Interactive Visual Analysis of Process Data

Ove Daae Lampe

Dissertation for the degree of Philosophiae Doctor (PhD)

Supervised by Helwig Hauser
Co-supervised by Christopher Giertsen

Institute for Informatics
University of Bergen
Norway

September 2011

To Elena,
Fillip and Patrik

Scientific Environment

This work has been performed at the Visualization group at the Informatics department at the University of Bergen, while beeing an employee at the Computing department at Christian Michelsen Research. I have also been ascosiated with the ICT Research school. This work is part of the research collaboration project eLAD (Petromaks Project 176018/S30, 2007-2011), partnered by International Research Institute of Stavanger, IRIS, Christian Michelsen Research, CMR and the Institute for Energy Technology, IFE. Parts of this research was performed while on a research stay in, and in close collaboration with, the VIDi group at the University of California Davis.

UNIVERSITY OF BERGEN
Department of Informatics

WWW.CMR.NO

ICT

Research School in
Information and Communication Technology

Acknowledgments

First of all I would like to thank my loving wife for the support and understanding I got during my PhD project. I would also like to thank my two kids, Fillip and Patrik, for sacrificing their time for countless evenings (especially close to deadlines) without their dad. I am very grateful to Helwig Hauser for taking me on as a student, and as my supervisor. Hauser has provided me with genuinely helpful guidance through this process, and introduced me into the world of visualization. He has also set the standard for me to strive to follow up on, of which I hope the quality of thesis reflects. I further would like to thank Kåre Villanger and Tor Langeland from CMR in both supporting me in starting with this PhD as well as during this project. I am also very grateful to CMR for providing me with this opportunity, and also for the financial support which made the decision to go "back to school" so much easier for my family to accept. I also acknowledge the financial support for this PhD project from the collaboration project eLAD (Petromaks Project 176018/S30, 2007-2011), partnered by International Research Institute of Stavanger, IRIS, Christian Michelsen Research, CMR and the Institute for Energy Technology, IFE. I would like to thank my parents, Aud Ingrid and Iver Daae Lampe for supporting me, and caring for my family at home during my long hours leading up to deadlines, and when away at conferences. I would also like to thank my brother Claus and his wife Evy-Helen Daae Lampe for being there when I needed to unwind from my PhD work. I especially thank Solvår Hjellestad for being a great inspirational source and who taught me how to love natural sciences through our classes at Rå School. During my visits to Iris in Stavanger, Eric Cayeux was always welcoming, and I would like to thank him for teaching me almost everything I know about drilling. I would also like to thank Gerhard Nygaard at Iris for the support he gave me. Furthermore I would like to thank Mike Herbert at ConocoPhillips for an open and standing welcome to his operations center to observe how they perform their integrated operations on drilling. I am grateful for the opportunity to work at UC Davis, staying in California with my family for three months, given to me by Kwan-Liu Ma of UC Davis. There I also received a great amount of help from Carlos Correa, co-authoring a paper with me, and from Tarik Crnovrsanin in welcoming me to this new city. I would also like to thank Rolf Wilhelm Rasmussen for being my mentor in my early career in the industry, and introducing me to great quality coding, plug-in based architectures and not the least Python. Furthermore I would like to thank: Johannes Kehrer for putting up with me, sharing an office with me all these years, but also for being a good friend and always (OK, almost always)

providing meaningful insights on our research topics. Stig Sandø and Knut Arild Erstad for being good friends and sharing many of my interests. Ivan Viola for proving that academic scientists are cool, when I first entered this world, and for the countless of helpful discussions we have shared since that. Daniel Patel, Endre Lidal, Åsmund Birkeland, Veronica Soltzenova, Paolo Angelelli, Armin Pobitzer and Julius Parulek, Cagatay Turkay, Mattia Natali, Andrea Brambilla and Jean-Paul Balabanian for being good friends, and making my stay at HiB welcoming, interesting and fun. Ole-Morten "Drastiske" Hansen for hours of unwinding nerd activities. Sveinung Sivertsen, Magne and Janne Torsvik, Eivind Samdal, Simon Våge for being good friends. Mark Nijhof for the many nights out at the cinema, and his wife Mona for proofing my English writing. Rune Dalmo for the many hours coding together, producing amazing results and learning OpenGL. Vegar Kleppe who always has interesting feedback on *anything* GPU related, and for critical insight on several new ideas. And, finally, Randall Munroe and Jorge Cham for providing material for procrastination.

Abstract

Data gathered from processes, or process data, contains many different aspects that a visualization system should also convey. Aspects such as, temporal coherence, spatial connectivity, streaming data, and the need for in-situ visualizations, which all come with their independent challenges. Additionally, as sensors get more affordable, and the benefits of measurements get clearer we are faced with a deluge of data, of which sizes are rapidly growing. With all the aspects that should be supported and the vast increase in the amount of data, the traditional techniques of dashboards showing the recent data becomes insufficient for practical use. In this thesis we investigate how to extend the traditional process visualization techniques by bringing the streaming process data into an interactive visual analysis setting. The augmentation of process visualization with interactivity enables the users to go beyond the mere observation, pose questions about observed phenomena and delve into the data to mine for the answers. Furthermore, this thesis investigates how to utilize frequency based, as opposed to item based, techniques to show such large amounts of data. By utilizing Kernel Density Estimates (KDE) we show how the display of streaming data benefit by the non-parametric automatic aggregation to interpret incoming data put in context to historic data.

