images and supporting videos are available at `http://www.vrvis.at/scivis/graphs-analysis/`.

3.7 Conclusion

Analysis of relationships between families of function graphs is a common task in many application domains. A novel combination of established visualization techniques, linked views and advanced brushing represents a valuable tool for interactive visual exploration and analysis of data sets that include multiple families of function graphs. All parameters are treated the same, providing support for iterative exploration of the data space. Multiple, linked views enable simultaneous viewing of independent and dependent parameters with immediate feedback. One can brush in the views showing dependent parameters to find the corresponding values of independent parameters and vice-versa.

Brushing proved to be the most effective of all the interaction techniques provided. The color gradient is used along the brush and in the linked views to establish visual identification of the correlated data items. The composite brushing (AND, OR, and SUB operations) allows us to refine the search and detect or extract patterns from the application domain. The line brush technique works very well on families of function graphs. It is intuitive, easy to use and very effective. Fig. 3.5 shows how a composition of nearly a dozen line brushes is used to identify a pattern in a traffic volume graph of a road.

The process of the composite brush construction captures the essence of visual analytics procedures: they are interactive and iterative. The initial brush provides the initial data selection in the current view. That selection is immediately displayed in the linked views where it can be analyzed from different perspectives to formulate a hypothesis. That hypothesis is then tested using a a new brush and so on until a new, possibly unexpected pattern is found. Fig. 3.2 shows a discovery of a pattern in D (constant high occupancy) that indicates a pattern in I (malfunctioning sensor). Such discoveries would be more difficult or even impossible without interactive composite brushes.

Future work will proceed in three directions. First, we will expand the data model to include input time series and time-dependent input parameters as well as first and second derivatives of times series. We will explore what impact this has on the required analysis procedures and try to find tools to support the new tasks. Finally, we will explore the use of large-scale displays and usability issues related to manageability and arrangement of large number of views.

Acknowledgments

The authors would like to thank the reviewers for their valuable comments and suggestions. The fuel injection simulation data is courtesy of AVL-List GmbH (`http://www.avl.com/`). Parts of this work was done through applied and basic research at the VRVis Research Center funded by an Austrian governmental research program called K plus (`http://www.kplus.at/`).

4

Interactive Visual Steering – Rapid Visual Prototyping of a Common Rail Injection System

Published as:

Krešimir Matković, Denis Gračanin, Mario Jelović, and Helwig Hauser:
Interactive Visual Steering - Rapid Visual Prototyping of a Common Rail Injection System
IEEE Transactions on Visualization and Computer Graphics, 14(6):1699-1706, 2008.

Interactive Visual Steering – Rapid Visual Prototyping of a Common Rail Injection System

KREŠIMIR MATKOVIĆ, DENIS GRAČANIN,
MARIO JELOVIĆ, AND HELWIG HAUSER

Abstract

Interactive steering with visualization has been a common goal of the visualization research community for twenty years, but it is rarely ever realized in practice. In this paper we describe a successful realization of a tightly coupled steering loop, integrating new simulation technology and interactive visual analysis in a prototyping environment for automotive industry system design. Due to increasing pressure on car manufacturers to meet new emission regulations, to improve efficiency, and to reduce noise, both simulation and visualization are pushed to their limits. Automotive system components, such as the powertrain system or the injection system, have an increasing number of parameters, and new design approaches are required. It is no longer possible to optimize such a system solely based on experience or forward optimization. By coupling interactive visualization with the simulation back-end (computational steering), it is now possible to quickly prototype a new system, starting from a non-optimized initial prototype and the corresponding simulation model. The prototyping continues through the refinement of the simulation model, of the simulation parameters and through trial-and-error attempts to an optimized solution. The ability to early see the first results from a multidimensional simulation space — thousands of simulations are run for a multidimensional variety of input parameters — and to quickly go back into the simulation and request more runs in particular parameter regions of interest significantly improves the prototyping process and provides a deeper understanding of the system behavior. The excellent results which we achieved for the common rail injection system strongly suggest that our approach has a great potential of being generalized to other, similar scenarios.

4.1 Introduction and Related Work

Increasing complexity and a large number of control parameters make the design and understanding of complex systems (such as automotive engines) impossible without simulations. Strict emission rules and regulations force car manufacturers to design improved engines, in very short time [BDHK05]. To meet those requirements, car manufacturers use simulations as a cost-efficient, and often the only possible way to design systems with desired characteristics. They use many types of simulation, including Computational Fluid Dynamics (CFD).

In this paper, we describe results from a recent project where the need for interactive steering emerged. We used interactive visual analysis to support an interactive design process. In contrast to the usual, very time consuming 3D CFD simulation, 1D CFD that is alternatively used in injection system simulation can be computed very fast. It is possible to run tens of thousands of simulations for a large set of parameters. However, the brute force approach, where a simulation runs for all possible parameter combinations, is often not feasible. Instead, interactive simulation steering helped us to insure a reasonably short design time. A pure numerical optimization is sometimes too complex and a user often gets only the final results, without proper insight.

The background of this work was the task to design an injection system. We developed a steering framework to support this task. Our interdisciplinary project setup provided us with valuable feedback during the design process in terms of the usefulness of the proposed approach and suggested improvements. We started from a simple model and gradually made it more and more complex.

One of the important parts of the automotive engine system is the injection system. The piezoelectric stack actuator is the main component of the injection system model [GC97]. When an input voltage is applied, the electric field across the ceramic layers of the stack actuator induces a mechanical strain. The strain results in an elongation of the stack that exhibits the rate-independent hysteresis between the electric voltage (force) and mechanical strain (displacement).

We identified tasks that can be generalized to other problems and illustrated how we designed and tuned the model. The model complexity did not allow us to run all possible simulations at the beginning and to analyze the results. Such an approach would also result in numerous unnecessary simulation runs and would waste time and computational resources. Furthermore, we did not have a complete model at the beginning. It was gradually built as we gained insight during the design process. Our approach, the use of interactive visualization and coordinated multiple views as a steering mechanism for simulation, proved to be very efficient. In this paper we show how the tight integration of visualization and simulation can significantly improve an engineer's workflow as compared to decoupled systems. The excellent results which we achieved for the common rail injection system and the very positive feedback from domain experts strongly suggest that our approach has a great potential and can be generalized to other, similar scenarios.

The decoupling of simulation and analysis can present significant obstacles and make it very difficult to effectively manage large amounts of simulation data [PJB97]. We should be able to interactively steer computations, change simulation parameters or representation and immediately see the simulation results. Computational steering and interactive visualization emerged in 1980s and

1990s as some of the most useful visualization paradigms for computational science [CRS96].

Many simulations are computationally intensive and may require interpolation for sensitivity analysis and optimization. Sensitivity analysis and optimization require the domain expert to interpolate the observed simulation data. This interpolating function is a metamodel of the underlying simulation model which is treated as a black box. An example is the Kriging interpolator representing a global metamodel that covers the whole experimental area [vBK04]. However, we can also iteratively refine the simulation model. That way we can refine both the simulation parameter values and the simulation model.

The simulation output data is often visualized using scientific visualization methods [HEvLS03]. Using visualization and spatial tools to understand complex systems is not a new idea. James C. Maxwell, one of the most important physicists of the nineteenth century [Wes99], often used visual-spatial thinking. An excellent example of his approach is a construction of 3D clay model of a surface based on Willard Gibbs' work. We've come a long way since those early beginnings. However, thinking about the science is still at the core of scientific visualization [Joh04]. The most important scientific visualization research problems include perceptual issues, human-computer interactions, global/local visualization, feature detection and visual abstraction, to name a few [Joh04].

Computational steering integrates modeling, computation, data analysis, visualization, and data input components of a simulation [PJB97]. However, integrating a simulation within a computational steering can be a very difficult problem. We need to address four facets of the problem [JPH$^+$99]: control structures, data distribution, data presentation, and user interfaces. Since computational steering is a highly interactive process, the user interface is a critical component of a computational steering environment [MvWvL99]. Kreylos et al. [KTH$^+$02] describe a system for real-time interactive visualization of computational fluid dynamics (CFD) simulations that allows a user to place and manipulate visualization primitives during an ongoing simulation process. Vetter and Reed [VR00] described performance monitoring, control, and interactive steering of computational grids. Wenisch et al. [WvTB$^+$07] demonstrated computational steering of CFD simulations on distributed computers.

There is extensive literature about user interface and visualization of simulation data from an engineering perspective. Laramee et al. [LGD$^+$05] use different flow visualization methods to show various aspects of the simulation data to support insight and visual analysis of the coolant flow through the cooling jacket of a car engine. Konyha et al. [KMH03] use 3D icons to analyze simulation data of chain and belt drives. Matković et al. [MJJ$^+$05] describe a method for the analysis of a fuel injection system that provides a highly abstract view of the injection system.

Using multiple, interactively linked views of the same data set allows the user to productively combine the information gathered from the different views [Hen98]. Doleisch et al. [DMG$^+$05] use multiple linked views for analysis of CFD simulation data. More advanced multiple view visualization systems can be freely configured [NS00] and provide flexible coordination of views [Wea04]. As the number of linked views and the amount of coordination increases, it may be necessary to visualize the visualization's structure and operation [Wea05].

4.2 Computational/Interactive/Simulation Steering

We used our previously developed coordinated multiple views visualization tool ComVis [MJJ$^+$05] and extended its functionality to interface it with the simulation tool HYDSIM which is a part of the AVL Workspace [AVL15]. In this way a steering framework has been established and used in the project.

The initial design goal for the visualization tool ComVis was rapid prototyping of new visualization techniques within the scientific context. As a consequence the tool was designed to be flexible in a way that it is easy to add new views and support new data types. The tool is intuitive to use and supports advanced interactions (multiple, iterative brushing). As a result, the tool is easy to use for domain experts from different domains (medical, engineering, etc.), and can use/read generally used data formats to provide access to existing data.

The simulation tool, HYDSIM, is a modular program for the dynamic analysis of hydraulic and hydro-mechanical systems. It is based on the theory of fluid dynamics (1D) and vibration of multi-body systems (2D). The user defines a model using 2D graph-like structures with icons and connecting elements. The defined model provides a general representation of the system topology. For each element (represented by an icon) the user can specify properties for the particular case. Once the user completes a definition of the model, the simulation provides output parameters values. In a typical workflow, a domain expert analyzes these results and, if necessary, modifies the simulation model and repeats the simulation until the desired results are achieved. Earlier we pursued an alternative approach to compute a very large set of simulations runs at once (offline) and analyze the results afterwards [KMG$^+$06, MJJ$^+$05]. Although this was a significant improvement compared to the traditional way, we still had to specify all combinations of input parameters in advance.

Our new framework makes it possible to define new simulations using the visualization tool. The visualization tool is used for the analysis and steering of the simulation. That makes it easy for the domain expert to generate new simulations and to refine or to filter the simulation dataset. Each simulation has a set of control (input) parameters and a set of output parameters that are computed for a given input. The main idea is to run many simulations with different control parameter settings (they are defined by lower limit, upper limit, and step size), and to use multiple, coordinated views to understand the model and to support the expert in injection system design.

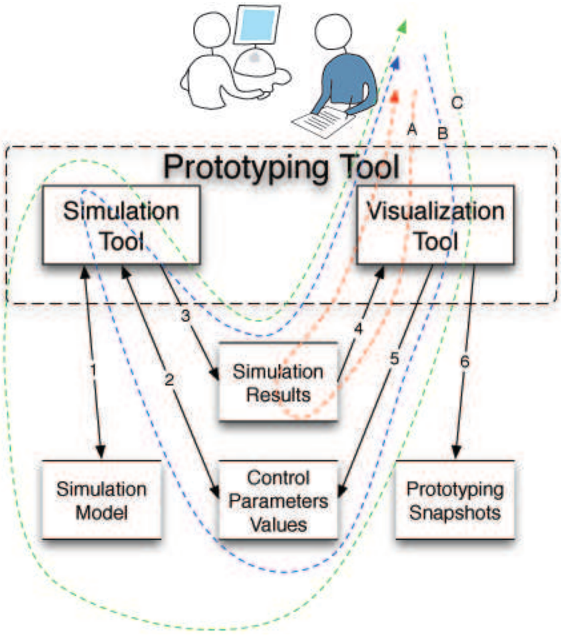


Figure 4.1: An iterative approach to prototyping. A combination of the simulation and visualization tools and related data allows us to design at different levels of abstraction. We distinguish three levels of the interactive steering process depicted with loops A, B, and C. The first loop, loop A, is based on the available simulation results. We explore them, get insight and store results (snapshots). If this is not sufficient, new simulation results can be generated (loop B). The simulation model is still not changed, only parameters are being refined. Finally, it is also possible to change the simulation model (loop C).

We provide four basic operations: refining or coarsening some control parameters (changing the step); narrowing down the control parameter interval (changing boundaries); adding new control parameters; and removing some existing control parameters. If we represent data in tabular form, the basic operations correspond to adding and removing rows (refinement and filtering parameters) or adding or removing columns (adding or removing parameters).

The domain expert estimates the coarse boundaries of the parameters, runs a sufficient number of simulations and sees what parameter values make sense and what values are not allowed based on the output values. In the case of fuel injection systems, the injected fuel mass was one of the output parameters often used to identify parameter values that are not allowed. If there is not enough injected fuel or if there is too much injected fuel, the engine will not run properly.

We use an iterative approach (Figure 4.1). The domain expert uses the simulation tool to create the initial simulation model, specify the initial control parameter values and produce simulation results (Figure 4.1, steps 1, 2, and 3). Only a part of the injection system is modeled in detail while the rest is replaced with modelled "ideal" values. These ideal values became the target when we refined the model. The goal is to create a simulation model and to determine the control parameter values that produce the simulation results that are as close as possible to the idealized result. We repeat the process at three different levels. The user first designs a very simple simulation model and sets the parameters. The user then extends the simulation model and provides the parameters for a more complex model that represents the second level. Finally, the user defines the complete simulation model that makes it possible to go back and forth between different levels (as it is often the case during prototyping) and to change already tuned parameters.

The domain expert carries out the interactive steering process at three different levels. The first level of iterative prototyping focuses on the already generated simulation results. The expert uses the visualization tool to investigate the simulation results and, by extensive use of brushing and linking, can get insight and create first reports (snapshots) about the current prototype results (Figure 4.1, steps 4 and 6).

If the current simulation results are not sufficient, the expert can proceed to the second level. The second level of iterative prototyping involves refining the control parameter values (Figure 4.1, step 5), generating new simulation results using the current simulation model and then returning to the first level of prototyping. The expert can do it in an interactive way and request new simulation results from the simulation tool. As new simulation results are computed, the data in the visualization tool is automatically updated. During this process visual analysis can proceed and benefit from better data resolution.

Based on the insight from the data, the expert may decide to refine the simulation model. At the third level of iterative prototyping the expert uses the simulation tool to update the simulation model and then returns to the second level of prototyping. As this is a larger step which makes it necessary to change internal data representation, the expert has to wait until initial setup is completed and the first results for new model are ready. This can take a few minutes. Once the first set of simulation results is computed and the visualization tool updates the internal structure, the whole process becomes interactive again. The simulation results are uploaded on the fly as they are computed.

In our implementation we always define the model using the HYDSIM tool. The HYDSIM creates simulation definition files and runs the simulations. As simulations are computed, output files are created, one directory for each simulation run. Our visualization tool, ComVis, reads the first simulation results, builds the internal data model, and visual analysis starts. The visualization tool checks for new output files and loads them when they are available. ComVis offers a possibility to specify new simulation parameters, as well. If the user requests new simulations from the visualization, ComVis creates HYDSIM input files and starts HYDSIM. HYDSIM generates new output files which are then automatically loaded into the visualization tool. Model changes are done in HYDSIM, and in these cases ComVis has to

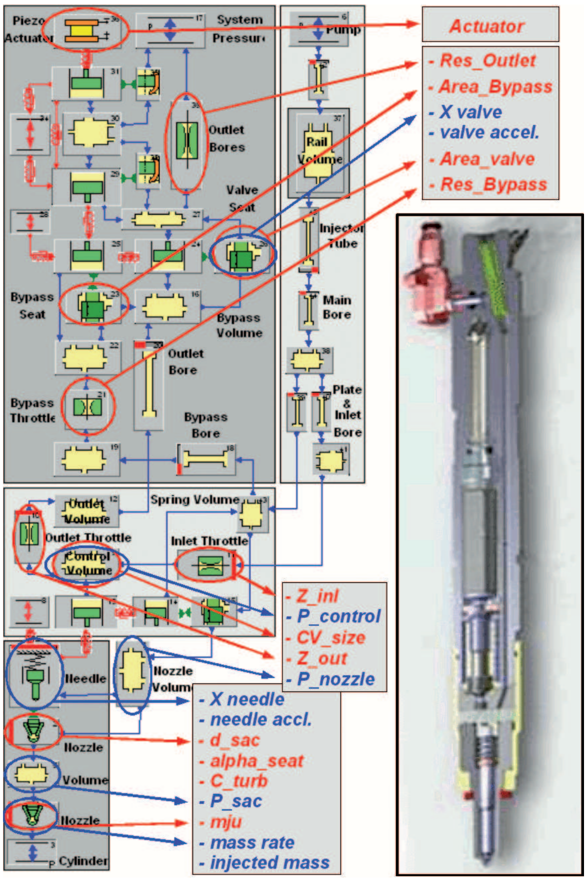


Figure 4.2: The final injection simulation model and the four main blocks. The blocks represent logical grouping that has no influence on the model topology. An actual injector, used for each cylinder in a car engine, is shown on the right. The control parameters are depicted in red, and the output parameters in blue. The exact description of the parameters is too long for this caption and can be found in the main text, instead.

recalculate internal data structures which usually takes a while (minutes). Once the new model is created, first simulations are computed, the internal data structures needed for visualization are created, and the process continues in a usual way.

4.3 Rapid Visual Prototyping and the Design of a Common Rail Injection System

We used the developed prototyping tool to design a common rail injection system. We selected this task for two reasons, i.e., the availability of fast simulation algorithms and the importance of injection in an overall Diesel engine efficiency and emission characteristic.

There are many (often conflicting) goals of a Diesel engine design, including high power, good fuel efficiency, meeting emission regulations, low noise levels, and drivability [Mah02]. The fuel injection system is the key Diesel engine component to achieve the goals. The common rail injection system has several attractive characteristics: injection pressure and quantity can be controlled with a high degree of flexibility, multiple fuel injections are possible within one injection cycle and the time and duration of the injections can be controlled precisely by the engine control unit based on the engine speed and load. These characteristics are key factors in meeting current and future (very demanding) emission regulations.

The common rail injection system consists of two parts, one hydro-mechanical and one electronic. The hydro-mechanical part determines the simulation model, while the electronic part determines the actuator control parameters. Our goal is to design both parts by iteratively adjusting the simulation model and the control parameter values.

We start by providing the end result of the design process (Figure 4.2). The reason for that is to provide the context and the basic expertise for the system design, something that experts already have. The main assembly components of the injector are the piezo actuator with the hydraulic amplifier (*Block III* from figure 4.2), the control valve with a certain control volume (*Block II*) and the nozzle (*Block I*). The piezo actuator governs the motion of the control valve. *Block IV* is the fuel supply from the common rail, which is not analyzed in this work.

Once the simulation model is created, the expert has to set up the control parameters (the parameters listed in red in Figure 4.2). For each set of the control parameters the output parameters are computed (the parameters listed in blue in Figure 4.2). All of the control parameters in our case are scalar values, and all of the output parameters are time series data. An additional control parameter is the actuator (the topmost element in the Figure 4.2) behavior. We model the actuator curve depicted in Figure 4.3 using a set of scalars, determining the start and duration of the pilot and main injection, their maximum amplitudes, and opening and closing times.

The model has 11 control parameters and setting them is the main task of the injection system design. The actuator curve parameters are also set, but they vary, based on the crankshaft load and speed, during the actual engine operation. The electronic control unit (ECU) of a car engine controls these curves. ECU has a lookup map of all possible curves and selects a curve based on the current crankshaft load and speed. If a car runs downhill at a certain speed, the crankshaft load can even be negative, so the actuator curves are chosen accordingly. On the other hand, for a high crankshaft load and a certain speed, the actuator curves and resulting injection have completely different shapes. However, a detailed discussion of injection curve modeling is out of the scope of this paper.

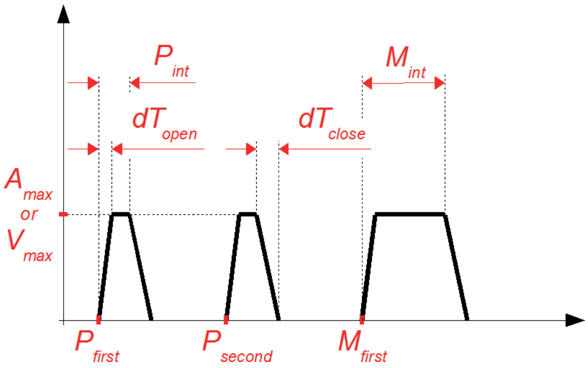


Figure 4.3: The actuator, the top most element in the model in Figure 4.2, is modeled using these parameters. This is the only set of control parameters that will be changed during the engine operation. Depending on the operation point (speed and load), the electric managing unit (EMU) will select the shape of actuator curves.

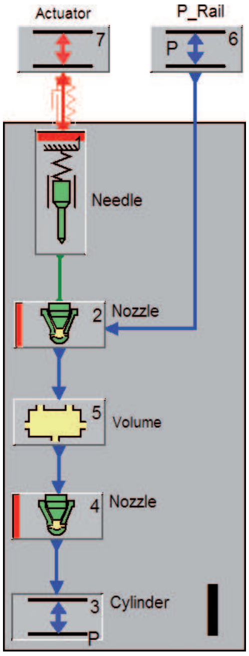


Figure 4.4: A simplified model used in the first iteration. We use only one section and the rest is represented by an ideal actuator. Once the simplified model is tuned, we gradually extend it to include the rest of the model, as shown in Figure 4.2.

One design option [KMG$^+$06, MJJ$^+$05] is to run the simulation for all possible combinations of parameters and to explore the system. If we use ten values per control parameter, there are 10^{11} possible combinations of control parameter values. Since we can run about ten simulations per minute (for the model in Figure 4.2), we

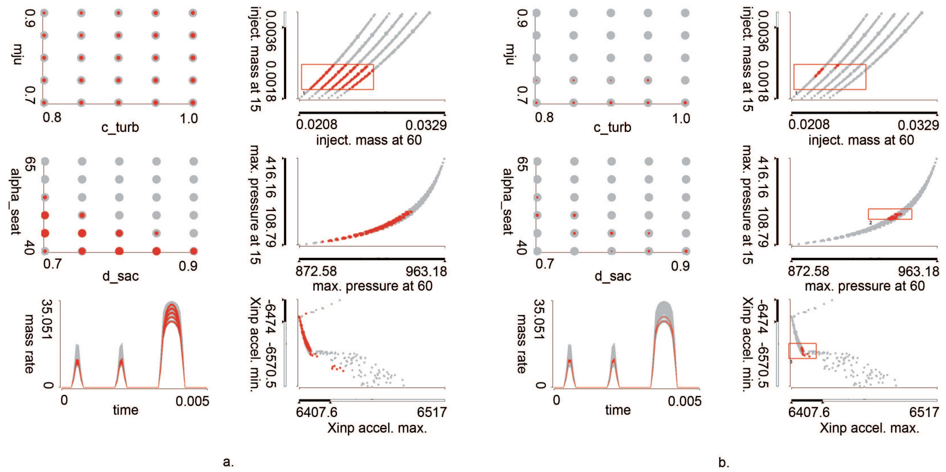


Figure 4.5: Multiple coordinated views show control and output parameters. **a)** We brush target values for the injected fuel mass. All combinations of *c_turb* and *mju* can produce the desired output (second scetterplot) while only some combinations of *d_sac* and *alpha_seat* are possible. **b)** Further refinement of targets using additional brushes for pressure and acceleration helps us to narrow possible parameters and to estimate input parameters for the first step.

would need 10^{10} minutes, or more then 19,000 years to complete all simulation runs. It is clear that such simulation time, even on a large cluster, is not feasible. Instead, we start with a simplified model, use interactive visualization to drill down the control parameter space, and once the initial control parameters are fixed, the simulation model is extended.

4.3.1 First Model

We start with a simplified model (Figure 4.4), *Block I* from Figure 4.2 with the actuator directly added on the top. This is supposed to be a direct actuator with simple characteristics. We tune the nozzle first.

We use only four control parameters (Table 4.1). It takes about 12 minutes to calculate 750 cases (60 simulations per minute for this simpler model). After this setup time, we explore the first data set to achieve a certain amount of injected fuel during the pilot and main injection. The target values were set according to Table 4.2.

We compute the calculated output parameter *injected mass* as a function of time. It is a cumulative mass over time. The value at the end corresponds to the totaly injected mass during injection. As we are interested in the injected mass after the first pilot injection and after the main injection, we aggregated the injected mass curves so to have the injected mass after first pilot and the total injected mass. We

Table 4.1: Control Parameters for the first case.

Parameter	Min	Max	Step
d_sac	0.7	0.9	0.05
alpha_seat	40	65	5
c_turb	0.8	1.0	0.05
mju	0.7	0.9	0.05

Table 4.2: Target Parameters for the first case.

Parameter	Target range (mg)
Pilot injected mass	2 – 2.5
Main injected mass	17 – 22

brush now the scatter plot depicting these two aggregated parameters. Figure 4.5 a shows this case.

At the same time, other coordinated views show the control parameters (Figure 4.5a, scatter plots in the first column) causing the targeted injected mass. We can clearly see that the wanted injected mass is possible only for some combinations of nozzle diameter (*d_sac*) and angle of needle seat (*alpha_seat*), and for all combinations of flow discharge coefficient (*mju*) and turbulent flow coefficient (*c_turb*). We refine the selection by selecting high pressure (as far as possible for the given target) and the desired needle acceleration. Figure 4.5b illustrates the selections (brushes 2 and 3). Note the zoomed-in scatter plot of accelerations which helped in the selection.

The allowed control parameter space has narrowed significantly (Figure 4.5b, scatter plots in the first column), and the expert can now select the first parameters (Table 4.3). Note that this is a quite coarse estimation. The parameters are fine-tuned at a later stage. However, even this coarse case shows which input ranges make no sense.

Table 4.3: Control Parameters selected for First Model.

Parameter	Target range (mg)
d_sac	0.75
alpha_seat	50
c_turb	0.9
mju	0.7

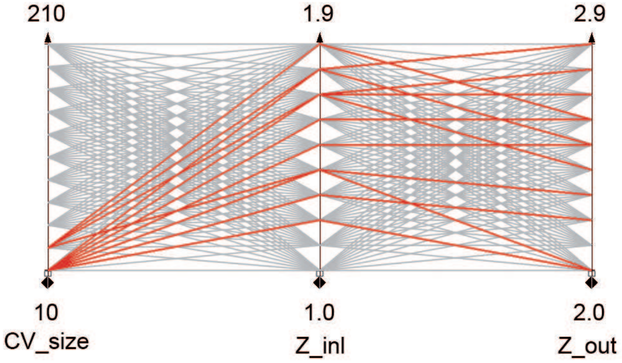


Figure 4.6: Second Model: The parallel coordinates view of the control parameters for the target values. Note that only low *CV_size* values are possible, while there are many combinations of *Z_inl* and *Z_out* which allow the desired output.

4.3.2 Second Model

We refine the simplified model (Figure 4.4) to create a more detailed model containing *Block II*. The actuator with the control valve is placed on top of *Block II* now. This actuator is described in Figure 4.3. We tune this part using the fixed control parameters for *Block I*. We are interested in the control volume size (*CV_size*) and in the inlet/outlet throttle flow resistance (*Z_inl* and *Z_out*). 1,100 simulations are computed (20 simulations per minute since the simulation time changes with the model complexity). For the same target values as in the first case, we set the control volume size to ten. Figure 4.6 shows the parallel coordinates view of the control parameters for the target values.

If we look at the mass rate curve view now, a new, interesting phenomenon can be observed (Figure 4.7). Note the width of curves at injections. There are some curves starting later and some starting on time. The actuator on the top is fixed, i.e., it has always the same input curve (Figure 4.3). This means that some of the parameter combinations can cause an injection delay. Injection delay is the time period between the injection reaction and the actuator action. In our case this is the difference between the start of the pilot (or main) injection as depicted in the curve view and the actuator starting times (P_first or M_first). This is a surprising finding, since we did not expect that this delay would show up at this stage of modeling.

The injection delay is an unwanted behavior and we have to be sure that it does not happen in the final model. Compared to the first, simplified model, the control volume which is placed just above needle top is not directly connected to the actuator any more. It is working in close correlation with two orifices (inlet and outlet) that supply the volume with fuel and drain it. We have to be sure that delay is as small as possible at this stage of modeling.

The correlation between the mass flow rate through the nozzle (*mass_rate*)

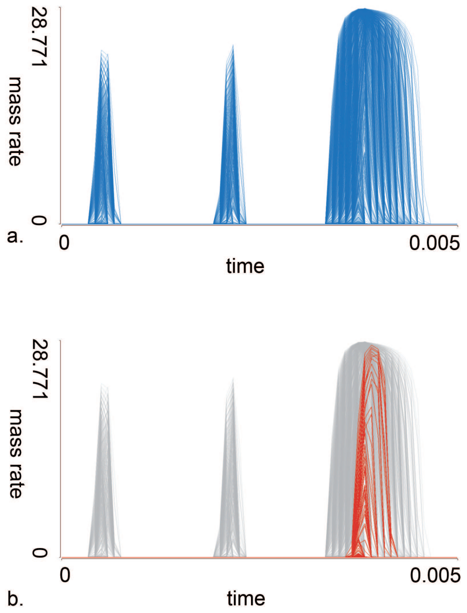


Figure 4.7: *Mass_rate* curves for the second model. **a)** Many curves of different widths, showing significant delays for some parameter combinations. **b)** The selection corresponds to the maximum values of *CV_size*, such curves would result in insufficient fuel mass. Note also that pilot injections are completely missing in this selection.

curves and the control volume size (*CV_size*) is easy to detect. Simple brushing shows that the larger the control volume size is, the narrower (less injected fuel) the curves are. Figure 4.7 depicts the curves selection for the maximum values of *CV_size*. The inlet and outlet flow resistance (*Z_inl* and *Z_out*) are more challenging and to investigate this problem we refine the model. New limits and step sizes are selected using the visualization tool and new simulations are initiated. The control volume size is set to the minimum, $CV\_size = 10$, and new flow resistance parameters *Z_inl* and *Z_out* are set (Table 4.4). The simulation tool is started and the visualization gets updated as new simulations are computed. The visualization tool checks if new data is available and automatically loads it. Since we do not change the model in this case, the update of the internal data and its representation is straightforward. During this process we continue the visual analysis and explo-

Table 4.4: Refined Control Parameters for Second Model in order to investigate injection delay.

Parameter	Range	Step
Z_inl	1,0 - 2.0	0.025
Z_out	2.0 - 3.0	0.025

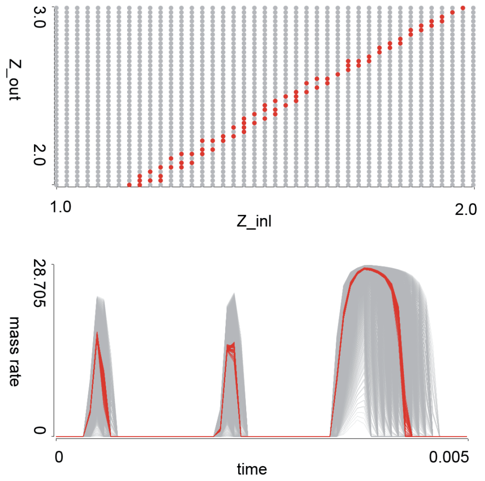


Figure 4.8: The scatterplot shows refined input parameters. We have selected desired output parameters (on views which are not displayed in the image) and the selection is shown in red. The *mass_rate* curves have a desired shape now. The linear correlation of *Z_inl* and *Z_out* is unexpected.

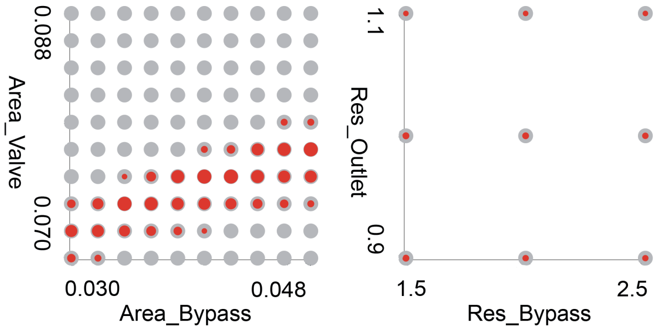


Figure 4.9: Two scatterplots showing the control parameters for the final model. We further refine the *Area_Bypass* and *Area_Valve* parameters. The red points correspond to selected desired output parameters.

ration. In this particular case, approximately 1,680 new simulations are computed, and iteratively loaded. Of course, the user can stop the simulation or request another refinement at any time.

We used the multiple view setup to observe what is happening. The target values for the injected mass, pressure in the SAC volume *P_sac*, and the needle acceleration are set, and flow resistance parameters show a linear dependency. Figure 4.8 shows the flow resistance parameters on the top and the mass rate curves in the bottom. Note the much denser parameter space due to the refinement. After a detailed exploration, we are able to remove the influence of the parameters on the delay. We understand what is going on, the delay turns out to be logical, and the parameters are set to 1.6 and 2.6 for *Z_inl* and *Z_out*.

4.3.3 Third Model

We are ready for the final step now where we further extend the model. It now corresponds to the model in Figure 4.2. Note that the control parameters for *Block I* and *Block II* are set (but they can be changed) and we tune the last part now. There are four parameters in the last block: the bypass flow resistance (*Res_Bypass*), the outlet flow resistance (*Res_Outlet*), the effective flow area at the bypass seat (*Area_Bypass*), and the effective flow area at the valve seat (*Area_Valve*).

Due to the model complexity we now calculate approximately ten simulations per minute. 900 parameter variations are set and we start the visual exploration. In contrast to the simple parameter refinement, the step change here represents a model change. Internal data structures in the visualization tool have to be changed. This is considered to be a larger step and the user has to wait a few minutes for the first results. Once the simulation software computes the initial results, parameter refinement can be done on the fly. During parameter refinement we continue the visual analysis and the data is automatically updated. The target values for the injected fuel mass (pilot and main) remain the same. The actuator is still fixed.

Figure 4.9 shows the parameters after the target values were selected. Two parameters have no significant influence, the target values can be achieved with all possible combinations of the flow resistances *Res_Bypass* and *Res_Outlet*. We set the values to 2.0 and 1.0, respectively. The other two parameters show a far more interesting behavior. A wide range of parameter values are initially investigated. It is successively refined as we realized where we need more information.

Figure 4.10 shows the parameters as computed at the end. We use two iterations, we refine the parameters once and then refine a subrange of parameters once more. This represents parameter refinement, and the data in the visualization tool is updated as new simulations are computed. During computation we continue the visual analysis. Output values are used to steer the refinement. Based on the output values we decide where to refine input parameters.

Figure 4.11 shows an example of the output parameter values as they are computed in various steps. Resulting outputs from various iterations are highlighted in the figure in order to illustrate results from various iterations.

Figure 4.12 shows the target injected mass, the corresponding pressure, control parameters, and mass rate curves. Note the scatterplot on the far left showing the same data (also in Figure 4.11) as the scatterplot from the very first model (Figure 4.2). The scatterplot in Figure 4.12 is zoomed in and shows the data from the final iteration only. We are far off the target values at the beginning and by successive refinement and simulation steering we achieve a finer granularity around the target area.

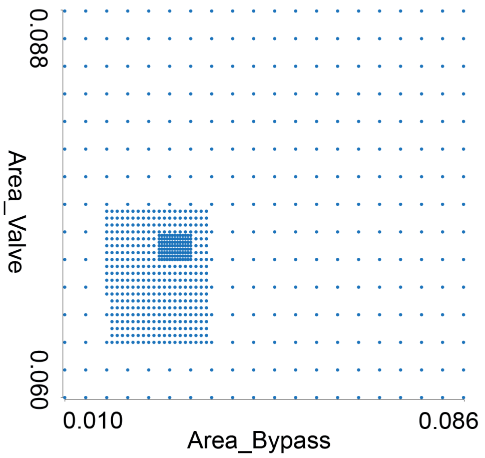


Figure 4.10: Combinations of the control parameters as created during the iterations. We have a coarse mesh of parameter values at the beginning, and we refine them twice during the process. All combinations of the control parameters are shown here, we can hide unwanted iterations during the analysis, if necessary.

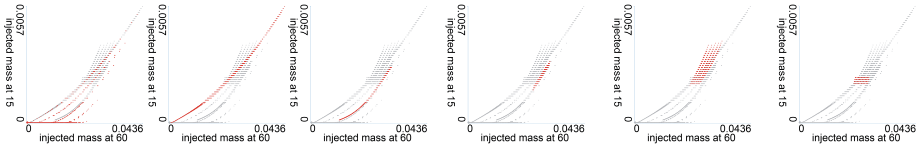


Figure 4.11: Six scatterplots showing output parameters as computed during six iterations of simulation steering. We can see there are many scattered values in the beginning. We then used interactive brushing in other views to get insight on how these output parameters are changing. Finally we can identify the desirable area, and the simulation results are refined in that area.

4.3.4 Final Model

We now set the final control parameters. Any of the data points of the injected mass shown in Figure 4.12 can be selected. They all result in a desired behavior. We have to select one set, however, as they can not be changed later. Due to the wanted pressure and needle acceleration, we select the effective flow area at the valve seat — *Area_Valve* and the flow discharge coefficient — mju to be 0.071 and 0.54, respectively. Our injector now is set.

The actuator on the top used to drive the injection in a real setup was fixed up to now. As stated before, the ECU of the engine will change the actuator during operation. Our parameters, on the other hand, remain the same. They cannot be changed at runtime.

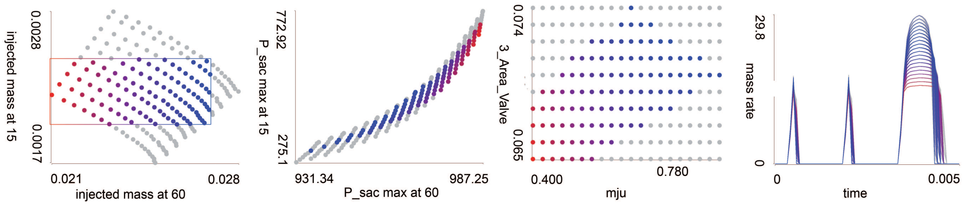


Figure 4.12: The final model. The target injected mass on the left is defined using a quite narrow range here. The corresponding pressure, input parameters and mass rate curves are shown. Everything seemed to be correct in this simulation model.

The last task is to check if the parameters also yield satisfying output for various actuator curves. We vary the actuator settings and again several times refine the parameters. Eventually we are satisfied and want to see all the response curves for all the combinations of actuator parameters (1,600 combinations are chosen).

To our great surprise, we see that some curves exhibit a very unusual behavior. The curve views in figure 4.13 show the response curves for *Needle_acceleration*, and pressures in the nozzle and control volumes, *P_nozle* and *P_control*, with the undesired peaks marked with red ellipses. Those peaks indicate system oscillations at specific points. Any oscillation in system is dangerous and undesired. The amplitude of any oscillation may rise above system limits for some unknown situations.

Now we have to find the reason for these oscillations in order to predict and avoid such a behavior. Furthermore, especially for the fuel injection system, any kind of secondary oscillations may open the nozzle at the wrong time and lead to fuel inflow in the combustion chamber and an undesirable combustion process.

To investigate the oscillations further, we isolate the peaks using a line brush in the curve view. The tool allows to simply draw a line across the curves, and all curves crossing the line will be selected. The composite brushing functionality is supported as well.

The scatter plot in Figure 4.13 shows pilot and main injection intervals, *P_Int* and *M_Int*, with the peaks selected. An unexpected and very interesting finding is the pattern at which peaks appear at 3 *M_Int* values. It shows oscillating behavior in the parameter space. We can easily skip those values, and program the ECU not to use these parameter values.

However, puzzled by this discovery, we want to investigate this phenomenon further. We go back one more time. The parameters with most influence up to now: *mju*, *Area_valve*, *Area_Bypass* and *M_int* are varied once more. Undesirable peaks are present again, but the control parameters are chosen to be far away from the settings which caused them. The previously set parameters are changed, and the injector is finally set. Table 4.5 shows the final values of the control parameters.

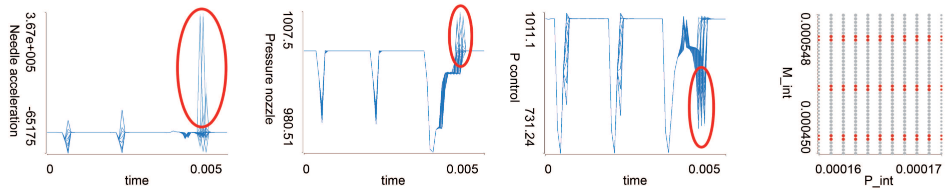


Figure 4.13: With the final model fixed, the actuator curves are varied. Quite surprisingly, there are unwanted peaks in the output curves for multiple parameters. Red ellipses show these peaks. It is not intuitively clear why and when they occur, and we explored it in more details. The assumption that all parameters are set and that actuator variations are just a routine fails and we had to go several steps back and run new simulations. Corresponding values of *M_int* are shown in the scatter plot on far right. Note the puzzling oscillating behavior.

4.4 Conclusion

The coupling of interactive visualization with the simulation back-end facilitates fast prototyping of a system under development. We start from a non-optimized initial prototype and the corresponding simulation model and through an iterative process, going back and forth between different levels of abstraction, we refine the simulation model and design a system that meets the requirements. In doing so we are significantly reducing the number of simulation runs.

The brute force approach requires to run simulations for all possible combinations of the control parameter values which is not computationally feasible. The described approach requires only several thousands simulation runs to find a design that meets the requirements.

The interdisciplinary setup of this project allowed to develop this steering solution in the context of a real-world problem. It was very rewarding to see how the tool facilitated new discoveries (Section 4.3.4), even quite surprising ones. The discoveries provided much better insight and allowed us to anticipate and address oscillation problems and thus create a much better design. Engineers still only seldom use interactive visualization and usually analyze simulation results using static 2D charts, depicting few simulation runs simultaneously in most cases. They also use automatic optimization methods, but our approach offers completely new view and insights.

The three levels of iteration (simulation data, control parameters values, simulation model) provide different levels of interactivity. While viewing the simulation data is done in real-time, changing the simulation model introduces a noticeable delay. However, since we can go back and forth between different levels, instead for waiting for the simulation model update to propagate to the simulation data, we use the simulation data level and continue our analysis until new simulation data are

Table 4.5: Final control parameters. The *Block IV* parameter values are not analyzed at this stage (listed for the sake of completeness).

Param.	Name	Description	Final Value
I_1	*d_sac*	Sac diameter	$0.75mm$
I_2	*alpha_seat*	Needle seat angle	$50 degrees$
I_3	*c_turb*	Turbulent coefficient	0.9
I_4	*mju*	Flow discharge coefficient at nozzle holes	0.6
II_1	*CV_size*	Size of control volume	$10mm^3$
II_2	*Z_inl*	Inlet flow resistance in control volume	1.6
II_3	*Z_out*	Outlet flow resistance out of control vol.	2.6
III_1	*Res_Bypass*	Flow resistance through bypass	2.0
III_2	*Res_Outlet*	Flow resistance through outlet	1.0
III_3	*Area_Bypass*	Bypass effective area	$0.032mm^2$
III_4	*Area_Valve*	Valve effective area	$0.07mm^2$
IV_1	*HPP_Length*	Length of high pressure pipe (fixed)	$300mm$
IV_2	*RV_Size*	Common Rail volume size (fixed)	$30cm^3$

generated. This approach is rather general and applicable to a wide range of design problems.

We plan to explore a variety of design problems and related solutions to identify some design patterns. We will further improve the interactivity of the developed tool. Some semi-automatic support for drill-down, possibly involving approaches to (semi-) automatically detect a region which seems to be out of the range of interest will be researched as well. Finally, we will explore a collaborative, multi-user version of the tool to “share” the design process among several experts.

4.5 Acknowledgements

Parts of this work have been done through the scope of applied and basic research at the VRVis Research Center (funded by an Austrian governmental research program called Kplus, http://www.kplus.at/) and at the Center for HCI at Virginia Tech (http://www.hci.vt.edu/).

5

Visual Analytics for Complex Engineering Systems: Hybrid Visual Steering of Simulation Ensembles

Published as:

> Krešimir Matković, Denis Gračanin, Rainer Splechtna, Mario Jelović, Benedikt Stehno, Helwig Hauser, and Werner Purgathofer:
> **Visual Analytics for Complex Engineering Systems: Hybrid Visual Steering of Simulation Ensembles**
> *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1803-1812, 2014.

Visual Analytics for Complex Engineering Systems: Hybrid Visual Steering of Simulation Ensembles

KREŠIMIR MATKOVIĆ, DENIS GRAČANIN, RAINER SPLECHTNA,
MARIO JELOVIĆ, BENEDIKT STEHNO, HELWIG HAUSER,
AND WERNER PURGATHOFER

Abstract

In this paper we propose a novel approach to hybrid visual steering of simulation ensembles. A simulation ensemble is a collection of simulation runs of the same simulation model using different sets of control parameters. Complex engineering systems have very large parameter spaces so a naïve sampling can result in prohibitively large simulation ensembles. Interactive steering of simulation ensembles provides the means to select relevant points in a multi-dimensional parameter space (design of experiment). Interactive steering efficiently reduces the number of simulation runs needed by coupling simulation and visualization and allowing a user to request new simulations on the fly. As system complexity grows, a pure interactive solution is not always sufficient. The new approach of hybrid steering combines interactive visual steering with automatic optimization. Hybrid steering allows a domain expert to interactively (in a visualization) select data points in an iterative manner, approximate the values in a continuous region of the simulation space (by regression) and automatically find the "best" points in this continuous region based on the specified constraints and objectives (by optimization). We argue that with the full spectrum of optimization options, the steering process can be improved substantially. We describe an integrated system consisting of a simulation, a visualization, and an optimization component. We also describe typical tasks and propose an interactive analysis workflow for complex engineering systems. We demonstrate our approach on a case study from automotive industry, the optimization of a hydraulic circuit in a high pressure common rail Diesel injection system.

5.1 Introduction

Recent advances in computation technologies provide an opportunity to compute large simulation ensembles — multiple simulation runs of the same simulation model using different sets of control parameters. Parameter spaces of complex engineering systems, if not carefully sampled, can result in prohibitively large simulation ensembles.

Current emission regulations and efficiency goals are great challenges for automotive systems designers. In order to meet strict time constraints and reduce the time to market, system designers of modern automotive systems need powerful design tools to understand the systems, their behavior, and their responses to changes of the design parameters. In this paper, we present a case study dealing with an injection system, i.e., one of the key components of modern car engines. Target users of the proposed solution are designers of complex systems that are based on simulation ensembles. This paper is a result of a long-term collaboration between visualization and simulation experts. We, a team of visualization, simulation, and injection experts, developed the proposed approach, inspired by the actual application in the automotive industry. Our collaboration included numerous interviews and common sessions. We had regular meetings on a weekly basis for more than six months. One of the injection experts with more then 15 years experience in simulation, also coauthors the paper. Additionally we observed and interviewed four more simulation experts. Hence, when we say *we* throughout the paper, we mean the whole team, the visualization and the simulation experts. In our opinion, neither group alone could come to such a solution. The new approach is the result of a long-term research effort to address and overcome the described obstacles in the design of a complex system. Although we developed the newly proposed approach with experts from the automotive industry we are confident that the proposed approach can be used in other domains where complex simulation data in high dimensional parameter spaces have to be explored and understood.

Properly specifying the simulation parameters is a tedious task, and, at the same time, crucial for the effective utilization of simulation. There is an inherent trade-off between the simulation accuracy and its speed. Better accuracy requires more simulation points (i.e., simulation runs) to better cover the parameter space. More simulation runs increase the simulation time and lengthen the design process. A system designer needs help to navigate the simulation space and explore the most promising combinations of simulation parameters. With proper support, the designer can be more efficient and productive.

Simulation results often have a complex structure and a simplified representation. A common workflow includes the extraction of certain scalar features prior to the analysis. These features are then studied in the automatic and interactive analysis. Current state of the art techniques also support the consideration of complex data in interactive studies [KMG$^+$06], but these techniques do not support an automatic analysis at the same time. Our work targets the interactive hybrid visual steering of a simulation ensemble which combines simulation and optimization with interactive visual steering to provide an integrated design environment.

The identified tasks for an integrated, hybrid steering environment are summarized in Table 5.1. These tasks are abstractions of the observed real-world practices and concrete tasks/activities in the automotive design workflow. Supporting these tasks is the key requirement that guided the development of our integrated, hybrid

Table 5.1: Hybrid steering tasks abstractions.

A	**Explore and Analyze the Ensemble**	
A1	**Parameters' Sensitivity**	Identify simulation results for certain control parameters and explore the parameters' sensitivity.
A2	**Model Reconstruction**	Identify control parameters for a desired output.
A3	**Comparison**	Compare output results related to different areas of the parameter space.

R	**Compute the Regression Model**	
R1	**Model Validation**	Show the model accuracy across the parameter space.
R2	**Model Definition**	Partition the parameter space and define relevant parts for the regression model building.
R3	**Automatic Optimization**	Automatic optimization using the regression model.

D	**Generate the Data**	
D1	**Initial Parameter Space Sampling**	Select regions in the parameter space to be initially sampled.
D2	**Interactive Refinement**	Select regions in parameter space which have to be resampled.
D3	**Automatic Optimization Refinement**	Choose refinement regions based on automatic optimization.

steering environment and our hybrid visual steering approach.

The main contributions of this paper are: **(1)** A case study demonstrating Hybrid Visual Steering, a novel simulation ensembles steering and exploration approach. This approach combines interactive exploration and analysis with automatic optimization based on regression models; **(2)** The task abstractions (Table 5.1) and the supporting visualization system, including two improved views, the Parameters Exploration View and Regression Exploration View. **(3)** The tight integration of all relevant components in an interactive workflow; and **(4)** An evaluation of the proposed approach based on a case study from the automotive industry including user feedback.

We build on our previous work [KMG^{+}06, MDG^{+}13, MGJ^{+}10, MGJH11, MGJH08] which integrates multiple simulation runs and visualization and focuses exclusively on the interactive exploration and steering. Here we introduce the adap-

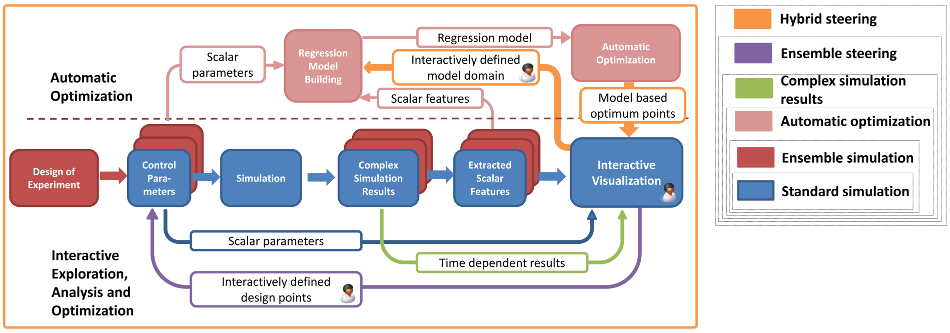

Automatic Optimization
Scalar parameters
Regression Model Building
Regression model
Interactively defined model domain
Automatic Optimization
Scalar features
Model based optimum points
Design of Experiment
Control Para-meters
Simulation
Complex Simulation Results
Extracted Scalar Features
Interactive Visualization
Scalar parameters
Time dependent results
Interactively defined design points
Interactive Exploration, Analysis and Optimization
Hybrid steering
Ensemble steering
Complex simulation results
Automatic optimization
Ensemble simulation
Standard simulation

Figure 5.1: Overview of the proposed approach. **Standard simulation**: A collection of control parameter values is used for a single simulation run to determine and visualize simulation results (extracted scalar features). **Ensemble simulation**: Design of experiment methods are used to create several collections of control parameters. The resulting output values are aggregated and visualized together with the control parameter values. **Automatic optimization**: Aggregated parameter values are used to create a regression model which is used for optimization using the defined optimization constraints. This approach is usually decoupled from visual analysis, or visualization is used to show optimization results only. **Complex simulation results**: All complex simulation results are visualized. **Ensemble steering**: During visual exploration additional control parameter values for new simulation runs are selected by means of visualization. **Hybrid steering**: A unified approach which enables the exploration of parameters, complex results, extracted features and optimization results. Furthermore, it uses results from automatic optimization to guide the user during interactive visual steering. The hybrid steering also supports regression model building and optimization constraints and goals specification, all within the same framework.

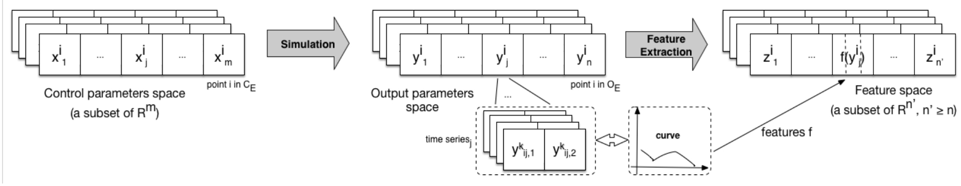


Figure 5.2: Simulation ensemble data model: control data points, output data points and features. For each output data point $\mathbf{y}^i$, there can be an output value y^i_j that is a time series (a curve). The time series is replaced by one or more scalar values (feature f) in the feature space.

tive exploration of the simulation space, based on regression modeling and the use of optimization to find an optimum within a subset of the simulation space. This new approach covers the spectrum between a fully automatic simulation and the manual adjustment of simulation parameters.

5.2 Related Work

Simulations are usually computationally intensive and are often combined with interpolation for sensitivity analysis and optimization. An example is the Kriging interpolator, representing a global metamodel that covers the whole experimental area [vBK04]. However, we can also iteratively refine the simulation model, in addition to the refinement of the simulation parameter values.

If data analysis is a postprocessing step after a simulation batch is completed, errors invalidating the results of the entire simulation may be detected too late [PJB97]. Computational steering and interactive visualization started in 1980s and 1990s as useful visualization paradigms for the computational sciences [CRS96] enabling users to interactively steer computations, change simulation parameters and instantly see the simulation results. The simulation results are usually presented using scientific visualization methods [HEvLS03].

Computational steering integrates modeling, computation, data analysis, visualization, and data management components of a simulation [PJB97]. However, integrating simulation within computational steering can be a very difficult problem. We need to address four facets of the problem [JPH$^+$99]: control structures, data distribution, data presentation, and user interfaces. Since computational steering is a highly interactive process, the user interface is a critical component [MvWvL99]. This early simulation steering approaches usually deal with a single simulation run which lasts for a long time. The idea is to monitor simulation execution and to change some parameters if preliminary results seem to be wrong. We do rely on basic simulation steering principles, but we deal with simulation ensemble steering. Our simulation can be computed relatively fast, and we steer the ensemble creation, not a single simulation run.

In each iteration of computational steering the user can define a region of interest in the parameter space that should be explored in more detail. Additional simulation runs are needed to cover that region, constituting a new “simulation experiment”. The design of such an experiment, i.e., the selection of the simulation points in the region of interest, is very important since we would like to reduce the number of simulation runs while providing a good coverage of the region of interest [Kle07, Ony09].

While the support for a user controlled simulation is at the very core of computational steering, there is very limited support for user controlled optimization [BP10]. Very often there is no clear or unique optimal solution. A user has to analyze, in an interactive fashion, trade-offs and interdependencies between objectives [PGR99, SBC$^+$04, TMH09]. Using an analytical representation of the objective function the user can explore the values of the objective function in the region of interest [MM06]. Such values can be dynamically updated in all views and brushes (selections) [PTMB09]. All these solutions are not integrated in an interactive steering environment. They focus on optimization based on a batch of precomputed simulation runs. In our case, we use optimization as a guideline in interactive steering in a fully integrated workflow.

The simulation data consists of discrete simulation points while the region of interest is usually a continuous space. We can use the simulation points to “span” that space using a surrogate (regression) model that approximates simulation results over the entire region of interest. The number of grid points in full grid methods depends exponentially on the number of dimensions. However, using sparse grids can reduce the dimensionality problem under some smoothness conditions. The sparse grid method, originally developed for the solution of partial differential equations [Zen91], is also used for interpolation and approximation. The properties of the hierarchical representation and approximation properties of sparse grids are discussed by Bungartz and Griebel [BG04b]. Improvements over the classical sparse grid approach include spatially adaptive refinement, modified ansatz functions, and efficient regularization techniques [PPB10].

Simulation steering and dealing with ensemble simulations require control over multiple heterogeneous simulation runs. World lines [WFR$^+$10] integrate simulation, visualization and computational steering to deal with the extended solution space by representing simulation runs as causally connected tracks that share a common time axis. The user has to select parameter combinations for new runs, there is no automatic support in selection of the new design points. Konyha et al. [KMG$^+$06] and Matkovic et al. [MGJ$^+$10, MGJH11] use interactive visual analysis for engineering problems with large parameter spaces. This is a purely interactive solution without an automatic support for steering. Berger et al. [BPFG11] employ statistical learning methods to predict results in real-time at any user-defined point and its neighborhood. The user is guided to potentially interesting parameter regions and the uncertainty of predictions is shown using 2D scatterplots and parallel coordinates. Booshehrian et al. [BMPM12] present a parameter space exploration approach from the fishery domain. These systems are not coupled with simulation, they operate on a set of predefined simulation runs. Engel et al. [EGG$^+$12] describe a novel interactive visual framework for dimensionality reduction of high-dimensional single particle mass spectrometry data. Bergner et al. [BSM$^+$13] present ParaGlide, a visualization system designed for interactive exploration of parameter spaces of multidimensional simulation models. They do initiate new data generation from the visualization, but the selection of points is based solely on user input, there is no support from automatic methods.

Machine learning techniques such as support vector machines [BGV92, CV95, SS04b] or relevance vector machines [Tip01] can be used to create linear, quadratic or nonlinear surrogate models. The validation of a surrogate model is difficult in general [PBK10]. We assume that our regression models are validated.

Although the related work covers parts of our proposed solution, none of these approaches, according to our best knowledge, integrates all components in a unified framework.

5.3 Simulation and Visualization

Figure 5.1 illustrates our Hybrid Visual Steering approach and its evolution. The basic workflow in simulation includes model definition, setting of control parameters, simulation, feature extraction from complex simulation results, and the visualization of the extracted features. Feature extraction is necessary if the simulation produces complex data that is not suitable for standard direct visualization. The blue parts in Figure 5.1 correspond to such a traditional procedure. Advances in computation make it possible to compute many runs for the same simulation model with different sets of control parameters — a simulation ensemble.

In this case the parameter space has to be sampled and different combinations of control parameters have to be chosen. This is a well-known problem (design of experiment) for which there are several available techniques. A simulation ensemble results in a combination of multiple complex simulation results and multiple scalar features (red parts in Figure 5.1). If an automatic optimization is desired, a regression model can be computed based on the control parameters and the extracted scalar features. This step is usually decoupled from the visualization (light red parts in Figure 5.1). In our previous work we have demonstrated that also complex simulation results can be integrated in the visual analysis [KMG$^+$06, MGJ$^+$10] (green parts in Figure 5.1). Ensemble steering makes it possible to select new sets of control parameters from the visualization [MGJH08] (purple parts in Figure 5.1) in an iterative, interactive manner.

When the simulation space is very large, the iterative design process can be time consuming and tedious. We would like to help the domain expert by automatizing this process as much as possible. Therefore, we couple automatic optimization with the visualization in a hybrid visual steering environment (orange parts in Figure 5.1). Our framework supports all identified tasks for a complex system design.

Integrated design environments are not readily available for industrial design. Tools are used separately or as partially integrated tools which significantly reduces efficiency. The integrated design environment we developed for the common rail injection design resulted, according to the domain expert, in a speed up factor of at least ten compared to the conventional approach where all tools are used separately. We also talked with four more domain experts at the AVL company working on optimization, timing drive, hybrid vehicle, and crankshaft design. They informally evaluated our integrated design environment prototypes and estimated a similar potential for speed up.

5.3.1 Formal Background

We often model the simulation process as a function S that maps the control parameters $\mathbf{x} = (x_1, \ldots, x_m)$ (a control data point in $\mathbb{R}^m$) to the output values $\mathbf{y} = (y_1, \ldots, y_n)$ (an output data point in $\mathbb{R}^n$) where m is the number of control parameters and n is the number of outputs. Due to the physical constraints of the simulated system, the set of feasible control data points C is a subset of $\mathbb{R}^m$ and the set of feasible output data points O is a subset of $\mathbb{R}^n$. A simulation ensemble E is a set of pairs of data points $(\mathbf{x}, \mathbf{y})$, $\mathbf{x} \in C$ and $\mathbf{y} \in O$.

Traditional data analysis approaches, such as statistics or OLAP techniques [Tho97], usually use a relatively simple multi-dimensional model [CMS99, Sam06] (simple with respect to the types of data dimensions) using data tables to capture relations among control parameter values for the same simulation run. While the control parameters are almost always numerical, scalar values, the output values also often include time/data series so O is no longer a subset of $\mathbb{R}^n$.

Since simple data tables [CMS99] are not sufficient, we need an adequate simulation ensemble data model to deal with more complex data. Our simulation ensemble data model uses a two-level data hierarchy for the output data points (Figure 5.2).

For each output data point $\mathbf{y}^i$ and y^i_j that is a data series, we have a separate set of "sub-points" with its own length and number of dimensions (parameters). Our discussion is limited to two dimensions. We can select one of the data series dimensions as the independent variable (e.g., time) and the other dimension as the dependent variable. Such data series can be considered a function of one variable and represented as a curve. The curve is replaced by one or more scalar values (features f) to create a feature point $\mathbf{z}^i$ in the feature space F, a subset of $\mathbb{R}^{n'}$, $n' \geq n$.

In other words, the scalar values y^i_s from $\mathbf{y}^i$ are directly used in the feature space as the corresponding values z^i_s in $\mathbf{z}^i$ while each data series value y^i_j from $\mathbf{y}^i$ is replaced by one or more scalar values (features f) in $\mathbf{z}^i$.

5.3.2 Regression Model

If we can approximate the mapping S from the control parameter values to the simulation results using a regression model, we can estimate the simulation results much faster, compared to actually running the simulation. In order to estimate S, a number of simulation runs are executed to get a set of input-feature pairs (simulation ensemble), $\{(\mathbf{x}^i, \mathbf{z}^i)\}$ as training data. A regression model R is built from the training data as the surrogate model of the simulation.

After the regression model is trained, we can then use it to estimate the simulation results for arbitrary input values. Or, more importantly, it can be applied in an optimization process, of which the goal is to obtain a set of control parameters so that the simulation results satisfy a set of desired constraints, such as the maximization of a linear combination of the output variables.

5.4 Integrated Interactive Steering Workflow

Hybrid Visual Steering integrates simulation, optimization, and visualization in a unified framework enabling the user to conduct simulation ensemble steering. After the computation of a set of initial runs the user explores the simulation ensemble, and detects a region of interest. New simulation runs are conducted to adaptively increase the resolution of the simulation points in that region and augment the simulation ensemble. Automatic optimization supports the user in identifying regions of interest. However, since the proposed optimum values are based on regression models (approximation) built based on scalar features extracted from actually more complex simulation data (again, an approximation), we can not rely on them. The user needs a hint on the regression model accuracy in the detected region, and dependent on this accuracy more or fewer additional runs will be computed. The overall workflow can be summarized as:

- Conduct simulation runs based on the design of experiment in the preceding iteration (for the first iteration create an initial design of experiment).
- Integrate the new simulation runs with the existing data (if there are any).
- Visually analyze the data and select an objective function.
- Set a regression model to explore the objective function.
- Identify a region of interest.
- Use optimization to determine (seemingly) optimal value(s).
- If the result of this optimization is not satisfactory, create a new design of experiment with an increased resolution.

The realization of a hybrid visual steering framework also represents a technical challenge. All components do exist in current design processes, but they are not coupled in a unified framework. Furthermore, not all of the components support the identified tasks, and they have to be extended. The following components are needed to realize an integrated hybrid visual steering environment:

- **Design of Experiment (DOE) Component**: supports the specification of a subspace of possible input values (a shape in a multi-dimensional space) and the specification of a distribution of points within this space.
- **Simulation Component**: simulates the phenomena of interest at a comparably accurate level.
- **Analysis and Exploration Component**: supports feature extraction, advanced interaction, and the visualization of complex and scalar simulation results. In the case of steering, it also supports the interactive selection of subspaces in the parameter space, and the specification of new design points. Finally, it also controls regression model building, evaluates it, and shows optimization results.

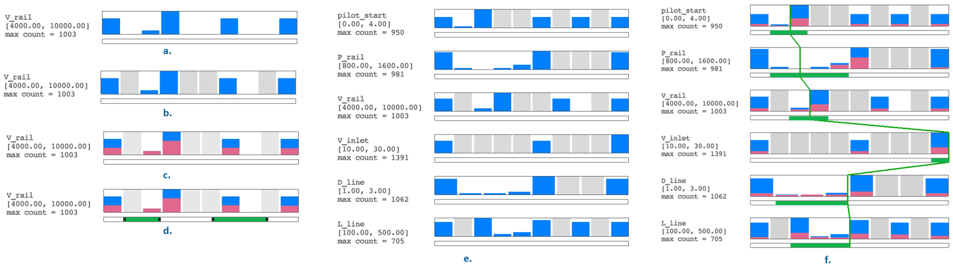


Figure 5.3: Parameters Exploration View: **a)** Histogram for one parameter. Note that bins which are shown empty might also contain just a few items which are not visible if the display resolution is too low. **b)** Empty bins are shown in a different color (grey). This design was preferred by domain experts over highlighting non-empty bins. **c)** Histogram showing brushed runs (red) in relation to the overall distribution (blue). **d)** Constraints bar below the histogram that is used to specify optimization constraints. **e)** Six parameters shown in the view. **f)** Six parameters with constraints and optimum values. Some items are brushed and they are shown in red.

- **Regression Model Building Component**: builds a regression model from (a subset of) the already computed simulation runs.
- **Automatic Optimization Component**: computes an optimum using the regression model, subject to the interactively specified constraints.

We use AVL's [AVL15] Design Explorer which supports DOE definition, regression model building, and automatic optimization and AVL's CruiseM for simulation. We extend the existing interactive analysis and exploration component to support all analysis tasks listed in Table 5.1. We exploit the well-known principle of coordinated multiple views and integrate all components in a unified framework.

5.5 Tasks and Steering Design

We have identified three main groups of tasks (Table 5.1). Each group of tasks has specific requirements. The main questions are how to visualize control parameters and simulation results, how to design the interaction, both for ensemble exploration and for optimization constraints as well as goals definition, and how to specify new points in the parameter space. Based on our accumulated experience and the current state of the art in exploratory visualization, we rely on the coordinated multiple views principle. The main idea is to depict multiple dimensions using several views and to allow the user to interactively select (brush) subsets of the data in a view. Consequently, all the corresponding data items in all linked views will be consistently highlighted.

5.5.1 Control Parameters Visualization

There are several possibilities for sampling the parameter space. The parameter space can be continuous, or discrete. However, even if the parameter space is continuous, we only select a limited, discrete set of parameters, resulting in a discrete parameter sample.

As we deal with a multi-dimensional parameter space, well-known techniques from visualization can be used. In the case of a continuous parameter space, parallel coordinates are often used [PBK10]. Projections to 2D using scatterplots are also frequently used. Furthermore, histograms and bar charts are also regularly used to show values of different parameters [BMPM12].

We needed a view which supports the identified tasks (Table 5.1, tasks A1–A3, R1–R3, D1–D3). We can abstract the tasks on a finer level when it comes to the parameter space visualization and identify general task requirements:

- Show the distribution of each parameter (A1–A3, R2, D1, D2).
- Show the distribution of brushed simulation runs (A1, A2, A3).
- Support the interactive specification of optimization constraints for the parameters (R3).
- Initiate regression model building (R2).
- Show the automatically computed optimum values (A3, R3).
- Keep track of the automatic optimization process (R3).

None of the standard views supports all these requirements. We designed a new view — the Parameters Exploration View which meets the design requirements. It is inspired by the attribute explorer [ST98]. We add additional bars for specifying optimization constraints and additional interaction capabilities.

The basic idea is to show each parameter as a histogram with a user defined bin count. Figure 5.3a shows the histogram for one parameter. Basic information, like the parameter name, its range, and the maximum number of counts across all bins are shown on the left. We do not show this information under the histogram as we stack histograms vertically. There are five bins in the histogram in Figure 5.3 that seem to be empty. As the number of runs per bin can vary, and the histograms have a limited vertical space, some non-empty bins can occupy only one or even less than one pixel. We realized that it is important to mark really empty bins.

During an informal user study with five engineers at the AVL company, we presented them with two alternatives, explicitly marking empty bins and explicitly marking non-empty bins. All engineers preferred the solution where empty bins are marked with a gray rectangle. Figure 5.3b shows such a solution. We see that there are only four empty bins. There is a bin with just a few runs, which is not marked as empty. This information cannot be seen in Figure 5.3a.

The view is fully integrated in the coordinated multiple views environment and also shows the brushed runs. Figure 5.3c shows one histogram showing brushed runs (red) in addition to the overall distribution (blue).

When designing a complex system the user wants to test various hypotheses and adjust optimization constraints. The optimization constraints change during the analysis process. We add a constraints bar below the histogram (green bars in Figure 5.3d) to support the interactive specification of constraints. The constraints bar is used to specify constraints for each parameter. As we binned the parameters a simple click in the constraints bar sets the constraints to the specified bin. The user can extend the constraints bar by additional clicks or simply by dragging one side of any rectangle in the bar. The dragging includes additional bins per default, but it can also be set to specify ranges not aligned with the borders of the bins.

Figure 5.3e shows the Parameters Exploration View for six control parameters. The computation of a new regression model is also initiated from the Parameters Exploration View. In the left part of the view, which we call view control (not shown in Figure 5.3), there are two buttons which start the process. One initiates the use of all simulation runs and the other computes the regression model based on the brushed runs only. The user can also choose a regression building method.

We also want to show the computed optimum values. We depict them as a polyline, passing through all histograms, similar to a parallel coordinates polyline. The line passes through the constraints bar as well. The lines are depicted in two colors, depending if they are a result from a regression model that was computed based on all runs or only on a subset of all runs. As the user hovers over the optimum line, corresponding constraints are shown in the constraints bar. The user can also hide the optimum lines if they are not of interest. There is a list of all computed optimum values (not shown in the figures) with user specified names, so the user can activate hidden optima on demand. Figure 5.3f shows the view for six parameters with an automatically computed optimum. The view does not only show brushed values but can be used to brush as well. A simple click on a bin selects the corresponding parameter values.

5.5.2 Simulation Results Visualization

Both parameters and the simulation results have to be shown together. As described in Section 5.5.1, we deal with a complex data model where scalars and curves are considered as elementary data units. In the case of an ensemble, this means that for each dimension we have a collection of scalars (as usual) or a collection of curves. We call all curves in an ensemble that belong to the same dimension a family of curves [KMG$^+$06].

When it comes to the exploration and analysis of ensemble simulation data (tasks A1, A2, and A3), we have to show the simulation results. Standard views, such as scatterplots, parallel coordinates, or histograms are most often used during our study. For more complex outputs, in particular for the families of curves, we use a curve view which depicts all curves simultaneously also employing a certain density mapping, when needed. Brushing is extensively used in the exploration of such ensemble simulation data. Having all complex outputs depicted, we can easily realize the reconstruction of a model (task A2) which is a tedious or impossible task using analytical methods only.

We use regression models that are evaluated and tuned. There are multiple methods for evaluating and tuning a regression model (a detailed description is beyond the scope of this paper). We are dealing with a high-dimensional space and many time-dependent state variables are represented by aggregates. Although the model is tuned, it is practically impossible to find a model which fits all simulation points. Therefore, we also need to show the accuracy of the regression model (task R1). If an automatically computed optimum belongs to an area where the regression model is not a good approximation of the simulation results, we need more simulation runs.

In order to show the model's accuracy, we simultaneously show simulation results and approximated results (from the regression model) in the Regression Exploration View. The Regression Exploration View uses a projection of the high-dimensional output (solution) space (simulation results and regression model results) onto a plane determined by two selected output values (a scatter plot). During the design process a designer should always explore several such projections (different pairs of output values).

There are several ways to visualize pairs of points in the scatterplot. We showed different designs to five experts from the AVL company. We did not end with a uniquely preferred solution, as different tasks require different approaches. Throughout the following examples the simulation based points are orange and the regression model based points are blue. Of course, the user can also configure these colors.

The first idea is to show the pairs of points and connecting lines. Figure 5.4a shows such a case. In an ideal case the orange and the blue points would overlap and there would be no lines. In a realistic case the lines depict the accuracy of the regression model. The lines help in deciding if a computed point is a good enough approximation in a certain region. In case of many points (Figure 5.4a) there is a large overlapping problem and the user gets just a rough impression of accuracy. However, during a drill-down process in the analysis, the number of relevant pairs is reduced and the view becomes more useful. Figure 5.4b shows the same view for a subset of points. The lines are easier to identify now, and there is in general less clutter.

We also enable the user to show only simulation or only approximation points, and then, to use color coding and point size to encode the accuracy (the line length in the other solution). The view is less cluttered as it has less points (Figures 5.4c and 5.4d). We also offer a design where only color or only point size is used, but all domain experts preferred the combined approach. Figure 5.4e shows the same data as in Figure 5.4c using color only. The user can turn the context on or off. The context use was very task specific, users sometimes preferred to see the context and sometimes they wanted no context.

Furthermore, the users are also considering the direction and magnitude of the lines. Inspired by the work of Turkay et al. [CPH12, TLLH13], we also show only the deviation lines. In this case all start points are moved to the origin, and the lines and the end points are shown. In an ideal case there would be no lines, again. With this setup, it is easy to brush all lines that have a certain slope and/or magnitude. Figure 5.4f shows such a visualization. Figure 5.5 shows two different areas, one where the model is more accurate (Figure 5.5a) and one where differences between the simulation and the regression are much bigger (Figure 5.5b). Figure 5.5b also shows the context.

5.6 Case Study

We use an example from the automotive industry to evaluate our proposed approach. Together with five domain experts we conducted several analysis sessions in order to optimize a common rail Diesel injection system. All of these experts have a long (10–20 years) experience in automotive simulation. The sessions were conducted at AVL [AVL15], one of the worldwide leaders in powertrain measurement and simulation systems.

Modern simulation software can be used to simulate injection systems and to help engineers to understand and tune injection parameters. Many phenomena can not be easily measured experimentally and the only way to get more information is through computational simulation. We used the AVL CruiseM simulation software to simulate a complete injection system [AVL15]. We simulate an injection system consisting of four injectors. The injectors are produced by only few manufacturers. Hence the engine manufacturers have to tune them according to their specific needs. The whole system, including the rail and high-pressure pipes, is also different for each engine. The simulation model has more than 200 elements. Each element in the simulation model has several control and state variables. In this case study we focus on 6 parameters and explore different simulation results.

The main principle of a common rail system is the use of a high-pressure rail, common to all cylinders. The high pressure in the rail is used to precisely inject fuel into the cylinders. Electronically controlled actuators open and close the injectors. Sometimes, in one cycle, the main injection is preceded by a pilot injection, or even several of them. All this happens at least several hundred times per second. A more detailed description of common rail injection is beyond the scope of this

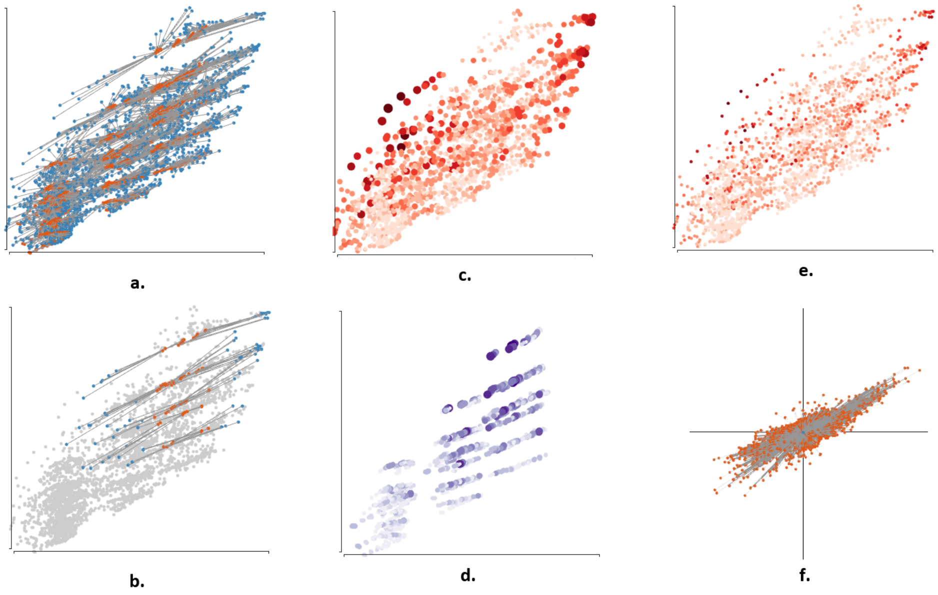


Figure 5.4: Regression Exploration View: A projection of the high-dimensional data points on a plane determined by two output values, X_max and Q_inj_max. The simulation results are shown in orange. The regression model results are shown in blue. **a)** Simulation points and corresponding regression model points are connected by a line to make the difference between them visually explicit. The view is cluttered due to the large number of points. **b)** Only brushed points are visualized. As there are much less points, clutter is also reduced. The context shown in gray can be turned off. **c)** and **d)** Alternative visualizations where only simulation points (**c**), or only regression points (**d**) are shown. The length of the lines (inverse accuracy of the model) is coded using color and point size at the same time. All points are shown (as in **c**). **e)** Only color is used to code the accuracy, the view **c** is preferred by domain experts. **f)** All lines are drawn from the origin which eases to brush points having a large deviation, or which deviate in a certain direction.

paper [BDHK05, BH97].

Due to high pressures and quick changes in the system, a modern common rail injection system operates in a condition which cannot be described sufficiently precise using classical fluid mechanics. Furthermore, in a common rail system each cylinder and injector is influenced by the others through the rail. This requires a careful rethinking of traditional system design. Figure 5.6 shows a modern injection system for a four-cylinder engine.

When tuning an injection system, engineers have a set of goals they have to meet. If the process values are not within a certain range, the engine will either run inefficiently or not at all. These results include mass injection rates, injection pres-

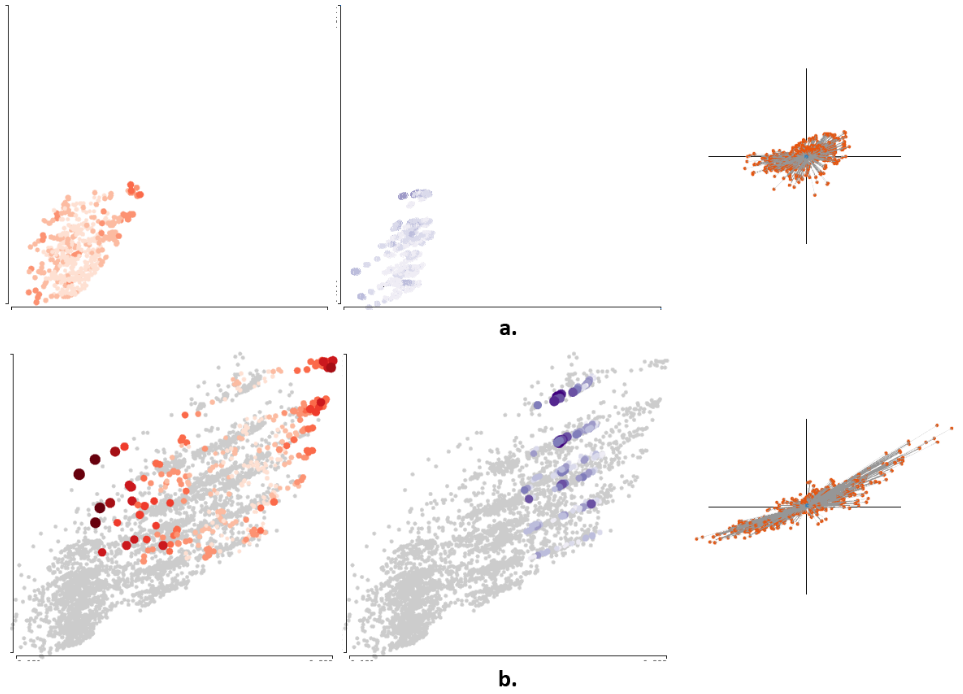


Figure 5.5: Two interesting areas are brushed. **a)** The regression model is quite accurate here. **b)** A much less accurate area is identified.

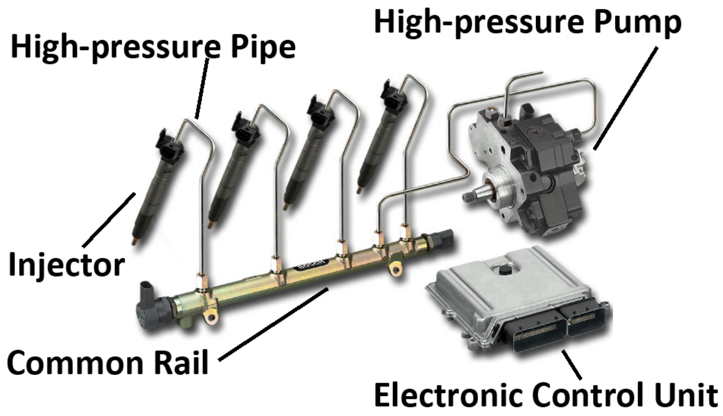


Figure 5.6: A modern common rail injection system used in Diesel engines of cars. There is one injector for each cylinder, and a common rail which utilizes high pressure (over 1500 bar) to precisely control the injectors' opening and closing.

sures (for each injector and the overall system), and the pilot and main injections time intervals. Corresponding control parameters for the desired ranges have to be explored (task A2, model reconstruction). A poor injection automatically leads to poor combustion, thereby increasing consumption, pollution, and power loss. An additional challenge in the design of a common rail injection system is to understand and prevent the appearance of unwanted pressure oscillations. Modern systems operate at a pressure of over 1500 bar. The oscillations can lead to amplitude jumps of over 400 bar. Such large oscillations can cause an undesirable behavior of individual injectors and introduce differences among the injectors. This results in a reduced efficiency and an increased emission — exactly the opposite of the design goals.