Contents

Scientific Environment iii

Acknowledgments v

Abstract vii

Related Publications xiii

I Overview

1 Introduction 3
- 1 Problem Statement . . . 4
- 2 Contributions . . . 5
- 3 Thesis Structure . . . 5

2 Related Work on Interactive Visual Analysis of Process Data 7
- 1 Process Visualization . . . 7
- 2 Streaming Data . . . 8
- 3 Interactive Visual Analysis . . . 10
  - 3.1 Interactive Visual Exploration and Analysis . . . 11
  - 3.2 Interaction and Multiple Coordinated Views . . . 11
- 4 Comparative Visualization . . . 12
- 5 Kernel Density Estimates in Visualization . . . 12

3 Interactive Visual Analysis of Process Data 15
- 1 Kernel Density Estimates for Continuous Data . . . 15
  - 1.1 Kernel Density Estimation . . . 16
  - 1.2 The Line Kernel . . . 17
  - 1.3 Curve Density Estimates . . . 20
- 2 Interaction with Kernel Density Estimates . . . 22
  - 2.1 Screen-Space Bandwidth Automation . . . 23
  - 2.2 Model Sketching . . . 25
  - 2.3 Differential Analysis . . . 26
- 3 Curve-centric Volume Reformation . . . 28

4 Demonstration **33**
1 Drilling for Oil and Gas . . . 33
1.1 Streaming Drilling Data . . . 33
1.2 Positional Uncertainty . . . 36
2 Differential Analysis of AIS Data . . . 38

5 Conclusion and Future Work **41**

II Scientific Results

A Interactive Visualization of Streaming Data with Kernel Density Estimation **45**
1 Introduction . . . 47
2 Related Work . . . 48
3 Kernel Density Estimation . . . 50
4 Reconstructing the Distribution of a Third Attribute . . . 52
5 Reconstructing Time . . . 53
6 Interactivity and Analysis . . . 55
7 Demonstration . . . 58
7.1 AIS Ship Traffic . . . 58
7.2 Drilling operations . . . 59
7.3 Commercial Air Traffic . . . 60
8 Technical Details and Accuracy . . . 61
8.1 Kernel Density Estimation on the GPU . . . 61
8.2 Error Estimation . . . 63
9 Summary and Conclusions . . . 65
10 acknowledgements . . . 66

B Curve Density Estimates **67**
1 Introduction . . . 69
2 Related Work . . . 72
3 Curve Density Estimates (CDE) . . . 74
4 Technical Details . . . 78
5 Applications . . . 79
5.1 High Frequency Curves . . . 79
5.2 Prediction Curves . . . 81
5.3 Process Visualization . . . 83
6 Summary and Conclusions . . . 84
7 Acknowledgements . . . 84

C Curve-Centric Volume Reformation for Comparative Visualization **85**
1 Introduction . . . 87

2 Related Work . . . 88
3 Theory . . . 90
3.1 Moving Coordinate Frames . . . 90
3.2 Curve-Centric Reformation . . . 93
3.3 Radial Ray-Casting . . . 95
3.4 A Common Axis for Comparative Visualization . . . 97
4 Application Cases . . . 98
4.1 Well-Centric Visualizations for the Petroleum Industry . . . 99
4.2 Streamline-Centric Visualization . . . 102
5 Summary and Conclusion . . . 103
6 Acknowledgments . . . 107

D Interactive Difference Views for Temporal Trend Discovery in Multivariate Movement Data 109
1 Introduction . . . 110
2 Related Work . . . 112
3 Interactive Difference Views . . . 114
3.1 Interactive and Iterative Visual Analysis . . . 115
3.2 Quantitative Difference Visualizations . . . 116
3.3 Large Datasets . . . 118
4 Answering the Application Questions . . . 119
5 Summary and Conclusions . . . 123
6 Acknowledgements . . . 124

E Interactive Model Prototyping in Visualization Space 125
1 Introduction . . . 126
2 Related Work . . . 127
3 The Basic Idea . . . 128
3.1 Visualization . . . 131
3.2 Model Sketching and Fitting . . . 131
3.3 Quantification and Model Prototyping . . . 133
4 Visualization and Interaction . . . 134
4.1 Visual Representations . . . 134
4.2 Convergence . . . 135
5 Case Study . . . 138
5.1 Process Data . . . 138
5.2 Temperature . . . 142
6 Discussion, Conclusions, and Future Work . . . 144
7 Acknowledgements . . . 147

Bibliography 149

Related Publications

This thesis is based on the following publications:

A. O. Daae Lampe and H. Hauser. **Interactive Visualization of Streaming Data with Kernel Density Estimation**. In *Proceedings of the IEEE Pacific Visualization Symposium 2011*, pages 171–178, Hong Kong, March 1–4, 2011.

B. O. Daae Lampe and H. Hauser. **Curve Density Estimates**. In *Proceedings of Eurographics/IEEE-VGTC Symp. on Visualization (EuroVis 2011)*, 30(3), pages 633–642, Bergen, Norway, June 1–3, 2011.

C. O. Daae Lampe, C. Correa, K. L. Ma and H. Hauser. **Curve-Centric Volume Reformation for Comparative Visualization**. In *IEEE Transactions on Visualization and Computer Graphics (IEEE Vis. 2009)*, 15(6), pages 1235–1242, 2009.

D. O. Daae Lampe, J. Kehrer, and H. Hauser. **Visual analysis of multivariate movement data using interactive difference views**. In *Proc. Vision, Modeling, and Visualization (VMV 2010)*, pages 315–322, 2010.

E. O. Daae Lampe and H. Hauser. **Interactive Model Prototyping in Visualization Space**. In submission to *SIGRAD 2011 in Stockholm, Sweden.*

The listed papers were all written during the Ph.D. project of the thesis author. The thesis author is also the main author of all the publications. Furthermore, all papers where co-authored by the supervisor of this thesis, Helwig Hauser. Hauser provided, through guidance, inspiration for most of the novel contributions presented here. Paper C where co-authored by, then post doc. at the VIDI group at UC Davis, Carlos Correa, and Kwan-Liu Ma, Professor of the VIDI group at UC Davis. This paper was written during a research stay at the VIDI group, and the ideas and contributions where formed through a tight collaboration between the authors. Carlos Correa contributed in the implementation and research phase by contributing data and with insights into volumetric deformations. In the writing phase of the paper, Correa wrote the largest part of the related work section, where the rest of the paper was written to the greater extent by the main author. Paper D was also co-authored by Johannes Kehrer who helped with several aspects of the interactive difference views, and with the write up.