In this case study, we are interested in the high-pressure pipe geometry and the common rail itself. We have studied the geometry of the common rail and the high pressure pipes and the influence of the common-rail pressure and the pilot injection timings on the overall system performance.

There are four injectors with pilot and main injections, four high-pressure pipes and the common rail. We need advanced tools to comprehend the behavior of such a complex system. We assume that the individual injectors are tuned, and we study (for two pilot injections and the main injection) the injection pressure, the amount of injected fuel, the needle opening velocity and the needle closing velocity.

We are exploring the system at the individual cylinder level and at the overall system level. At the cylinder level we aim at the following:

- Small differences among injection pressures of the individual pilot injections.
- Small differences among the amounts of the injected fuel during the individual pilot injections.
- Maximum possible needle opening and closing velocities for two pilot injections and the main injection.
- Good damping of pressure oscillations that can occur within the high-pressure pipe.

On the overall system level we are looking for minimum possible differences among injection pressures for the individual cylinders and among the injected amounts of fuel for the individual cylinders. Besides these goals, the injection curves have to be of certain shapes, depending on the engine operation regime.

5.6.1 Iterative Analysis

After creating the model we need to decide which parameters will be varied. We focus on the six most relevant parameters: *L_line* and *D_line* (the length and diameter of the high pressure pipe), *V_rail* and *rail_pressure* (the volume of the rail and the pressure inside the rail), *V_inlet* (the volume of a junction between the rail and the high pressure pipe) and *pilot_start* (the starting time of the first pilot injection measured in degrees of crankshaft rotation).

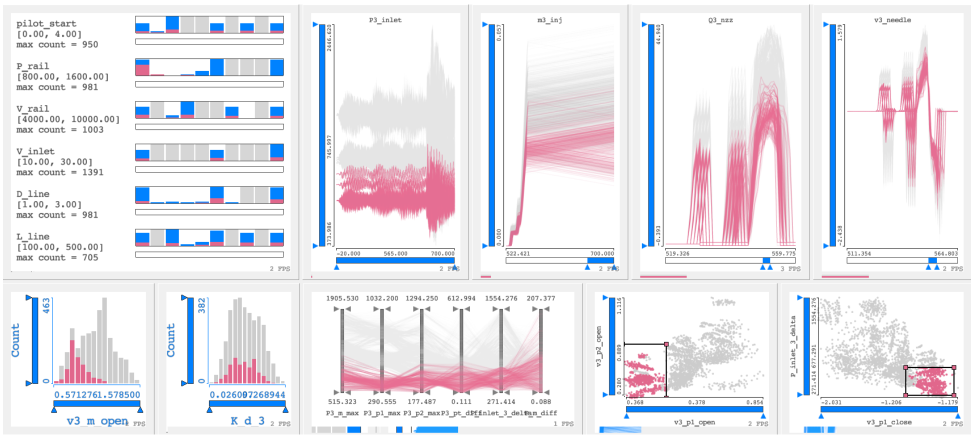


Figure 5.7: The initial view configuration (more than 2700 records). The Parameters Exploration View is shown in the upper left corner. The other views depict selected state parameters studied during the analysis.

We vary each control parameter and compute 2700 simulation runs. One simulation run takes approximately 200 milliseconds on a standard single core desktop PC. Several simulations can run simultaneously when multiple cores are available.

More than 30 output values (time series — functions of the crank angle) are computed per run. Once the simulation values are computed, more than 30 features per run are computed as well. Some of the features are computed from one curve (e.g., its maximum), and some are based on several curves. Table 5.2 shows a small subset of the computed scalar features.

All values of each run are computed for two revolutions of the crankshaft (720 degrees). The result is a complex data set where each record has some scalar attributes and some time series attributes (all state parameters). Since the computation of the regression model and the optimization expect scalar values, extracted scalar

Table 5.2: Scalar features of time series simulation values.

State parameter	Explanation
P_m_diff	Absolute difference between the maximum pressure during the main injection for the first and the third cylinder.
m_m_diff	Absolute difference between the injected fuel mass during the injection into the first and the third cylinder.
K_d_3	A damping value.
P3_pt_diff	Absolute difference between the maximum pressure for the first and the second pilot injection (third cylinder).
m3_pt_diff	Absolute difference between the injected fuel mass during the first and the second pilot injection (third cylinder).

features are used to approximate the model.

After the computation of the initial set of runs we started the data exploration and analysis. Not all of the runs generate feasible results. Some the high pressure parameter combinations result in a too low injected fuel mass, for example. Figure 5.7 shows the initial setup of our analysis. The Parameters Exploration View is placed in the upper left corner and it remained there during the whole session. The other views are often reconfigured, depending on the current analysis goals.

Three of the five users had experience with interactive visual analysis prior to this case study. One user reported after the study that *"Seeing all results at once, although some are not feasible, represents a great advantage. I can see results in context now, and not each run separately as up to now."* In order to explore how outputs change when control parameters change (task A1) we use brushing. By moving the brush across the parameter space all corresponding outputs are highlighted consistently. Another user commented *"Seeing how outputs change when I move the brush adds additional quality to conventional sensitivity analysis. I get a much better impression now."*

The model reconstruction task focuses on finding parameters which result in desired values for scalar outputs, and certain curve shapes for curve outputs. Curve shapes are crucial for an efficient injection. For model reconstruction (task A2), we simply select the desired values of the scalar outputs and the desired curve shapes. This is practically impossible to do analytically becomes very intuitive now. We brush desired shapes and undesired shapes as well. Some oscillations occur for some combinations of control parameters and it is important to understand when this happens. We also use multiple composite brushing. The user can select different subsets in different colors and interactively change any selection. This proved to be perfect for various comparisons (task A3), as stated by one user *"Comparing several scenarios is straightforward using multiple brushes. Similar comparisons are simply impossible using a conventional workflow where we analyze each run separately, and compare the results afterwards."*

We identify the area in the parameter space where a possible optimum could be. Automatic optimization is used to find the optimum instead of trying to regularly fill in the range with new design points (which would result in many additional simulation runs). The automatic optimization can be started from within the tool. The parameters view is used to specify which runs are included and the regression model is created. From now on all scalar values can be computed using the regression model. We specify the optimization constraints and compute the optimum based on the regression model. All these steps are performed in the integrated environment that was appreciated by a user: *"I could never set up optimization so fast. I also see all results together with the initial runs."*

We run the simulation (not the regression model) for the optimum point. The simulation tool is started automatically, a new run is computed, and the simulation results are loaded in the visualization tool. All outputs are computed by the simulation for the computed optimum point. Of course, these values differ from the values computed using the regression model. While the regression model is only an approximation we know it is precise enough to indicate the region of the optimum in the specified subspace. However, we are aware that the optimum is computed using an approximation, so we propose to compute additional points in the neighborhood of the optimum. For each additional point a simulation run is needed. The number of additional points can be specified dependent on model accuracy in the optimum's neighborhood. In this case we generate 244 new points around the optimum. The points are mostly in the constrained space, but we allow a slight deviation if the optimum is on a border. The points are generated using the full-factorial algorithm [Puk06].

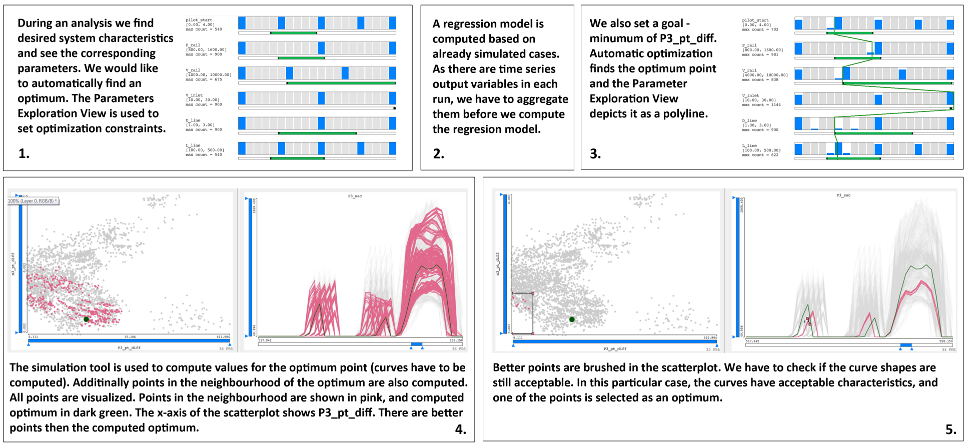
The computed optimum point is depicted in the Parameters Exploration View and in the other views. If better points exist, they are displayed and can be selected by the user. One user commented: *"The suggestions of where to refine the parameter space based on optimization speeds up the steering. The model accuracy shown indicates the quality of the optimization results. Seeing all runs all the time is simply unmatched in a conventional workflow. I can also see all curves which are normally not available when an optimization based on scalar features is conducted. I would estimate a speedup of at least an order of magnitude when designing complex systems."*

Figure 5.8 illustrates a typical workflow from an analysis session.

5.6.2 Regression Model Modification

The following example illustrates how the regression models influence the computed optimum. At one stage, we had 3188 simulation runs in the ensemble, as we had continuously added runs to the ensemble. We computed an optimum and additional points in the neighborhood. We wanted to minimize *P_m_diff*. The scatterplot in Figure 5.9a shows the scatterplot where *P_m_diff* is on the x axis. The computed minimum is dark green and all points from the optimum's neighborhood are pink. We clearly see that there are better points in the neighborhood. The same data in the parallel coordinates (Figure 5.9b) shows two clusters among the pink items on the first axis, *P_m_diff*. The optimum is rather high, and it seems as if the upper cluster is attracting the optimum.

It is possible that we have a high gradient which influences the regression model. A new regression model is computed based on a subset of the points only. The points with high *P_m_diff* values are excluded, and new optimum is computed. The scatterplot (Figure 5.9c) and parallel coordinates (Figure 5.9d) show the original optimum and the new optimum in light green. The new model fits much better and the new optimum is smaller (remember that we wanted to minimize *P_m_diff*). We also computed the optimum using a model based on even more neighboring points but the outcome was almost the same. The curve shapes for the new optimum are



During an analysis we find desired system characteristics and see the corresponding parameters. We would like to automatically find an optimum. The Parameters Exploration View is used to set optimization constraints.

1.

A regression model is computed based on already simulated cases. As there are time series output variables in each run, we have to aggregate them before we compute the regresion model.

2.

We also set a goal - minumum of P3_pt_diff. Automatic optimization finds the optimum point and the Parameter Exploration View depicts it as a polyline.

3.

The simulation tool is used to compute values for the optimum point (curves have to be computed). Additinally points in the neighbourhood of the optimum are also computed. All points are visualized. Points in the neighbourhood are shown in pink, and computed optimum in dark green. The x-axis of the scatterplot shows P3_pt_diff. There are better points then the computed optimum.

4.

Better points are brushed in the scatterplot. We have to check if the curve shapes are still acceptable. In this particular case, the curves have acceptable characteristics, and one of the points is selected as an optimum.

5.

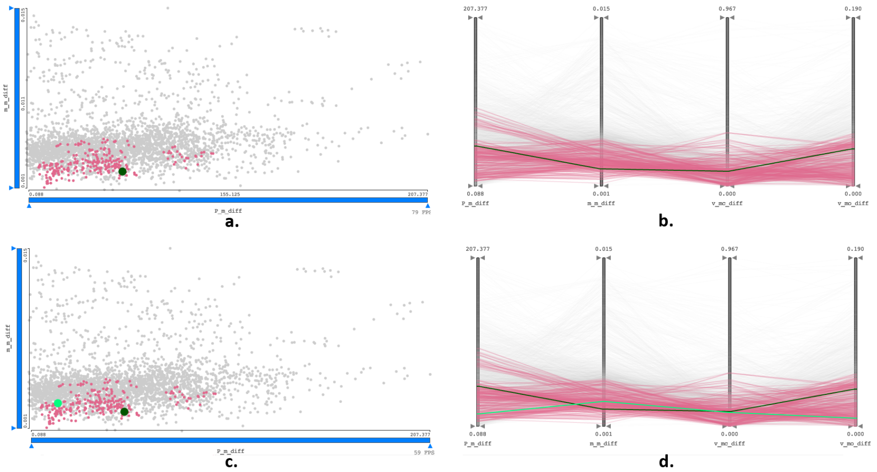
Figure 5.8: An example of a typical analysis iteration. **1.** The Parameters Exploration View is used to set the initial ranges of the control parameters. **2.** A regression model is computed based on the already computed simulation runs. Time series data are aggregated. **3.** The Parameters Exploration View shows the first computed optimum. The green poly-line connects the optimal control parameters. **4.** Left: the points in the neighborhood of the computed optimum are simulated and depicted in pink. Right: the corresponding curves. It is obvious that there are better points than the automatically suggested one. **5.** We additionally check if the curves have acceptable shapes. Left: the selected points and the optimum are highlighted. Right: the corresponding curves.

examined and verified by users: *"Hybrid steering makes us aware of automatic optimization limits. We can easily see if results are right or not, and we can quickly refine the model if needed. I used steering before, but the addition of automatic optimization improves it significantly."*

5.7 Discussion and Conclusions

The new Hybrid Visual Steering approach represents an integrated design environment for simulation, visualization, and optimization. The development of this approach would have been impossible without a close collaboration with domain experts through numerous sessions over several months. The improvements and time savings are significant when compared to the conventional approach. The integrated design environment manages complex data (no tedious file conversions), keeps track of the process and of all optima found during the process.

We illustrated the approach on the common rail injection system design but the approach is not limited to the injection design only. We talked to the domain experts working on different automotive systems (timing drive, crankshaft balancing, and



a. b. c. d.

Figure 5.9: **a)** A scatterplot showing the first optimum (*P_m_diff*), dark green. **b)** The parallel coordinates showing the first optimum, dark green. **c)** The scatterplot showing the first and the second optimums (*P_m_diff*), light green. **d)** The parallel coordinates showing the first and the second optimums, light green.

overall driving comfort) and they anticipate similar potential speed ups for their design problems. The approach can be applied whenever there is a complex system that can be represented by a simulation model with a multidimensional parameter space (either continuous or discrete), and relatively fast simulation. The simulation could be any algorithm that computes something based on control parameters. We are currently exploring applying the approach to image segmentation in the medical domain. We are also planning to extend the system for air flow simulation and traffic simulation.

The initial discussions we had with experts from all these domains make us confident that Hybrid Visual Steering can be applied in many scientific and engineering domains. Of course, the individual components of the system (simulation tool, regression model building tool, optimization, and visualization) should be modified according to the specific domain, but the main methodology and the workflow (see Section 5.4) remains the same. The modifications should be done in consultation with the domain experts as every domain has its own requirements, conventions and standards.

The interplay between the parameters of complex systems is so intricate that the expert's intuition and knowledge can not be represented by an automatic system. Hence it is important to have an interactive system. Only human experience, knowledge, and imagination, supported by automatic methods can yield the best-possible results.

In this paper we described our experiences with the injection system design. Initially, the common rail injection system design process was done using a number of isolated tools. It was necessary to create an integrated design environment (Figure 5.1) that supports simulation, visualization and optimization. The evaluation of the proposed approach by five domain experts demonstrates the viability of the proposed approach, advantages over the existing design practice, and its usefulness in everyday industrial design. The integrated design environment which was deployed in the context of the case study is currently used by AVL. The intent is to make it a standard part of AVL's commercially available software suite.

Acknowledgments

Part of this work was done in the scope of the K1 program at the VRVis Research Center.

6

Interactive Visual Analysis of Multiple Simulation Runs Using the Simulation Model View: Understanding and Tuning of an Electronic Unit Injector

Published as:

Krešimir Matković, Denis Gračanin, Mario Jelović,Andreas Ammer, Alan Lež, and Helwig Hauser:
Interactive Visual Analysis of Multiple Simulation Runs Using the Simulation Model View: Understanding and Tuning of an Electronic Unit Injector
IEEE Transactions on Visualization and Computer Graphics, 16(6):1449 - 1457, 2010.

Interactive Visual Analysis of Multiple Simulation Runs Using the Simulation Model View: Understanding and Tuning of an Electronic Unit Injector

KREŠIMIR MATKOVIĆ, DENIS GRAČANIN, MARIO JELOVIĆ,
ANDREAS AMMER, ALAN LEŽ, AND HELWIG HAUSER

Abstract

Multiple simulation runs using the same simulation model with different values of control parameters usually generate large data sets that capture the variational aspects of the behavior of the modeled and simulated phenomenon. We have identified a conceptual and visual gap between the simulation model behavior and the data set that makes data analysis more difficult than necessary. We propose a simulation model view that helps to bridge that gap by visually combining the simulation model description and the generated data. The simulation model view provides a visual outline of the simulation process and the corresponding simulation model. The view is integrated in a Coordinated Multiple Views (CMV) system. We use three levels of details to efficiently use the display area provided by the simulation model view. We collaborated with a domain expert and used the simulation model view on a problem in the automotive application domain, i.e., meeting the emission requirements for Diesel engines. One of the key components is a fuel injector unit so our goal was to understand and tune an electronic unit injector (EUI). We were mainly interested in understanding the model and how to tune it for three different operation modes: low emission, low consumption, and high power. Very positive feedback from the domain expert shows that the use of the simulation model view and the corresponding analysis procedures within a CMV system amount to an effective technique for interactive visual analysis of multiple simulation runs. We also developed new analysis procedures based on these results.

6.1 Introduction

The importance of computational simulation and simulation models in engineering cannot be overemphasized. The design and development of new products mostly follows the standard simulation workflow. First a model is developed for the phenomenon under consideration and then that model is used as a basis for simulation [BINN10].

The common characteristics shared by most of the models, simple or complex, is that they are created from basic building blocks with well defined behaviors specified by a set of control parameters. The state parameters in such a model show how a block behaves given the values of the control parameters. A simulation usually determines the values of the state parameters at different instances of time. The values of the state parameters are exchanged among the blocks in the simulation model.

The connections and dependencies among the blocks and the overall structure of the model determine the model's behavior. A visual representation of the model, usually in the form of a 2D graph, captures these dependencies and allows an engineer to have good understanding of the model.

A simulation produces a set of values for the state parameters (simulation results) that captures the behavior of the model (given the values of the control parameters). However, this connection between the model (and its visual representation) on one side and the numerical data (as produced by the simulation) on the other side is often missing from the analysis. The engineer can use various visualization techniques and data views to get an insight into the simulation results and relate those results to the underlying model. However, there is still a gap, both cognitive and visual, that needs to be closed (in the context of interactive visual analysis). To mitigate this problem, it is possible to integrate the simulation results (a single simulation run) within the display of the simulation model (e.g., as done in Simulink [Sim]) or by "anchoring" the information display on the system model [MHSG02].

When dealing with multiple simulation runs, the same model is used with varying values of the control parameters. Closing the gap in such a scenario presents an even greater visualization challenge and no solution has yet been proposed. There can be thousands or tens of thousands of runs that can generate a huge amount of complex data. We need a visualization and analysis solution than can cope with this challenge and bridge the gap between the model and the simulation results.

In this application paper, we propose a new view, the simulation model view, that provides an additional context for the simulation results to close the gap between the model and data for multiple simulation runs. The view provides a 2D graph where each node represents a building block of the simulation model. Each block has the control parameters that are used to tune the simulation and the state parameters that are determined through the simulation run. The values of both the control and state parameters are displayed directly within the node in the simulation model view.

If there are multiple simulation runs, the simulation model view blocks should show multiple values of the parameters. Moreover, the descriptive parameters are often time-dependent making the problem even more complex. The simulation model view is integrated in a coordinated multiple views (CMV) system. The benefits of multiple linked views and composite brushing facilitate the use of the simulation model view, especially when dealing with multiple simulation runs.

We have evaluated our approach in the context of an application from the automotive industry. Diesel engine powered heavy-duty trucks need to meet lower exhaust emission levels to comply with emission regulations [CMN$^+$04]. In addition, there are demands to increase the engine torque, the rated power output, and to reduce the engine fuel consumption. One of the key engine components that determines the emission levels and the engine performances is the Fuel Injection Equipment, more specifically the Electronic Unit Injector (EUI). Figure 6.1 shows a schematic of an advanced two-actuator EUI.

We modeled, simulated, and analyzed the Delphi E3 EUI [GTB03]. The Delphi E3 EUI (Figure 6.2, left) has two independent, fast response precision actuators that can change the injection pressure level and adjust fuel delivery timing and duration. This approach provides the unique ability to achieve full pressure control at low and high engine speeds.

The main goal of the evaluation was to understand the injector and to tune it for three different operation modes: low emission, low consumption, and high power. Several domain experts (one of them is a coauthor of this paper) used the proposed approach to analyze the effectiveness of the approach. The simulation model view, as a part of a CMV system, received a very positive feedback, and domain experts were able to do the analysis much more efficiently.

The remainder of the paper is organized as follows. Section 6.2 provides an overview of the related work. Section 6.3 describes the application domain. The new simulation model view is described and discussed in Section 6.4. Section 6.5 provides an illustration of the visual analysis using the simulation model view. Section 6.6 concludes the paper.

6.2 Related Work

Visualization of large, high-dimensional, and time-dependent data sets is an important, large, and very active area of research [Tuf01]. Large data sets need to be presented in a visual form and analysts need to interact with the data [Kei01]. Data visualization techniques should be well suited for the given data set, have limited visual overlap, be easy to learn, and recall. One goal is to reduce the cognitive load when performing analysis tasks while providing good integration with traditional techniques (including simulation) to improve the data exploration process.

A combination of different views, combined with advanced interactive brushing, supports iterative visual analysis by providing means to create complex, composite brushes [DGH03]. Those brushes span multiple views and they are constructed using different combination schemes.

The information mural view [JS98] provides a miniature version of the information space using visual attributes (gray-scale shading, intensity, color, and pixel size) and antialiased compression techniques. The view alleviates problems due to the limited number of pixels on the screen and the resulting information bandwidth constraints.

The table lens view [PR96, TR97] uses a focus+context (fisheye) technique for tabular information and displays important label information and multiple distal focal areas. The view is used for visualizing and making sense of large tables using a graphical mapping scheme for displaying table contents by fusing symbolic and graphical representations into a single, user customizable coherent view. The user can control the number of colors used and the color mapping.

6.2.1 Time-Dependent Data

Time-dependent data is a very important category of data sets. Brushing the time axis to display details of the selected time frame is one very common and useful interaction technique used with static representations. Müller and Schumann provide an overview (taxonomy) of the visualization methods for time-dependent data [MS03] and discuss general aspects of time-dependent data. The time factor requires a special treatment during visual exploration. They distinguish between two cases based on the time dependency of the visual representations, time-dependent (dynamic) and time-independent (static) representation. In addition, they discuss data versus event visualization and conventional versus multivariate display. Multivariate data visualization techniques include the *ThemeRiver*, *Spiral Graph* and several special visual metaphors such as *Calendar View*, *SpiraClock*, *Lexis Pencils* and others.

The *ThemeRiver* visualization [HHWN02] shows thematic changes in a large collection of documents in the context of a time line. The collection (time line, selected content, and thematic strength) is shown as a river (flow, composition, and changing width). Colored currents in the river represent individual themes.

Aigner et al. [AMM$^+$08] provide an overview of visual methods for analyzing time-oriented data and discuss general aspects of time-dependent data.

6.2.2 Application Domain

An advanced electronic unit injector with two electronically controlled valves can provide a very flexible choice of fuel injection characteristics. Single-cylinder engine tests have demonstrated the potential of such EUI systems for a heavy-duty diesel engine [GTB03].

The optimization of the Fuel Injection Equipment system is very important in order to understand the evolution of pressure during the injection event, the multiple injection interactions with injector and cam features [CMN$^+$04]. The modeling of the simulation of the Delphi E3 EUI shows how the full integration of the modeling work in the design process contributes to the understanding and the optimization of injection system features and its engine environment design.

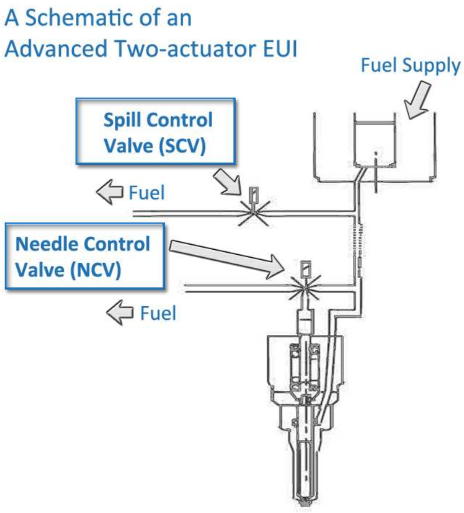

A Schematic of an
Advanced Two-actuator EUI
Fuel Supply
Spill Control Valve (SCV)
Fuel
Needle Control Valve (NCV)
Fuel

Figure 6.1: Controlling fuel injection with an EUI. SCV controls pressure gradient and level. NCV controls needle opening and closing. Both of them are electronically controlled.

6.3 The Electronic Unit Injector

Strict emission regulations and the need to make engines as efficient as possible represent two main constraints in automotive engine design today. Engineers work hard on improving existing engines in order to meet these constraints. An optimally tuned injection system is one of the key components of an efficient modern engine.

There are several different types of injection systems in cars and vehicles. Currently, the two most important ones for Diesel engines are common rail and unit injector systems. The common rail systems have a fuel pressurized to the injection pressure in a fuel rail which feeds the cylinder. The rail is common to all cylinders.

The unit injector systems have the high pressure fuel pump integrated with the injector. There is one injector/pump per cylinder that is installed into the engine cylinder-head assembly.

The predecessor of the modern unit injector system was the patent from 1911 which shows the working principle of the unit injector. It took a long period of time until the technology was advanced enough to enable reliable and cheap production of unit injectors. The production started with unit injectors for large locomotive engines and heavy duty engines. In 1998 unit injectors started appearing in passenger cars.

In this work, we analyzed the Delphi E3 Diesel EUI [CMN^{+}04, GTB03]. This is an advanced Diesel fuel injection system with two independent, fast-response precision actuators that can change the injection pressure level and adjust the fuel

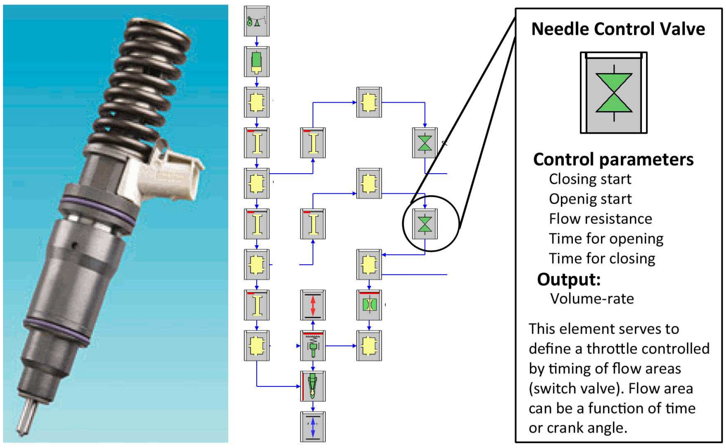

Needle Control Valve
Control parameters
Closing start
Openig start
Flow resistance
Time for opening
Time for closing
Output:
Volume-rate
This element serves to define a throttle controlled by timing of flow areas (switch valve). Flow area can be a function of time or crank angle.

Figure 6.2: The Delphi E3 Diesel EUI (left), the corresponding model view (middle), and a description of one of the NCV blocks from the model (right).

delivery timing and duration. This technology gives the unique ability to achieve full pressure control at low and high engine speeds.

The main parts of an EUI are: the nozzle, the needle with its return spring, the needle control valve (NCV), the spill control valve (SCV), the plunger, the plunger spring, and the electrical connector.

We first briefly describe the basic functionality of the unit injector. Figure 6.1 shows a basic schematic of the injector. The fuel comes into the pressure chamber. At the beginning of the pumping (used to increase the pressure) fuel escapes through the normally open SCV. The fuel can freely flow to the fuel gallery which is connected to the fuel tank.

When the electronically controlled SCV closes, the fuel pressure builds up in the system (and when it opens the fuel spills and the pressure drops).

The electronically controlled NCV controls whether the pumped fuel pressure is applied to the nozzle needle. The needle has a spring which pushes it down, and the fuel pressure can be used to support the spring and to apply much higher force to the needle (in the closing direction). The NCV allows the timing of the opening and the timing of the closing of the nozzle needle to be determined electronically.

If the NCV is activated throughout a period when the SCV is closed then the nozzle opens and closes according to the nozzle opening pressure and the nozzle closing pressure, set by the nozzle needle return spring only. This mode of operation and injection characteristic is the same as that produced by the single-actuator EUI system.

If the SCV is closed before the NCV is activated, then the fuel pressure can be pumped up to a much higher level before the NCV is then activated to allow the nozzle needle to open. Because the pressure applied to the needle (in the closing direction) is controlled by the SCV, a much higher pressure is needed to open the needle. If, towards the end of injection, the NCV is deactivated before the SCV is opened, the needle can be closed very fast with a high needle closing pressure.

The two EUI valves have the capability to precisely control multiple injection events. Accordingly, the needle opening pressure, the injected quantity, the hydraulic separation, and the needle closing pressure are dependent on the way the SCV and the NCV are activated. All of those events can be controlled for a pilot, main and a post injection. However, the discussion of these three types of injection is beyond the scope of this paper. The results presented in this paper apply to all types of injection.

6.4 The Interactive Model View

Every simulation begins with a model definition. There are different ways of how such a model can be defined, based on its complexity and the tools used. Many simulation tools allow to compose a model from basic building blocks. Each block has some control parameters and computes a couple of state variables. The blocks are connected and create the joint simulation model. The blocks exchange state parameters with their neighboring blocks (represented by connection lines).

Figure 6.2 (left) shows a real EUI while Figure 6.2 (middle) shows a part of the simulation model of that injector. The model was created using the AVL HYDSIM software [AVL15], a 1D CFD simulation tool. It is well suited for the modeling of injection systems, where all flow phenomena primarily occur along lines and valves (1D Flow). The calculations are very fast and the computed results are comparable with much more time-consuming 3D calculations (in case of injection systems).

The engineer carefully chooses the blocks and sets the value of the parameters so that the model represents a real injector as closely as possible. A lot of experience is needed in order to model real injectors (or any other complex device). Each block has several control parameters that can be set, which makes the overall system design very challenging.

Figure 6.2 (right) shows one of the NCV blocks and its parameters. There is an icon for each block type which helps the engineers to quickly identify a block. Based on the control parameters (determine the block's behavior and characteristics) and the defined model, the simulation software computes the values for the state parameters for each block. Those can be scalar values, but mostly they are time-dependent values, i.e., time series. As we analyze the simulated injection, we are interested in injection over time, usually over one cycle (one full crankshaft revolution [MGKH09]).

In the case of multiple simulation runs, the engineer defines the model, first, and then runs the simulation for various combinations of the control parameters. The amount of computed data increases drastically this way. We need advanced tools to support the engineers in the process of analyzing this data. The current state of the art approaches [KMG$^+$06, MGKH09] use an interactive visual analysis methodology and complex automatic optimization in order to understand the data.

We use a Coordinated Multiple Views (CMV) system that supports multiple, linked views. A user can choose from over a dozen predefined settings and views or configure all the views and set various parameters, e.g. point size or color, to better display the parameters.

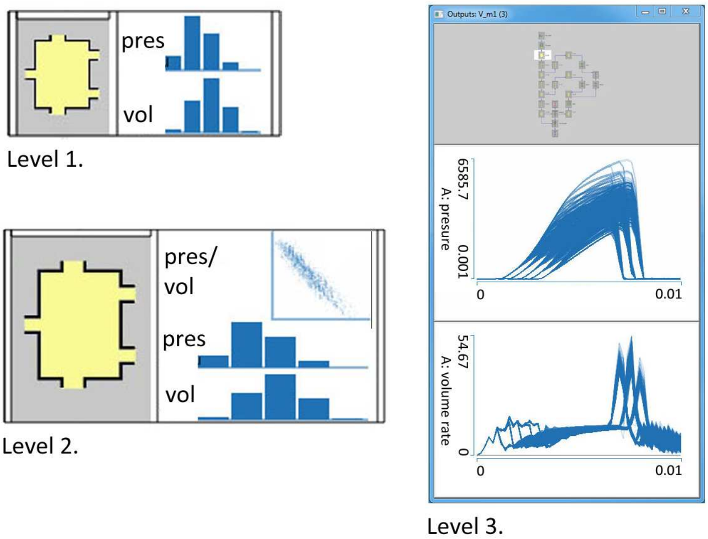
A time-dependent parameter across all simulation runs is called a family of curves. Our CMV system supports this data model and uses the curve view to display a family of curves. The curve view displays all curves in a family simultaneously using transparency to display the density of the curves. When combined with linked views and brushing techniques it can be used to display curves in focus and those forming the context.

Brushing is one of the essential features of every CMV systems [DGH03]. Our CMV system supports both single and multiple (iterative and composite) brushing. The composite brushing enables a user to combine several brushes using Boolean operations in a iterative way. The user selects a Boolean operation to be applied to the current selection and the new brush. The user can easily broaden (OR) or narrow (AND, SUB) the selection in an intuitive way. In the curve view we can use a line brush to select a subset of curves. Similarly we can use an angular brush to select a subset of slopes and the corresponding subset of curves [HLD02].

Although such an approach is of great help for the engineers, there is a problem that the analysis of the results is usually decoupled from the original model. The engineers use different, mostly also linked views to display the results, but there is no notion (except for the labels) of where the results come from (in terms of the corresponding blocks in the model). For example, if an output dimension is called “volume rate”, the engineer has to link it (mentally and by training) to the corresponding valve, nozzle, or orifice.

6.4.1 Blocks with Three Levels of Detail

In order to close the gap between the simulation model and the simulation data, we propose to integrate the simulation model view into a CMV system. We suggest to enhance the block icons so that they display control parameters and state variables from multiple simulation runs. As the available screen space is very limited, we propose a three levels of detail approach (Figure 6.3). We have decided to use the left side of the icon for the control parameters and right side for the state variables (Figure 6.4). We use the three different levels of detail to achieve a compromise between the amount of displayed information and the available space for each block.



Level 1.

Level 2.

Level 3.

Figure 6.3: Three levels of details for representing blocks in the simulation model view. The first level shows only histograms, the second level adds a 2D scatter plot and the third level shows curve views. The space required grows as the level increases, so the third level is shown in a separate floating view.

The First Level

The control parameters are usually scalar values that can be displayed using a simple histogram. Note that we have multiple runs, which means that the control parameters values can be varied between the simulation runs. The user wants to see which control parameters were varied. Each histogram bin displays the number of runs with a certain value of the respective control parameters. The bins can be equally high (if we run the same amount of runs for each value of control parameters) or they can differ. The constant parameters are simply shown as text. Due to a limited display size we show no more than three parameters at once. The user can easily select which three parameters should be shown.

The right side (showing the values of the block's state parameters) is more complex since state parameters usually have time-dependent values. This means that the results from a single simulation run are already time series. For multiple runs we then have a family of curves, one curve for each run. One possibility of displaying a family of curves is to use a curve view [KMG$^+$06]. However, due to a limited display size, it is not possible to show small curve views in a block.

We again use a histogram. We have to aggregate each curve in order to get a scalar value. We allow minimum, maximum, average, and integral aggregates for that operation. The user can select the desired aggregate type for each histogram. Just as on the control parameter side we allow up to three histograms. The user can select which state parameters are shown. Figure 6.3 shows an example for the first level view where two state parameters are shown, pressure (`press`) and volume rate (`vol`).

The Second Level

At the second level, we double the block size in both dimensions. We have more display area but still there is not enough space to display all curves. Aggregates are used at this level, as well. Since there is a little more space now, the user can choose to show up to twice as many (six) histograms. A smaller number of histograms are displayed in a higher resolution. This depends on the data, the number of parameters, and the task the user wants to solve. Figure 6.3 shows an example for the second level.

Besides histograms, a scatterplot of two state variables can also be shown. Interestingly, in the presented case the engineer never used this option. In general, we used scatterplots a lot, but we did not want them in the blocks. It was not interesting to see correlation of two state parameters in the same block. We used scatterplots, however, as separate views to compare related state parameters from different blocks.

The Third Level

Due to the limits in the available display area, it is not possible to increase the block size further. Instead, we introduce a new floating view that consists of a map of the model with the originating block and all state parameters displayed using the curve views in a vertical layout (optionally, the user might select a horizontal layout). Figure 6.3 shows the third level of detail. This is the highest level, i.e. all information is displayed and the user interacts with the curve views. Due to the size of the floating view it is impossible to integrate it in the model view directly.

Therefore, we provide a map as the first view which helps the user to relate it easily to the originating block. The map can be hidden if the user needs more space for the curves. The floating view label remains the only link to the block in this case. The number of state parameters shown are easily set by the user.

Figure 6.4 shows the interactive model view with the blocks showing their values. Some of the blocks are displayed using the first or the second level of details. The blocks used for analysis (Section 6.5) are labeled using capital letters **A** through **I**. The corresponding control parameters (red) and state parameters (blue) for the labeled blocks are shown on the right. The interactive model view is fully integrated in a CMV system. This means that the user will use other views (such as scatterplots, parallel coordinates, histograms, curve views, ...) to display selected parameters and to analyze them.

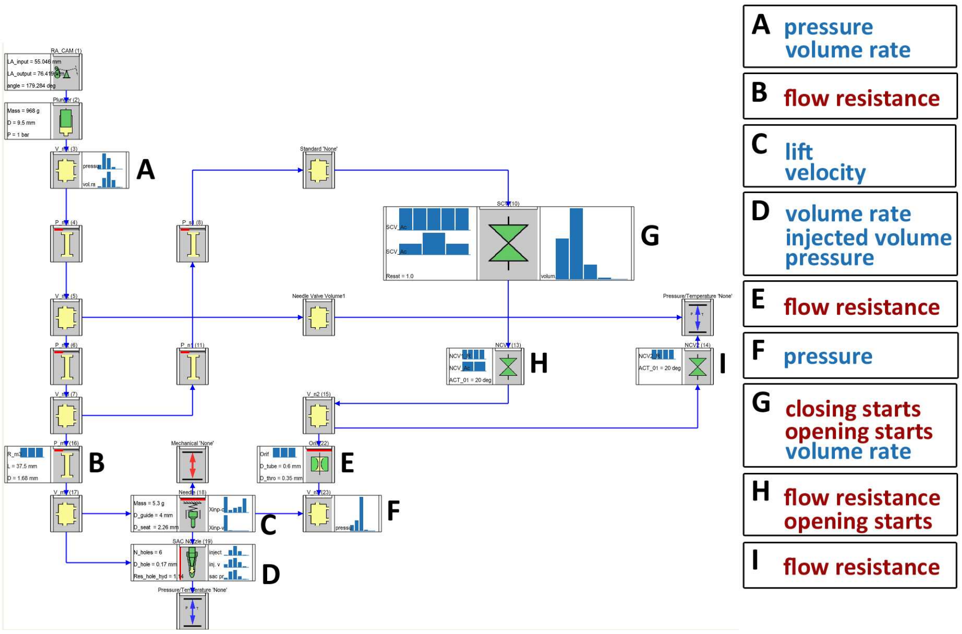


Figure 6.4: An example of the simulation model view. We can see all the blocks and their control (red) and state (blue) parameters. The first and second level of details are used.

The main idea of the CMV system is to identify some feature in one view (and brush something) and then highlight other parameters from the brushed records in other views, as well as in the enhanced blocks. Figure 6.5 shows a scatterplot where the user has brushed high average values of block **A** volume rate and low values of block **D** volume rate integral. The zoom into the model view shows the corresponding values of all control parameters and state parameters directly highlighted in the blocks. The integrated simulation model view helps the user to link the values to the blocks. This way the user better understands the model and related processes.

6.5 Interactive Visual Analysis of an Electronic Unit Injector

We now describe the use of the proposed approach in the analysis of an EUI. The analysis was done with a domain expert (one of the coauthors) from a company dealing with the development of simulation software and offering services in the car engine design support.

We focus on the shape of the injection curve as generated by an EUI. The mixing process of fuel and air heavily depends on the way the spray develops when injected into the cylinder. The engineers try to shape the injection rate curves in order to

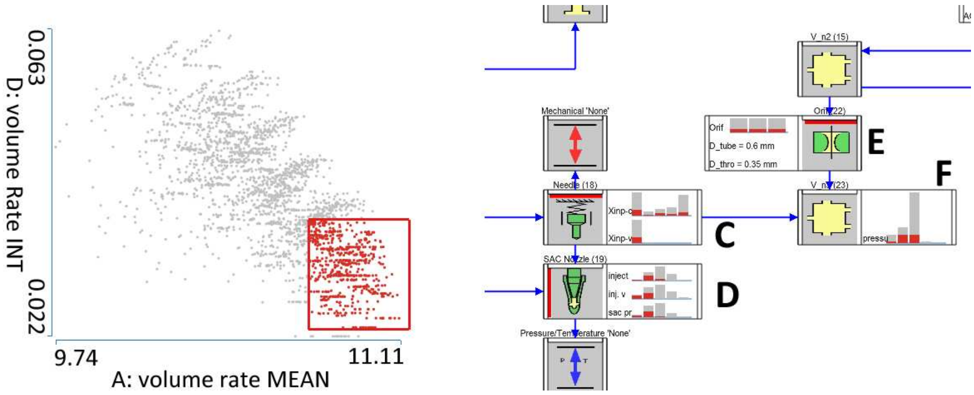


Figure 6.5: Whenever the user brushes an interesting subset of the data in any of the views, the simulation model view updates accordingly (being an integral part of the CMV system). Here we can see that the lower right part of the scatterplot (block **D** in Figure 6.4) has been brushed and all histograms reflect the brushed data in the simulation model view.

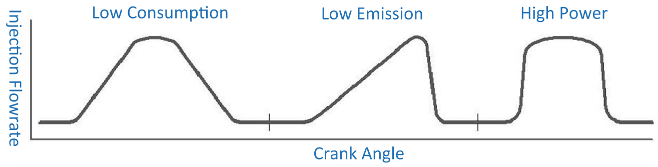


Figure 6.6: Ideal injection curves for the low consumption, low emission, and high power modes of operation.

achieve desired engine performance. The injector can be tuned for various goals. The three typical modes of operation are low consumption (fuel efficiency), low emission, and high power. Figure 6.6 shows characteristic injection shapes for these three modes for heavy-duty Diesel engines.

We are interested in how injector points can be adapted to enable the three different engine modes.

In order to understand the injector and to find out about opportunities for improving it with respect to the operation modes, we considered 4,320 simulation runs for different values of the control parameters. Due to some additional constraints introduced by the domain expert, we had a total of 2,880 simulation runs.

The simulation model view introduced in Section 6.4 was used. The domain expert was mostly interested in flow resistance tuning (this is the most influential parameter for the injector behavior) and we have varied the following flow resistance parameters: the flow resistance for blocks **B**, **E**, **H**. **I**. Furthermore we have varied three parameters controlled by the electronic control unit (ECU), the closing starts for block **G** and the opening starts for blocks **G** and **H**.

Table 6.1: Control parameters for the 4,320 possible simulation runs (2,880 simulation runs performed).

Block	Parameter	Values
H	flow resistance	1.0; 2.0; 3.0; 4.0
I	flow resistance	1.0; 2.0; 3.0; 4.0
E	flow resistance	1.0; 2.0; 3.0
B	flow resistance	1.0; 2.0; 3.0
G	closing starts	20; 25; 30; 35; 40
G	opening starts	-15; -20; -25
H	opening starts	-15; -20

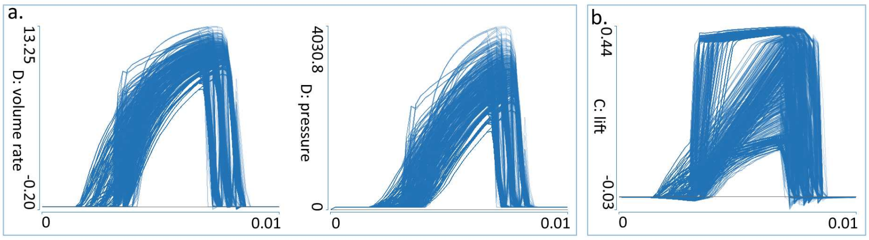


Figure 6.7: The third level details for block **D** (a) and block **C** (b). The curve views are used to interactively explore combinations of control parameters.

Table 6.1 shows the control parameters and values chosen for each of them. We will explore which factors facilitate or hinder the possibility of achieving the desired form of injection.

In general, the first stage of analysis is experimenting with the injector design and geometric properties. Once these properties are set, injectors are produced and they can not be changed any more. We are not dealing with the physical design of the injector, so it is used as is.

Our point of interest is on hydraulic flows through different parts of injector and correlations between them. During the second stage of the analysis (described in this paper), the designer fixes the geometry and explores the parameters controlled by ECU.

Table 6.2 shows the outputs (state parameter's values) that have been considered in the analysis. Note that all outputs are time-dependent, i.e., they are not single scalar values but rather functions of time. After we ran all 2,880 simulations we had a dataset consisting of 2,880 records. Each record has a set of independent scalar dimensions (Table 6.1) and nine dependent attributes which are time series (Table 6.2). Such a dataset follows a more complex data model than usual data models, where each record has scalar attributes only.

Our first task was to explore possibilities of designing an EUI suitable for high power. Such a scenario is typical when designing high-power special vehicles (military or heavy duty commercial trucks).

Table 6.2: State parameters used in the analysis.

Block	Parameter	Units
A	pressure	bar
A	volume rate	mm^3/deg
G	volume rate	mm^3/deg
C	lift	mm
C	velocity	m/s
D	volume rate	mm^3/deg
D	injected volume	mm^3
D	pressure	bar
F	pressure	bar

6.5.1 The High Power Mode

If we want to achieve the high power mode of operation, the injection curve has to be shaped almost as a square (steep rise and steep decrease), as shown in Figure 6.6. Additionally, the injection pressure must be as high as possible in order to inject a sufficient amount of fuel.

We start with the simulation model view (Figure 6.4). It shows the model with all the blocks and their control parameters and state parametersÕ aggregates on the right.

We are interested to see state parameters values in two blocks, **C** and **D**. We select the third level for the blocks and the curve views are configured. Figure 6.7 shows only the curve views of interest.

We explore different shapes and try to understand which control parameters combination can produce the desired behavior. We start with the beginning of the injection (needle opening — the point where the injections curves start to rise) and do not analyze the closing (the part where injection curves fall) at the moment.

As stated before, we want very high injection gradient (steep curves) at high injection pressure. High pressure will cause more fuel to be sprayed into the combustion chamber. We brush the curves using the line brush and refine the selection by limiting the crossing angle (Figure 6.8a). In this way we select only curves which have a fast needle opening.

At the same time, we are interested in cases where injection pressure is high at the beginning of injection. High pressure will cause a stronger penetration of spray into combustion chamber — a desired characteristic of the high power mode. In order to refine the selection, we combined the previous selection with a new brush on the pressure curves (Figure 6.8b).

Note that we have some slowly increasing curves in the injection rate view. We exclude them (Figure 6.8c) using a difference brush.

The interactive simulation model view is visible all the time and corresponding control parameters and values of state variables are highlighted during the analysis.

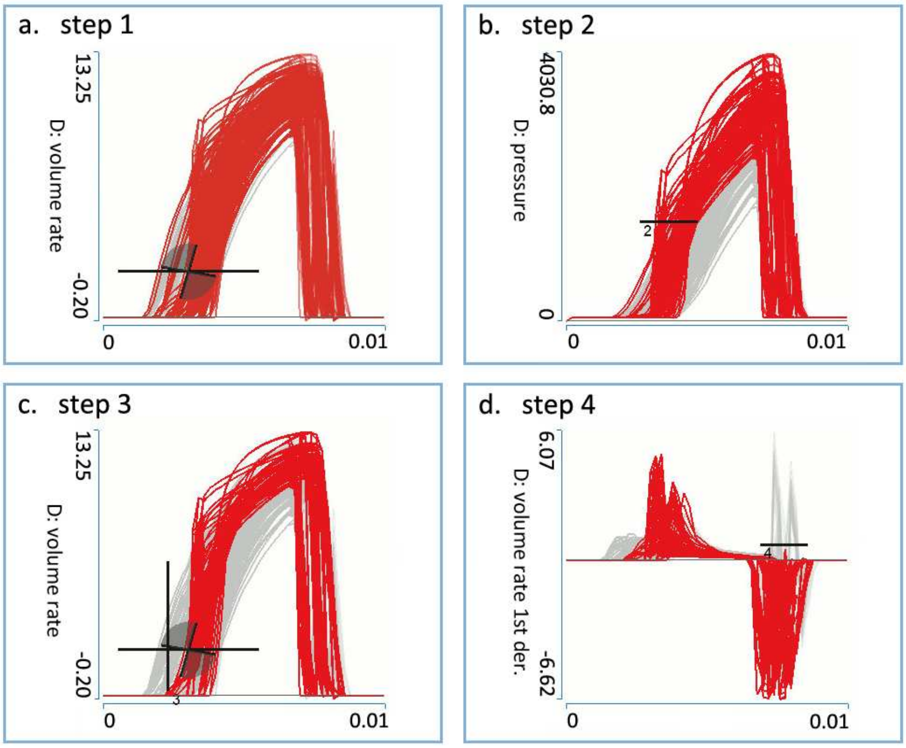


Figure 6.8: Searching for the cases with high injection gradient at high pressures (injection rate at opening has steep raise) in four steps — a) step 1: select by limiting the crossing angle on injection rate view; b) step 2: refine the selection with a new brush on the pressure curves; c) step 3: exclude slowly increasing curves by using a difference brush; and d) step 4: exclude the second needle opening by using a difference brush.

Before analyzing the control parameters causing the desired shape, we want to make sure there is no second needle opening. This is a phenomenon that happens sometimes. The needle is opened once more at the end which leads to an unwanted, uncontrolled subsequent injection. This process must be avoided because it leads to the rapid deterioration of the quality of the combustion process inside the cylinder. In order to examine such cases (not easily visible in the curve view) we used the first derivative of the injected rate curves.

Figure 6.9 (left) shows the curve views. We can see positive derivative (the left part of the curve view), and negative and positive derivative on the right side of the curve view. However, there are also positive derivatives, i.e., needle openings (the right part of the view). It is the second needle opening, which is not controlled, and has to be avoided. We can brush the unwanted cases (Figure 6.9).

We know that the moment, when an unwanted behavior happens, is the moment

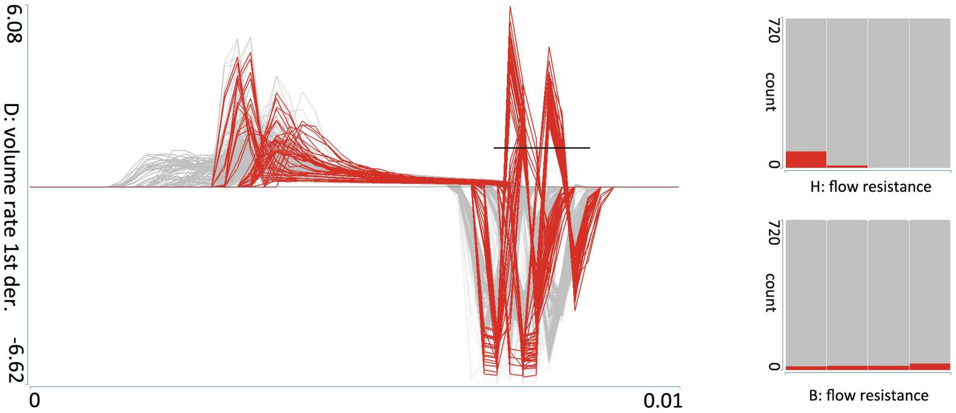


Figure 6.9: Eliminating the curves with unwanted, uncontrolled subsequent injection in block **D** using a visualization of the first derivative. The histograms for flow resistance in blocks **B** and **H** are shown.

when block **G** (SCV) starts to open but is not fully open and blocks **H** and **I** (NCVs) are already fully open.

Additional investigation of the control parameters that are controlling hydraulic behavior of the system shows that flow resistance in block **H** is small and flow resistance in block **B** is high in the runs where secondary needle opening occurs ((Figure 6.9 right).

Why does the second needle opening happen? Blocks **H** and **I** (NCVs) are suddenly open and their flow resistance is small. On the other side of valve is low pressure and pressure near the valve drops quickly (Figure 6.10). That causes a shock wave to propagate back through the system toward the other side (red line) to block **G** (SCV). Block **G** is not fully open and nozzle inlet has higher flow resistance. Shock waves hit these “hydraulic barriers” and reflect back toward NCV block **H** (blue line).

There is again a small resistance toward the control volume. The reflected wave will result in a pressure drop in the control volume and the needle jumps up. The interactive simulation model view is especially useful in analysis of such complex phenomenon where experts have to understand many states of different blocks connected in a certain way.

Once we have analyzed the unwanted behavior we can continue with the original analysis. We will use the difference brush to subtract the unwanted cases from the last stage of the analysis (Figure 6.8d).

We see that parameters with the dominant influence on the behavior of the system in order to achieve a square shape of injection rate are: the flow resistance within block **E** (orifice), which must be small, and the flow resistance through the passage towards volume above the needle (block **B**) which must be small, too. Less

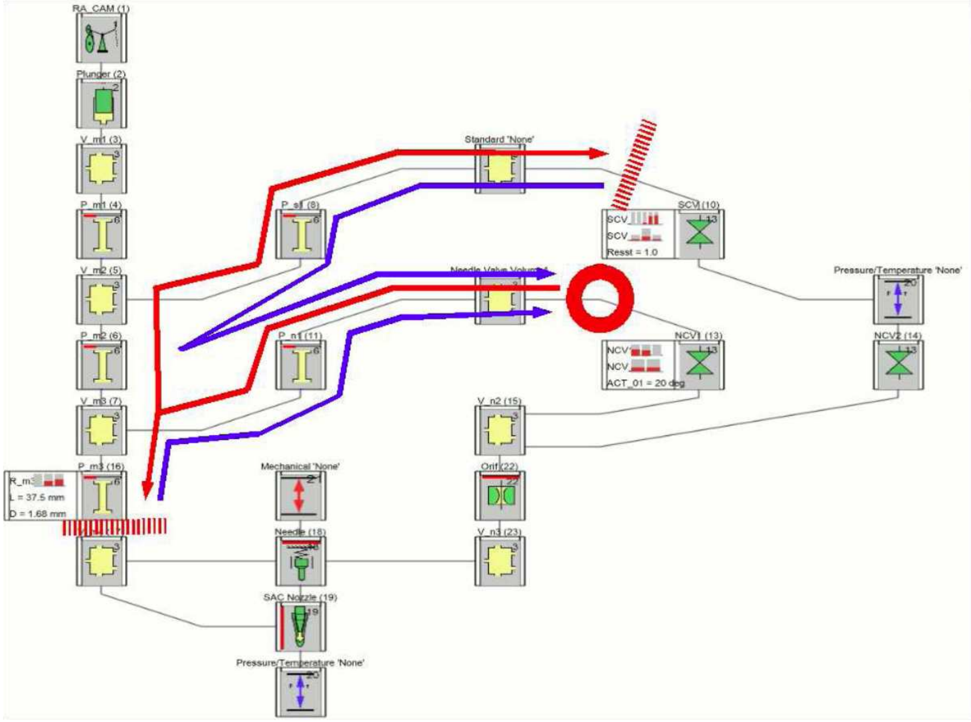


Figure 6.10: A shock wave propagates back through the system toward the other side (red line) to block **G** (SCV). Block **G** is not fully open and nozzle inlet has higher flow resistance. Shock waves hit these “hydraulic barriers” and reflect back toward NCV block **H** (blue line).

resistance at the nozzle enables a more free and rapid injection start without losses (Figure 6.11).

On the other hand, the pressure above the needle in the control volume will be higher, because there is no damping between the rest of the system and the control volume caused through orifice. When the needle control valve opens, the pressure in the control volume drops faster again, because of less resistance in the orifice. Faster pressure collapse in the control volume will result in a faster needle opening and a faster injection (closer to the square shape).

Besides square shape and high pressure we want a high amount of injected fuel, as well. We focused on block **D** next. We have selected the volume rate through the nozzle, the cumulative volume flow through the nozzle, and pressure in the block **D** as state parameters for this block (Figure 6.12, right top).

The aggregates of these state variables are displayed in the model view using histograms (Figure 6.13). The second histogram shows the distribution of the amount of the injected fuel (this is the maximum aggregate) which corresponds to the total amount of fuel injected since injected fuel output is computed as a cu-

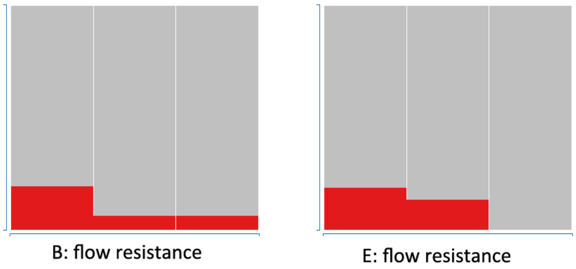


Figure 6.11: We can see that the highest block **E** flow resistance values are not allowed if we want maximum power and that there are more combinations having lower **B** and block **E** flow resistance values.

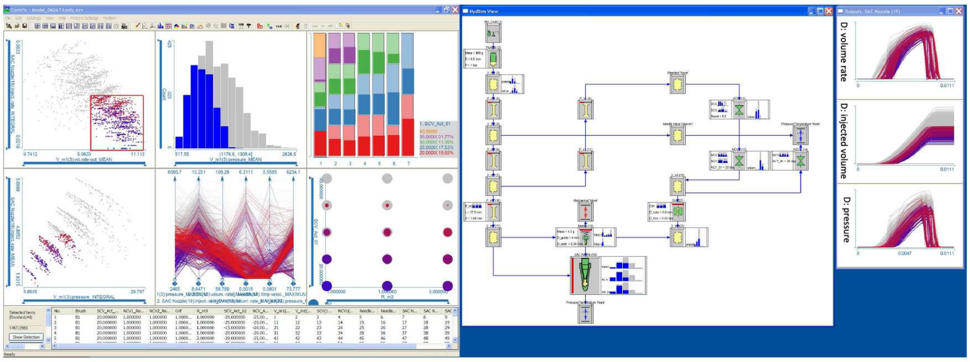


Figure 6.12: A snapshot from an interactive visual analysis session with the CMV system as used in this study. On the left, six linked "standard" views are shown with a brush applied to the scatterplot in the upper left. In the middle, the simulation model view is shown with linked histograms, reflecting the same selection. On the right, details for **D** block are shown (as third level of detail view).

mulative value. This is visible when we open the third level and see the curves themselves. If we brush the injected rates now, we can see that they are correlated to the closing start at block **G**.

The second histogram also shows that for the most cases that provide a square shape, the amount of the injected fuel is more or less average. Square shapes of curves are primarily between injection sets in the middle part of the pressure generation curve. It must be so, because for a square shape it is necessary to start with a higher pressure, which causes delay in the start of the injection. The injection is shorter and maximum values of injected amounts are not achieved. The high fuel amount criterion is not fulfilled, but the shape of the curve and high injection pressure are achieved.

If we select all desirable simulation runs (a square injection curve, high pressure and a significant amount of injected fuel), we reduce our data set to 147 simulation

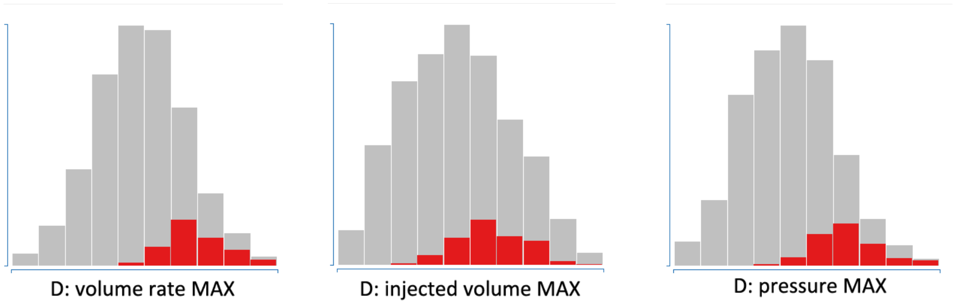


Figure 6.13: The second histogram in block **D** summary shows that for most cases that provide square shapes, the amount of the injected fuel is more or less average.

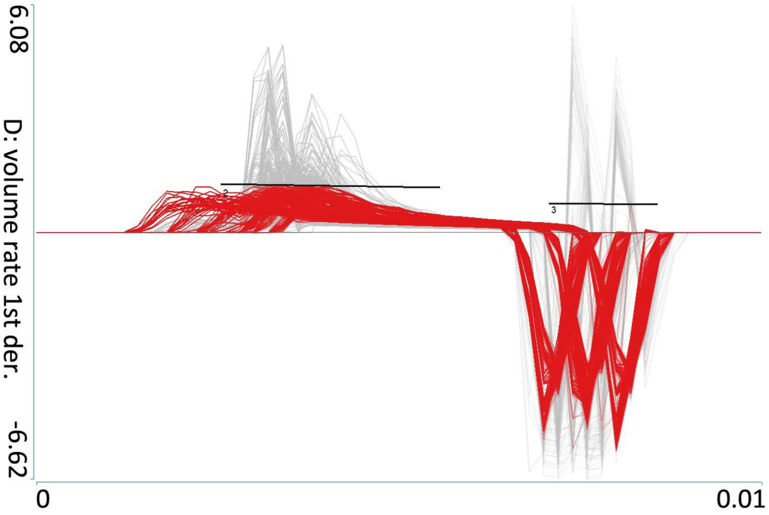


Figure 6.14: Using the first derivative of block **D** volume rate, we subtract unwanted curves: those with steep rising and with uncontrolled second needle opening.

runs (records).

The described example shows us how the engineer starts from the model view, configures the third level views for the analyzed blocks and then selects the curves having the desired shape. Several other views, configured independently from the simulation model view, were used as well.

Figure 6.12 shows a snapshot of the complete CMV system during the analysis. Some interesting findings (the second needle opening is a nice example for an unexpected finding — *detect the expected and discover the unexpected* [TC05, TC06]) illustrate how interactive visual analysis makes it possible to gain a deeper understanding by supporting additional, non-planned exploration.

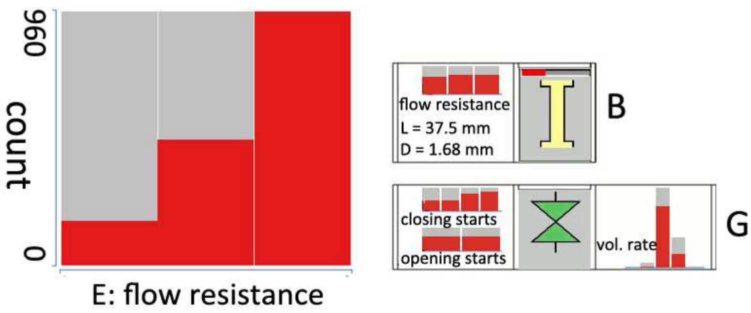


Figure 6.15: For a high-power low-emission engine a compromise is needed. Higher block **E** (orifice) flow resistance is welcome but this is in contradiction with the regime of maximum power when as low as possible resistance at this position is needed. Block **B** flow resistance should be low in the high power case and it does not influence the low-emission shape. All values of **B** flow resistance are almost equally probable for the low-emission shape. Block **G** closing starts and opening starts control parameters values make it possible to achieve a sharp or a ramped termination of the volume rate.

6.5.2 The Low Emission Mode

The emission regulations are becoming stricter every day. Heavy-duty Diesel engines have to meet very strong emission criteria in the near future. The fuel injection system together with the pressure charging, the cooling system, the exhaust after-treatment and other engine subsystems play a key role in achieving low emission.

The fuel injection system has to offer a range of different improvements in areas of flexible injection characteristics, e.g., a multiple injection, high injection pressures and different shapes of injection rates for every regime of operation. The high requirements for heavy-duty engines must be achieved without compromising their current performance and fuel economy. We focus on the emission-reducing capability of a prototype injector in this section.

We start with the model view again, and use it to configure a curve view in the CMV system. We want to achieve a different injection profile now, a ramp injection shape for the low emission mode of operation (Figure 6.6). We use the first derivative of the injection to subtract the unwanted curves: curves with steep rising (high derivative) and curves with uncontrolled second needle opening (undesired behavior). The large number of cases following this shape did not come as a surprise (Figure 6.14). Due to the physics of the vents and basic injector geometry, the ramp shape is the most natural shape for the unit injector [CMN^{+}04, GTB03].

The model view shows us that the damping at the entrance to the control volume above the needle (block **E**) has a significant impact on the adaptability of the injector to the regime of low emission. Higher flow resistance is welcome but this is in

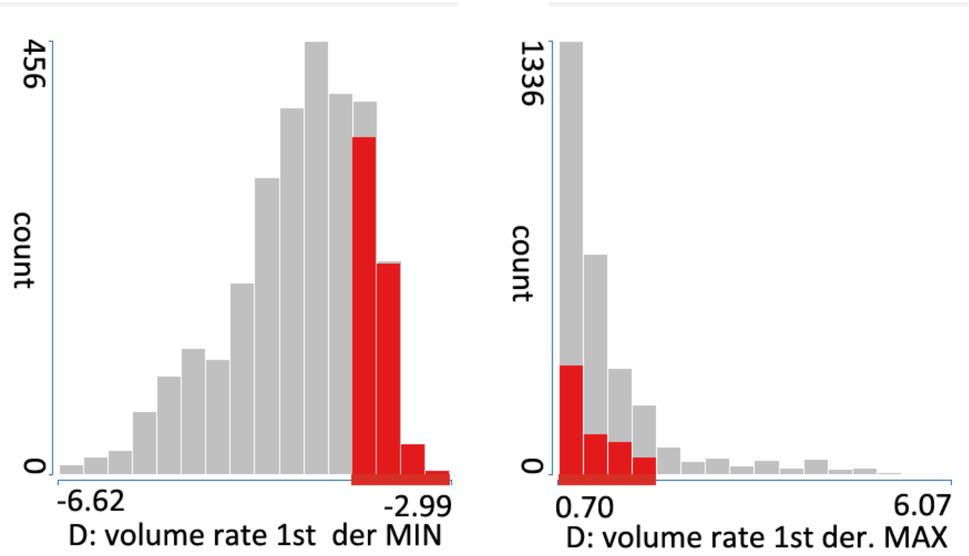


Figure 6.16: For the desired shape of the injection curve both gradients of volume rate during needle opening (right) and during the needle closing (left) must be near zero.

a contradiction with the regime of the maximum power when we need as low as possible flow resistance at this position. Obviously, a compromise is needed if a high-power low-emission engine is a goal (Figure 6.15).

Let us examine block **B** flow resistance now. Remember it was preferably low in the high power case. We have better luck now, flow resistance does not influence the low emission shape. All values of flow resistance are almost equally probable for all cases with the low emission shape.

We have also analyzed an additional point in the model. We are again interested in checking if it would be possible to tune the engine for low emission and high power. A slow increase of flow resistance in blocks **H** and **I** (NCVs) will result in a better behavior in case of low emission mode but again, this increase is in a contradiction with the high power mode where this control parameter has to be slightly decreased. The domain expert has to find a compromise.

6.5.3 The Low Consumption Mode

Direct injection systems for Diesel engines must deliver high performance and the maximum torque while keeping fuel consumption low. The desired injection rate shape curve for the low consumption mode of operation is shown in Figure 6.6.

There are several approaches how to meet the requirement on the injection curve shape. One possibility is to tune the signals sent to blocks **G**, **H**, and **I**. The ECU controls the valves and sends these signals. Different settings of block **G** (SCV)

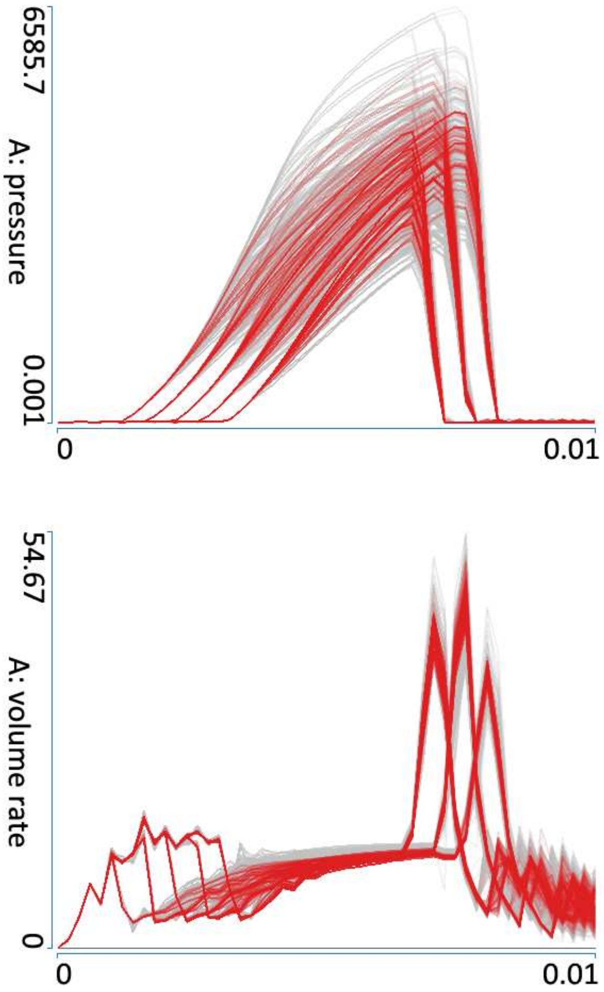


Figure 6.17: **A** (pump chamber) block third level. The low consumption regime has all basic characteristics with average values and no extremes.

with a constant settings for blocks **H** and **I** (NCVs) may be used at the end of the injection to produce different injection pressure levels during the needle closing. This makes it possible to achieve a sharp or a ramped termination of the volume rate.

Another approach is to see if it can be achieved with altering other control parameters. If we want to have the desired shape of injection curve, the gradients of block **D** volume rate curves during needle opening (Figure 6.16 right) and during the needle closing have to be near zero (Figure 6.16 left). As a consequence, the range of values for the volume rate derivation is narrow (approximately between -4 and 2) and the changes in the volume rate are relatively small.

Once we have selected these cases, we can explore the control parameters of the selected simulation runs using brushing (Figure 6.17).

The large view of block **A** (pump chamber) in Figure 6.17 shows us that the low consumption regime is some kind of a middle regime. It means that all basic characteristics (like pressure) have middle values (no extreme or limit values). The high power and the low emission modes of operation (Figure 6.6) are two extreme regimes and the low consumption mode of operation is somewhere in the middle. The low consumption curve shape can also be considered as a combination of the high power and low emission curve shapes.

6.5.4 Discussion

These three scenarios of interactive visual analysis performed by the domain expert provide a basis for an initial evaluation of the proposed approach. From the domain expert point of view, the challenge is how to understand a complex simulation model with lot of complex interactions among simulation blocks. To achieve that there must be full interaction between the creation of the simulation model, simulation runs and investigation of the simulation results. The process must be fast and transparent.

The advantages of our approach that the model, simulation and results are all integrated within a single application (the CMV system) that provides visual understanding of the simulation process. This, in turns, allows the expert to less effort find an optimal solution with full understanding of the design. When dealing with multiple simulation runs, the CMV system helps the domain expert to close the gap between the model and data (Section 1), something that has not been available before. Possible improvements include integration of other advanced optimization tools, better support for high-resolution displays, and support for design and analysis of 3D geometry models.

6.6 Conclusion

Using multiple simulation runs helps engineers to gain a deep insight into the simulated phenomenon. As the model complexity grows it becomes impossible to mentally link the simulation results with the originating blocks of the model. We have integrated a model view into a CMV system and made it possible for engineers to quickly get an overview of the control and state parameters in the model itself. As the space in the model view is very limited we propose a three levels of detail approach where higher levels show more information but require more display area.

The newly introduced interactive simulation model view is fully integrated in the CMV system and a selection in any of the views highlight the information displayed in the blocks. We have illustrated the usefulness of proposed approach in a case study on understanding and tuning an EUI for Diesel engines.

A very positive feedback from the domain expert in the Diesel engine simulation domain (who is also a coauthor of this paper) indicates that such an approach would be useful for other domains as well. Every simulation starts with a model definition and the possibility to show results from multiple runs within the simulation model blocks helps the experts to analyze and understand the underlying system much more efficiently.

Acknowledgments

The simulation data is courtesy of AVL List GmbH (`http://www.avl.com`). Part of this work was done in the scope of the SemSeg project and VSOE VCV program at the VRVis Research Center (`http://www.VRVis.at`) and the research at the Center for HCI at Virginia Tech (`http://www.hci.vt.edu`). The project SemSeg acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number 226042.

7

Interactive Visual Analysis of Complex Scientific Data as Families of Data Surfaces

Published as:

Krešimir Matković, Denis Gračanin, Borislav Klarin, and Helwig Hauser:
Interactive Visual Analysis of Complex Scientific Data as Families of Data Surfaces
IEEE Transactions on Visualization and Computer Graphics, 15(6):1351-1358, 2009.

Interactive Visual Analysis of Complex Scientific Data as Families of Data Surfaces

KREŠIMIR MATKOVIĆ, DENIS GRAČANIN, BORISLAV KLARIN,
AND HELWIG HAUSER

Abstract

The widespread use of computational simulation in science and engineering provides challenging research opportunities. Multiple independent variables are considered and large and complex data are computed, especially in the case of multi-run simulation. Classical visualization techniques deal well with 2D or 3D data and also with time-dependent data. Additional independent dimensions, however, provide interesting new challenges. We present an advanced visual analysis approach that enables a thorough investigation of *families of data surfaces*, i.e., datasets, with respect to pairs of independent dimensions. While it is almost trivial to visualize one such data surface, the visual exploration and analysis of many such data surfaces is a grand challenge, stressing the users' perception and cognition. We propose an approach that integrates projections and aggregations of the data surfaces at different levels (one scalar aggregate per surface, a 1D profile per surface, or the surface as such). We demonstrate the necessity for a flexible visual analysis system that integrates many different (linked) views for making sense of this highly complex data. To demonstrate its usefulness, we exemplify our approach in the context of a meteorological multi-run simulation data case and in the context of the engineering domain, where our collaborators are working with the simulation of elastohydrodynamic (EHD) lubrication bearing in the automotive industry.

7.1 Introduction

Simulation is used in science and engineering to study a wide range of problems and to understand underlying models and investigated phenomena. Interactive visual analysis helps professionals to explore simulation results and to understand and explain the data. The readily available computing power also allows to investigate multiple simulation runs for a given case scenario. The input parameters (independent variables) are varied and the values of the output parameters (dependent variables) are computed for each combination of the input parameters. The resulting collection of simulation runs needs to be analyzed by exploring individual simulations and how they relate, i.e., what are the emerging patterns.

One important motivation for the study of multi-run simulation data is to perform a sensitivity analysis of the computation. To do so, new ways of visual data exploration and analysis are needed — data from multiple simulation runs are very complex to study (compared to conventional simulations where just time and/or space are considered as independent variables). Traditional scientific data, and corresponding visualization and analysis methods, are usually tuned for this more traditional data model.

We present an advanced visual analysis approach that supports the analysis of families of data surfaces, i.e., datasets that are seen with respect to pairs of independent variables (dimensions). While it is straightforward and easy to visualize one such data surface, the visual exploration and analysis of a large number of such data surfaces poses significant stress to the user's perception and cognition. We propose an approach that carefully integrates different projections and aggregations of the data surfaces at three different levels (one scalar aggregate per surface, a 1D profile per surface, or the surface as such).

There is an increasing number of application domains, including industrial simulation and meteorology, for example, in which it becomes natural to automatically consider multiple simulation runs for analysis, and accordingly there is increased need for appropriate visualization solutions. Konyha et al. [KMG$^+$06] showed how to analyze data from multiple simulations of 1D CFD in an automotive injection system. They limited their approach to the investigation of families of curves, i.e., the consideration of the data with respect to one variable. A large number of important real-world problems can benefit from this approach. Other problems, however, have complex data sets that should be considered as families of surfaces. Examples include avalanche warning systems in mountains and the simulation/measurement of the seabed as used in modern tsunami warning systems.

In the following, we use one illustrative example, i.e., a study of a historic climate scenario, in the beginning in order to introduce the here proposed methodology. Although this meteorological example is not a real case study, it is useful as an easily understandable illustrative example. Later, we discuss a more detailed case study from the engineering domain, based on the simulation of elastohydrodynamic (EHD) lubrication bearing in the automotive industry, which stems from an actual inter-disciplinary collaboration amongst the authors of this paper. Along with these two cases, we demonstrate the necessity for a flexible system that allows to integrate a large number of different (linked) views for making sense of this complex data scenario and propose new interaction and analysis techniques which make it possible to deal well with such data.

7.2 Related Work

The body of literature about the visualization of large, high-dimensional, and time-dependent data sets is impressive and the field is still an area of active research [AMM$^+$08, BH07, Rob07, Tuf01].

The exploration of large data sets [Kei01] is based on the idea to present the data in a visual form that would allow analysts to interact with it. Data visualization techniques, when suited for the given data set, reduce the cognitive load while performing analysis tasks. A visualization technique should have limited visual overlap, fast learning, and good recall [Kei01]. Furthermore, good integration with traditional techniques (including simulation) improves the data exploration process.

Time-dependent data is a very important category of data sets. Brushing the time axis to display details of the selected time frame is one very common and useful interaction technique used with static representations [HS04, KMG$^+$06, Mon90].

Aigner et al. [AMM$^+$08] provide an overview of visual methods for analyzing time-oriented data and discuss general aspects of time-dependent data. The time factor requires a special treatment during visual exploration. Two cases are distinguished based on the time dependence of the visual representations, time-dependent (dynamic) and time-independent (static) representations. The examples of the multivariate data visualization techniques for multi-variate time-dependent data include the *ThemeRiver* [HHWN02] and *Spiral Graph* [WAM01].

Time-dependent (serial) data often exhibit some periodic behavior. Such serial periodic data are of special interest. For example, time continues forward serially but include recurring periods (weeks, months, and years) [CK98]. The challenge is how to simultaneously display serial and periodic attributes of a data set.

All of these methods consider each dimension in a multi-dimensional space to be a scalar value (numeric, categorical, nominal, or the like). In the case of time-dependent data they handle it as an isolated case, or aggregate the data in order to get scalar values.

The challenges that result from a complex internal data structure can be tackled, for example, by the interactive visual analysis of families of curves [KMG$^+$06]. That approach provides analysis procedures and practical aspects of interactive visual analysis that are specific to this type of data. Multiple linked views combined with advanced interactive methodology support iterative visual analysis by providing means to create complex, composite brushes that span multiple views and that are constructed using different combination schemes and that respect the 1D data series (curves) as a data (sub-)structure [DGH03].

Time-independent representations are well explored but still have room for new innovations. Some of the recent findings include Lexis pencils [FP98] that map various time-dependent variables to the faces of a pencil. Another interesting approach uses extruded parallel coordinates, linking with wings, and three-dimensional parallel coordinates, integrated in a single rendering system, that visualize trajectories of higher-dimensional dynamical systems [WLG97].

Interaction techniques allow the user to better understand the data due to the ability to interact with the data. One of the well established techniques is Focus+Context (F+C) visualization [Hau06]. When the amount of data is too large to be displayed, the user should be able to focus on specific data sets of interest while keeping track of the context (entire dataset). There are four groups of F+C techniques: distortion-oriented, overview methods, filtering, and in-place techniques [BH07]. Focus+Context visualization is often used in an multiple linked view setup that supports linking and brushing.

The display of surfaces from volume data is standard in visualization [Lev88]. Surfaces can succinctly represent complex three- and multi-dimensional data. These kinds of surfaces are created from sampled scalar data in three spatial dimensions.

The user's perception and cognition of surfaces can be improved by designing perceptually near-optimal visualizations [War04]. Such a design is achieved by collecting perceptual characteristics of visualization methods, and exploring them to discover principles and insights to guide the design of visualizations [HBW06].

7.3 Illustrative Example and Proposed Methodology

We first describe an example from the climate research field to illustrate the challenges and then introduce the newly proposed technology.

7.3.1 A sample Analysis of Multi-run Climate Data

Meteorological data provide a prime example of a collection of long-term multi-dimensional data sets. The relevance and broader impact of the results gathered from meteorological data are tremendous. The time scale ranges from hourly and daily weather forecasts to long-term climate change. There are many efforts to facilitate collection, storage and exchange of meteorological and related environmental data.

One such effort is the Potsdam Institute for Climate Impact Research (PIK — `http://www.pik-potsdam.de/`), where researchers in the natural and social sciences work together to study global change and its impacts on ecological, economic, and social systems. A combination of data analysis, computer simulations, and models is used to study meteorological and related data. The researchers at PIK collaborate with researchers worldwide in order to be able to predict future climate changes. Relevant progress with respect to better predictions can only be made if the past is well understood. As part of this research they investigate – amongst other cases, of course – climate scenarios around several meltwater outbreak events

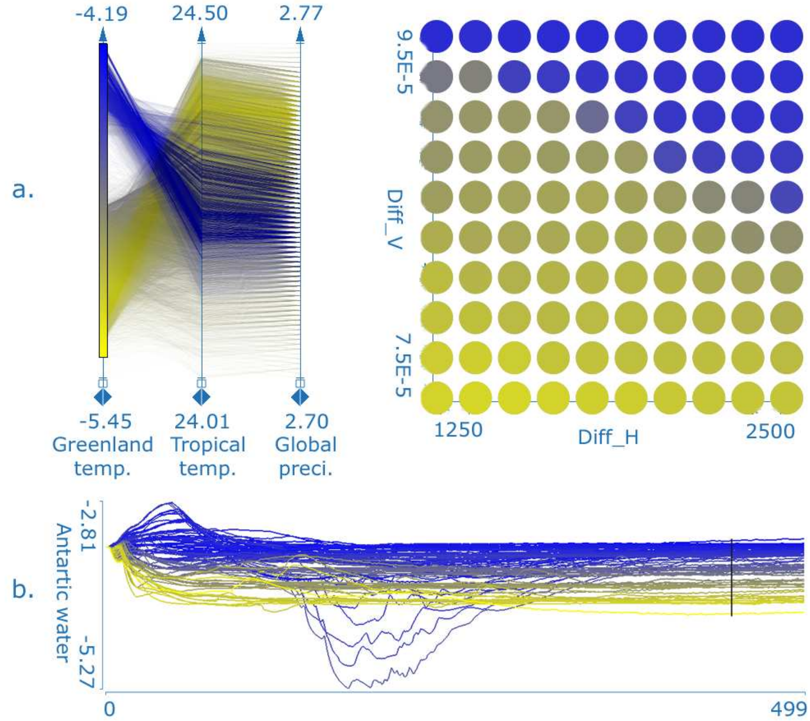


Figure 7.1: *a*. Two linked views, parallel coordinates depicting three simulated values and a scatterplot showing control parameters. A gradient brush over Greenland temperatures shows how it is related to $diff_v$ (Diff_V) and $diff_h$ (Diff_H) control parameters, and negative correlation between Greenland temperature and tropical temperature, and respectively with global precipitation. A plain table with 50000 rows was analyzed. *b*. A curve view showing the Arctic water parameter. Each curve represents 500 time steps of a simulation run with a particular set of control parameters. There are 100 curves depicted, one for each combination of $diff_v$ and $diff_h$

of proglacial Lake Agassiz, i.e., an immense glacial lake located in the center of North America and formed about 12,000 years ago [BG04a].