Part I

Overview

Chapter 1

Introduction

The interactive visual analysis of process data, or interactive process visualization, is the combination of traditional process visualization and techniques from interactive visual analysis. This thesis describes visual representations that support both streaming process data, and visual interaction techniques. To a greater extent than ever before we gather data using sensors monitoring complex systems, ranging from run sensors that monitor heart-rate, speed and position to monitor your training progress, to sensors placed on every cab in large cities, and to traditional process sensors monitoring the flow of different materials in a large chemical production plant. With an increased connectivity and a steep drop in price, the placement of sensors is becoming more and more ubiquitous, and their data can to a greater extent be gathered and utilized. What all these sensors have in common is their ability to monitor and report a value measured from some physical process. All these sensors also have a physical location, which might change over time, and they report their values given a timestamp which combined fixes their relevance in time and space. In some instances the sensors produce spatially connected and continuous values, and in some they do not, such as across boundaries. However, for almost all sensors monitoring physical processes they produce continuously connected values over time. Addressing the increased amount of sensors, the individual sensors abilities to produce more and more data and the widespread connectivity enabling the relay of this data to central servers; we see the need for new techniques to monitor their data. Traditional process visualization often consist of dashboards made to convey the status quo of the process, predominantly also made to show the physical location of the different sensors. From the cognitive and psychological viewpoint these dashboards are made to convey information as good as possible to the operators, while carefully avoiding overloading them. The amount of information an operator can handle is tightly coupled to the *pace* this operator is working in. Figure 1.1 illustrates how pace and the amount of information an operator can consume is connected. Where an analyst can interact with the data, making an educated guess, a firefighter would need a tight selection of the most important data, e.g., temperature or target location, and then without any interaction act instinctively. Visual analytics frameworks, on the other hand, are made to fully exploit interaction and are usually made with as many options as possible to fully analyse any number of situations. Figure 1.1 illustrates visual analytics

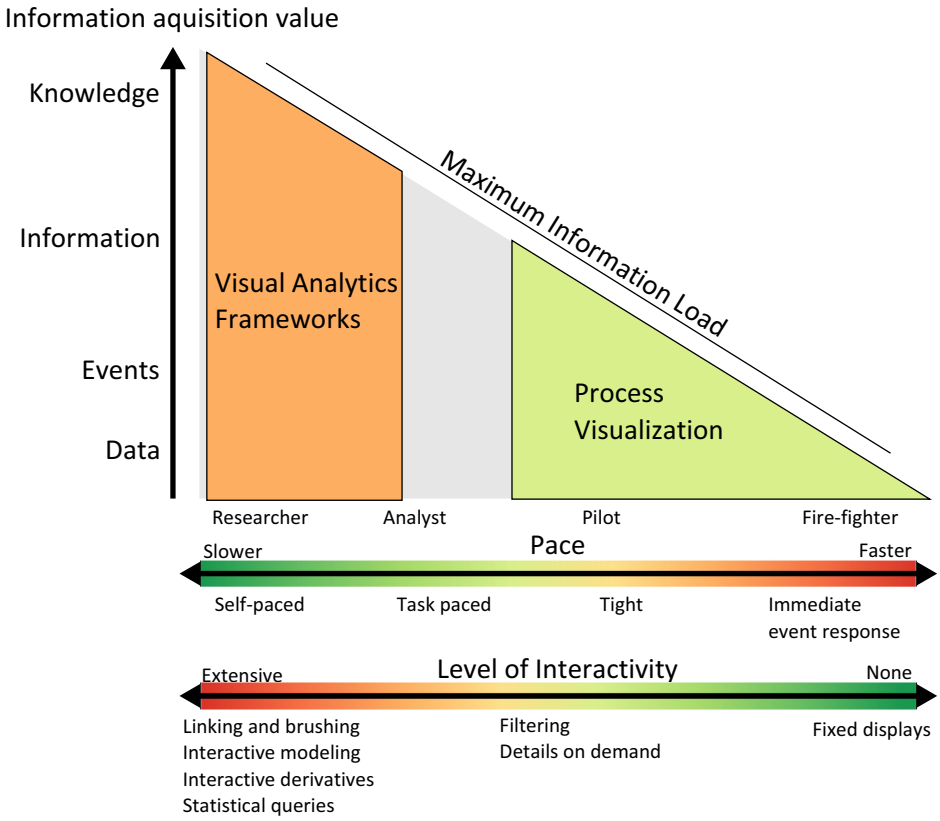

Figure 1.1: Figure showing the pace (time to solve a given task) and interactivity needed to increase information value.

frameworks in the other end of the temporal scale, indicating that they do take more focus and time to work with. With interactive process visualization, as coined from this thesis title, we intend to extend the range of process visualization towards that of visual analytics, thus, enabling the changing paces, or roles a process engineer must take.

1 Problem Statement

The motivation for this thesis was found through the eLAD project [91] in close collaboration with the involved industry partners. The eLAD project is supported by the research council of Norway and made up by Statoil and ConocoPhillips as industry partners, and Iris, IFE and CMR as research partners. The main

challenges which this thesis should address were related to the introduction of the wired pipe [16] technology into the drilling process and to facilitate integrated operations. The wired pipe technology widens the bandwidth of communication with sensors in the well (while drilling) from approximately seven bits per second, using the current technology, to several megabit/s. This increased bandwidth enables a large amount of new sensors to be developed and used, flooding the current workflow with data. The second challenge lies in supporting integrated operations, meaning, supporting the integration of the teams working offshore at the oilrig with the land based operation center. From this we extracted two major visualization challenges. First, to support the different paces for the operators (as illustrated in figure 1.1), which differ greatly with respect to the driller at the rig, and as compared to those who operate the onshore operations center. The visualization need visual representations that work in both a static and in an interactive setting. Second, accounting for either the massive increase in data from current sensors, or a large increase in different sensors, the visualization techniques should work on streaming data in an in situ setting.