The investigated data describes the simulated climate response to one of these outbreak events. The more than 4,000 year long lifespan of this Lake Agassiz provides an exciting case for simulation and data analysis. Approximately 8,000 years ago, the lake drained due to climate warming and melting of the surrounding Laurentide Ice Sheet (LIS). The here investigated multi-run climate simulation is based on the PIK Climber 2.3 model and simulates a cooling of about 3.6 K over the North Atlantic induced by this meltwater pulse from Lake Agassiz, routed through the Hudson strait.

To get a better understanding of the variability of their climate simulations with respect to external parameters of the simulation, they compute multiple runs of the simulation with varied parameters. They varied two diffusivity parameters (one horizontal, diff_h, and one vertical, diff_v, both with respect to the ocean part of this model, 10 variations each), a total of 100 (10×10) runs. Each simulation run spans 500 years of annual data.

As a result, they aggregated 35 different values from the more detailed raw simulation data. The aggregates include CO_2 concentration, global surface air temperature, surface air temperatures for both hemispheres, land surface air temperature, ocean surface air temperature, Greenland temperature, global precipitation, ice areas for both hemispheres, salinity information, various differential measures (heat transport, ice transport), etc. For each of the 100 simulation runs all of these result values are available for 500 years (one time step per year).

The conventional approach would be to represent the data corresponding to one run as a table with 500 rows, each corresponding to one time step. There would be columns for the time step (an independent variable differentiating the rows in one simulation run) and output dimensions like surface air temperature. As we have 100 simulation runs, a large table consisting of $100 \times 500 = 50,000$ rows can be created. In this case, there are three independent parameters, diff_h, diff_v, and time. Such a large table can be explored and analyzed using coordinated multiple views so that experts gain deeper understanding. Some interesting relations between different climate descriptors can be seen instantly, e.g., that the average Greenland temperature is negatively correlated with the average temperature in the tropical regions as depicted in Figure 7.1a. There is also the strong correlation between Greenland temperature and simulation control parameter diff_v.

If analysts are interested in the development of various results over time such an approach would be complicated to use (since coherency is lost in the sense of which data rows belong to which time step). All timesteps belonging to the same set of control parameters should also show up in the visualization coherently. Konyha et al. [KMG$^+$06] showed how an advanced data model with an explicit support of time series in the data can be exploited for advanced visual analysis. In our case, this would mean to merge all 500 rows from one simulation run into one row containing time series in dependant dimensions. There would be 100 rows in such a table, but each would contain many time series dimensions as advanced data types. Figure 7.1b shows an example of 100 time series (one per simulation run), representing the Arctic water parameter, visualized using a function graphs view. This view shows multiple function graphs simultaneously. If there are many curves, a density transfer function can be used to depict areas with less curves more transparently. A set of outliers that deserves further investigation (possibly indicating an error in the model) are shown in Figure 7.1b.

Generally, we can interpret such data as a collection of substructures, i.e., a collection of data subsets that form time series as addressed above (one per climate descriptor and run, and with 500 time steps each). For other application questions,

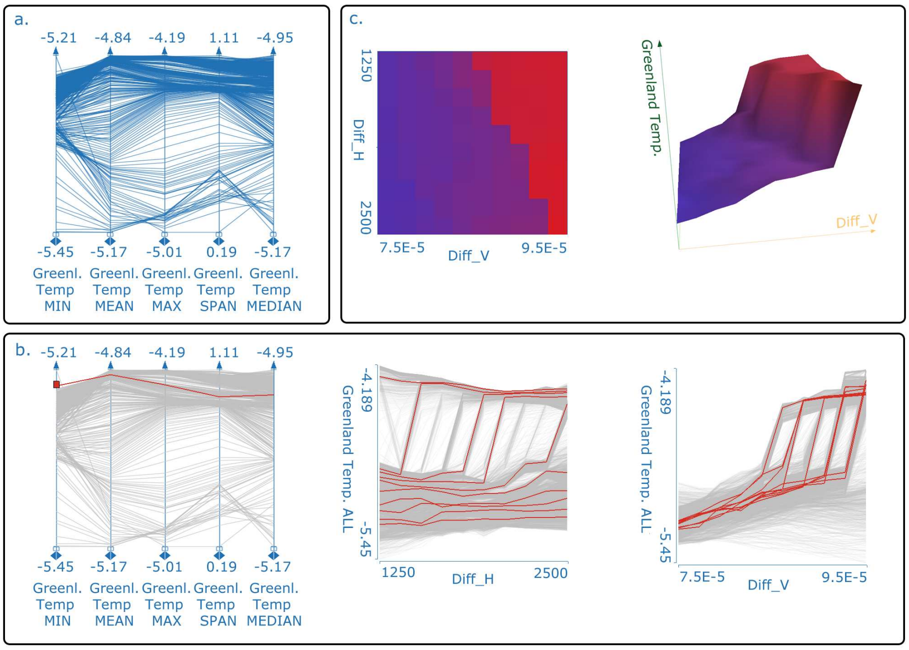


Figure 7.2: Different ways of depicting a family of surfaces. *a*. Standard scalar aggregates of Greenland temperatures can give a first overview of surface family. *b*. Only one surface is selected here. It is shown as a single polyline in the parallel coordinates, or as a collection of function graphs as seen along each of the axis. All respective (curve-typed) cross-sections through the surfaces are shown. *c*. The 3D surface view and the 2D height map view of the selected surface.

we consider different substructures: they can be small 2D data tables of their own (one per climate descriptor and time step, and with $10 \times 10 = 100$ values in the table, i.e., one per instance pair of the two varied diffusivity parameters). We call these small subsets *data surfaces*, or just *surfaces*, and all surfaces representing one climate descriptor a *family of surfaces*. We merge all original data items which have the same time step, and organize the 35 simulated dimensions as 35 families of 500 surfaces each. For each time step, a data attribute such as Greenland temperature, for example, is now a function of two variables, $f(x,y)$, in our case $f(\mathit{diff}_h, \mathit{diff}_v)$. This is true for each simulated value. We have 500 rows in the table now, each containing one scalar value – the time step, and 35 surfaces in the form $surface = f(\mathit{diff}_h, \mathit{diff}_v)$.

Data organized in this way offers new and unique analysis opportunities. However, existing interactive visual analysis technology does not support it. Visualizing one data surface is trivial, visualizing and exploring an entire family is a challenge. We introduce new methodology for the visual analysis of such data.

Analysts want to get deep insight into data and they want to understand the data and underlying simulation model. They want to differentiate the surfaces according to overall characteristics, or considering the variation of climate descriptor values in surfaces along their horizontal (or vertical) diffusivity axis. In order to support such procedures we propose to use various levels of aggregation of the surfaces. The simplest one is to capture one surface as one aggregation scalar, such as the surface's maximum, minimum, median, mean, or span. These scalars can be easily visualized using parallel coordinates, for example. Figure 7.2a shows parallel coordinates visualizing five standard Greenland temperature aggregates (for 500 surfaces each). In the next step, experts want to investigate these surfaces with respect to each of the two axes of the surfaces. They might be interested in how a climate descriptor value changes along the horizontal (or vertical) diffusivity axis. They want to compare how this changes along the time steps. As the view to the surface along one axis can be considered as a collection of curves, we use the curve view to depict the surfaces at this level. Figure 7.2b shows Greenland temperature as seen along each of the axes. We depict all respective (curve-typed) cross-sections through the surfaces in this case. Only one surface was selected, the rest are shown as context in the Figure 7.2b. Note the different shapes which the surfaces from the collection have. This single surface is represented with a single polyline in the parallel coordinates view. Finally, professionals want to see the surface itself at some point of analysis. Although we cannot efficiently visualize whole families of surfaces at once, we can visualize one surface (or a few of them using a real 3D view, or a 2D height map). Figure 7.2c shows the selected surface from the last step as 3D surface and height map.

7.3.2 Interactive Visual Analysis of Families of Surfaces

The analysis of families of surfaces is a very complex task and depends on the application domain. However, we can identify some standard analysis steps supported by interactive visual analysis. Furthermore, interactive visual analysis is the only way a domain expert can cope with the significant complexity of such data, especially if doing a cross-analysis of several families of surfaces. There are three levels in this process. At the top level, the user is interested in overall trends and in high level correlations. The most efficient way to perform such tasks is to represent one surface in the family by one aggregated scalar (or more scalars). Once this high level analysis is done, the user is ready to drill down. However, more data is needed, scalar aggregates do not suffice. We have identified various profiles of the surfaces, which, when used simultaneously, support the cross-surface analysis at this level. Finally, as we do have surfaces, all our collaborators needed them in order to understand the profiles better. It is much easier to understand a surface if it is visualized as a surface.

For multivariate, multidimensional data we use a coordinated multiple views system to pursue the analysis. The system is capable of depicting scalars using various standard views, function graphs which will originate from various profiles

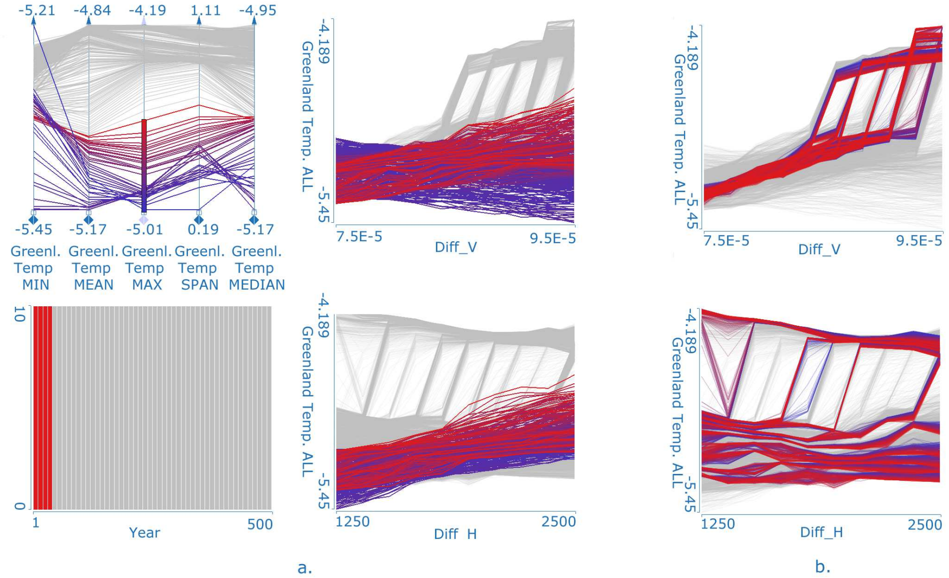


Figure 7.3: *a.* The parallel coordinates depict scalar aggregates of Greenland temperature. Low maximum values of Greenland temperature are selected and corresponding profiles along each axis are highlighted. The unusual shape might indicate an unstable phase of simulation (first 40 years). *b.* The same surface for 200 time steps.

at the second level in our case, and real 3D surfaces as 3D surfaces or 2D height maps.

Analysis Through Single Scalar Aggregates (Top Level)

At the early stage we want to get familiar with the data and to explore high level relations between various surface families and other dimensions in the data set. It is very convenient to have scalar aggregates. We have identified five most often used aggregates, i.e., the minimum, maximum, mean, median, and data span (max–min). All of these aggregates are automatically created for each family of surfaces, and all of them are available automatically for analysis. The analysis uses coordinated multiple views with multiple scalar dimensions at this stage.

We have identified several analysis patterns. Brushing through the parameter space (independent variables) in order to understand the influence and to perform sensitivity analysis is something which is often done first. The next step is selecting either wanted or undesired output values in order to eventually detect a pattern in input parameters causing these outputs. In our meteorological example we have often selected cases with low maximum and high minimum values. Sometimes, like in the EHD lubrication bearing case described in Section 7.5, the engineers are interested in the highest maxima, since, e.g., high pressure is undesirable and they want to see which cases cause the extreme. Figure 7.1a shows an example of analysis at this level for a single surface. A similar example consisting of more views and values for more families can be done in a similar way. Not all of the aggregates were used in the same way. Minimum and maximum are most actively used, user interacts with them directly and selects ranges here. The span is also sometimes used in an active way. The mean and median aggregates, on the other hand, are rarely being actively brushed, they serve as an overview, and provide important visual feedback (as a passive visualization with focus–context discrimination).

Analysis Through Aggregated Profiles

The complex data often requires deeper analysis, exploration of complex cross-family relations. Scalar aggregates are certainly not sufficient for this level. As surfaces are dependant on two variables, users are often interested in surface behavior with respect to one of the independent variables. Furthermore, a common task identified was to explore how various surface parameters behave with respect to each of the independent variables. The simple aggregates, like maximum over all, are not enough, users want to see the maximum with respect to one variable, while keeping the other independent. As we use various profiles along one axis, the resulting data is a function graph per surface. Just as we have identified standard scalar aggregates, we have seen that users most often use standard profiles. We suggest to use the maximum, minimum, mean, median, and all curves profiles. If we want to create curve profiles, we have to select either x or y in$f(x,y)$ first. Once the variable is selected, we can use a cutting plane parallel to one axis (dependent on the selection of the independent variable in focus) and create a collection of cuts. The intersection of the cutting plane and the surface defines a curve. We can use all possible cuts along one axis (since the data is discretized we get a finite number of cuts), or choose one particular cut from a predefined set. In order to depict curves at this stage we use the curve view.

The profiles are based on the surface value and comparison with other values on the curve. For example, the user selects to keep the x axis and for each x_i, we can choose the y value dependent on where the surface has the maximum. Therefore, the surface is represented by the curve where values represent the feature along the cut. In the case of the maximum, a surface $f(x_{i1}, x_{i2})$ is replaced by a curve $f(x_{i1}) = \max_{x_{i2}} f(x_{i1}, x_{i2})$. The same is true for other profiles.

Figure 7.3a illustrates a simple case. We have used parallel coordinates to depict scalar aggregates of Greenland temperature (in the upper left). Low maximum values of Greenland temperature are selected and we can observe interesting relations.

First, we see that all selected runs fall within the first 40 years of the simulation (histogram in Figure 7.3a). A possible explanation for this result might be an instability of the simulation in the early phase. Although this might seem to be of

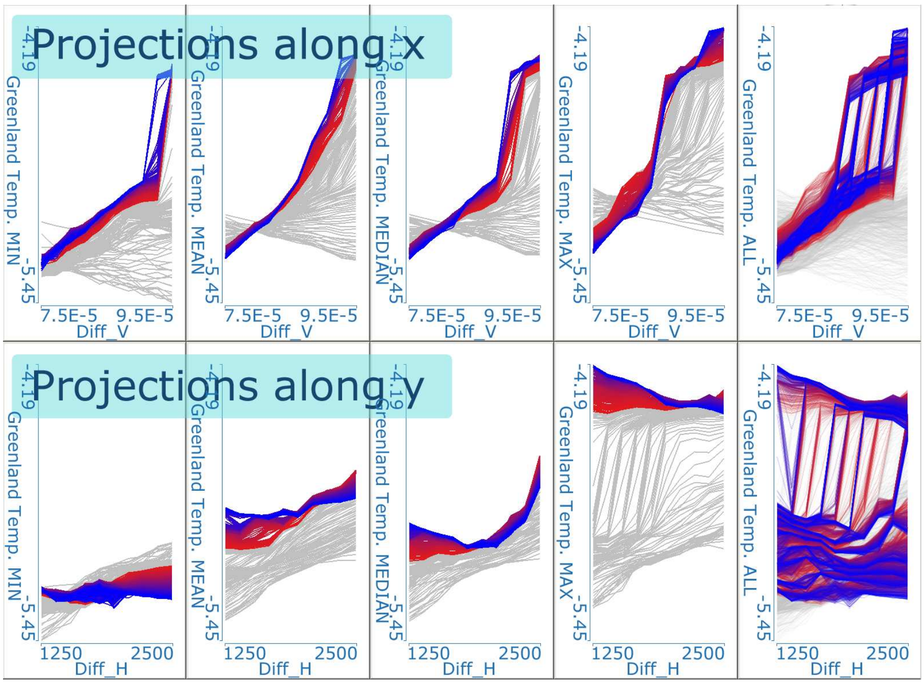


Figure 7.4: Automatically configured multiple profiles of a surface family used at the second level of analysis. Several such views were usually used, in addition to other views depicting scalars and surfaces.

limited importance, it can help the simulation team to detect errors in the model and to estimate when the simulation becomes stable. The surface family (depicted as all curves on the right) shows also that first simulations behave differently than the later ones. The blue curves in the upper right corner (they correspond to low maxima) have a reversed trend as well. They all correspond to early time steps. As time goes by the curves start to rise, and they get a more typical shape. Figure 7.3b shows the surfaces for the last 200 time steps.

The users often use more than one profile at this stage of analysis. We have added a pre-configured profiles view which depicts ten profiles per surface next to each other. The users simply need all the views on the family of surfaces to understand what is going on. Figure 7.4 illustrates an automatically configured view depicting the Greenland temperature family of surfaces using five standard profiles along each of the two axes. It is useful to have several such views, one for each surface family in the data, during the analysis. In this case we would see a case for large-scale high-resolution information displays — this is a clear case where more pixels mean more value in the analysis.

Analysis of Data Surfaces as Such (Lowest Level)

The scalar aggregates and curve profiles, as introduced above, offer a powerful tool for exploration and analysis of complex data. However, at some point, the users want to see the surface itself. It helps them significantly in forming the mental image of the shape, and they can interpret profiles much more easily once they also see the surface. Since we cannot visualize the whole family with all surfaces simultaneously, we use a 3D surface view and a 2D surface view in the advanced phases of the exploration process where we are able to narrow the focus to just a few surfaces from the family.

The idea of the 2D surface view is to represent a surface as a rectangle (2D surface) where a discrete point (x,y) is assigned a color value based on $z = f(x,y)$. Due to a large number of surfaces in a family, providing rectangles for all surfaces is prohibitive, except for very small families. Once the number of selected surfaces is small enough (e.g., after brushing and drill-down), the 2D surface view supports an in-depth analysis of individual surfaces. The pixel count is a limiting factor in such a display. If we use a reasonable size for an overview (something like thumbnail view for images) we can simultaneously depict up to 100 2D surfaces in a usual working environment.

If we reduce the number of surfaces even further, the 3D surface view offers the most intuitive representation of a data surface. Interestingly, all our collaborators really needed the 3D surface view at some point of the analysis. They often switched back to it to understand the family of surfaces better.

7.4 Data Model

The data set described in Section 7.3 is an example of data sets that are collections of data points (tuples) so that the data set under consideration is $D = \{\mathbf{x}^1, \dots, \mathbf{x}^i, \dots, \mathbf{x}^n\}$ where n is the size of the data set (the number of data points) and each data point $\mathbf{x}^i = (x^i_1, \dots, x^i_j, \dots, x^i_d)$ is a collection of attributes, one for each dimension. A tuple attribute x^i_j can be categorical, numerical, or a data series itself.

In the data set from Section 7.3, x^i_1 is the value of $diff_h$ for the data point i, x^i_2 is the value of diff_v for the data point i, x^i_3 is the year and so on for the 35 more attributes. Data points can be aggregated based on the same combination of diff_h and $diff_v$ values. We can then refine our data model so that for each combination of diff_h and diff_v values, the attribute values are data series over 500 years. The data model now has a two-level structure (Figure 7.5).

While these model refinements are rather trivial in this simple example, they illustrate the rationale for a two-level data model that allows us to aggregate data points based on the values in a selected dimension(s) thus restructuring the data set

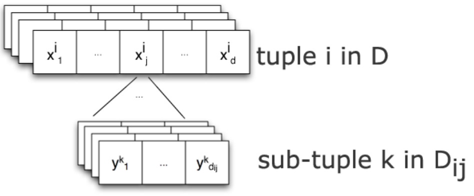


Figure 7.5: Generic data tuple. Each item can be scalar, but can also be a mapping. The data set which contains more tuples, contains a family or families of mappings.

to have a relatively small number of data points while preserving the information content. That provides new opportunities for visualization.

More formally, in our approach we are considering a two-level data set that consists of data points (tuple values) of d dimensions. For each tuple $\mathbf{x}^i$ and each data series attribute x^i_j in a data tuple, we have a separate set of "sub-tuples" with its own cardinality and dimension. The set of sub-tuples is defined as $D_{ij} = \{\mathbf{y}^1, \ldots, \mathbf{y}^k, \ldots, \mathbf{y}^{n_{ij}}\}$ where n_{ij} is the number of sub-tuples. A sub-tuple in D_{ij} has a form $(y_1, \ldots, y_{d_{ij}})$ where d_{ij} is a sub-tuple size and each sub-tuple attribute is either categorical or numerical. The sub-tuple $\mathbf{y}^k$ is then $(y^k_1, \ldots, y^k_{d_{ij}})$.

The dimensions of data series need not overlap with the top level dimensions. While each data series has its own cardinality and dimensions, our discussion is limited to three-tuples and less ($d_{ij} \leq 3$), i.e. a data series can be a sequence of numbers, a sequence of pairs of numbers or a sequence of three-tuples. Data series values can be represented using curves or surfaces. In the case of a sequence of numbers (y_1) we can use a sequence position as a function domain (independent variable) and numbers as function values (dependent variable).

In case of the sequence of pairs of numbers (y_1, y_2), one number is used as an independent variable and other as the dependent variable. We can use a function graph (curve) to represent data series. In case of the sequence of three-tuples (y_1, y_2, y_3), we can select one of the data series dimensions for a dependent variable and the remaining two dimensions for independent variables, thus defining a function of two variables that can be visualized as a surface.

7.5 Case Study — Analysis of a Slider Bearing

For the design of internal combustion (IC) engines, the reliability of the crank train slider and thrust bearings and the piston to linear contact is of central importance. Its design affects key functions such as durability, performance, wear and engine noise. Due to increasing specific loads, all physical effects have become important and they have to be considered by an advanced simulation tool: structural elasticity and dynamics, energy flow, mixed friction, and the influence of temperature and

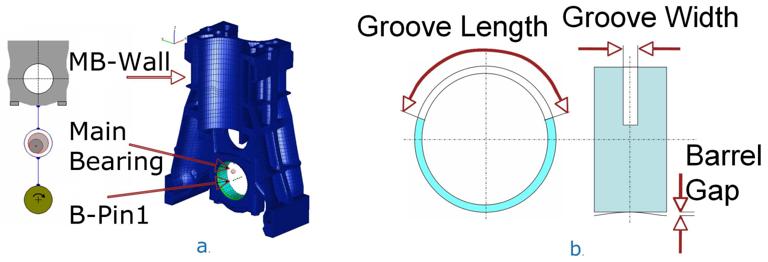


Figure 7.6: *a.* EXCITE Power Unit Main Bearing EHD Model used in the simulation. Topology view on the left showing main bearing wall, EHD joint, and journal and geometry view on the right. *b.* Parameters which were varied in the simulation. Front view of the bearing on the left, and side view on the right. The groove is used for oil supply.

pressure upon oil viscosity. The case study is based on an interactive visual analysis of the IC engine main slider bearing, simulated with the AVL EXCITE Power Unit solver [Off04, PK98]. The corresponding model is shown in Figure 7.6.

In the hydrodynamic bearing simulation, the physical behavior of the structural parts is described by the dynamics of the elastic bodies. Resources needed for the advanced slider bearing analysis are extensive and thus the analysis is very demanding. To allow an efficient sensitivity analysis in this case, two separate strands of analysis have been performed using two different modeling depths. In the first analysis a model of the entire four cylinders inline IC engine has been built up using a simplified main bearing model and a complex MBS model which includes all moving and supporting engine parts. With this model we calculated bearing loads acting on each bearing with sufficient accuracy. Afterwards, the most loaded bearing was selected for a detailed second analysis. Figure 7.7 shows the load computed. The valleys (marked with red circles) correspond to the firing of cylinders.

We have performed an elasto-hydrodynamic (EHD) analysis for this more detailed investigation, using an advanced slider bearing numerical model. It is represented by one main bearing wall section which carries loads applied to the journal as it is shown in Figure 7.6a. The aim of the analysis is to investigate a design space by varying several parameters in order to reduce bearing loads and damage, to increase bearing life time, and to reduce noise generation as well as friction losses.

7.5.1 Simulation Parameters and Results

Numerous control parameters can be defined for a simulation. We varied three design parameters, length and width of the oil groove (used for oil supply in the slider bearing), and height of the gap in the barrel shape of bearing profile (Figure 7.6b). Table 7.1 shows the parameters used and their units. There are $9 \times 5 \times 5 = 225$

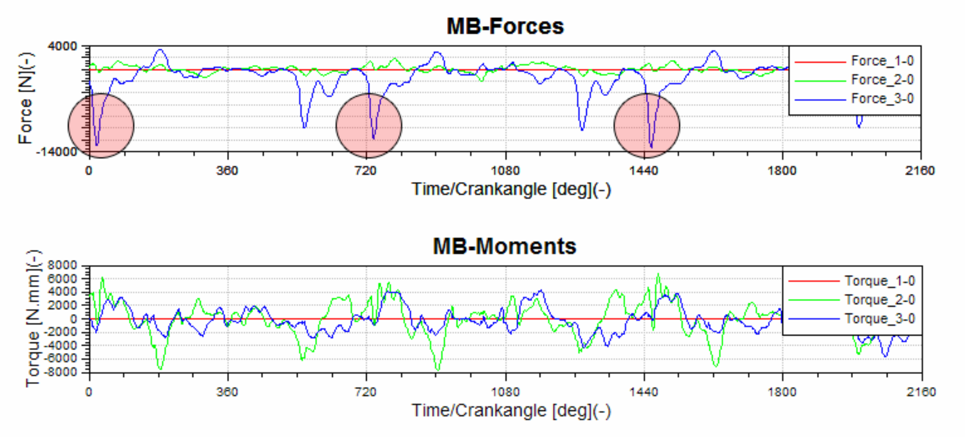


Figure 7.7: Bearing loads, acting on main bearing journal, simulated using a complex multi body simulation model which includes all moving and supporting engine parts. The valleys (red circles) correspond to cylinder firings.

Table 7.1: Control Parameters

Parameter	Values	Unit
Groove Length	90, 112.5, 135, 157.5, 180, 202.5, 225, 247.5, 270	degrees
Groove Width	3, 3.5, 4, 4.5, 5	mm
Barrel Gap	0, 2.5, 5, 7.5, 10	microns

possible parameters combinations, resulting in 225 simulation runs. A simulation run has a simulation period of two engine cycles which, for a four stroke engine, results in four complete revolutions of the crankshaft, or a revolution for 1,440 degrees around the rotational axis. Due to numerical instabilities in the first calculated engine cycle (because of imperfections of the predefined boundary conditions), only the second cycle is used for results evaluation (720 to 1,440 degrees of crankshaft rotation). In this cycle the numerical model is more stable and afterwards results are periodic and repeatable even if the simulation time is extended for more cycles. Seven response parameters were computed for each of the 225 simulation runs. Table 7.2 shows the computed parameters, the abbreviations used in the figures, and the units used.

The simulation tool computes the distribution of the values from Table 7.2 over the entire bearing shell surface. The values are computed for every degree of crankshaft revolution using regularly spaced points across the bearing shell surface. Therefore, each value in Table 7.2 can be seen as a data surface, spanned by two independent variables, bearing shell angle and bearing width. Each surface

Table 7.2: Output values

Parameter	Name	Unit
Bearing clearance height	CLEA	mm
Hydrodynamic pressure	PRES	MPa
Asperity contact pressure	PRSA	MPa
Total pressure	PTOT	MPa
Fill ratio	FILL	
Hydrodynamic shear stress	TAHS	MPa
Asperity contact shear stress	TAAS	MPa

is discretized with 85 points over the bearing shell angle and 11 equidistant nodes over the bearing width. To reduce the amount of data and to speed up the study, the surfaces extracted from the results data have 29 values of the crankshaft revolution regularly sampled within one engine cycle. The complete simulation (225 runs) takes 4–5 days on a typical PC.

7.5.2 Interactive Visual Analysis

Once the data was computed, we started the analysis. The approach with multiple simulations is a new trend here and our domain experts did not have a tool which would effectively support interactive visual analysis of multiple runs and of such complex data. Visualization of individual runs is done using various 2D and 3D charts of output parameters. Usually, a domain expert then compares results from various runs, but cannot interactively explore them. Furthermore, the experts are used to only treating crankangle (equivalent to time) as an independent variable. Since we actually have the output values distributed over all the surface of the bearing shell, it was natural to apply our surface analysis methodology. This way new insights emerged, and our collaborators adapted to the newly proposed method very fast.

The first task was to explore the distributions of asperity contact pressure. High asperity contact pressure yields to an increased load on the bearing and to wearing of the engine. Reducing the asperity contact can contribute the most to the slider bearing optimization.

We have used a scatterplot to depict groove width and height, and two histograms to depict barrel gap and crankangle (Figure 7.8, barrel gap histogram is not shown). We have many simulation runs, one for each combination of groove width, height, barrel gap, and crankangle, and therefore each point in the scatterplot represents multiple runs. The function graph views in Figure 7.8 show maxima of PRSA and PRES, as well as minima of CLEA and FILL data surfaces. The shape of the surfaces is completely invisible if aggregates are used only.

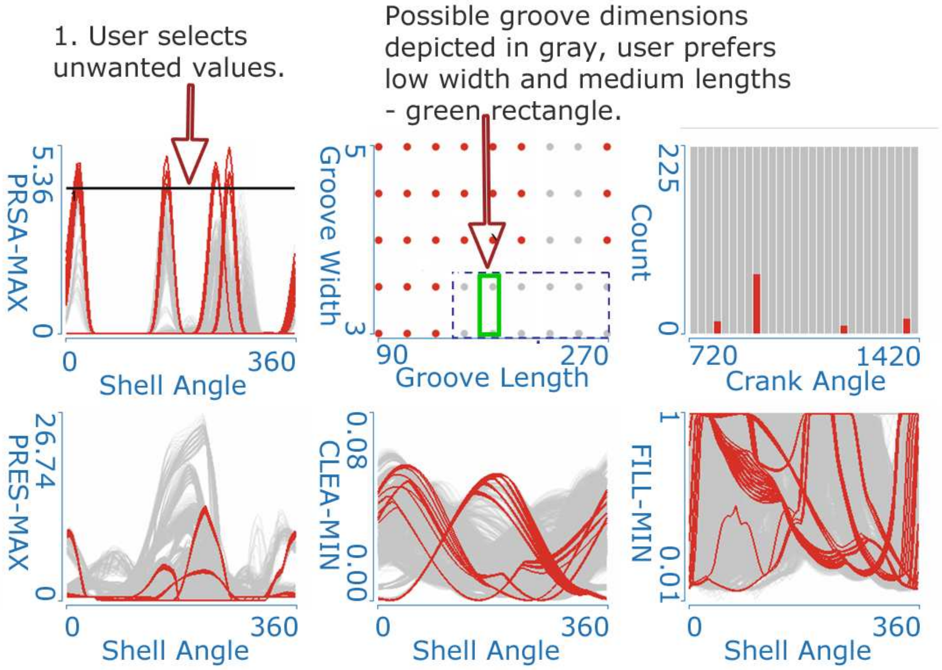


Figure 7.8: The first task was to identify cases with high asperity contact pressure as this leads to wearing of the bearing. We have selected PRSA maximum values in the maximum projection of PRSA surfaces and identify regions of control parameters which result in wanted, low maximum asperity pressure. Note that such a high pressure appears only at certain crankangles, and this information would have been lost in a conventional analysis. Note also the interesting shapes of PRES, CLEA, and FILL surface slices in the lower row of the figure.

The engineer selected maximum PRSA (unwanted behavior) and identified possible values of the control parameters which lead to lower maximum PRSA (gray points in the scatterplot in Figure 7.8). By analyzing the results from the simulation runs we saw that asperity contact is reduced in general by increasing oil groove length. At the same time asperity pressure can be reduced by decreasing oil groove width. The most interesting region is identified: groove length of 180 degrees which is preferable from production point of view and small groove width (the green rectangle in Figure 7.8). This was set as the starting condition for the following analysis.

In the next step of the analysis we look at the simulation runs near to the interesting region (the green rectangle in the Figure 7.8). We use the same setup for control parameters as in the previous step. Figure 7.9 shows four stages of the analysis. We first brush the region of interest in the scatterplot. This brush in the left scatterplot in Figure 7.9a remains the same throughout the analysis. In the first steps (Figures 7.9b–d) we also focus on one value of the barrel gap. Later we refine the selection using a histogram brush on the barrel gap histogram (second histogram in Figure 7.9a, active only for Figure 7.9e). Results for those six combinations of groove size (as selected in the scatterplot) and always the same barrel gap (as selected in the histogram) are showed for complete engine cycle – for all crankangles. To look at the results in more detail, the distribution of asperity contact and hydrodynamic pressure is also shown as a 3D surface, plotted over the bearing shell angle and the bearing width for all crank angles throughout Figure 7.9.

Figure 7.9b shows that maximum asperity contact pressure for selected cases is less than 50% of the maximum pressure in all simulation runs (PRSA-MAX graph) and maximum hydrodynamic pressure is also reduced by approximately 10% (PRES-MAX graph). Our domain expert likes the 3D surface view in wire-frame mode. He always uses it in order to see if there is some strange behavior. As this view depicts many data surfaces simultaneously all we see is a kind of envelope. He can hardly see the interior of the family of surfaces where many data surfaces are possible (and can be clearly seen in function graph views). As he is often interested in extremes he uses it often. By looking at the 3D surface distribution one can see that asperity contact appears at bearing edges and mainly at one side of the bearing due to the distribution of applied bearing moments (Figure 7.9b). The PRSA 3D surface view shows all peaks at one side. At the same time, the distribution of hydrodynamic pressure has a peak in the central bearing region (PRES 3D surface view and function graph view). When we apply the third brush (Figure 7.9c) and show only the maximum values of hydrodynamic pressure, one can see that the maximum appears at only one crank angle near to the top dead center, where maximum bearing forces are acting due to the combustion in the particular cylinder. At this crank angle, the distribution of hydrodynamic pressure (PRES) is regular and asperity contact pressure is near to zero (PRSA). Therefore, further decreasing of the hydrodynamic pressure will probably be not feasible only with selected parameters or without any increasing of the asperity contact pressure.

On the other side, when we look at the maximum asperity contact pressure in Figure 7.9b, (PRSA-MAX views) one can see three groups of characteristic peaks over the bearing shell angle. We select the highest, (3rd,) peak (brush 3 in Figure 7.9d) and explore the corresponding surfaces. It can be seen that maximum asperity contacts (the PRSA graph in Figure 7.9d) appear at one side of the bearing but are not the minimum or the maximum values from the all simulation runs. We can also see in the crankangle histogram that those peaks appear immediately after maximum hydrodynamic pressure after combustion with shifted maximum values to the bearing edge where solid contact is detected (3D surface views). Typically, the asperity contact pressure at the bearing edges can be reduced by using a barrel profile of the bearing shell. This means increasing the barrel gap as defined in Figure 7.6b. We tried to see what is happening as we increase the gap, and we expected the asperity contact to decrease. We extended the barrel gap histogram brush over larger values of the barrel gap, the rightmost histogram in Figure 7.9a shows this selection. Corresponding graphs are depicted in Figure 7.9e. Our results show that asperity contact is even increased due to increases of hydrodynamic pressure and the changed distribution, contrary to our first expectations.

Finally, it is important to see that the here presented sensitivity analysis can also be implemented as a first step in slider bearing optimization. We have efficiently identified the most influential parameters on the bearing behavior in an interactive way. Using our new methodology it is possible to check large amounts of results data and identify correlations between main design parameters and the slider bearing dynamic behavior. The newly proposed data model makes it possible to explore inter-surface relations efficiently. The paradigm shift from a crankangle–based concept to surface width/height based series was very fast and, once achieved, also intuitive. Next steps would be to make the simulation model more complex by varying bearing shell surface properties and oil quality, e.g. The same methodology can be used to explore future designs and automatic optimization results.

7.6 Conclusion

We introduce a new interactive visual analysis methodology to support the analysis and exploration of complex data originating from multiple-runs simulations. Such data is becoming increasingly popular and our approach significantly improves our ability to cope with increased complexity. We introduce a novel way of considering two independent variables in the data. Such data can be understood as families of surfaces. Here we focus on how to deal with the complexity of the data. We can easily depict a million of data items when using a scatter plot, but only about 10 000 items (in a legible and expressive way) using parallel coordinates (without advanced techniques such as reverting to a frequency-based representation [NH06]), and only a handful of surfaces when using the 3D surface view. The proposed approach is certainly not trivial and requires a certain learning curve. Domain experts were puzzled at the beginning, but then they appreciated the new technique. They realized quickly that we need to keep data surfaces coherent in order to understand complex relations and interplay of parameters.

We identify three levels of analysis, aggregation using scalars, profiling with respect to one variable, and finally the last stage where individual surfaces are used. Due to occlusion related problems the final level can be used at late stages of analysis when the user drills down to a single (or just a few) surface(s) in a family. This approach is widely applicable, here illustrated with two examples, meteorological data and the analysis of EHD bearing from automotive industry.

Here we deal with cases where the surfaces are regularly sampled. Surfaces with irregular sampling represent an interesting research challenge. The extension to alternative sampling strategies is an important direction of future work. In both here discussed cases (meteorology and bearing), the analysis is an a posteriori process, i.e., with no further influence on the computed simulation data. We illustrate the newly proposed technology on the cases with just a few control parameters. In other cases, one could think about a coupled setup (as in the case of computational steering [MGJH08]) where throughout the analysis new parameters are varied. Then it would be necessary to reload the analysis with respect to new surface structures in the data.

7.7 Acknowledgements

The authors thank Thomas Nocke and colleagues from PIK (www.PIK-Potsdam.de) for providing the climate simulation data, Stian Eikeland for helping with the data conversion, and Johannes Kehrer for fruitful discussions (both from the visualization group in Bergen, www.ii.UiB.no/vis). The EHD simulation data is courtesy of AVLList GmbH (www.AVL.com). Part of this work was done in the scope of the VSOE VCV program at the VRVis Research Center (www.VRVis.at) and at the Center for HCI at Virginia Tech (www.HCI.VT.edu).

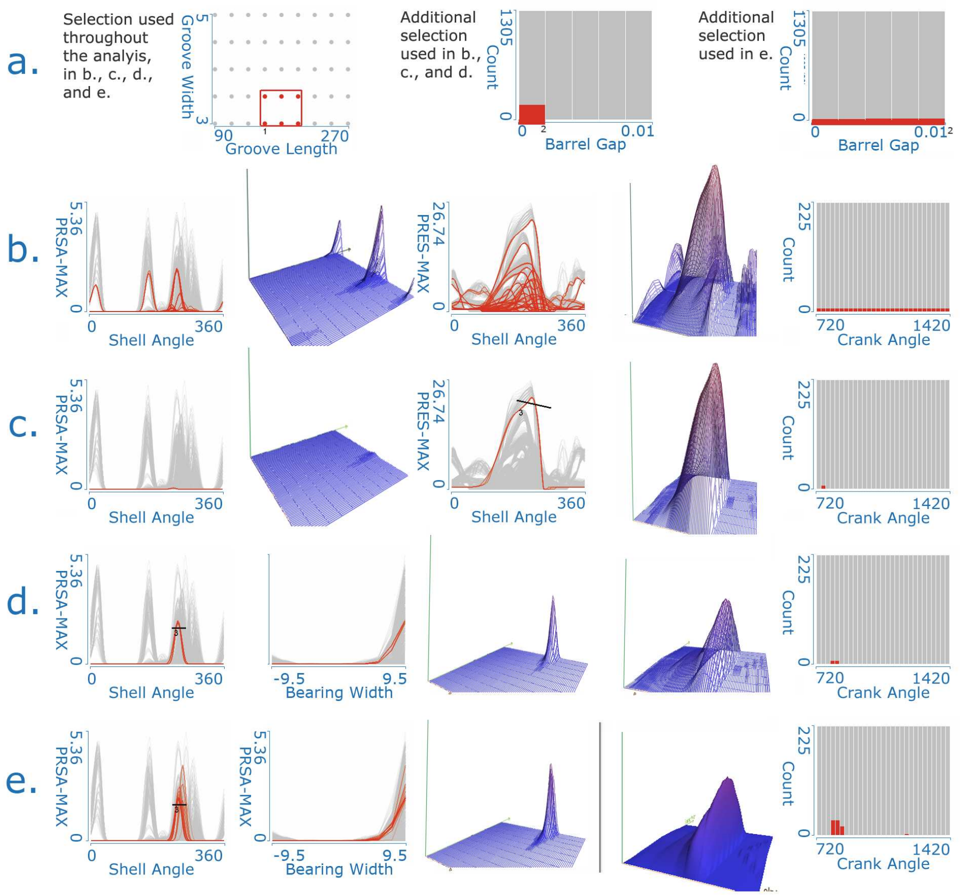


Figure 7.9: *a*. Groove dimensions are fixed using scatterplot. Barrel Gap is the same for *b*, *c*, and *d*. The extended case is used for *e*. Three different peaks in PRSA were identified and explored. Selection was refined by selecting high pressure and barrel gap influence was studied. Note the use of 3D surface view and 3D scatterplot which were almost always used by domain expert - mechanical engineer. *b*. The interesting region of groove size was selected for an in depth analysis. *c*. Selections were refined by selecting high PRESS-MAX. *d*. Selections were refined by selecting high PRSA-MAX. *e*. Finally, gap was extended using histogram selection and high PRSA-MAX was selected.

8

Path Line Attributes - an Information Visualization Approach to Analyzing the Dynamic Behavior of 3D Time-Dependent Flow Fields

Published as:

Kuangyu Shi and Holger Theisel and Helwig Hauser and Tino Weinkauf and Krešimir Matković and Hans-Christian Hege and Hans-Peter Seidel:
Path Line Attributes - an Information Visualization Approach to Analyzing the Dynamic Behavior of 3D Time-Dependent Flow Fields
Topology-Based Methods in Visualization II, Springer, 2009.

Path Line Attributes - an Information Visualization Approach to Analyzing the Dynamic Behavior of 3D Time-Dependent Flow Fields

KUANGYU SHI, HOLGER THEISEL, HELWIG HAUSER, TINO WEINKAUF,
KREŠIMIR MATKOVIĆ, HANS-CHRISTIAN HEGE, AND HANS-PETER SEIDEL

Abstract

We describe an approach to visually analyzing the dynamic behavior of 3D time-dependent flow fields by considering the behavior of the path lines. At selected positions in the 4D space-time domain, we compute a number of local and global properties of path lines describing relevant features of them. The resulting multivariate data set is analyzed by applying state-of-the-art information visualization approaches in the sense of a set of linked views (scatter plots, parallel coordinates, etc.) with interactive brushing and focus+context visualization. The selected path lines with certain properties are integrated and visualized as colored 3D curves. This approach allows an interactive exploration of intricate 4D flow structures. We apply our method to a number of flow data sets and describe how path line attributes are used for describing characteristic features of these flows.

8.1 Introduction

An effective visual analysis of the dynamic behavior of 3D time-dependent flow fields is still a challenging problem in scientific visualization. Although a number of promising approaches have been introduced in recent years, the size and complexity of the data sets as well as the dimensionality of the underlying space-time domain makes the data handling, the analysis and the visual representation challenging and partially unsolved. In particular, it also proves to be inherently difficult to actually comprehend (in detail) the important characteristics of 3D time-dependent flow data.

In addition to others (streak lines, time lines, etc.), there exist two important kinds of characteristic curves for time-dependent flow fields: stream lines and path lines. While stream lines describe the steady behavior of the flow at a certain time step, path lines describe the paths of massless particles over time in the flow. Hence, the analysis of the dynamic behavior of flow fields is strongly related to the analysis of the behavior of the path lines.

One common approach to analyzing flow fields is to partition the flow domain into areas of characteristically different flow properties. To do so, a variety of different features have been proposed, such as topological features, vortical structures, or shock waves. They reflect different properties of the flow and therefore focus on the representation of different inherent structures. In fact, not all features may give useful information for every flow data set, and the selection of the relevant features is often left to the user in an unsupported way. Moreover, among the features there may be correlations which are either general due to their definition, or they occur in certain areas of particular flows and give relevant information about the behavior of the flow. Therefore we believe that not only the introduction and visualization of new features leads to a deeper understanding of the dynamic behavior of the flow field, but also an effective analysis of the relations between the features and the applications of these results for a visual representation. Our paper is one step along the recently challenging path towards a better understanding of 3D time dependent flow fields.

Our approach starts with the extraction of a number of properties (features, scalar values, and time series) at each point of a regular sampling of the 4D space-time domain. We have focused on properties describing the (local or global) behavior of the path lines, being either classical and well-established values in vector algebra, or properties newly proposed in this paper. The result of this step is a *path line attribute data set*: a four-dimensional multivariate data set collecting all computed path line properties.

The visual analysis of multidimensional multivariate data is a well researched topic in information visualization. A variety of techniques has been developed to visualizing such data sets making inherent correlations visible. Because of this we attempt to use information visualization approaches to analyzing the path line attributes data set. The results of this analysis (i.e., selections of path lines with certain combinations of properties) are then used for a focus+context visualization of either the selected path lines or the interesting properties. This way the user is able to do a simultaneous exploration in the 4D space-time domain of the flow and in the abstract path line attribute space. We show that this can give new insight into characteristic substructures of the flow which leads to a better understanding of time-dependent flow fields.

The rest of the paper is organized as follows. Section 8.2 mentions related work in the visualization of 3D time-dependent flow fields. Section 8.3 presents the properties of path lines which we extract for the further analysis. Section 8.4 describes our information visualization approach and explains how to use it for a focus+context visualization of the flow data. Section 8.5 applies our approach to a number of data sets. Section 8.6 draws conclusions and mentions issues of future research.

8.2 Related Work

The idea to segment a flow domain into areas of certain flow properties has been used for 3D steady flow fields for a variety of features, such as topological features [GLL91, LDG98, MBS$^+$04, TWHS03, WTHS04] or vortex regions [Hun87, SH95, PR99]. [SS07] provides a general framework of this in the context of topological features. [PVH$^+$03] gives an overview on flow visualization techniques focusing on feature extraction approaches.

The extension of these techniques to 3D time-dependent fields is usually done by observing the feature regions over time, see [TSH01, TS03, GTS04] for topological features and [BS94, BS95, BP02, TSW$^+$05] for vortex features. Although these approaches provide insight into the flow behavior at arbitrary time steps, the analysis of the dynamic behavior based on path lines make specialized approaches necessary. [WS05] visualizes a number of carefully selected path lines to get static representations of the dynamic flow. [TWHS05] considers a segmentation of the flow domain based on local properties of the path lines. [WEE03, WSEE05] apply texture based visualization approaches to capture path line characteristics.

The idea of connecting information visualization and scientific visualization approaches is considered to be one of the "hot topics" in visualization [Joh04]. Salzbrunn et al. published an approach of streamline predicates for steady flow [SS07]. The work closest to ours is the SimVis approach [DGH03, DMG$^+$05] which uses approaches of information visualization to analyzing various kinds of simulation data. The main difference to our approach is that SimVis works on multiple scalar data describing certain properties of the simulation. Contrary to this, our approach works on dynamic flow data, focusing on local and global properties of path lines, i.e. on a multi-variate properties data set, derived from a 3D unsteady flow field.

8.3 Path Line Attributes

Given a 3D time-dependent vector field $\mathbf{v}(\mathbf{x},t)$, $\mathbf{x}$ describes the 3D domain and t is the temporal component. Stream lines and path lines are generally different classes of curves [TWHS05]. Stream lines are the tangent curves of $\mathbf{v}$ for a fixed time t, while path lines describe the paths of massless particles in $\mathbf{v}$ over time.

Given a point $(\mathbf{x},t)$ in the space-time domain, the stream line starting at $(\mathbf{x},t)$ can be written in a parametric form

$$\mathbf{s}_{\mathbf{x},t}(\tau) = \mathbf{x} + \int_0^\tau \mathbf{v}(\mathbf{s}_{\mathbf{x},t}(s),t)\,ds \tag{8.1}$$

while the path line starting at $(\mathbf{x},t)$ has the parametric form

$$\mathbf{p}_{\mathbf{x},t}(\tau) = \mathbf{x} + \int_0^\tau \mathbf{v}(\mathbf{p}_{\mathbf{x},t}(s),s+t)\,ds. \tag{8.2}$$

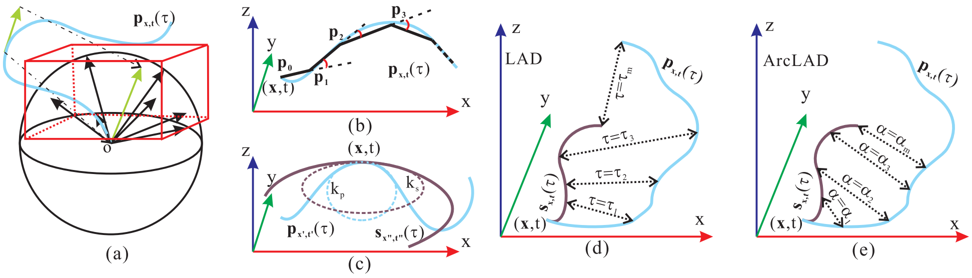


Figure 8.1: a) Mapping the direction vectors along a path line to a unit sphere and calculating the bounding box approximation of the opening cone; b) Winding angle along a path line; c) Curvature difference between the path line and stream line pass through a specified point. d) LAD that records the Euclidean distance between the point of a path line the corresponding stream line at the same time τ; e) ArcLAD that records the Euclidean distance between the point of a path line and the corresponding stream line at the same arc length α from the start point.

Table 8.1: Scalar Attributes

Id	Name	Description
nonStraightV	Non Straight Velocity	$\frac{\int_0^\tau \|\mathbf{v}(\mathbf{p}_{\mathbf{x},t}(s),s+t)\|\,ds - \lvert\mathbf{p}_{\mathbf{x},t}(\tau)-\mathbf{x}\rvert}{\tau}$
distSE	Relative start end distance	$\frac{\lvert\mathbf{p}_{\mathbf{x},t}(\tau)-\mathbf{x}\rvert}{\tau}$
avDir	Average direction	$\frac{\mathbf{p}_{\mathbf{x},t}(\tau)-\mathbf{x}}{\|\mathbf{p}_{\mathbf{x},t}(\tau)-\mathbf{x}\|}$
avParticleV	Average particle velocity	$\frac{\int_0^\tau \|\mathbf{v}(\mathbf{p}_{\mathbf{x},t}(s),s+t)\|\,ds}{\tau}$
lyapunov	Lyapunov exponent	$\frac{log(\sqrt{\lambda_{max}(A^TA)})}{\tau}, A=\nabla_{\mathbf{x}}\mathbf{p}_{\mathbf{x},t}(\tau)$[GRH07, SLM05]
wind	Winding Angle	$\Sigma_{i=0}^{n-2}\angle(\,(\mathbf{p}_{i+1}-\mathbf{p}_i),(\mathbf{p}_{i+2}-\mathbf{p}_{i+1})\,)$,Fig. 8.1b
lad	Local acceleration displacement	$\|\mathbf{p}_{\mathbf{x},t}(\tau)-\mathbf{s}_{\mathbf{x},t}(\tau)\|$,Fig. 8.1d
curvDiff	Curvature difference	$(\kappa_{\mathbf{s}}-\kappa_{\mathbf{p}})^2, \kappa_{\mathbf{p}}=\frac{\|\dot{\mathbf{p}}\times\ddot{\mathbf{p}}\|}{\|\dot{\mathbf{p}}\|^3}, \kappa_{\mathbf{s}}=\frac{\|\dot{\mathbf{s}}\times\ddot{\mathbf{s}}\|}{\|\dot{\mathbf{s}}\|^3}$,Fig 8.1c
div	Local divergence	div($\mathbf{v}$)

Path lines depict the trajectory of massless particles in a time-dependent flow. To characterize path lines, we consider two kinds of information: scalar values that describes local or global properties of a path line, and time series that collects information along a path line.

For scalar attributes, we compute a number of scalar properties of the path line starting at a given point $(\mathbf{x},t)$ which reflect either local or global properties of the path lines. In the latter case, the value depends on the considered integration time. Since we are interested in the global behavior of the path lines, the integration time can be chosen rather large (relative to the time interval in which $\mathbf{v}$ is defined). In particular, we compute the scalar values in Table 8.1.

For time series we have investigated the attributes in Table 8.2.

Table 8.2: Time Series Attributes

Id	Name	Description
DistEu	Euclidean distance to start	$DistEu(\tau)=\|\mathbf{p}_{\mathbf{x},t}(\tau)-\mathbf{x}\|$
LAD	Local acceleration displacement	$LAD(\tau)=\|\mathbf{p}_{\mathbf{x},t}(\tau)-\mathbf{s}_{\mathbf{x},t}(\tau)\|$,Fig. 8.1d
ArcLAD	Arc local acceleration displacement	Fig. 8.1e
Dir	Direction vector	
OpeningCone	Opening cone	Fig. 8.1a
Curvature	Curvature	
Velocity	Velocity	

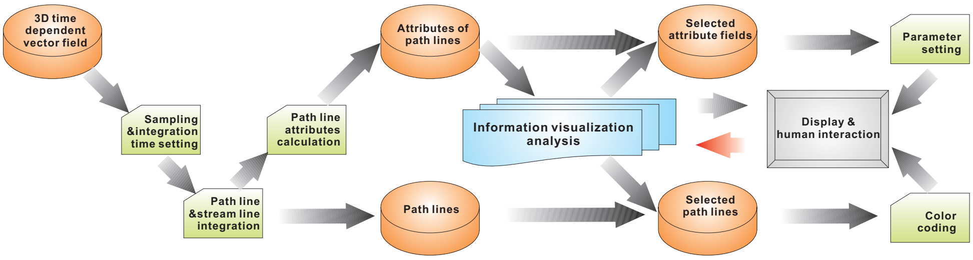

3D time dependent vector field
Sampling &integration time setting
Path line &stream line integration
Path line attributes calculation
Attributes of path lines
Path lines
Information visualization analysis
Selected attribute fields
Selected path lines
Display & human interaction
Parameter setting
Color coding

Figure 8.2: Pipeline for analyzing path line attributes.

8.4 System overview

Fig. 8.2 shows the pipeline of our path line attribute analysis approach. We start with a 3D time-dependent flow field $\mathbf{v}$ to be analyzed. As a first step, we apply a *sampling* of the space-time domain to obtain the points for which we compute the path line attributes. Note that since the data lives in a 4D domain, even a rather small sampling density may give a high amount of sample points. Therefore, the sampling density should be a compromise between the spatio-temporal accuracy of the analysis and the available computing resources. If the analysis delivers interesting features in certain smaller regions of the domain, this region can be analyzed using a higher sampling density to make sure the sampling rate is above the Nyquist frequency . At this state of the approach we also have to set the *integration time* for the path lines. Also this setting is a tradeoff between the fact that we want to have the path lines to be analyzed as long as possible and the property that most of the path lines should be integrated over the same time without leaving the domain.

The next step of the approach is the integration of the stream lines and path lines starting from the sampled points over the set integration time. For our examples we have used a 4th order Runge-Kutta integration. From these integrations we compute all path line attributes introduced in section 8.3.

The set of all path line attributes is the input of our information visualization core module which will be described in section 8.4.1 in more detail. Interactive visual analysis on the basis of state-of-the-art information visualization techniques and brushing in linked views is used to extract relevant correlations, interesting feature combinations, or general properties of the data. Note that the brushed features are not necessary physical variable. The result of this analysis is used to steering the visualization of the path lines and their attributes. If the interactive visual analysis delivers interesting features in a certain scalar path line attribute, we can visualize it using standard volume rendering techniques like direct volume rendering or slicing. Furthermore, the interactive visual analysis delivers a selection of interesting path lines having a certain combination of properties. They are visualized as 3D line structures with a color coded time component.

Our implementations of the visualization of the selected path lines and the selected attributes are based on Amira [SWH05], whereas our information visualization analysis is based on the ComVis system which is described in section 8.4.1.

8.4.1 The ComVis system

ComVis is an interactive visualization tool. It supports conventional information visualization views such as 2D and 3D scatter plots, parallel coordinates, histograms, as well as a special curves view which is used for displaying function graphs. This combination of views makes it possible to analyze a wide variety of data where in the same row of a multi-variate table some values are scalar (just as it is usual) and others correspond to a function graph (common in various kinds of scientific data)[MJJ$^+$05]. The tool offers multiple linked views parallel to each other. Each view can be of any of the above mentioned view type. ComVis pays great attention to interaction. Due to advanced brushing and linking proved to be very powerful analytical tool. Users can brush the visualized data in any view, all linked views reflect the data selections by appropriate focus+context visualization. Furthermore, the user can use a simple, yet powerful line brush in the curves view. The line brush selects all curves which intersect the line. All brushes can be scaled and moved interactively. The multiple brush mode makes it possible to flexibly combine various brushes. The user selects brushes and boolean operations between them. AND, OR, and SUB are supported. Furthermore, the tool creates a composite brush in an iterative manner. This means that the user selects a current operation (AND, OR, or SUB) and draws a brush. The previous brushing state is combined with the new brush accordingly. The new state is computed, and it is used when the user draws another brush. In this way the user immediately gets visual feedback, and can very easily broaden the selection (using OR), or can further restrict the selection (using AND or SUB). Once the user is satisfied with a selection (or in the meantime), a tabular representation of the selected data can be shown and exported to file on demand.

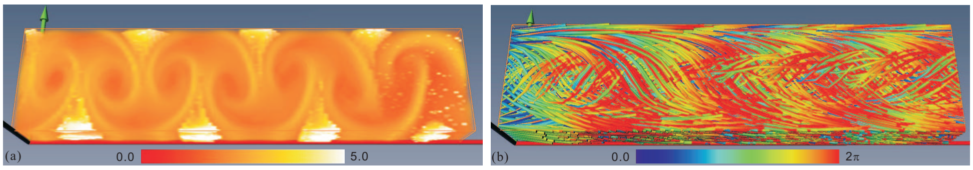


Figure 8.3: Flow behind a cylinder: a) Direct volume rendering of the *lyapunov* attribute field at time 0; b) All considered path lines.

8.5 Applications

We applied our approach to a number of data sets. Not surprisingly, not all attributes are interesting in all data sets, and different path line attributes turn out to be important for different data set. However, we can also identify several interesting coherencies between different path line attributes which seem to hold even for different data sets. Accordingly, we are optimistic that the here described analysis indeed provides a useful basis for future generalization of this approach.

8.5.1 3D time-dependent cylinder flow

Figures 8.3 and 8.4 present some results of analyzing a 3D time-dependent flow behind a circular cylinder. The cylinder is put in the origin with radius 0.5 and height 8.0, while the data set domain D is $[3.15, 19.74] \times [-2.06, 2.06] \times [0.09, 1.89] \times [0, 2\pi]$. This data set was kindly provided by Gerd Mutschke (FZ Rossendorf) and Bernd R. Noack (TU Berlin). We considered path lines at a $28 \times 14 \times 7 \times 6$ (191MB attribute file to ComVis) sampling and used an integration time of 1.5π (for the data set given in a 2π time slab). Figure 8.3a shows the direct volume rendering of one of the attribute fields *lyapunov*. In figure 8.3b, all path lines integrated from the sampled points are displayed. As we can see from figure 8.3a, there are certain patterns in the *lyapunov* attribute field. Low *lyapunov* values indicate stability of the path line. We use the information visualization approach to select the area with low *lyapunov*, as shown in the upper left of figure 8.4a. The visualization of the selected path lines is shown in figure 8.4b. Fig. 8.4c shows the seed area of the selected path lines at the time 0.

When investigating the visualized result, we can see that there are further different patterns in the low *lyapunov* path lines. It is obvious when we investigate the ComVis result of time series *LAD*, after choosing the cluster as shown in the upper right of figure 8.4d. We get the path line cluster whose *LAD* time series have small values at the end of the integration time. Fig. 8.4e and 8.4f present the visualization of the selected path lines and their seed areas. We notice that they stay in the middle of the domain and along the flow direction directly behind the cylinder.

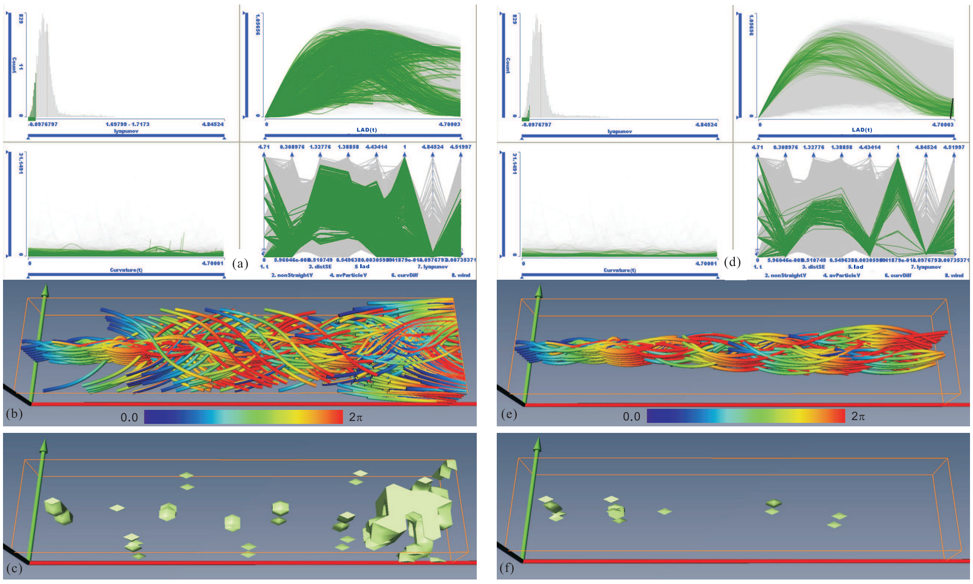


Figure 8.4: Feature low *lyapunov* and *LAD*: a) Selecting low *lyapunov* area in ComVis; b) Visualization of selected path lines with low *lyapunov*; c) Visualization of the seeding area of the selected low *lyapunov* path lines at time 0; d) Selecting low *lyapunov* and parabola *LAD* area in ComVis; e) Visualization of selected path lines with low *lyapunov* and low *LAD*; f) Visualization of the seeding area of the selected low *lyapunov* and low *LAD* path lines at time 0.

8.5.2 Hurricane Isabel

Fig. 8.5 shows a visual analysis of the hurricane Isabel data set, which has been previously analyzed in a number of papers [GM04, DMH04]. We sample the domain with path lines at a resolution of $24 \times 24 \times 6 \times 6$ (253MB attribute file to ComVis), and set the maximum integration time to 30 hours (the whole data set covers 48 hours). Fig. 8.5a shows the visualization of all considered path lines. Fig. 8.5b show a direct volume rendering of *nonStraightV* at time 0 (the starting time of the simulation).

For this data set, we start the information visualization analysis, with the observation of the *avParticleV* vs. *distSE* scatter plot (upper right of Fig. 8.5c), showing a number of points on the diagonal but also a number scatter points clearly above it. We expect the points on the diagonal to represent path lines with a rather straight-line-like behavior, whereas the locations of the points above the diagonal may indicate a swirling behavior. Since *nonStraightV* is equivalent *avParticleV* vs. *distSE*, we selected all points above the diagonal, by considering points with a rather high *nonStraightV* (upper left of Fig. 8.5c). The parallel coordinate representation (lower right of Fig. 8.5c) shows that the selected path lines have a rather low *curvDiff*. This indicates that in these regions stream lines and path lines are locally rather similar.

The curvature plot of the selected path lines doesn't have extreme values (lower left of Fig. 8.5c). The selected path lines are visualized in Fig. 8.5d, clearly showing that we have selected the ones swirling around the moving eye of the hurricane. Fig. 8.5e shows the areas where the selected path lines originate at time $t = 0$, while Fig. 8.5f shows the starting areas of the selected path lines for all time steps.

8.5.3 Airfoil

Figures 8.6 - 8.7 show a comparative visual analysis of 8 different data sets of a flow around an airfoil. The difference between these 8 data sets are the air injection frequency. The injection frequencies are 0(base), 0.2, 0.44, 0.6, 0.88, 1.0, 1.5 and 2.0. The goal of our analysis is to find the best air injection frequency which contributes the best lift power. It is known that abnormal vortex structures reduce the lift of the airfoil. Therefore, our visual analysis focuses on the areas with vortices where the probability of abnormal flow is high. We reduce our consideration to a small area around the areas with vortices.

We sample the interesting area with path lines at a resolution of $36 \times 12 \times 8 \times 10$ for each data set, and set the maximum integration time to 30 seconds (the whole data set covers different time domains for different frequencies and the path line integration will usually leave the domain within 30 seconds for each frequency). Fig. 8.6 shows the visualization of all considered path lines for different frequencies. We observe that most path lines behave well showing a rather straight behavior. The abnormal flows correspond to those non straight path lines. As our experience on these attributes, the $nonStraightV$ is a good attribute to reflect the characteristics of straightness of path lines. So we compare this attribute computed at same location and same time for different frequency data sets in ComVis.

Fig. 8.7 shows the comparative result of the analysis of the $nonStraightV$ for these 8 different frequencies. Relative analysis is popular in airfoil analysis since the relative flow behavior for different parts of an airfoil determines the lift power. We apply a relative selection here and select those path lines for each data set with 70 percent highest $nonStraightV$ attributes. Those selected path lines and the corresponding seeding areas are visualized. We can see that these selected non straight path lines are closed to the area with vortices. And we can clearly observe that for a frequency 0.6, there are fewest non straight path lines and the non straight seeding areas are the smallest. So we find that for frequency 0.6, the probability of abnormal flow is less compared to others. We have tested several other percentage of the highest $nonStraightV$. All the results present the equivalent information. We conclude that 0.6 is the best air injection frequency among the 8 tests. The experience from the industry partner confirms this result successfully.

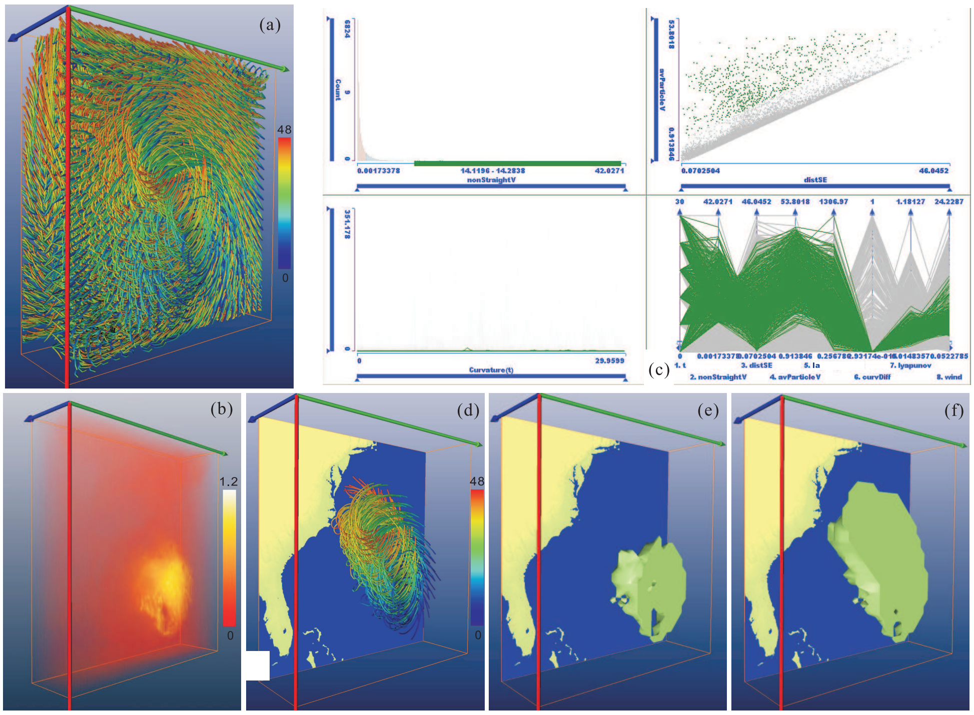


Figure 8.5: Analysis and visualization of data set Hurricane Isabel: a)A visualization of all considered path lines. b) Direct volume rendering of the *nonStraightV* attribute field at time 0; c) Selecting the area with high *nonStraightV* which corresponds to swirling behavior in ComVis; d) Visualization of selected path lines of swirling behavior; e) Visualization of the seeding area of the selected swirling path lines at time 0; f) Visualization of the seeding area of the selected swirling path lines at all time steps.

8.6 Conclusions

To getting insight into the dynamic behavior of path lines of 3D time-dependent flow fields is still a challenging problem for the visualization community. Path lines elude a straightforward extension from stream line based methods because path lines can be integrated only over a finite time, and they may intersect each other (at least when only considering their 3D reference locations). This paper is the - to the best of our knowledge - first approach to getting insight into the behavior of path lines by applying an approach from information visualization. In particular, we made the following contributions:

- We identified a number of local and global attributes of path lines which we expect to contain relevant information about the path line behavior.
- We interactively analyzed these attributes by using an approach from infor-

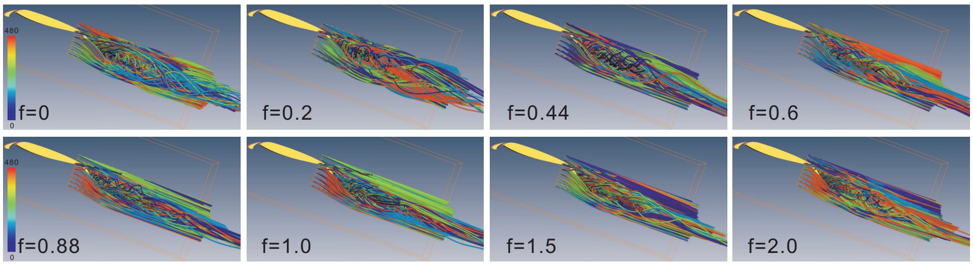


Figure 8.6: The path lines started from the focus area of the airfoil flow field for different air injection frequency.

mation visualization. The results were used to steering a 3D path line visualization.

- We applied our approach to a number of data sets, in order to get new insight into the path line behavior.

During our analysis it turned out that not all path line attributes gave useful results for all data sets. However, inherent and data independent correlations in the attribute data set can be expected, making a reduction of the attribute set possible. In particular, we have the impression that the investigation of path line attributes can indeed lead to a useful and practicable way of accessing/segmenting interesting flow features in time-dependent data sets, including swirling/vortical/rotating flow subsets, (e.g., via attributes *wind* and *nonStraightV*) , quasi-steady flow structures, (e.g., via attributes *LAD* and *ArcLAD*, etc.), etc. We are optimistic with respect to these expectations, not at the least because it was, for example, fairly straight forward and quite easy to accomplish to extract the rotating main vortex of hurricane Isabel, which – to the best of our knowledge – cannot so easily be accomplished with any of the previously published vortex extraction methods.

8.7 Acknowledgments

We thank Bernd R. Noack for the supply of the cylinder data which was kindly provided by Gerd Mutschke. The Hurricane Isabel data produced by the Weather Research and Forecast (WRF) model, courtesy of NCAR and the U.S. National Science Foundation (NSF). Also many thanks to Jan Sahner for support and fruitful discussion. Parts of this work have been done in VRVis Research Center which is funded by an Austrian research program called Kplus. Also, parts of the work was funded by the Max-Planck center of visual computing and communication. All visualization of this paper have been created using ComVis (see `http://www.vrvis.at/via/products/comvis.html`) and Amira – a system for advanced visual data analysis [SWH05] (see `http://amira.zib.de/`).

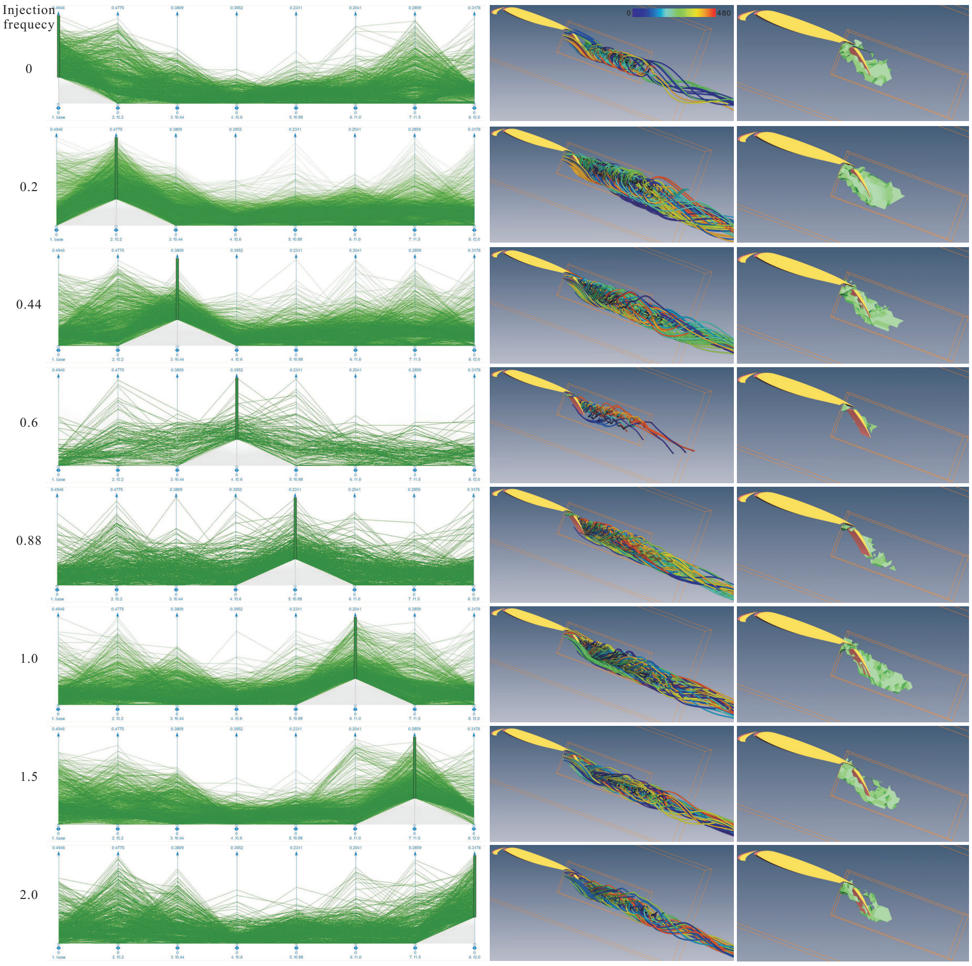


Figure 8.7: The comparative analysis result for the attribute *nonStraightV* of of the airfoil flow field for different air injection frequency. The pictures in the first column depict the selections of 70 percent highest *nonStraightV* for different frequencies in ComVis. The pictures in the second column depict the corresponding selected path lines for the first column. The pictures in the third column are the corresponding seeding area for the selections in the first column.

9

A Statistics-based Dimension Reduction of the Space of Path Line Attributes for Interactive Visual Flow Analysis

---

Published as:

> Armin Pobitzer and Alan Lež and Krešimir Matković and Helwig Hauser:
> **A Statistics-based Dimension Reduction of the Space of Path Line Attributes for Interactive Visual Flow Analysis**
> *Proceedings of Pacific Visualization Symposium (PacificVis), 2012.*

A Statistics-based Dimension Reduction of the Space of Path Line Attributes for Interactive Visual Flow Analysis

ARMIN POBITZER, ALAN LEŽ, KREŠIMIR MATKOVIĆ, AND HELWIG HAUSER

Abstract

Recent work has shown the great potential of interactive flow analysis by the analysis of path lines. The choice of suitable attributes, describing the path lines, is, however, still an open question. This paper addresses this question performing a statistical analysis of the path line attribute space. In this way we are able to balance the usage of computing power and storage with the necessity to not loose relevant information. We demonstrate how a carefully chosen attribute set can improve the benefits of state-of-the art interactive flow analysis. The results obtained are compared to previously published work.

9.1 Introduction

When analyzing the dynamics of unsteady flow, the investigation of particle movements is a canonical choice. In order to enable further analysis based on particle paths, these trajectories need to be characterized. Possible ways to describe the paths include deriving measures for their global and local behavior and properties of the field around the moving particles. A large number of such feature detectors is available and has been used in different contexts [PVH$^+$03, JMT05, STH$^+$09, SGSM08].

Previous work by Bürger et al. [BMDH07], Shi et al. [STH$^+$09] and Lež et al. [LZM$^+$11] has shown the great potential of the combination of *Interactive Visual Analysis* (IVA) and feature extraction. However, the question of how to choose an adequate attribute set to investigate is left open, although it is non-trivial. Feature detectors are usually designed to target one specific aspect of the flow behavior. An ad hoc choice of suitable attributes is therefore dependent on correct prior knowledge (or assumptions) on what type features to expect. This has the implication that unexpected behavior is possibly hard to detect. Therefore, a objective and complete investigation of the data set in question would require to look for "all possible" features (say vortices, vortex core lines, path lines with low average speed,…) and their detectors at once.

This leaves us with a large amount of possibly interesting features and even more detectors that should be considered. Computing all of them is tedious at least and results in a high computation time and storing a large number of attributes per trajectory. It can be expected that this brute force approach would generate a considerable information overhead, since many of the attributes are computed from the same velocity field. In general, different feature detectors may systematically correlated to each because they either describe the same aspect of the flow or are related to each other by physical principles (e.g., velocity and vorticity by the vorticity equation). From the practical side, a systematic analysis of a data set gets increasingly challenging the more dimensions it contains. Hence, a canonic question in this context is: Is there a common subset of the path line attributes that captures "all" complexity of the data sets? Or, in short, what is the intrinsic dimensionality of the path line attribute space?

The problem of analyzing high dimensional data sets is a classic challenge, that both statistics [JW07] and visualization [FH09] deal with, as well as others. Roughly speaking, the main distinction between these two approaches to multivariate data analysis, is the role of the user: while statistics relies on automatic methods, visualization-based approaches try to exploit a larger amount of user interaction [SS04a, PBH08]. The benefits and drawbacks of the two approaches can be considered complementary. One should therefore aim to combine the strengths of both, namely rigidness and inherent objectiveness of statistical methods for determining the intrinsic dimensionality of the data and the flexibility and possibility of integration user knowledge of IVA in the analysis stage.

In statistics, a number of dimension reduction methods are available [Fod02], two of the most prominent being *principal component analysis* (PCA) [Pea01] and *exploratory factor analysis* (EFA) [Spe07]. In short, the first finds orthogonal principal components (linear combinations of the observed variables) that account for the maximal amount of variance. Although efficient for the mere dimension reduction purpose, one of the draw-backs of this method is that the principal components are usually hard to interpret and the computation of them involves possibly all observed variables. Modifications of PCA that attempt to avoid this have been proposed [ZHT06], imposing additional constrains on the algorithm. Exploratory factor analysis gives the number of statistical variables, called *factors*, needed to explain the common variance between variables and how these factors *load on* (are correlated to) the observed variables. These factors are assumed to be not directly measurable, hence the actual interesting information being the *loadings* (correlation coefficients) on the observed variables. Taking the highly loaded variables for every factor yields a set of variables that are interpretable, account for the complexity of the data, and the set is (if the data allows for this) of considerable smaller dimensionality. Hence, exploratory factor analysis is a more promising choice for the dimension reduction for our purposes. For a more detailed overview of the similarities and differences between PCA and EFA, we refer to Suhr [Suh05].

In this paper we investigate several CFD data sets with exploratory factor analysis, with the goal to find a common set of variables that can be used as a starting point for a deeper analysis of CFD data sets. This variable set should capture the underlying physical processes in fluids with a little as possible redundancy. The analyzed data sets span different geometries, constant/non-constant inflows as well as different simulation methods to prevent the variable set from being specific for one type of simulation/geometry/application.

We compare the results from IVA applied on the attribute set found in our investigation to previously published results. Our results match and also partially exceed previous ones. In contrast to previous work we have a predefined attribute set, which makes a more systematic analysis possible.

The remainder of this paper is organized as follows: first we briefly discuss previous work, and then we describe our statistical analysis and present its results. We give a demonstration of the results achievable from the combination of our findings and IVA, comparing these results to previous ones.

9.2 Related Work

Because of their tight relation to the dynamic behavior of the flow, visualization by means of particle trajectories is a well established branch of flow visualization [MLP+10].

Theisel et al. [TWHS05] introduce the classification and segmentation of path lines according to attracting, repelling and saddle-like behavior for visualization purposes. This classification allows the authors to identify a path line-based topology for two-dimensional unsteady flows. Salzbrunn and Scheuermann [SS06] introduce a mathematical framework based on Boolean algebra that allows to define a topology based on so-called streamline predicates. These predicates are user chosen and can be seen as path line attributes with Boolean range. Later, Salzbrunn et al. [SGSM08] extend this approach to path lines.

Bürger et al. [BMDH07] investigate the opportunity to combine several feature detectors making use of interactive visual analysis, focusing on vortical features. Shi et al. [STH+09] present a similar approach together with more general path line attributes, using both local and global descriptors for the path line behavior. Lež et al. [LZM+11] enhance the utility of path line based IVA by the possibility for direct path line brushing via projections.

The problem of dimension reduction in high-dimensional data sets is a well established research field within statistics. Pearson published his seminal work on *principal component analysis* [Pea01] in 1901. Spearman laid the foundation for *factor analysis* with his 1907 article on the "true measurement of correlation" [Spe07]. Since, a large number of related methods and algorithms has been presented. For an overview we refer to Fodor's survey on this topic [Fod02].

In the context of information visualization, the possibility for user-guided dimension reduction has been investigated. Yang et al. present a method called *Visual Hierarchical Dimension Reduction* (VHDR) [YWRH03]. VHDR clusters the data dimensions according to similarity measures, generating a dimension hierarchy. The user selects clusters and specifies "representative dimensions" for those clusters. Finally, a projection step is applied.

Seo and Shneiderman present the *rank-by-feature* framework [SS04a], that allows the user to rank the dimensions by some simple statistics for one-dimensional and two-dimensional representations of the original dimension and dimension pairs, respectively. Piringer et al. [PBH08] extend this approach to the investigation of user-specified subsets of the original data set instead of dimensions only. Recently, Turkay et al. [TFH11] presented a visualization model that allows for interaction in both *item* and *dimension space*. This eases the understanding of the relation of different data dimensions and the according analysis of high-dimensional data sets, yielding means of interactive dimension reduction.

This paper is targeting dimension reduction for interactive flow analysis along the lines of the works of Bürger et al. [BMDH07], Shi et al. [STH$^+$09], and Lež et al. [LZM$^+$11].