2 Contributions

In this thesis, in particular in chapter 3 of part I and in the papers of part II, we describe the following novel contributions:

1. A line kernel to represent continuous data in density/frequency based visualizations.
2. An interaction technique for KDE based visualizations, connecting the bandwidth to the zoom level.
3. Differential analysis enabled by interactively defined difference views.
4. A new and easily reproducible visualization technique that can display single or multiple curves irrespective of frequency and zoom.
5. Curve-centric volume reformation, a technique to align 3D volumetric data to a 1D parametric curve, which also includes the definition of a modified Frenet frame and a realtime inside-out raycasting.
6. A technique for interactive model prototyping, which includes a novel detrending of the value space to provide feedback on the quality of fit.

3 Thesis Structure

This thesis is divided into two main parts. This first part summarizes and abstracts the findings that were worked out through this project and documented in several individual publications. In the second part these individual publications follow. These papers are provided ad verbatim, in their published form, with the

exception of their style, which is conformed to fit this thesis, and their bibliographies which were collated. Paper E will, naturally, undergo some changes, since it is still in submission.

In this part we provide a chapter covering work related to our contributions. This chapter is a supplement to the more detailed and specific related work sections found in the individual papers in part II. Following, in chapter 3, a detailed look into the novel contributions made in this thesis is described. Then, a demonstration of the applicability of the contributions, with particular focus on the related industry, is made in chapter four, before a conclusion is provided in chapter five.

Chapter 2

Related Work on Interactive Visual Analysis of Process Data

This chapter provides a brief overview of work related to interactive visual analysis of process data. To cover this combined terminology, we first cover process data, and the visualization thereof, before focussing on the streaming aspect. We then go into interactive visual analysis and look at several concepts there. Two other important aspects, covered in this thesis, are comparative visualization and kernel density estimation, which are both also covered here.

1 Process Visualization

Usually, the main purpose of process visualization is to enable an operator to monitor the status of a process, and to support decision making. A traditional operator is often given the role of a *process pilot*. Braseth et al. [18] recognized that this role changes during the course of operation. When, e.g., unexpected problems arise, their role might more closely resemble that of a fire-fighter. When the problems are resolved, the operator might even adopt a researcher role to find out what went wrong in the first place. The decisions handled by the operator are ranging from the small subconscious ones to more deliberate ones. How these decisions are made, defined by Rasmussen [96] as the SRK framework, can be separated in three levels of control, namely: skill based (S), rule based (R) or knowledge based (K).

- The skill based behavior is the low level response to observations, processed continuously from a stream of data. Examples of skill based behavior are maintaining the position of a car in a lane, or maintaining the pressure level and rotation speed in a drilling operation. This behavior is mostly subconscious, and can thus be maintained with minimal attention. We have a high parallel capacity for such tasks.
- Rule based behavior is defined as the response to "familiar" situations, where an event should be interpreted, by pattern recognition, and the normal response should be initiated. Two examples of such behavior is stopping the car at a red light, or shutting down a pump when the receiving tank is full. The parallel capacity for such behavior is moderate.

- Knowledge based behavior, or problem solving, is the complex process of gathering information, integrated from multiple sources, to formulate a plan and to execute the proper response. This response should be based on as much relevant information as possible, to get the full picture, such that the operator can relate to previous experiences or knowledge about the processes. This behavior is mentally demanding and usually requires the operator's full attention, such that parallel tasks are detrimental to the outcome.

Braseth et al. [18] recognized the temporal aspect in these different roles, and the maximum information load possible, which is showed in Fig. 1.1.

The cognitive and psycological sciences defines several different methodologies of *interfaces design*. One methodology is user-centered design, which focuses primarily on the tasks to be performed and the operator that should perform them. A differing methodology is the ecological interface design (ECI) which was introduced by Vicente and Rasmussen [127]. ECI sets focus at the work-domain and the environment. ECI builds upon two main concepts, the abstraction hierarchy [97], which defines how to model the work environment, and the SRK framework which model the operator's different roles. Since its introduction ECI has seen several implementations ranging from less complex processes, to large and complex process monitoring [134], such as nuclear plant management [17].

2 Streaming Data

With an increasing ability to collect data in almost all fields, we find ourselves with data growing faster than our ability to process and analyze it. The prevalent technique, both in statistics and visualization, of storing and then analyzing in-place, is often unusable in the context of streaming data. Szewczyk defined streaming data [114] as a continuous sequence of ordered observations of indeterminate length. This definition, while simple, carries some important implications. Since the data arrives continuously and there is no known end to it, one cannot read all of it, and then process it. Rather, one must process the current data, and then eventually, before one runs out of memory, discard old data to fit the new. If either the old data is not removed, or if the process stalls, e.g., due to a too high CPU load during analysis, newly arriving data will simply be lost.

Szewczyk recognized three factors that define streaming data, one, the continuously arriving data, two, the unknown sample size, and three the permanent loss of data if not explicitly kept and stored. Aside from the obvious limitation of storage space, the defining factor that determines if streaming data can be analyzed and visualized is the time spent to process new samples. Unless every new sample can be analyzed in real-time, i.e., faster than the samples are arriving, new samples will either have to be discarded, or left out from the analysis. Processing time, in other words, is bounded by acquisition time. Process visuali-

zation, and data from processes, thus cannot automatically qualify as streaming data, since often the size can be manageable in terms of storage. This data does, however, share the other two aspects of streaming data, namely, continuously arriving data and an unknown sample size. However, for many practical applications for real-time process visualization the limitation on storage space still applies, since process visualization is often either done, offline, on the historical stored data, where time is not an issue, or directly on the stream. Algorithms that are applicable to streaming data need to first be applicable to data with an unknown size, and second fit within the limitation on processing time. An example of an algorithm that is not applicable to streaming data is the following implementation of the sample mean, $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, which needs the full dataset $[x_i]_{i=1,\ldots,n}$. This algorithm can be rewritten as a recursive algorithm, $\bar{x}_n = \bar{x}_{n-1} + \frac{1}{n}(x_n - \bar{x}_{n-1})$, and can thus be applied to a stream, by only storing $\bar{x}_n$.

Wong et al. [139] showed how the Haar wavelets can be applied to streams, to compress large data amounts, without loosing important features. Wong et al. [139] also investigated how to perform PCA and MDS on streams, where the eigenvectors are only calculated for parts of the data, while the rest are projected using the same eigenvectors as the first part. Zhou et al. [145] presented M-Kernel Merging (MKM), to evaluate the kernel density estimation on streaming data. MKM works by applying binning to the incoming samples, and these bins are represented as a single kernel weighted by the bin count. When the total bin count exeeded a given threshold, two bins would be merged and a new bandwidth and weight would be calculated for the resulting bin. Considering that the kernel density estimation can already be defined for a fixed evaluation grid in a recursive term, and thus being applicable for streaming data, MKM enables both dynamic grids, and a kernel size/bandwidth that can change along the stream.

Applications, algorithms and visualizations that operate on streaming data can, in our opinion, be separated in two distinct forms: one which operates solely on the current window[1], and the other which operates on persistent results. Where windowed algorithms, in this sense, only contains information extracted from the current window, a persistent algorithm will contain information which evolved over the entire stream. As an example of a window based algorithm, consider a window of the ten latest samples, and their average, calculated in full whenever one sample enters (and one leaves) the current window. As an example for a persistent algorithm, we consider the recursive average, where, whenever a new sample arrives, the existing average is updated (using the recursive algorithm above), and will thus hold information about the entire stream.

Other methods to increase the persistence of the stream includes decimation, aggregation, modeling and abstraction. A decimation technique, could by uti-

[1] A windowed function, or an apodization function, is a mathematical function that remain constant in a given interval, and zero outside. In streaming data algorithms, this usually mean the recent subset of samples, that we can effectively handle.

lizing statistical sampling reduce the amount of samples that are kept, or use the Haar wavelets to compress the stream, as proposed by Wong et al. [139]. Aggregation techniques summarize several samples to reduce both the size and complexity. BinX [14] employed temporal binning at different levels of aggregation while calculating the mean and higher moments. Aggregation could also be applied to other measures than the temporal axis, e.g., count the hits to a web server per city. Modeling techniques can by understanding the underlying rules/physics that govern the process which the measured data is coming from, extract information on a higher level, e.g., discard all data from a "normal" operation, and show model parameters, instead of raw data, for the rest (as shown in paper E). Abstraction includes the creation of higher concepts, often of higher value than individual samples, from the data stream. Detecting and storing an event with extracted details, e.g., an ongoing denial of service attack, is an example of abstraction. Abstractions are often more closely related to the mental model we use, i.e., looking at millions of SYN requests (spawning half-open TCP connections), as one single DOS instead of individual requests.

Hao et al. [52] utilized a pixel based displays to show a time window into streaming data. Since streaming data can come in at high speeds, Hao et al. also investigated different layouts to minimize the overall movement of the display, when introducing new data.

3 Interactive Visual Analysis

Thomas and Cook [117] defined the field of visual analytics as following:

> Visual analytics is the science of analytical reasoning facilitated by interactive visual interfaces.

And as further defined by Keim et al. [72], visual analytics is a highly interdisciplinary field ranging from information analytics and statistical analytics to cognitive and perceptual sciences. Direct interaction with both the data and visual representations are key elements in In interactive visual analysis (IVA). Shneiderman coined the visual information seeking mantra [106] for what he recognized as a repeating pattern for the common interaction while seeking information in data. Keim et al. [71] found that when analysing large amounts of data, an overview might not be an option. Keim et al. further expanded on Shneidermans mantra, by including automated analysis, to:

> Analyze First, Show the Important, Zoom, Filter and Analyze Further, Details-on-Demand

In the following sections we will describe work related to two important aspects of IVA: interactive visual analysis and exploration, interaction techniques and visual representations.

3.1 Interactive Visual Exploration and Analysis

In 1977 Tukey [123] presented in the seminal work on *exploratory data analysis*. Up to that date, much of the statistical visualization consisted of static figures of results. Tukey suggested to connect the visualizations, using interaction, directly to the data. In visual exploration, and particularly when dealing with large multidimensional data, one are not looking for the answer to one particular question, but rather picking up evidence during the interactive exploration. This evidence, important pieces of information or findings, was termed by Yang et al. [142] as *nuggets* of information. Liu and Stasko [80] investigated how the internal mental model, or nuggets, relate to external visualizations. Yang et al. [142] showed that providing techniques to externalize these nuggets back into the visualization enables both reasoning and collaboration. This externalization was provided through their *nugget management system*, where the nuggets could be added as evidence for, or against, a larger proof or hypothesis. Shrinivasan and van Wijk presented the *Knowledge View* [107] that enabled externalization through a mind map, where the nodes where linked to the interaction steps needed to reproduce the externalized findings.

3.2 Interaction and Multiple Coordinated Views

Interaction is one of the primary distinctions between regular analysis and IVA. The role of interaction, as a mean of connecting the user to the data, was classified by Yi et al. [143] based on the users *intent*. They identified the following seven intentions that the user had with respect to the data and its visual representations:

- *select*, to mark something as important,
- *explore*, show me something different,
- *reconfigure*, show me a different arrangement,
- *encode*, show me a different representation,
- *abstract/elaborate*, show me more or less detail,
- *filter*, show me something conditionally,
- *connect*, show me related items

A central method for the interactive exploration/analysis of multivariate data, is that of linking & brushing, compare to Eick and Wills [33] or Tweedie et al. [125]. Brushing is the defintion of a selcection, or marking a subset of data elements as interesting. Linking is the highlighting of the corresponding subset in other views, of the same data, to highlight their relations. This highlighting is often performed utilizing techniques described as *focus+context*, detailed by Hauser [55], which presents how to emphasize a subset, the *focus*, while maintaining the overview, the *context*.