9.3 Statistical Analysis

In this section we firstly give a description of the statistical methodology we use, and then describe the actual analysis of the data.

9.3.1 The Statistical Model

As explained in the introduction, we chose exploratory factor analysis (EFA) to investigate the dimensionality of the path line attribute space. It is worthwhile noticing that modern EFA is more a group of methods than one single algorithm. Since we want to find the minimal number of factors explaining the variation in the data set, we have to choose *Principal Factor Analysis* (also known as *Common factor analysis*) [Har76]. In order to increase the numerical stability, an iterative algorithm is used [Har76]. Since we are interested in factors that can be related back to one (or more) attributes that we can compute, we discharge the usual assumption of uncorrelated factors and use the so-called *varimax criterion* instead [Har76]. In short, this criterion tries to maximize the variation in the factor loadings onto the variables. This yields often the situation that each variable virtually loads one factor only [Har76]. For a thorough discussion of these algorithmic choices, and possible alternatives, we refer to Harman's book [Har76].

One crucial aspect of a factor analysis is the criterion that determines how many factors have to be retained. The eigenvalues of the respective factors give informa-

data set	param.	num. of factors (Kaiser/screen/prop.)	factor: variable with highest loading (loading)
flow through a box	time	6/7/11	1:λ_2 (0.95), 2:inv (0.99), 3:startEnd (0.97), 4:vel (0.87), 5:usH (0.85), 6:windang (0.65), 7:dpos (0.81), 8:vel (0.71)
	space	5/5/9	1:λ_2 (0.96)/Hunt's Q (-0.94), 2:inv (0.98), 3:vel (0.86), 4:uSH (0.86)/SH (0.85), 5:startEnd (0.92), 6:pos (0.68)
t-junction	time	5/8/6	1:inv (0.98), 2:startEnd (1.00)/avspeed (0.99), 3:λ_2 (0.94)/Hunt's Q (-0.93), 4:SH (1.00), 5:pos (0.88), 6:avspeed (0.71)
	space	5/7/7	1:startEnd (0.99)/avspeed (0.97), 2:inv (0.89), 3:SH (0.96)/uSH (0.94), 4:λ_2 (0.95)/Hunt's Q (-0.93), 5:inv (0.78), 6:pos (0.86), 7:avspeed (0.58)
breaking dam	time	5/5/7	1:vort (0.95), 2:SH (0.94)/uSH (0.93), 3:inv (1.01), 4:startEnd (0.95)/ avspeed (0.94), 5:λ_2 (0.89)/Hunt's Q (-0.85)
	space	4/4/8	1:vort (0.97), 2:inv (1.01), 3:dpos (0.67)/vel (0.63), 4:uSH (0.89)/SH (0.87), 5:Hunt's Q (0.94), 6:pos (0.73)
exhaust manifold	time	4/8/8	1:vort (0.94), 2:SH (0.92)/uSH (0.92), 3:inv (0.92), 4:startEnd (0.94)/avSpeed (0.93), 5:λ_2 (0.89)/Hunt's Q (-0.85)
	space	4/8/7	1:vort (0.87), 2:avspeed (0.97)/dpos (0.94), 3:avspeed (0.97)/dpos (0.94), 4:inv (0.78), 5:avspeed (0.98)/dpos (0.93)
turb. chan. flow	time	5/7/5	1:inv (0.99), 2:λ_2 (1.00), 3:vel (0.93), 4:vel (0.92), 5:vel (0.94)
	space	5/7/6	1:inv (0.99), 2:λ_2 (0.96)/Hunt's Q (-0.95), 3:vel (1.00), 4:vel (1.00), 5:vel (1.00), 6:windang (1.00)
rot. vortex rope	time	7/6/7	1:inv (0.98), 2:vort (0.99), 3:SH (0.99)/normhel (0.99), 4:λ_2 (0.99)/ Hunt's Q (-0.99), 5:avspeed (1.00)/vel (0.96), 6:avspeed (1.00)/vel (0.95)
	space	7/7/9	1:inv (0.97), 2:vort (0.99), 3:SH (0.99)/normhel (0.99), 4:λ_2 (0.99)/Hunt's Q (-0.99), 5:vel (1.00)/avspeed (0.96), 6:avspeed (1.00)/vel (0.95), 7:windang (0.91)

Table 9.1: Summative result of the statistical analysis on the different data sets, according to their parametrization. The found patterns are discussed in Sec. 9.3.4.

tion on how much of the variance is explained by the single variable, compared to a uniform distribution of the variance. Hence, Kaiser [Kai60] suggests to retain all factors associated to an eigenvalue greater than 1. Cattell suggests the use of the plot of the eigenvalues against their index to determine the right number of factors to retain [Cat66]. The factors that lay on the *scree* (i.e., the base of a steep incline or cliff) of the plot are considered neglectable, therefore this criterion is commonly referred to the *scree plot test* [Cat66]. Finally, if the goal is to guarantee that the retained contain a certain percentage of variance, one can simply include factors until their relative weight exceeds a desired threshold. Kaiser's criterion has the advantage of being objective, but has proven to be unreliable in extracting the true number of underlying factors [CO05]. Better results are obtained using the scree test [CO05]. Here the drawback lies in fact that this is a "soft" criterion that relies on the users interpretation of the scree plot. Finally, retaining factors accounting for more than 100% of the variance will not add information about the data set but noise. Hence, we consider the maximum of the factor number suggested by Kaiser's criterion and the scree plot test, with 100% of explained variance (or proportion) as a limiting bound. For a more thorough discussion of how to choose the correct number of factors to retain, we refer the reader to the article of Costell and Osbourne [CO05].

All statistical computations for this paper have been carried out with SAS© software.

9.3.2 The Data Sets

In total we analyzed 5 different data sets with different geometries and simulation methods and 1 analytic data set. For the greatest possible generality, we use only the velocity fields to calculate the path lines and their attributes. This means that the similar factor patterns across the data sets are due to the common underlying principles of fluid dynamics and not due to similarity in the data sets. The data sets investigated are the following:

Flow through a box: This data set is the simulation of flow through a box. The data set consists of 100 time steps. The inlet is on the top of the box. The data set consists of 17120 cells organized in a Cartesian grid.

T-junction: This data set is the simulation of flow through a T-junction with two inlets and one obstacle inside. The data set consists of 100 time steps. One inlet is in horizontal direction, another one in vertical direction. The obstacle is placed under the vertical inlet. The fluid flows through the horizontal inlet first, while the inflow from the top begins after some time. The data set consists of 30930 cells organized in a Cartesian grid.

Breaking dam: This data set is a flow simulation of a bursting dam with a box-shaped obstacle. The data set consists of 48 time steps. The burst occurs in the first time step. The data set consist of 76505 cells, organized in a Cartesian grid.

Exhaust manifold: This data set is a flow simulation of an exhaust manifold. The data set consists of 69 time steps, covering one inflow from every of the three exits from the cylinders. The data set consists of 36524 cells organized in an unstructured grid.

Turbulent channel flow: This data set is a direct numerical simulation (DNS) of a fully developed turbulent channel flow at frictional Reynolds number Re_τ of 180. The flow domain is bounded by two infinitely large parallel solid walls, and the flow is driven by a constant mean pressure gradient in the stream-wise (x) direction. The boundary conditions are non-slip on the solid walls and periodic else. The data are produced by a *Spectral Element Method* (SEM) solver. The data set consists of 2097152 cells organized in a rectilinear grid.

Rotating vortex rope: This data set is an analytic model of a rotating vortex tube as used by Fuchs et al. [FPH$^+$08] with parameters $R = 0.25$, $k = 2$, $\omega = 0.5$ and $s = 3$. The data set consists of 100 time steps and 504063 cells organized in a regular grid.

9.3.3 The Path Lines and their Attributes

For all data sets, we seed a path line at every cell center and integrate it until the particle leaves the flow domain or the time span described by the data set elapses. The particles are saved at the same time steps the original data sets consist of. Besides the positions we compute both attributes depending on these positions and attributes depending on the velocity field itself, evaluated at the particle positions. The fact that we save particle positions (and attributes) at the original time steps, avoids temporal interpolation of these fields. The investigated attributes with their dimensionality are:

Attributes from positions: position (3), quadratic statistical invariants (3) [LD00], temporal derivative of position (3), torsion (1), curvature (1), winding angle (1), arc length (1), average speed (3), distance actual position to end position (1)

Attributes from velocity: velocity (3), λ_2 (1) [JH95], Hunt's Q (1) [HWM88], normalized helicity (1) [LDS90], scalar field corresponding to the *Eigenvector method* (Sujudi and Haimes) [SH95] (1), scalar field corresponding to the *Eigenvector method for unsteady flow* [FPH$^+$08] (1), scalar field corresponding to the *Cores of swirling particle motion* [WST07] (1), vorticity (3)

It is worthwhile noticing that all attributes that would be constant along the path line (average speed, arc length,...), have been computed from the actual position in the time step to the last time steps. For example, the average speed at time step 0 is the average over the whole path line, at time step *i* the average over the part of the path line starting at its position in time step *i* to its end. This means that we have a time series for all of the attributes. The information of the usual definition is stored at time step 0 and is therefore easily retrievable. In the statistical analysis we consider the components of attributes independently, since no assumptions on the dependencies of the dimensions of the same attribute can be made. If one of the dimensions is characteristic for the data set, however, we include all of them since the meaning of a dimension can change from data set to data set. For example, the x-velocity may be the stream-wise velocity in one data set and the span-wise in another.

Finally, we investigate all data sets in the above described configurations, as well as sampled evenly with respect to the arc length. This is achieved by a reparametrization. The arc length parametrized representation has the advantage that certain shape descriptors become unique, e.g., the combination of curvature and torsion (c.f. *Frenet-Serrat formulae* [Hsi81]).

9.3.4 Results

The main results of our statistical analysis are summarized in Table 9.1. The first column gives the data set in question, the second the parametrization type (time or space). The third column gives the number of retained factors according to Kaiser's criterion, the scree plot test, and the 100% proposition criterion, respectively. The last column gives the factors we consider, according to the principle explained in Sec. 9.3.1. Next to the index of the factor we give the attribute with the highest loading and in brackets the numerical value of the loading. If there is another attribute that is in a 5%-range, we include it as well. For the full statistical output we refer to the included extra material. The abbreviations used in the table as well as in the full output (in alphabetic ordering) are: arc (arc length), avspeed (average speed), curv (curvature), dpos (time-derivative of the position), HuntsQ (Hunt's Q), inv (quadratic statistical invariants), l2 (λ_2), normhel (normalized helicity), pos (position), SH (eigenvector method according to Sujudi and Haimes), swirl (cores of swirling particle motion), tors (torsion), uSH (eigenvector method for unsteady flow, "unsteady Sujudi and Haimes"), vel (velocity), vort (vorticity), windang (winding angle). The statistical analysis has to answer two questions: first, which dimensionality has the attribute space, and second, which attributes represent these dimensions best?

The dimensionality The average number of factors retained by Kaiser's criterion is 5.25 ($SE = 0.28$, $CV = 0.18$), while the scree plot test retains 6.4 ($SE = 0.34$, $CV = 0.18$) factors, on average. From the proportion criterion we see that 7.5 factors ($SE = 0.47$, $CV = 0.22$) are on average sufficient to explain 100% of the variance in the data set. Our criterion, which balances maximum dimension reduction (Kaiser), a soft user-influenceable criterion (scree plot), and the goal of explaining "all" variance of the data set, retains on average 6 factors, being more stable than the other criteria ($SE = 0.28$, $CV = 0.16$).

The representative attributes For our final suggestion of 6 factors we order the attributes according to their frequency. The four most frequent attributes are inv (12), λ_2 (10), Hunt's Q (8) and avspeed (7). We include all but Hunt's Q, since this attribute is coupled to λ_2 in 7 out of 8 occurrences, and λ_2 is known to outperform Hunt's Q[JH95]. Two attributes have frequency 6: startEnd, and vel. We include both. From the now retained five attributes, two, namely inv and startEnd, depend on pos (frequency 3), so we decide to include this attribute to make the 6 attributes we investigate as self-contained as possible.

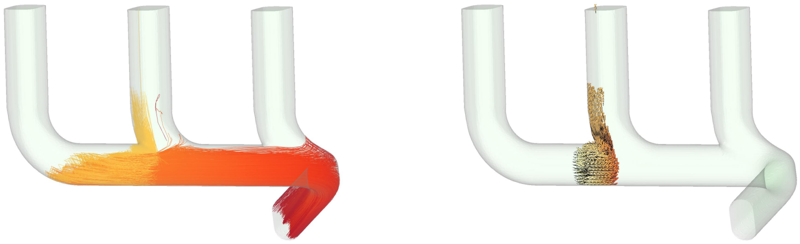

(a) path lines (b) seeding positions

Figure 9.1: (a) path lines meeting the analytic condition defined in Sec. 9.4.2 and (b) their seeding positions. The color coding gives the temporal evolution of the path line (from yellow to red).

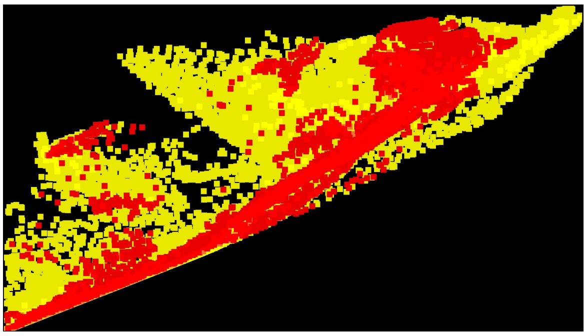
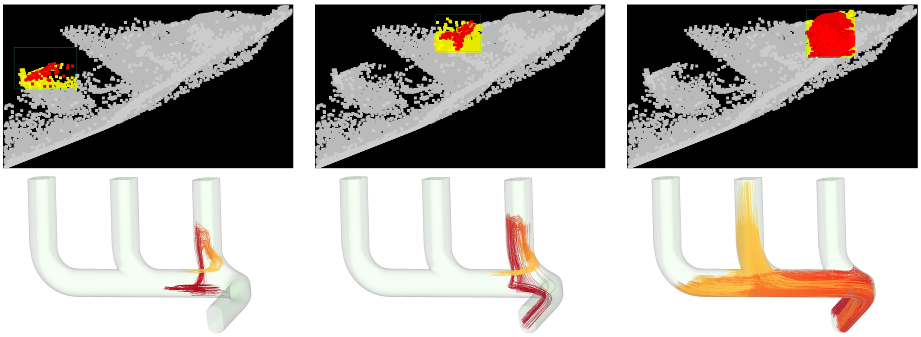


Figure 9.2: Scatter plot of the start-end distance (horizontal) and path line length (vertical). Red points represent path lines starting in or in the vicinity of the middle pipe. For further discussion see Sec. 9.4.2.

Hence, 6 good candidates for representing the path line attribute space are: the quadratic statistical invariants (inv), λ_2, the average speed (avspeed), the start to end distance (startEnd), the velocity (vel) and the positions (pos).

We evaluate our factor choice by rerunning the analysis with the number of factor to retain fixed to 6 and checking the obtained factor loading pattern for cross-loadings and the amount of variance explained as suggested by Costello and Osborne [CO05]. On average 2.4 ($SE = 0.76$, $CV = 1$) attributes out of 28 exhibit crossloadings, and the average variance explained is 0.95 ($SE = 0.02$, $CV = 0.07$) which shows that the proposed factor structure is both expressive and stable for the investigated data sets. The full output of the statistical evaluation can found in the supplemental material.

A remark on time- vs. arc length-parametrization We observe that the arc length-parametrized data set allows a representation with the same number of factors, or fewer, following Kaiser's criterion or the scree plot test. With the 100%



(a) cluster 1 (b) cluster 2 (c) cluster 3

Figure 9.3: Investigation of some of the clusters in Fig. 9.2. The top row shows the actual selections, while the bottom row gives the associated path lines in their 3D context (color coding according to temporal evolution from yellow to red). For the discussion of the figures, we refer to the main text (Sec. 9.4.2).

proportion criterion, no clear trend is observable. It is worthwhile noticing that an arc length parametrization represents the geometry of the path line more faithfully, but lacks information on the dynamics (uniform speed with respect to arc length!). Hence, the trend to be expressible by fewer factors may actually originate for that fact that this representation causes an information loss. On the other hand, we see that the shape descriptors inv perform well under both parametrization. Hence, we may conclude that the geometry-wise advantage of an arc length parametrized data set is too small to outweighed the possible risk of information loss.

9.4 Demonstration

After we determined both dimensionality and representative attributes, we now demonstrate how an interactive visual flow analysis based on our findings can look like. First, we describe the framework used. Then we analyze two different data sets. Both data sets have been investigated in previously published work, which allows us to assess the results we achieve.

9.4.1 The framework

The framework used for this paper is the SimVis software [Dol07]. This software is an *interactive visual analysis* environment, tailored to meet the special requirements of computational fluid dynamics. Apart from multiple linked views, consisting of different information visualization views (e.g., histograms, scatter plots, parallel coordinated), the system provides a passive 3D view for focus+context visualization of the flow domain. Besides this, the frame work offers the opportunity to derive new flow attributes on the fly. For further details we refer to Doleisch's paper on the SimVis software [Dol07] and the references therein. One of the views, that makes the framework especially useful in the context of path line attributes, is the *curve view* [KMG$^+$06, MKO$^+$08]. The curve view is used to display large families of function graphs at once (cf., e.g., Fig. 9.7) plotting the function values against time. Lines are selected by brushing a certain value range for a specific time step. Functions with multiple components can be analyzed component-wise.

9.4.2 Exhaust manifold

This data set has been investigated by Lež et al. [LZM$^+$11]. Their paper is not targeting the question of which attributes to choose for a interactive flow analysis, however, the authors suggest several attribute combinations that they found useful for the case study. These attribute combinations are start to end distance and path line length (arc length), maximum velocity and mean velocity along path line, and maximal curvature and maximal torsion. All those attributes are constant along the path lines. As also mentioned in the original paper by Lež et al., one of the goals in the design of exhaust manifolds is the decrease of flow resistance (so-called back pressure). Hence, the detection of path lines/particles causing back pressure is a natural task in this context.

In order to make the visual analysis based on the different attribute sets comparable, we identify an analytically defined set of path lines that we try then try to retrieve using both the original variable combinations and the here proposed attribute set. We restrict the analysis to particles seeded in the middle tube and its imminent vicinity. We identify particles possibly causing back pressure as those which 1) move upstream (i.e. $\max_t(pos_x(t_0) - pos_x(t)) > 0$, x denoting the axis aligned with the stream and assuming the stream to have positive sign) and 2) are upstream from the middle pipe at some point in time (i.e. $\max_t(pipeboundary_x - pos_x(t)) > 0$, with $pipeboundary_x$ being the position on the x-axis where the inflow pipe is connected to the outlet and under the same assumptions as before). Hence, the path lines in question are those where both parameters are positive. See Fig. 9.1 for an overview over the path lines identified and their seeding positions. Obviously, an ad hoc analytic definition of interesting path lines is only possible in relatively clear and intuitive situation as this. We use this for the sake of comparability only.

First, we investigate the attribute combination start to end distance and path line length. We have preselected particles in the middle pipe and its immediate vicinity. In Fig. 9.2 we see a scatter plot of the two attributes. The red dots represent the path lines to investigate, the yellow dots give the context (i.e. the remaining path lines). In the scatter plots opacity scaling according to point density is used. In their paper, Lež et al. suggest investigating "unusual clusters", and we can visually identify several of them. We select those clusters one after the other and monitor the path lines associated to them (Fig. 9.3). We see that none of the visually distinguishable main

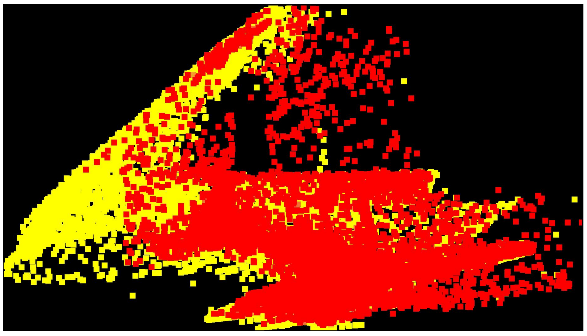


Figure 9.4: Scatter plot of the maximum velocity (horizontal) and the mean velocity along the path line (vertical). As in Fig. 9.2, red points represent path lines starting in or in the vicinity of the middle pipe. For further discussion see the main text in Sec. 9.4.2.

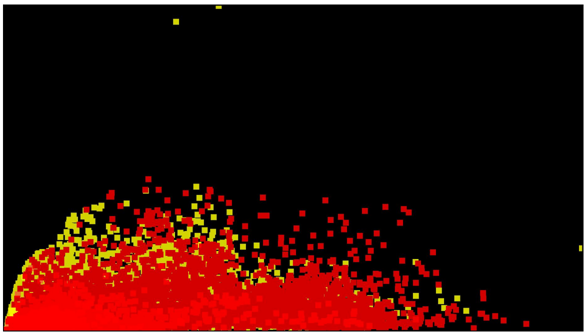
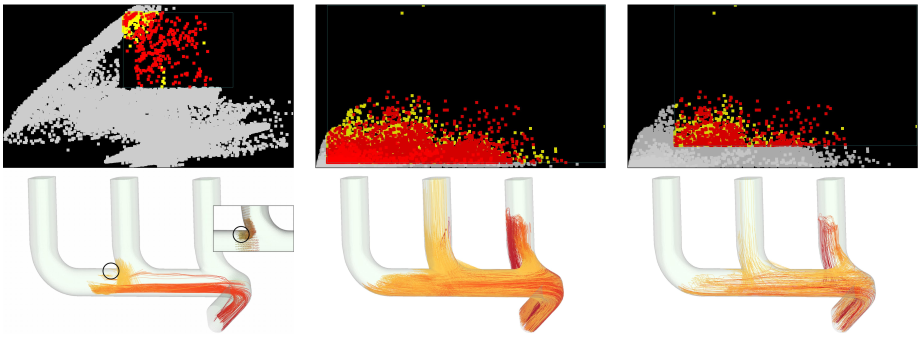


Figure 9.5: Scatter plot of the maximum curvature (horizontal) and the maximum torsion along the path line (vertical). As in Fig. 9.2 and 9.4, red points represent path lines starting in or in the vicinity of the middle pipe. For further discussion see the main text in Sec. 9.4.2.

clusters gives satisfactory results: on the one hand the clusters in Fig. 9.3(a) and Fig. 9.3(b) describe the same path line behavior, the cluster in Fig. 9.3(c) contains both path lines we are interested in (left branch) as well as path lines that seem not to be associated with back pressure (lighter path lines in the right branch). Further refinement of the query could help this, but no visual clues on how to do this are present in the scatter plot.

The next attribute combination investigated is maximum velocity and mean velocity along the path line. Fig. 9.4 shows a scatter plot of these two variables, the colors have the same meaning as before. In this case the visual detection of unusual clusters is harder. The most apparent abnormality seems to be the high share of path lines in question in the center of scatter plot. As Fig. 9.6(a) shows these path lines are indeed associated with the behavior we want to track. However, we systematically miss out on path lines seeded in a specific region (marked up with the

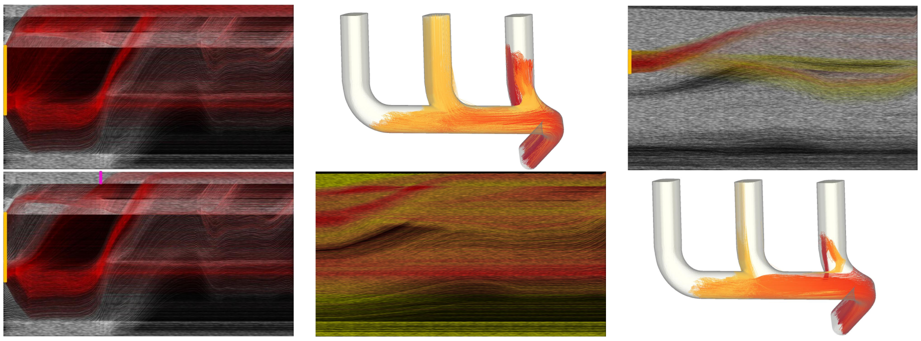


(a) investigation of maximum velocity vs. mean velocity

(b) investigation of maximum curvature vs. the maximum torsion

(c) investigation of maximum curvature vs. the maximum torsion

Figure 9.6: (a) The visually detectable "abnormality" in the scatter plot is selected. The path lines show the targeted behavior, but not all of them can be detected (cf. inset). (b) and (c) Due to the lack of visual clues, different thresholds are assessed, not showing the desired effect. For further discussion we refer to the main text (Sec. 9.4.2). For all three figures: Color coding of the path lines according to their temporal evolution (from yellow to red).



(a)

(b)

(c)

Figure 9.7: Intermediate steps of the interactive flow analysis based on the attribute set proposed in this paper. The time line is left to right, top to bottom. Regular selections are marked in orange, not-selections in pink. The final result can be found in Fig. 9.8. A detailed description of the analysis steps is given in Sec.9.4.2.

circle).

Finally, we investigate the combination of maximum curvature and maximum torsion along the path line (see Fig. 9.5 for the respective scatter plot). Here, no clusters are visible. This means we would have to rely on thresholding. This thresholding gives, however, not the desired results, as seen in Fig. 9.6(b). Choosing a higher threshold refines the selection, but it fails to discriminate different types of flow behavior (Fig. 9.6(c)).

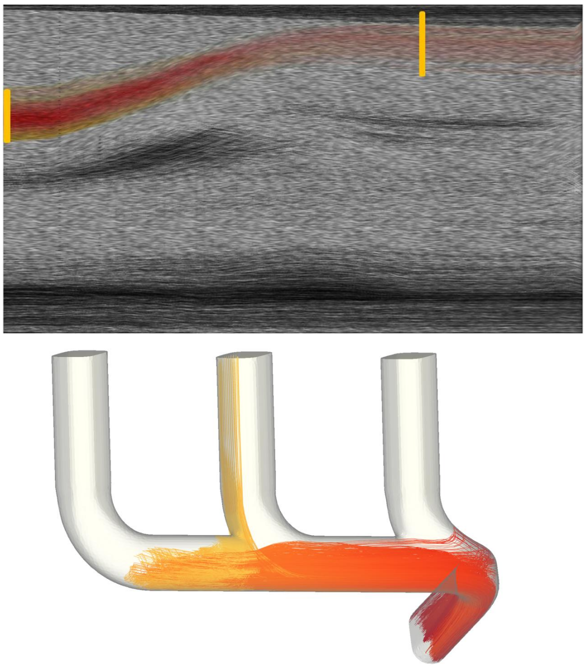


Figure 9.8: The upper branch in the inv2 line plot together with the respective path lines (together with the previous described selection, see Sec.9.4.2).

As a summary, we conclude that, following the state-of-the-art approach as described by Lež et al., we could find only a part of the path lines targeted.

Now we use the attribute set suggested by our statistical analysis. As remarked earlier, all of these attributes are time series. Hence, we make extensive use of the curve view. First we look at the stream-wise position (in the same sense as used earlier). As in the first investigation, we select particles that originate from the middle pipe and its vicinity (Fig. 9.7(a) top). In order to cause back pressure, particles have to still be in the pipe, at the next stroke of the engine. Hence, we discharge ("not-selection") particles that are in the outlet at the time step the next stroke occurs (selection in Fig. 9.7(b) top). In the top of Fig. 9.7(c) we see the path lines corresponding to this selections. The particles that move "upstream" exhibit the same pattern as the once found by the analytic definition.

However, our selection is, at the current point, still containing a number of path lines with clearly different (so to say "correct") flow behavior. Hence, we move to a different attribute to refine our selection. In the bottom of Fig. 9.7(a) we see the time series for the second quadratic statistical invariant (in the following: inv2). We see (at least) two clearly distinguishable patterns: path lines with a medium-high value of inv2 in the beginning of the time series, and others with a rather low value. We select the ones with the higher values and see (cf. Fig. 9.7(c) bottom) that now nearly all path lines exhibit the expected behavior. A small number of path lines is

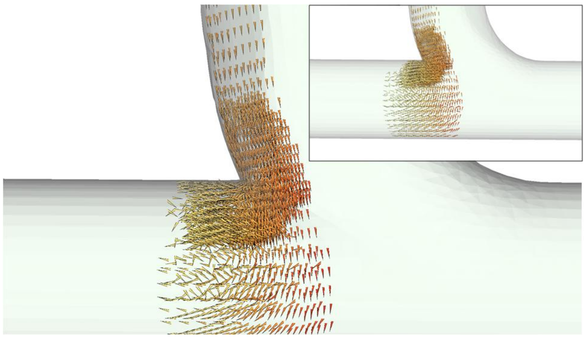
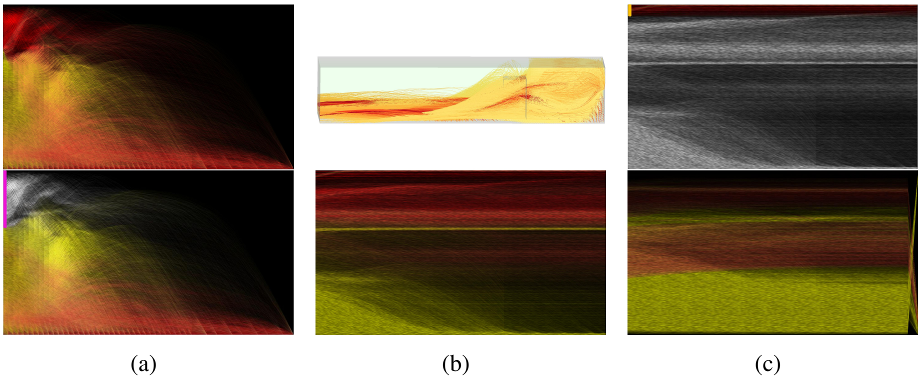


Figure 9.9: Close-up of the seeding positions of the path lines in Fig. 9.8 compared to the seeding points of the reference path lines. We see that the difference between those two sets is hardly detectable.



(a) (b) (c)

Figure 9.10: Intermediate steps of the interactive flow analysis based on the attribute set proposed in this paper. The time line is top to bottom, left to right. The final result can be found in Fig. 9.11 and Fig. 9.12. A detailed description of the analysis steps is given in Sec.9.4.3.

not of the expected type, representing particles being sucked in the rightmost tube. In fact, also the selected time series of inv2 have two branches (upper and lower, see Fig.9.7(b) bottom).

As Fig.9.8 shows the two branches are indeed associated to the different types of path line behavior present. Comparing the seed points of the path lines found by our analysis to the seed points of the reference path lines, we see a clear correspondence of the two sets (Fig. 9.9, in contrast to the situation in Fig. 9.6(a) bottom).

We see that the interactive flow analysis of the data set based on our suggestions is able to find the targeted path lines. In addition, the process is intuitive in the sense that different flow behavior is reflected by clearly distinguishable clusters in the attributes. We discussed our results with a domain expert, who confirmed the expressiveness of our results.

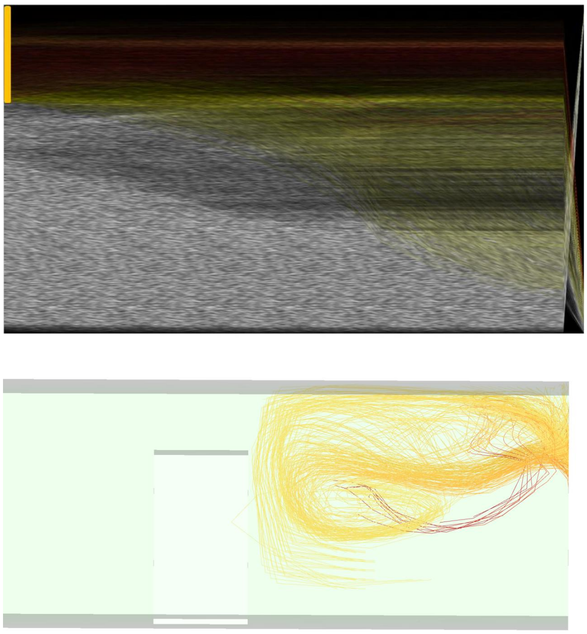


Figure 9.11: Final result of the interactive flow analysis based on the her proposed feature set. We are able to identify the recirculation area in front of the of the obstacle described by Pobitzer et al. [PPF$^+$12]

9.4.3 Breaking dam

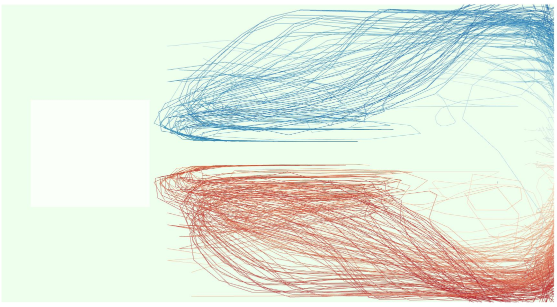


Figure 9.12: Top view on the path lines depicted in Fig. 9.11. The color coding is according to the attribute inv3, blue being low and red being high values.

This data set has been investigated by Pobitzer et al. [PPF$^+$12] in the context of *finite-time Lyapunov exponents* (FTLE). One of the interesting features is a separation structure in front of the obstacle, separating particles passing at the two sides of the obstacle. Another structure is a recirculation zone in front of the obstacle.

Due to its definition, the FTLE approach is not suitable to investigate the internal structure of the recirculation. We therefore investigate this question by means of interactive flow analysis of the attribute set proposed earlier in this paper.

Since we want to target recirculation behavior, we preselect particles seeded upstream from the obstacle. Since the recirculating particles do not pass the obstacle, we can assume that the distance from start to end is not too big. Hence, we exclude the path line cluster associated to high distances from our analysis, using a "not"-selection (Fig. 9.10(a)). The top of Fig. 9.10(b) shows the path lines corresponding to this selection.

Now we investigate one of the other attributes, namely the first quadratic statistical invariant (in the following: inv1). The attribute is chosen since it provides clearest clustering with the current selection (Fig. 9.10(b) bottom). The most distinct cluster is a rather small group of almost horizontal lines in the top. We recall that inv1 is a shape descriptor and ideal recirculation can be thought of as a circular motion. Hence, an almost constant shape descriptor could indicate the wanted behavior. After selecting this curve cluster, we investigate the second quadratic statistical invariant (in the following: inv2). Again, this attribute has been selected for analysis by the same principle as before. We detect two possible clusters and by the same reasoning as before, we select the family of more or less constant lines (cf. bottom of Fig. 9.10(c)). In Fig. 9.11 we see the selection and the resulting path lines. We conclude that we found the recirculation zone Pobitzer et al. found the boundary of in their paper [PPF$^+$12]. Investigating the remaining attributes, we see a clear split in the second quadratic statistical invariant, color coding the path lines according to this attribute, yields Fig. 9.12, revealing that the left-right separation structure is also present inside the recirculation, an insight the FTLE-based analysis of Pobitzer et al. failed to convey.

9.5 Conclusions

In this paper we address the problem of selecting an expressive subset of the path line attribute space for interactive visual flow analysis. Investigating a number of CFD data sets using factor analysis we found that there are common patterns both in dimensionality and attributes associated to them across data sets. We identify 6 path line attributes that represent those factors. The analysis based on the attributes suggested in this papers proves to match, and in part also exceed, previous work, showing how the benefit from the already proven concept of interactive flow analysis can be utterly increased by carefully selecting appropriate attributes. Prior knowledge of which attributes to investigate reduces both computational and storage overhead. In addition, a lower-dimensional data set is easier to handle in the context of IVA and allows for a systematic investigation.

Usually, one of the aims of factor analysis is to identify the underlying factors, at least qualitatively (as mentioned earlier, are the factors assumed to not be measurable directly). Looking at the attributes suggested, we can informally identify one attribute associated to shape (inv), one related to vortices (λ_2), and a bigger group of attributes related to motion (avspeed, startEnd, vel and pos). This may indicate that the motion is the most complicated aspect of path lines, or, more optimistically, better attributes for describing it could be found.

9.6 Acknowledgments

The authors want to thank Carl Erik Wasberg from the Norwegian Defence Research Establishment (FFI) for providing the DNS data set of the turbulent channel flow. The CFD simulation of a flow through a box, T-junction, breaking dam and the exhaust manifold are courtesy of AVL List GmbH, Austria. Thanks to Michael Mayer from AVL List GmbH, Austria, for his valuable feedback. Finally, the authors thank SimVis GmbH, Austria, for providing the framework.

The project SemSeg acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number 226042.

Bibliography

[AMM$^+$08] W. Aigner, S. Miksch, W. Müller, H. Schumann, and C. Tominski. Visual Methods for Analyzing Time-Oriented Data. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):47–60, 2008. Cited on pages 122 and 147.

[AS94] C. Ahlberg and B. Shneiderman. Visual information seeking: tight coupling of dynamic query filters with starfield displays. In *CHI '94: Conference companion on Human factors in computing systems*, pages 313–321, New York, NY, USA, 1994. ACM Press. Cited on pages 35 and 45.

[AVL15] AVL. AVL List GmbH. `http://www.avl.com/`, 2015. [last accessed 25 April 2015]. Cited on pages 21, 29, 39, 74, 102, 106, and 125.

[BDHK05] F. Boecking, U. Dohle, J. Hammer, and S. Kampmann. Passenger Car Common Rail Systems for Future Emissions Standards. *MTZ*, 66(7–8):552–557, 2005. Cited on pages 6, 71, and 107.

[BG04a] E. Bauer and A. Ganopolski. Simulation of the cold climate event 8200 years ago by meltwater outburst from Lake Agassiz. *Paleoceanography*, 19(3):1–13, 2004. Cited on page 149.

[BG04b] H.-J. Bungartz and M. Griebel. Sparse grids. *Acta Numerica*, 13:147–269, 2004. Cited on page 98.

[BGV92] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A Training Algorithm for Optimal Margin Classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory (COLT '92)*, pages 144–152, New York, 1992. ACM. Cited on page 98.

[BH97] W. Boehner and K. Hummel. Common Rail Injection System for Commercial Diesel Vehicles. *SAE Transactions*, (SAE 970345), 1997. Cited on pages 6, 57, and 107.

[BH07] R. Bürger and H. Hauser. Visualization of multi-variate scientific data. In *EuroGraphics State of the Art Reports (STARs)*, pages 117–134, 2007. Cited on pages 147 and 148.

[BINN10] J. Banks, J. S. Carson II, B. L. Nelson, and D. M. Nicol. *Discrete-Event System Simulation, Fifth Edition*. Prentice Hall, 2010. Cited on page 119.

[BMDH07] R. Bürger, P. Muigg, H. Doleisch, and H. Hauser. Interactive cross-detector analysis of vortical flow data. In *Proceedings of CMV 2007*, pages 98–110. IEEE, 2007. Cited on pages 183, 185, and 186.

[BMMS91] A. Buja, J. A. McDonald, J. Michalak, and W. Stuetzle. Interactive data visualization using focusing and linking. In *VIS '91: Proceedings of the 2nd IEEE Conference on Visualization '91*, pages 156–163. IEEE Computer Society Press, 1991. Cited on page 27.

[BMPM12] M. Booshehrian, T. Möller, R. M. Peterman, and T. Munzner. Vismon: Facilitating Analysis of Trade-Offs, Uncertainty, and Sensitivity In Fisheries Management Decision Making. *Computer Graphics Forum*, 31(3pt3):1235–1244, June 2012. Cited on pages 98 and 103.

[BP02] D. Bauer and R. Peikert. Vortex Tracking in Scale-space. In *Data Visualization 2002: Proc. of the 4th Joint* EUROGRAPHICS – IEEE TCVG *Symp. on Visualization (VisSym 2002)*, pages 233–240, Aire-la-Ville, Switzerland, 2002. Eurographics Association. Cited on page 171.

[BP10] W. Berger and H. Pringer. Interactive Visual Analysis of Multiobjective Optimizations. In *Proceedings of the 2010 IEEE Symposium on Visual Analytics Science and Technology*, pages 215–216, 24–29 October 2010. Cited on page 97.

[BPFG11] W. Berger, H. Piringer, P. Filzmoser, and E. Gröller. Uncertainty-Aware Exploration of Continuous Parameter Spaces Using Multivariate Prediction. *Computer Graphics Forum*, 30(3):911–920, June 2011. Cited on page 98.

[BS94] D. C. Banks and B. A. Singer. Vortex tubes in turbulent flows: identification, representation, reconstruction. In *Visualization '94: Proceedings of IEEE Conference on Visualization '94*, pages 132–139, CP14, Oct 1994. Cited on page 171.

[BS95] D. C. Banks and B. A. Singer. A predictor-corrector technique for visualizing unsteady flow. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):151–163, Jun 1995. Cited on page 171.

[BSM+13] S. Bergner, M. Sedlmair, T. Möller, S. Nabi Abdolyousefi, and A. Saad. ParaGlide: Interactive Parameter Space Partitioning for Computer Simulations. *IEEE Transactions on Visualization and Computer Graphics*, 19(9):1499–1512, September 2013. Cited on page 98.

[Cat66] R. B. Cattell. The scree test for the number of factors. *Multivariate Behavioral Research*, 1(2):245–276, 1966. Cited on page 187.

[CK98] J. V. Carlis and J. A. Konstan. Interactive visualization of serial periodic data. In *UIST '98: Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology*, pages 29–38. ACM Press, 1998. Cited on page 147.

[Cle85] W. S. Cleveland. *The Elements of Graphing Data*. Wadsworth Publishing Co., Belmont, CA, USA, 1985. Cited on page 46.