Coordinated multiple views is a methodology that use multiple views on the same data, and has these views connected. This connection is often realized using linking and brushing, or other focus+context methods. There are several applications that support this visual querying, e.g., the XmdvTool [132], the SimVis framework [31], Spotfire® [1], Tableau® [115] or ComVis [83].

4 Comparative Visualization

Pagendarm and Post [92] separated comparative visualization where images are placed side by side, leaving the interpretation of differences to our cognitive system, from that of direct comparative visualization. In direct comparative visualization the resulting image shows the evaluated difference in one target image. Pagendarm and Post [92] furthermore identified two primary approaches of performing direct comparative visualization, the first as image level comparison, and the second as data level comparison. In image level comparison two images, resulting from their own visualization pipelines, are overlaid using either a blending or difference operator. In data level comparison, data from two different sources are converted to a common representation and then both are fed into the same visualization pipeline. Later Verma and Pang [126] added feature level comparsion to this list of primary approaches. In feature level comparison, different methods of extracting similar features from the same dataset are overlaid into the same image, highlighting the differences in a direct manner.

In paper C in part II of this thesis, different data-sources are visualized using different visualization pipelines, and as such cannot be called a direct comparative visualization. However, in combination with a proposed helper line and a shared parallel axis, a direct comparison can be made in the one shared dimension. In paper D a direct comparative visualization is used to show the differences between two density fields representing two different categories. This difference operator is implemented on the evaluated kernel density estimation, but before the visual representation, and is thus partly between the data level and the image level comparative visualization approaches. To visualize the result from a difference operator, which also can yield negative values, a diverging colormap, compare to [19], is applied.

5 Kernel Density Estimates in Visualization

Kernel density estimation, KDE, has a long history dating back to its introduction first by Rosenblatt in 1956 [101] and later, in 1962, independently also by Parzen [93], after which it also recieved its name as the Parzen window, or also as the Parzen-Rosenblatt window.

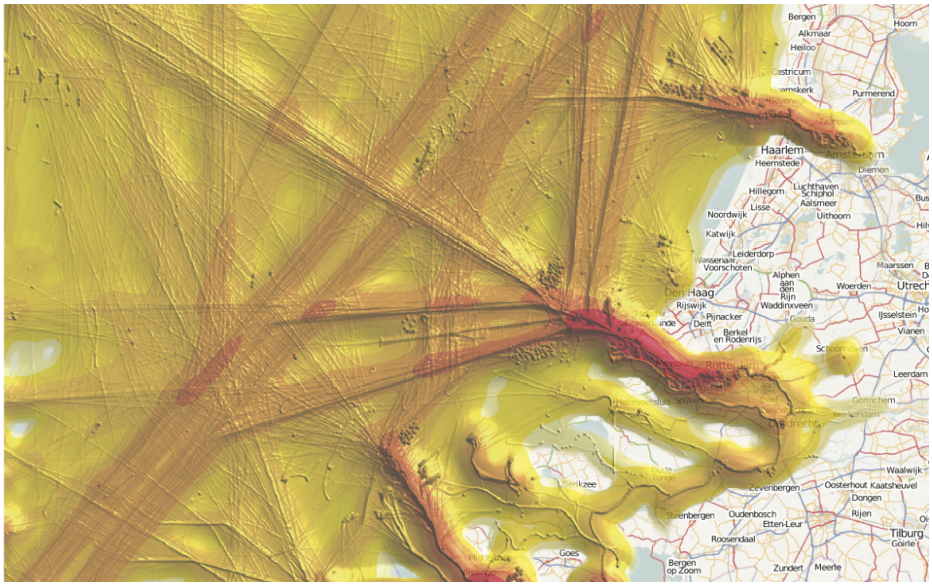

Figure 2.1: Visualization of vessel movements, courtecy of Willems et al., showing how to combine two different bandwidths (kernel sizes), to show overall density in addition a more detailed view.

There is a tight connection between density estimation and visualization, since often the density estimation is created to inspect a distribution visually. Scott wrote a good overview of multivariate density estimation in visualization [104]. Different density maps, similar to those of KDE, have been used in cartography for several applications. Fisher introduced a technique called hotmap [39] to visualize the frequency of views of different map tiles. In a similar application, Whittaker and Scott presented the use of the average shifted histogram [135], as an effective approximation for KDE. Also for a geographic use, Willems et al. introduced KDE to visualize vessel movements [136]. In this work they combined the usage of two different kernel sizes, to first show the overal density of vessel movements, and secondly to show detailed movements, as also shown in figure 2.1. Later, Scheepens et al. [103] refined this work to include a GPU pipeline for rendering, and tools to enable multivariate analaysis.

Minnotte and Scott presented the mode tree [86] which is a visualization of the continiously changing modes when the bandwidth is increased. The mode tree was further refined by Minnotte et al. [85] to move in the direction of continous representation of modes, constructed from multple distributions. Florek and Hauser [42] presented an improved technique on how to visually analyze the

bivariate mode tree. Florek and Hauser [41] also investigated how to improve the quantitative visual properties of bivariate KDE through indicators and iso-lines.

As very similar to KDE, we can also consider radial basis functions (RBF). Jang et al. investigated the representation of volumetric datasets by weighted radial basis functions (RBF) [61, 21], and introduced an effective algorithm on how to render these. Crawfis and Max [26] used KDE to reconstruct uniformly sampled 3D volumes.