[CMN+04] P. Chaufour, G. Millet, S. Neyrat, M. Hedna, and E. Botelle. Advanced modelling of a heavy-truck unit-injector system and its applications in the engine design process. *Diesel Fuel Injection and Sprays*, 2004. Cited on pages 121, 122, 123, and 138.

[CMS99] S. K. Card, J. D. Mackinlay, and B. Shneiderman. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999. Cited on pages 2, 25, 48, and 100.

[CMS03] S. K. Card, J. Mackinlay, and B. Shneiderman, editors. *The Craft of Information Visualization: Reading and Reflections*. Morgan Kaufmann Publishers, 2003. Cited on page 25.

[CO05] A. B. Costello and J. W. Osborne. Best practices in exploratory factor analysis: Four recommendations for getting the most from your analysis. *Practical Assessment, Research & Evaluation*, 10:173–178, 2005. Cited on pages 187, 188, and 191.

[Cox69] H. S. M. Coxeter. *Introduction to Geometry (2nd ed.)*. Wiley & Sons, 1969. Cited on page 30.

[CPH12] C.Turkay, J. Parulek, and H. Hauser. Dual analysis of DNA microarrays. In *Proceedings of the 12th International Conference on Knowledge Management and Knowledge Technologies*, pages 26:1–8, New York, 2012. ACM. Cited on page 106.

[CRS96] J. X. Chen, D. Rine, and H. D. Simon. Advancing interactive visualization and computational steering. *IEEE Computational Science & Engineering*, 3(4):13–17, Winter 1996. Cited on pages 73 and 97.

[CV95] C. Cortes and V. Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, 1995. Cited on page 98.

[DGH03] H. Doleisch, M. Gasser, and H. Hauser. Interactive feature specification for focus+context visualization of complex simulation data. In *Data Visualization 2003: Proc. of the 5th Joint* EUROGRAPHICS – IEEE TCVG *Symp. on Visualization (VisSym 2003)*, pages 239–248, Aire-la-Ville, Switzerland, 2003. Eurographics Association. Cited on pages 4, 43, 51, 121, 126, 147, and 171.

[DMG$^+$05] H. Doleisch, M. Mayer, M. Gasser, P. Priesching, and H. Hauser. Interactive feature specification for simulation data on time-varying grids. In *Proceedings of the Conference Simulation and Visualization 2005 (SimVis 2005)*, pages 291–304, 2005. Cited on pages 46, 73, and 171.

[DMH04] H. Doleisch, P. Muigg, and H. Hauser. Interactive Visual Analysis of Hurricane Isabel with SimVis. Technical Report TR-VRVis-2004-058, VRVis Research Center in Vienna, Austria, 2004. Cited on page 176.

[Dol07] H. Doleisch. SimVis: Interactive Visual Analysis of Large and Time-Dependent 3D Simulation Data. In *Proceedings of Winter Simulation Conference 2007*, pages 712–720, 2007. Cited on pages 192 and 193.

[EGG$^+$12] D. Engel, K. Greff, C. Garth, K. Bein, A. Wexler, B. Hamann, and H. Hagen. Visual Steering and Verification of Mass Spectrometry Data Factorization in Air Quality Research. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2275–2284, December 2012. Cited on page 98.

[FH09] R. Fuchs and H. Hauser. Visualization of multi-variate scientific data. *Computer Graphics Forum*, 28(6):1670–1690, 2009. Cited on page 184.

[FMH08] W. Freiler, K. Matković, and H. Hauser. Interactive Visual Analysis of Set-Typed Data. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1340–1347, Nov 2008. Cited on page 3.

[Fod02] I. K. Fodor. A survey of dimension reduction techniques. Technical Report UCRL-ID-148494, Lawrence Livermore National Laboratory (LLNL), 2002. Cited on pages 184 and 185.

[FOX] FOX Toolkit, http://www.fox-toolkit.org. Cited on page 35.

[FP98] B. Francis and J. Pritchard. Visualisation of historical events using Lexis pencils: a case study. In D. Unwin and P. Fisher, editors, *Case* *studies of visualisation in the social sciences*. JISC Advisory group on Computer Graphics, 1998. Cited on page 147.

[FPH+08] R. Fuchs, R. Peikert, H. Hauser, F. Sadlo, and P. Muigg. Parallel vectors criteria for unsteady flow vortices. *IEEE Transactions on Visualization and Computer Graphics*, 14(3):615–626, 2008. Cited on page 189.

[FS95] K. Fishkin and M. C. Stone. Enhanced dynamic queries via movable filters. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 415–420. ACM Press/Addison-Wesley Publishing Co., 1995. Cited on page 35.

[FWR99] Y.-H. Fua, M. O. Ward, and E. A. Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. In *VIS '99: Proceedings of the conference on Visualization '99*, pages 43–50, Los Alamitos, CA, USA, 1999. IEEE Computer Society Press. Cited on page 46.

[GC97] M. Goldfarb and N. Celanovic. A lumped parameter electromechanical model for describing the nonlinear behavior of piezoelectric actuators. *ASME Journal of Dynamic Systems, Measurements, and Control*, 119:478–485, September 1997. Cited on page 72.

[GK03] V. González and A. Kobsa. Benefits of information visualization systems for administrative data analysts. In *Proceedings of Information Visualization 2003*, pages 331–336. IEEE Computer Society, 2003. Cited on page 45.

[GLL91] A. Globus, C. Levit, and T. Lasinski. A tool for visualizing the topology of three-dimensional vector fields. In *VIS '91: Proceedings of the 2nd IEEE Conference on Visualization '91*, pages 33–40, 408, Oct 1991. Cited on page 171.

[GM04] K. Gruchalla and J. Marbach. Immersive Visualization of the Hurricane Isabel Dataset. In *Proceedings of the 15th IEEE Visualization Conference (Vis'04), contest entry*, 2004. Cited on page 176.

[GMA+11] H. Gan, K. Matković, A. Ammer, W. Purgathofer, D. Bennett, and M. Terblanche. Interactive visual analysis of a large ICU database: a novel approach to data analysis. *Critical Care*, 15(1), 2011. Cited on page 3.

[GRH07] M. A. Green, C. W. Rowley, and G. Haller. Detection of Lagrangian coherent structures in 3D turbulence. *Journal of Fluid Mechanics*, 572:111–120, Feb 2007. Cited on page 172.

[GRW$^+$00] 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 of the IEEE Conference on Visualization 2000*, pages 489–492, 2000. Cited on pages 4 and 46.

[GTB03] G. Greeves, S. Tullis, and B. Barker. Advanced two-actuator EUI and emission reduction for heavy-duty diesel engines. *SAE Transactions*, 112(3):914–931, 2003. Cited on pages 121, 122, 123, and 138.

[GTS04] Chr. Garth, X. Tricoche, and G. Scheuermann. Tracking of vector field singularities in unstructured 3D time-dependent datasets. In *VIS '04: Proceedings of the IEEE Conference on Visualization 2004*, pages 329–336, 2004. Cited on page 171.

[Har76] H. H. Harman. *Modern Factor Analysis*. The University of Chicago Press, Chicago, IL U.S.A., 3rd edition, 1976. Cited on page 186.

[Hau06] H. Hauser. *Generalizing Focus+Context Visualization, in Scientific Visualization: The Visual Extraction of Knowledge from Data*, chapter Generalizing Focus+Context Visualization, pages 305–327. Springer, 2006. Cited on page 148.

[HBMS03] H. Hochheiser, E. H. Baehrecke, S. M. Mount, and B. Shneiderman. Dynamic querying for pattern identification in microarray and genomic data. In *Proceedings IEEE International Conference on Multimedia and Expo*, Baltimore, MD, 2003. IEEE Computer Society. Cited on page 26.

[HBW06] D. H. House, A. S. Bair, and C. Ware. An approach to the perceptual optimization of complex visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):509–521, 2006. Cited on page 148.

[Hen98] C. Henze. Feature Detection in Linked Derived Spaces. In *VIS '98: Proceedings of the IEEE Conference on Visualization '98*, volume 0, page 87, Los Alamitos, CA, USA, 1998. IEEE Computer Society. Cited on page 73.

[HEvLS03] H. Hagen, A. Ebert, R. H. van Lengen, and G. Scheuermann. Scientific visualization: methods and applications. In *SCCG '03: Proceedings of the 19th spring conference on Computer graphics*, pages 23–33, New York, NY, USA, 2003. ACM. Cited on pages 73 and 97.

[HGP99] P. Hoffman, G. Grinstein, and D. Pinkney. Dimensional anchors: a graphic primitive for multidimensional multivariate information visualizations. In *Proceedings of the 1999 Workshop on New Paradigms in*

Information Visualization and Manipulation in Conjunction with the Eighth ACM Internation Conference on Information and Knowledge Management, NPIVM ’99, pages 9–16, 1999. Cited on page 46.

[HHWN02] S. Havre, E. Hetzler, P. Whitney, and L. Nowell. ThemeRiver: Visualizing thematic changes in large documents collections. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):9–20, 2002. Cited on pages 122 and 147.

[HLD02] H. Hauser, F. Ledermann, and H. Doleisch. Angular brushing of extended parallel coordinates. In *INFOVIS ’02: Proc. of the IEEE Symposium on Information Visualization (InfoVis’02)*, page 127, Washington, DC, USA, 2002. IEEE Computer Society. Cited on page 126.

[HS04] H. Hochheiser and B. Shneiderman. Dynamic query tools for time series data sets: Timebox widgets for interactive exploration. *Information Visualization*, 3(1):1–18, 2004. Cited on pages 46, 52, and 147.

[Hsi81] C.-C. Hsiung. *A first course in differential geometry*. Wiley, New York, 1981. Cited on page 190.

[Hun87] J. C. R. Hunt. Vorticity and vortex dynamics in complex turbulent flows. *Canadian Society for Mechanical Engineering, Transactions (ISSN 0315-8977)*, 11(1):21–35, 1987. Cited on page 171.

[HWM88] J. C. R. Hunt, A. A. Wray, and P. Moin. Eddies, stream and convergence zones in turbulent flows. In *2. Proc. of the 1988 Summer Program*, pages 193–208, 1988. Cited on page 189.

[ID90] A. Inselberg and B. Dimsdale. Parallel coordinates: a tool for visualizing multi-dimensional geometry. In *VIS ’90: Proceedings of the First IEEE Conference on Visualization ’90*, pages 361–378, 1990. Cited on page 46.

[JH95] J. Jeong and F. Hussain. On the identification of a vortex. *Journal of Fluid Mechanics*, 285:69–84, 1995. Cited on pages 189 and 190.

[JMT05] M. Jiang, R. Machiraju, and D. Thompson. Detection and visualization of vortices. In *The Visualization Handbook*, pages 295–309. Academic Press, 2005. Cited on page 183.

[Joh04] C. Johnson. Top scientific visualization research problems. *IEEE Computer Graphics and Applications*, 24(4):13–17, 2004. Cited on pages 73 and 171.

[JPH$^+$99] C. Johnson, S. G. Parker, C. Hansen, G. L. Kindlmann, and Y. Livnat. Interactive simulation and visualization. *IEEE Computer*, 32(12):59–65, December 1999. Cited on pages 73 and 97.

[JS98] D. F. Jerding and J. T. Stasko. The information mural: A technique for displaying and navigating large information spaces. *IEEE Transactions on Visualization and Computer Graphics*, 4(3):257–271, July-September 1998. Cited on page 121.

[JW07] R. A. Johnson and D. W. Wichern. *Applied Multivariate Statistical Analysis*. Pearson Prentice Hall, Upper Saddle River, NJ U.S.A., 6th edition, 2007. Cited on page 184.

[KAF$^+$08] D. Keim, G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, and G. Melançon. Visual Analytics: Definition, Process, and Challenges. In A. Kerren, J. T. Stasko, J.-D. Fekete, and Ch. North, editors, *Information Visualization*, volume 4950 of *Lecture Notes in Computer Science*, pages 154–175. Springer Berlin Heidelberg, 2008. Cited on page 4.

[Kai60] H. F. Kaiser. The application of electronic computers to factor analysis. *Educational and Psychological Measurement*, 20(1):141–151, 1960. Cited on page 187.

[Kan01] E. Kandogan. Visualizing multi-dimensional clusters, trends, and outliers using star coordinates. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 107–116, 2001. Cited on page 46.

[Kei00] D. A. Keim. Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):59–78, 2000. Cited on page 46.

[Kei01] D. A. Keim. Visual exploration of large data sets. *Communications of the ACM*, 44(8):38–44, August 2001. Cited on pages 121 and 147.

[KH13] J. Kehrer and H. Hauser. Visualization and Visual Analysis of Multifaceted Scientific Data: A Survey. *IEEE Transactions on Visualization and Computer Graphics*, 19(3):495–513, March 2013. Cited on page 4.

[KKEM10] D. A. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann. *Mastering the Information Age - Solving Problems with Visual Analytics*. Eurographics Association, 2010. Cited on page 4.

[Kle07] J. P. C. Kleijnen. *Design and Analysis of Simulation Experiments*. International Series in Operations Research & Management Science. Springer, New York, 2007. Cited on page 97.

[KMDH11] J. Kehrer, P. Muigg, H. Doleisch, and H. Hauser. Interactive Visual Analysis of Heterogeneous Scientific Data across an Interface. *IEEE*

Transactions on Visualization and Computer Graphics, 17(7):934–946, July 2011. Cited on page 4.

[KMG$^+$06] Z. Konyha, K. Matković, D. Gračanin, M. Jelović, and H. Hauser. Interactive visual analysis of families of function graphs. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1373–1385, 2006. Cited on pages 6, 12, 74, 79, 94, 95, 98, 99, 104, 126, 127, 146, 147, 150, and 193.

[KMH03] Z. Konyha, K. Matković, and H. Hauser. Interactive 3D visualization of rigid body systems. In *Proceedings of the IEEE Conference on Visualization 2003*, pages 539–546. IEEE Computer Society, 2003. Cited on pages 45 and 73.

[KTH$^+$02] O. Kreylos, A. M. Tesdall, B. Hamann, J. K. Hunter, and K. I. Joy. Interactive visualization and steering of CFD simulations. In *Data Visualization 2002: Proc. of the 4th Joint* EUROGRAPHICS – IEEE TCVG *Symp. on Visualization (VisSym 2002)*, pages 25–34, Aire-la-Ville, Switzerland, 2002. Eurographics Association. Cited on page 73.

[LD00] C.-H. Lo and H.-S. Don. Invariant representation and matching of space curves. *Journal of Intelligent and Robotic Sysstems*, 28:125–149, June 2000. Cited on page 189.

[LDG98] H. Löffelmann, H. Doleisch, and E. Gröller. Visualizing Dynamical Systems near Critical Points. In *Spring Conference on Computer Graphics and its Applications*, pages 175–184. ISBN, 1998. Cited on page 171.

[LDS90] Y. Levy, D. Degani, and A. Seginer. Graphical visualization of vortical flows by means of helicity. *AIAA Journal*, 28:1347–1352, August 1990. Cited on page 189.

[Lev88] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8(3):29–37, 1988. Cited on page 148.

[LGD$^+$05] R. S. Laramee, C. Garth, H. Doleisch, J. Schneider, H. Hauser, and H. Hagen. Visual analysis and exploration of fluid flow in a cooling jacket. In *Proceedings of the IEEE Visualization 2005*, pages 623–630, 2005. Cited on pages 45 and 73.

[LZM$^+$11] A. Lež, A. Zajic, K. Matković, A. Pobitzer, M. Mayer, and H. Hauser. Interactive exploration and analysis of pathlines in flow data. In *Proceedings of WSCG 2011*, pages 17–24, 2011. Cited on pages 183, 185, 186, and 193.

[Mah02] B. Mahr. Future and potential of diesel injection systems. In *Proceedings of the Thiesel 2002 Conference on Thermo- and Fluid-Dynamic Processes in Diesel Engines*, September 2002. Cited on page 78.

[MBS$^+$04] K. Mahrous, J. Bennett, G. Scheuermann, B. Hamann, and K.I. Joy. Topological segmentation in three-dimensional vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 10(2):198–205, March 2004. Cited on page 171.

[MDG$^+$13] K. Matković, M. Djuras, D. Gračanin, R. Splechtna, B. Stehno, and H. Hauser. Interactive Visual Analysis in the Concept Stage of a Hybrid-Vehicle Design. In *Proceedings of the EuroVA 2013 - EuroVis Workshop on Visual Analytics*, 2013. Cited on page 95.

[MGA$^+$12] K. Matković, H. Gan, A. Ammer, D. Bennett, W. Purgathofer, and M. Terblanche. Interactive Visual Analysis of Intensive Care Unit Data - Relationship between Serum Sodium Concentration, its Rate of Change and Survival Outcome. In P. Richard, M. Kraus, R. S. Laramee, and J. Braz, editors, *GRAPP/IVAPP*, pages 648–659. SciTePress, 2012. Cited on page 3.

[MGF$^+$09] K. Matković, D. Gračanin, W. Freiler, J. Banova, and H. Hauser. Large Image Collections - Comprehension and Familiarization by Interactive Visual Analysis. In A. Butz, B. Fisher, M. Christie, A. Krüger, P. Olivier, and R. Theron, editors, *Smart Graphics*, volume 5531 of *Lecture Notes in Computer Science*, pages 15–26. Springer Berlin Heidelberg, 2009. Cited on page 4.

[MGJ$^+$10] K. Matković, D. Gračanin, M. Jelović, A. Ammer, A. Lež, and H. Hauser. Interactive Visual Analysis of Multiple Simulation Runs Using the Simulation Model View: Understanding and Tuning of an Electronic Unit Injector. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1449–1457, 2010. Cited on pages 12, 95, 98, and 99.

[MGJH08] K. Matković, D. Gračanin, M. Jelović, and H. Hauser. Interactive visual steering - rapid visual prototyping of a common rail injection system. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1699–1706, 2008. Cited on pages 6, 12, 95, 99, and 164.

[MGJH11] K. Matković, D. Gračanin, M. Jelović, and H. Hauser. Interactive Visual Analysis Supporting Design, Tuning, and Optimization of Diesel Engine Injection. In *VisWeek 2011: Discovery Exhibition*, 2011. Cited on pages 95 and 98.

[MGKH07] K. Matković, D. Gračanin, Z. Konyha, and H. Hauser. Color Lines View: An Approach to Visualization of Families of Function Graphs.

In *Proceedings of 11th International Conference on Information Visualization 2007, IV '07*, pages 59–64, July 2007. Cited on page 4.

[MGKH09] K. Matković, D. Gračanin, B. Klarin, and H. Hauser. Interactive visual analysis of complex scientific data as families of data surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 15:1351–1358, 2009. Cited on pages 12, 125, and 126.

[MGS+14] K. Matković, D. Gračanin, R. Splechtna, M. Jelović, B. Stehno, H. Hauser, and W. Purgathofer. Visual Analytics for Complex Engineering Systems: Hybrid Visual Steering of Simulation Ensembles. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1803–1812, 2014. Cited on pages 6 and 12.

[MHSG02] K. Matković, H. Hauser, R. Sainitzer, and M. E. Gröller. Process visualization with levels of detail. In *INFOVIS '02: Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02)*, page 67, Washington, DC, USA, 2002. IEEE Computer Society. Cited on page 120.

[MJJ+05] K. Matković, M. Jelović, J. Jurić, Z. Konyha, and D. Gračanin. Interactive visual analysis and exploration of injection systems simulations. In *Proceedings of the IEEE Conference on Visualization 2005*, pages 391–398, October23–28 2005. Cited on pages 45, 57, 58, 73, 74, 79, and 174.

[MJK+05] K. Matković, J. Jurić, Z. Konyha, J. Krasser, and H. Hauser. Interactive Visual Analysis of Multi-Parameter Families of Function Graphs. In *CMV '05: Proceedings of the Coordinated and Multiple Views in Exploratory Visualization (CMV'05)*, pages 54–62, 2005. Cited on pages 12 and 46.

[MKO+08] Ph. Muigg, J. Kehrer, St. Oeltze, H. Piringer, H. Doleisch, B. Preim, and H. Hauser. A four-level focus+context approach to interactive visual analysis of temporal features in large scientific data. *Computer Graphics Forum*, 27(3):775–782, 2008. Cited on page 193.

[MLP+10] T. McLoughlin, R. S. Laramee, R. Peikert, F. H. Post, and M. Chen. Over two decades of integration-based, geometric flow visualization. *Computer Graphics Forum*, 29(6):1807–1829, 2010. Cited on page 185.

[MM06] K. Miettinen and M. M. Mäkelä. Synchronous approach in interactive multiobjective optimization. *European Journal of Operational Research*, 170(3):909–922, May 2006. Cited on page 97.

[Mon90] M. Monmonier. Strategies for the visualization of geographic time-series data. *Cartographica*, 27(1):30–45, 1990. Cited on page 147.

[MS03] W. Müller and H. Schumann. Visualization for modeling and simulation: Visualization methods for time-dependent data - an overview. In *WSC '03: Proceedings of the 35th Conference on Winter Simulation*, volume 1, pages 737–745. Winter Simulation Conference, 2003. Cited on page 122.

[MvWvL99] J. D. Mulder, J. J. van Wijk, and R. van Liere. A survey of computational steering environments. *Future Generation Computer Systems*, 15(1):119–129, 1999. Cited on pages 73 and 97.

[MW95] A. Martin and M. Ward. High dimensional brushing for interactive exploration of multivariate data. In *Proc. of IEEE Visualization '95*, pages 271–278, 1995. Cited on page 4.

[MWSB12] K. Matković, Ch. Winding, R. Splechtna, and M. Balka. Interactive Visual Analysis of Ethological Studies: Getting Insight from Large Ensembles of Animals' Paths. In K. Matković and G. Santucci, editors, *EuroVA 2012: International Workshop on Visual Analytics*. The Eurographics Association, 2012. Cited on page 4.

[NH06] M. Novotný and H. Hauser. Outlier-preserving focus+context visualization in parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):893–900, 2006. Cited on page 163.

[NS00] C. North and B. Shneiderman. Snap-together visualization: a user interface for coordinating visualizations via relational schemata. In *Proceedings of the working conference on Advanced visual interfaces*, pages 128–135, 2000. Cited on pages 46 and 74.

[Off04] G. Offner. Quality and Validation of Cranktrain Vibration Prediction - Effect of Hydrodynamic Journal Bearing Models. In *Multi-body Dynamics - Monitoring and Simulation Techniques III*, 2004. Cited on page 158.

[Ony09] L. C. Onyiah. *Design and Analysis of Experiments: Classical and Regression Approaches with SAS*. Statistics: Textbooks and Monographs. CRC Press, Boca Raton, FL, 2009. Cited on page 97.

[Ope] OpenGL, http://www.opengl.org. Cited on page 35.

[PBH08] H. Piringer, W. Berger, and H. Hauser. Quantifying and comparing features in high-dimensional datasets. In *Proceedings of InfoVis 2008*, pages 240–245, 2008. Cited on pages 184 and 186.

[PBK10] H. Piringer, W. Berger, and J. Krasser. HyperMoVal: Interactive Visual Validation of Regression Models for Real-Time Simulation. *Computer Graphics Forum*, 29(3):983–992, June 2010. Cited on pages 98 and 103.

[Pea01] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6):559–572, 1901. Cited on pages 184 and 185.

[PGR99] H. Pirkul, R. Gupta, and E. Rolland. VisOpt: a visual interactive optimization tool for P-median problems. *Decision Support Systems*, 26(3):209–223, September 1999. Cited on page 97.

[PJB97] S. G. Parker, C. J. Johnson, and D. Beazley. Computational steering: Software systems and strategies. *IEEE Computational Science & Engineering*, 4(4):50–59, October–December 1997. Cited on pages 72, 73, and 97.

[PK98] H. H. Priebsch and J. Krasser. Simulation of Vibration and Structure Borne Noise of Engines - A Combined Technique of FEM and Multi Body Dynamics. In *CAD-FEM Users' Meeting, Bad Neuenahr*, 1998. Cited on page 158.

[PKH04] H. Piringer, R. Kosara, and H. Hauser. Interactive focus+context visualization with linked 2D/3D scatterplots. In *CMV '04: Proceedings of the Second International Conference on Coordinated & Multiple Views in Exploratory Visualization (CMV'04)*, pages 49–60, 2004. Cited on page 46.

[PLMH12] A. Pobitzer, A. Lež, K. Matković, and H. Hauser. A statistics-based dimension reduction of the space of path line attributes for interactive visual flow analysis. In *Proceedings of IEEE Pacific Visualization Symposium (PacificVis), 2012*, pages 113–120, Feb 2012. Cited on page 12.

[PPB10] D. Pflüger, B. Peherstorfer, and H.-J. Bungartz. Spatially adaptive sparse grids for high-dimensional data-driven problems. *Journal of Complexity*, 26(5):508–522, October 2010. Cited on page 98.

[PPF$^+$12] A. Pobitzer, R. Peikert, R. Fuchs, H. Theisel, and H. Hauser. Filtering of FTLE for visualizing spatial separation in unsteady 3D flow. In R. Peikert, H. Hauser, H. Carr, and R. Fuchs, editors, *Topological Methods in Data Analysis and Visualization II*. Springer, 2012. Cited on pages 198 and 199.

[PR96] P. Pirolli and R. Rao. Table lens as a tool for making sense of data. In *Proceedings of the workshop on Advanced Visual Interfaces (AVI*

'96), pages 67–80, New York, NY, USA, 1996. ACM Press. Cited on page 122.

[PR99] R. Peikert and M. Roth. The "Parallel Vectors" operator-a vector field visualization primitive. In *Vis '99: Proceedings of IEEE Conference on Visualization '99*, pages 263–532, Oct 1999. Cited on page 171.

[PTMB09] H. Piringer, Ch. Tominski, Ph. Muigg, and W. Berger. A Multi-Threading Architecture to Support Interactive Visual Exploration. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1113–1120, November–December 2009. Cited on page 97.

[Puk06] F. Pukelsheim. *Optimal Design of Experiments*, volume 50 of *Classics in Applied Mathematics*. SIAM, Philadelphia, 2006. Cited on page 112.

[PVH+03] F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch. The State of the Art in Flow Visualisation: Feature Extraction and Tracking. *Computer Graphics Forum*, 22(4):775–792, 2003. Cited on pages 171 and 183.

[Rob07] J. C. Roberts. State of the Art: Coordinated & Multiple Views in Exploratory Visualization. In Gennady Andrienko, Jonathan C. Roberts, and Chris Weaver, editors, *Proc. of the 5th International Conference on Coordinated & Multiple Views in Exploratory Visualization*. IEEE CS Press, 2007. Cited on pages 4 and 147.

[Sam06] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Elsevier, Amsterdam, 2006. Cited on pages 2 and 100.

[SBC+04] J. Seo, M. Bakay, Y.-W. Chen, S. Hilmer, B. Shneiderman, and E. P. Hoffman. Interactively optimizing signal-to-noise ratios in expression profiling: project-specific algorithm selection and detection p-value weighting in Affymetrix microarrays. *Bioinformatics*, 20(16):2534–2544, 2004. Cited on page 97.

[SGSM08] T. Salzbrunn, C. Garth, G. Scheuermann, and J. Meyer. Pathline predicates and unsteady flow structures. *Visual Computer*, 24(12):1039–1051, 2008. Cited on pages 183 and 185.

[SH95] D. Sujudi and R. Haimes. Identification of swirling flow in 3-D vector fields. In *Proceedings of 12th Computational Fluid Dynamics Conference.* American Institute of Aeronautics and Astronautics, Jun 1995. Cited on pages 171 and 189.

[She68] D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference*, pages 517–524. ACM Press, 1968. Cited on page 29.

[Shn96] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages*, page 336, 1996. Cited on pages 4 and 52.

[Shn02] B. Shneiderman. Inventing Discovery Tools: Combining Information Visualization with Data Mining. *Information Visualization*, 1(1):5–12, March 2002. Cited on page 4.

[Sim] Simulink – simulation and model-based design, http://www.mathworks.com/products/simulink/. [last accessed 26 june 2010]. Cited on page 120.

[SLM05] S. C. Shadden, F. Lekien, and J. E. Marsden. Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3-4):271 – 304, 2005. Cited on page 172.

[SND04] P. Saraiya, C. North, and K. Duca. An evaluation of microarray visualization tools for biological insight. In *INFOVIS '04: Proceedings of the IEEE Symposium on Information Visualization*, pages 1–8, 2004. Cited on page 45.

[Spe07] C. Spearman. Demonstration of formulæ for true measurement of correlation. *The American Journal of Psychology*, 18(2):161–169, 1907. Cited on pages 184 and 185.

[SS04a] J. Seo and B. Shneiderman. A Rank-by-Feature Framework for Unsupervised Multidimensional Data Exploration Using Low Dimensional Projections. In *InfoVis 2004: Proceedings of the IEEE Symposium on Information Visualization 2004*, pages 65–72, Washington, DC, USA, 2004. IEEE Computer Society. Cited on pages 184 and 186.

[SS04b] A. J. Smola and B. Schölkopf. A Tutorial on Support Vector Regression. *Statistics and Computing*, 14(3):199–222, 2004. Cited on page 98.

[SS06] T. Salzbrunn and G. Scheuermann. Streamline predicates. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1601–1612, 2006. Cited on page 185.

[SS07] T. Salzbrunn and G. Scheuermann. Streamline Predicates as Flow Topology Generalization. In H. Hauser, H. Hagen, and H. Theisel, editors, *Topology-based Methods in Visualization*, Mathematics and Visualization, pages 65–77. Springer Berlin Heidelberg, 2007. Cited on page 171.

[ST98] R. Spence and L. Tweedie. The attribute explorer: information synthesis via exploration. *Interacting with Computers*, 11(2):137 – 146, 1998. Cited on pages 46 and 103.

[STH$^+$09] K. Shi, H. Theisel, H. Hauser, T. Weinkauf, K. Matković, H.-C. Hege, and H.-P. Seidel. Path line attributes - an information visualization approach to analyzing the dynamic behavior of 3D time-dependent flow fields. In H.-C. Hege, K. Polthier, and G. Scheuermann, editors, *Topology-Based Methods in Visualization II*, Mathematics and Visualization, pages 75–88, Grimma, Germany, 2009. Springer. Cited on pages 12, 183, 185, and 186.

[Suh05] D. D. Suhr. Principal component analysis vs. exploratory factor analysis. In *Proceedings of the Thirtieth Annual SAS®Users Group International Conference*. SAS Institute Inc., 2005. Cited on page 184.

[SWE05] T. Schafhitzel, D. Weiskopf, and T. Ertl. Interactive exploration of unsteady 3D flow with linked 2D/3D texture advection. In *CMV '05: Proc. of the Coordinated and Multiple Views in Exploratory Visualization (CMV'05)*, pages 96–105, 2005. Cited on page 46.

[SWH05] D. Stalling, M. Westerhoff, and H.-C. Hege. Amira: A highly interactive system for visual data analysis. In *The visualization handbook*, pages 749–767. Elsevier, 2005. Cited on pages 174 and 179.

[TC05] J. J. Thomas and K. A. Cook, editors. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. National Visualization and Analytics Center, 2005. Cited on pages 43, 45, and 137.

[TC06] J. J. Thomas and K. A. Cook. A visual analytics agenda. *IEEE Computer Graphics and Applications*, 26(1):10–13, January/February 2006. Cited on pages 4, 45, and 137.

[TFH11] C. Turkay, P. Filzmoser, and H. Hauser. Brushing dimensions – a dual visual analysis model for high-dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2591–2599, 2011. Cited on page 186.

[The00] H. Theisel. On Properties of Contours of Trilinear Scalar Fields. In P. J. Laurent, P. Sablonniere, and L. L. Schumaker, editors, *Curve and Surface Design: Saint-Malo 99*. Vanderbilt University Press, 2000. Cited on page 46.

[Tho97] E. Thomsen. *OLAP Solutions: Building Multidimensional Information Systems*. John Wiley & Sons, Inc., New York, 1997. Cited on pages 2, 46, and 100.

[Tip01] M. E. Tipping. Sparse Bayesian Learning and the Relevance Vector Machine. *Journal of Machine Learning Research*, 1:211–244, June 2001. Cited on page 98.

[TKT$^+$00] J. G. Trafton, S. S. Kirschenbaum, T. L. Tsu, R. T. Miyamoto, J. A. Ballas, and P. D. Raymond. Turning pictures into numbers: Extracting and generating information from complex visualizations. *Int. J. Hum.-Comput. Stud.*, 53(5):827–850, 2000. Cited on page 45.

[TLLH13] C. Turkay, A. Lundervold, A.J. Lundervold, and H. Hauser. Hypothesis Generation by Interactive Visual Exploration of Heterogeneous Medical Data. In *Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data*, volume 7947 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2013. Cited on page 106.

[TMH09] S. Tarkkanen, K. Miettinen, and J. Hakanen. Interactive Poster: Interactive Multiobjective Optimization — A New Application Area for Visual Analytics. In *Proceedings of the 2009 IEEE Symposium onVisual Analytics Science and Technology (VAST 2009)*, pages 237–238, October 2009. Cited on page 97.

[TMS$^+$08] R. Trobec, K. Matković, K. Skala, S. Samarin, M. Depolli, and V. Avbelj. Visual Analysis of Heart Reinervation after Transplantation. In *Proceedings of the MIPRO 2008*, pages 283–288, 5 2008. Cited on page 4.

[TR97] T. Tenev and R. Rao. Managing multiple focal levels in table lens. In *Proceedings of the IEEE Symposium on Information Visualization 1997*, pages 59–63,122, 20–21October 1997. Cited on page 122.

[TS03] H. Theisel and H.-P. Seidel. Feature Flow Fields. In *Data Visualization 2003: Proc. of the 5th Joint* EUROGRAPHICS – IEEE TCVG *Symp. on Visualization (VisSym 2003)*, pages 141–148, Aire-la-Ville, Switzerland, 2003. Eurographics Association. Cited on page 171.

[TSDS96] L. Tweedie, R. Spence, H. Dawkes, and H. Su. Externalising abstract mathematical models. In *CHI '96: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 406–412, New York, NY, USA, 1996. ACM Press. Cited on page 46.

[TSH01] X. Tricoche, G. Scheuermann, and H. Hagen. Topology-Based Visualization of Time-Dependent 2D Vector Fields. In D. S. Ebert, J. M. Favre, and R. Peikert, editors, *Data Visualization 2001: Proc. of the 3rd Joint* EUROGRAPHICS – IEEE TCVG *Symp. on Visualization (VisSym 2001)*, Eurographics, pages 117–126. Springer Vienna, 2001. Cited on page 171.

[TSW$^+$05] H. Theisel, J. Sahner, T. Weinkauf, H.-C. Hege, and H. P Seidel. Extraction of parallel vector surfaces in 3D time-dependent fields and application to vortex core line tracking. In *Vis 2005: Proceedings of the IEEE Conference on Visualization 2005*, pages 631–638, Oct 2005. Cited on page 171.

[TSWB94] L. Tweedie, B. Spence, D. Williams, and R. Bhogal. The Attribute Explorer. In *CHI '94: Conference companion on Human factors in computing systems*, pages 435–436, 1994. Cited on page 46.

[Tuf90] E. Tufte. *Envisioning Information*. Graphics Press, Cheshire, Connecticut, 1990. Cited on page 25.

[Tuf01] E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, Connecticut, second edition, 2001. Cited on pages 25, 46, 121, and 147.

[TWHS03] H. Theisel, T. Weinkauf, H.-C. Hege, and H.-P. Seidel. Saddle connectors - an approach to visualizing the topological skeleton of complex 3D vector fields. In *Vis 2003: Proceedings of the IEEE Conference on Visualization 2003*, pages 30–, Washington, DC, USA, 2003. IEEE Computer Society. Cited on page 171.

[TWHS05] H. Theisel, T. Weinkauf, H.-C. Hege, and H.-P. Seidel. Topological methods for 2D time-dependent vector fields based on stream lines and path lines. *IEEE Transactions on Visualization and Computer Graphics*, 11(4):383–394, 2005. Cited on pages 171 and 185.

[twi] Twin cities traffic data archive, http://www.d.umn.edu/ tkwon/tmc-data/tmcarchive.html. Cited on page 47.

[vBK04] W. C. M. van Beers and J. P. C. Kleijnen. Kriging interpolation in simulation: a survey. In *WSC '04: Proceedings of the 36th conference on Winter simulation*, pages 113–121. Winter Simulation Conference, 2004. Cited on pages 73 and 97.

[VR00] J. S. Vetter and D. A. Reed. Real-Time Performance Monitoring, Adaptive Control, and Interactive Steering of Computational Grids. *International Journal of High Performance Computer Applications*, 14(4):357–366, 2000. Cited on page 73.

[WAM01] M. Weber, M. Alexa, and W. Müller. Visualizing Time-Series on Spirals. In *InfoVis 2001: Proceedings of the IEEE Symposium on Information Visualization 2001*, pages 7–13, 2001. Cited on page 147.

[War04] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers, second edition, 2004. Cited on page 148.

[WB97] P. C. Wong and R. D. Bergeron. 30 years of multidimensional multivariate visualization. In *Scientific Visualization, Overviews, Methodologies, and Techniques*, pages 3–33, 1997. Cited on page 46.

[Wea04] C. Weaver. Building Highly-Coordinated Visualizations in Improvise. In *InfoVis 2004: Proceedings of the IEEE Symposium on Information Visualization 2004*, pages 159–166, 2004. Cited on pages 46 and 74.

[Wea05] C. Weaver. Visualizing coordination in situ. In *InfoVis 2005: Proceedings of the IEEE Symposium on Information Visualization 2005*, pages 165–172, 2005. Cited on pages 46 and 74.

[WEE03] D. Weiskopf, G. Erlebacher, and T. Ertl. A texture-based framework for spacetime-coherent visualization of time-dependent vector fields. In *Vis 2003: Proceedings of the IEEE Conference on Visualization 2003*, pages 107–114, Oct 2003. Cited on page 171.

[Wes99] T. G. West. Images and reversals: James Clerk Maxwell, working in wet clay. *ACM SIGGRAPH Computer Graphics*, 33(1):15–17, February 1999. Cited on page 73.

[WFR$^+$10] J. Waser, R. Fuchs, H. Ribičić, B. Schindler, G. Blöschl, and M. E. Gröller. World Lines. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1458–1467, November–December 2010. Cited on page 98.

[WLG97] R. Wegenkittl, H. Löffelmann, and E. Gröller. Visualizing the behavior of higher dimensional dynamical systems. In *Proceedings of the IEEE Visualization (VIS '97)*, pages 119–125, 1997. Cited on page 148.

[WS99] J. J. Van Wijk and E. R. Van Selow. Cluster and calendar based visualization of time series data. In *INFOVIS '99: Proceedings of the 1999 IEEE Symposium on Information Visualization*, page 4, Washington, DC, USA, 1999. IEEE Computer Society. Cited on pages 26 and 31.

[WS05] A. Wiebel and G. Scheuermann. Eyelet particle tracing - steady visualization of unsteady flow. In *Visualization, 2005. VIS 05. IEEE*, pages 607–614, Oct 2005. Cited on page 171.

[WSEE05] D. Weiskopf, F. Schramm, G. Erlebacher, and T. Ertl. Particle and texture based spatiotemporal visualization of time-dependent vector fields. In *Vis 2005: Proceedings of the IEEE Conference on Visualization 2005*, pages 639–646, Oct 2005. Cited on page 171.

[WST07] T. Weinkauf, J. Sahner, and H. Theisel. Cores of swirling particle motion in unsteady flows. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1759–1766, 2007. Cited on page 189.

[WTHS04] T. Weinkauf, H. Theisel, H.-C. Hege, and H.-P. Seidel. Boundary switch connectors for topological visualization of complex 3d vector fields. In *Data Visualization 2004: Proc. of the 6th Joint* EUROGRAPHICS – IEEE TCVG *Symp. on Visualization (VisSym 2004)*, pages 183–192, Aire-la-Ville, Switzerland, 2004. Eurographics Association. Cited on page 171.

[WvTB+07] P. Wenisch, C. van Treeck, A. Borrmann, E. Rank, and O. Wenisch. Computational steering on distributed systems: Indoor comfort simulations as a case study of interactive cfd on supercomputers. *International Journal of Parallel, Emergent and Distributed Systems*, 22(4):275–291, 2007. Cited on page 73.

[YRW07] D. Yang, E. A. Rundensteiner, and M. O. Ward. Analysis guided visual exploration of multivariate data. In *VAST 2007: Proceedings of IEEE Symposium on Visual Analytics Science and Technology 2007*, pages 83–90, 2007. Cited on page 4.

[YWRH03] J. Yang, M. O. Ward, E. A. Rundensteiner, and S. Huang. Visual hierarchical dimension reduction for exploration of high dimensional datasets. In *Data Visualization 2003: Proc. of the 5th Joint* EUROGRAPHICS – IEEE TCVG *Symp. on Visualization (VisSym 2003)*, pages 19–28. Eurographics Association, 2003. Cited on page 186.

[Zen91] C. Zenger. Sparse Grids. In W. Hackbush, editor, *Parallel Algorithms for Partial Differential Equations, Proceedings of the Sixth GAMM-Seminar, Kiel, 1990*, volume 31 of *Notes on Numerical Fluid Mechanics*, pages 241–251. Vieweg-Verlag, 1991. Cited on page 98.

[ZHT06] H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286, jun. 2006. Cited on page 184.

[ZZR03] L. Zhang, A. Zhang, and M. Ramanathan. Enhanced Visualization of Time Series through Higher Fourier Harmonics. In Mohammed J. Zaki, Jason T. L. Wang, and Hannu T. T. Toivonen, editors, *Proceedings of the 3rd ACM SIGKDD Workshop on Data Mining in Bioinformatics*, 2003. Cited on page 26.