Chapter 3

Interactive Visual Analysis of Process Data

With the aim to support process engineers to switch from the role of process pilots to the role of an analyst, or even a researcher, we have defined several methods that support these tasks. In the remainder of this thesis, these methods are presented as the following novel contributions:

1. A line kernel to represent continuous data in density/frequency based visualizations (related publication: Paper A in Part II).
2. An interaction technique for KDE based visualizations, connecting the bandwidth to the zoom level (Paper A).
3. Differential analysis, enabled by interactively defined difference views (Paper D).
4. A new and easily reproducible visualization technique that can display single or multiple curves irrespective of frequency and zoom (Paper B).
5. Curve-centric volume reformation, a technique to align 3D volumetric data to a 1D parametric curve, which also includes the definition of a modified Frenet frame and a realtime inside-out raycasting (Paper C).
6. A technique on interactive model prototyping, which includes a novel detrending of the value space to provide feedback on quality of fit (Paper E).

The following sections in this chapter are structured as following: first we cover the progress we have made in improving kernel density estimates (KDE), and then we cover the novel contributions on interactive techniques in KDE, before going into the concept of curve-centric volume reformation.

1 Kernel Density Estimates for Continuous Data

In the papers D, B, A and E (in part II of this thesis), a common theme has been the usage of kernel density estimation (KDE) to investigate temporal data. A big difference between the existing KDE and our focus towards temporal data, is the connectivity between the samples. E.g., samples from a satisfaction survey yield discrete samples (individual customer opinions), but a sampled time-series of a measurement from a physical process should be interpreted as snapshots of a continuous change. In the following we first provide a brief and general

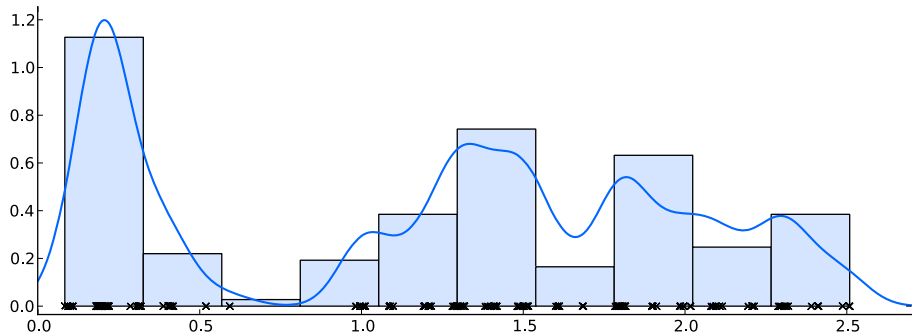

Figure 3.1: Data samples from the Fisher Iris dataset [40], shown as a rug plot (with a small random displacement to also show overlapping values), as a normalized histogram and as a one-dimensional kernel density estimate (the blue line).

introduction to visualizations based KDE, followed by a description of our novel contributions.

1.1 Kernel Density Estimation

A short description of a kernel density estimate (KDE) is to consider it as a continuous histogram. Instead of defining a discrete number of bins and counting how many samples that belong to each of them, a KDE is defined as the sum of a series of kernel instances. Each of these represents one of the samples in the original data set. Fig. 3.1 shows an example KDE, compared to the corresponding histogram. While these two visualizations correspond well here, we show how histograms can suffer from aliasing effects in paper A.

KDE was introduced by Rosenblatt and Parzen over 50 years ago [101, 93]. Given a set of n (1D) data samples x_i the kernel density estimator $\widehat{f}_h(x)$ is computed as

$$\widehat{f}_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i), \tag{3.1}$$

based on a kernel function K and a bandwidth parameter h. While KDE is defined for an arbitrary kernel function, K, we have primarily focused on the normal distribution. From the KDE in its one-dimensional form (Eq. 3.1), we extend to the N-dimensional form (as shown by Scott [104]) by

$$\widehat{f}_{\mathbf{H}}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_i) \tag{3.2}$$

with $\mathbf{H}$ being a symmetric and positive definite bandwidth matrix and $K_{\mathbf{H}}$ being defined as

$$K_{\mathbf{H}}(\mathbf{x}) = |\mathbf{H}|^{-\frac{1}{2}} \mathcal{K}(\mathbf{H}^{-\frac{1}{2}}\mathbf{x}).$$

$\mathcal{K}$ is a multi-variate kernel function that integrates to 1. This equation will for a set of samples, $\mathbf{x}_0, ..., \mathbf{x}_n$, create a continuous estimate of density, i.e., a probability density estimate with the unbounded integral of one.

Since the individual kernels integrate up to one, the normalization term of Eq. 3.2 is $1/n$. If we remove this term, the unbounded integral of Eq. 3.2 is n. In paper A we show different cases when this is desirable. E.g., if the individual samples represent a dollar spent at a given location, then the resulting KDE will show the distribution of dollars spent over the total area. In the same example, a bounded integral will result in the given amount spent within that bounded area[1]. To facilitate these desirable properties of the KDE we iterated on equation 3.2 (in paper A). We removed the normalization, and introducing a scaling factor per kernel, c_i. To revisit our previous example, the scaling factor allows for a single sample/kernel to represent multiple dollars, instead of a single dollar. The iterated equation for this 2D KDE is

$$\widehat{g}_{\mathbf{H}}(\mathbf{x}) = \sum_{i=1}^{n} c_i K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_i). \tag{3.3}$$

This new definition enables the aggregation of other attributes than just sample density, e.g., time in seconds, as shown in later examples, or dollar per square mile as illustrated in Fig. 3.2. This figure, 3.2, illustrates individual contributions as samples, each scaled by the contributed amount, c_i. As opposed to histograms and normalized density estimations the scale can also be negative. In the case of Figure 3.2 , much to our surprise this contribution dataset also contained negative amounts. These negative amounts represented the contributions that were rejected and sent back, and there are, in several areas, more geospatially located rejections than acceptions (shown as blue).

1.2 The Line Kernel

For most sensors utilized to measure physical processes, a value is read out at discrete intervals and streamed to its host. Most physical processes, however, are continuous in nature, which means, e.g., if a temperature is measured at five degrees one second and at ten degrees the next second, it is more correct to represent this change with a smooth continuous change from five to ten, than a discrete jump. To represent the density of temporally connected samples given

[1] In actuality the bounded integral, will not yield exactly the same result as the sum of the samples. KDE distributes each sample over a small area, and thus, samples outside the bounded area can contribute in, and vice versa.

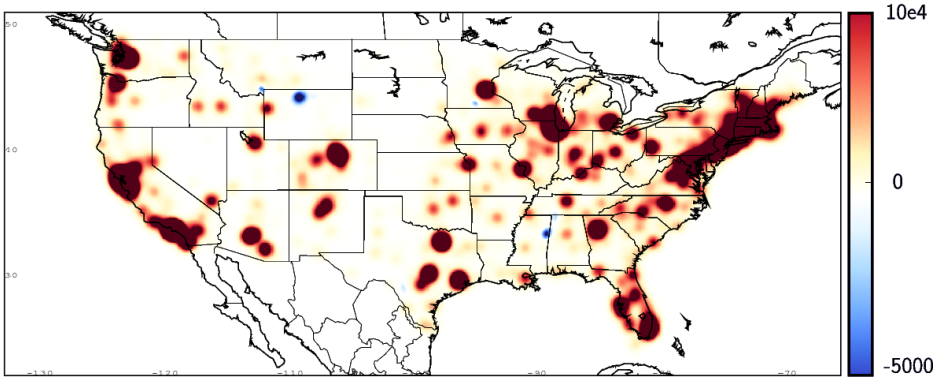

Figure 3.2: Visualizing over 165 000 monetary contributions to the Obama 2008 campaign. Interesting areas with negative aggregates, i.e., locations where the amount of contributions are less than zero dollars per square km (shown as blue).

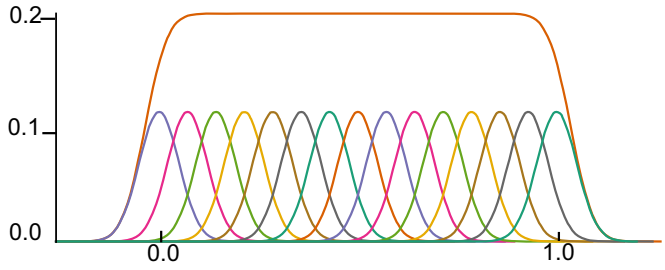

Figure 3.3: The line kernel in 1D, shown as upper orange curve, is given as the integral of individual kernels along the line between its start (here 0) and its end (here 1), is defind by Eq. 3.4.

over a specific domain one can either resort to linear interpolation and subsampling or, as we argue for in this section, utilize a specialized line kernel. In papers A and B of part II, we show several cases of the beneficial properties of this line kernel.

The line kernel accomplishes two main advantages over drawing a regular line. The first advantage is its normalization; where a regular line will draw more pixels when the line gets longer, and thus add more to an integral or sum, the line kernel is always normalized. The second advantage is the singular degeneration a line will have if the two points fall on the same pixel. A regular line drawing will draw zero pixels in this case. The line kernel, however, converges towards a regular 2D normal distribution when the two samples converge.

In paper A, we describe this as a line kernel L_k defined by two consecutive data samples, and their positions $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$:

$$L_k(\mathbf{x}) = \int_0^1 c_i K_{\mathbf{H}}\big(\mathbf{x} - ((1-\phi)\mathbf{p}_i + \phi\mathbf{p}_{i+1})\big)\,\mathrm{d}\phi, \tag{3.4}$$

with $K_{\mathbf{H}}$ being the 2D normal distribution kernel (or any other proper 2D kernel). This kernel, L_k, as shown as a 1D function in Figure 3.3, is the integral of a series of small kernels moving from $\mathbf{p}_i$ to $\mathbf{p}_{i+1}$. To enable a proper reconstruction from uneven sampling in time, we insert the elapsed time between the two samples in the scaling factor c_i. The realization of this integral, in paper A, was achieved by using preintegrated look-up tables for rendering, but in paper B we defined a direct and analytical definition. To find this analytical definition, we first define a 1D version of equation 3.4. This 1D version is defined by reducing the dimensionality of the kernel to 1D, also shown in figure 3.3. The analytical definition is then found by considering a single point $x < p_0$ with $p_1 \approx \infty$ of this 1D version of Equation 3.4. The evaluated value for this x is the integral of an infinite series of normal distributions with their mean increasing from p_0. By turning this definition around, we observe that this is equal to the bounded integral of a single normal distribution with mean p_0 from $-\infty$ to x. The integral of the normal distribution is a cumulative distribution function (cdf). This distribution function is defined by

$$\mathrm{cdf}(\mathrm{x}, \mu, \sigma) = \frac{1}{2}\left(1 + \mathrm{erf}\big(\frac{x-\mu}{\sqrt{2\sigma^2}}\big)\right), \tag{3.5}$$

with erf the error function, found when integrating the normal distribution,

$$\mathrm{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2}.$$

When considering $p_1 < \infty$ we have to remove the contributions of all kernels larger than p_1, which leaves:

$$L_{k1D}(x) = \frac{1}{|p_1 - p_0|}\big(\mathrm{cdf}(\mathrm{x}, \mathrm{p}_0, \sigma) - \mathrm{cdf}(\mathrm{x}, \mathrm{p}_1, \sigma)\big). \tag{3.6}$$

This 1D line kernel is then expanded back to 2D by first creating a new parameterization with u extending along the line segment and v orthogonal to it[2], and then by applying the following product with the normal distribution kernel:

$$L_k(\mathbf{x}) = c_i L_{k1D}(u) \cdot N(v) \tag{3.7}$$

Figure 3.4 shows a single line kernel of the line segment between points $[0, 0]$ and $[1, 1]$, and the parameterization directions for u and v as utilized in Eq. 3.7.

[2] more details on this parameterization in paper A in part II

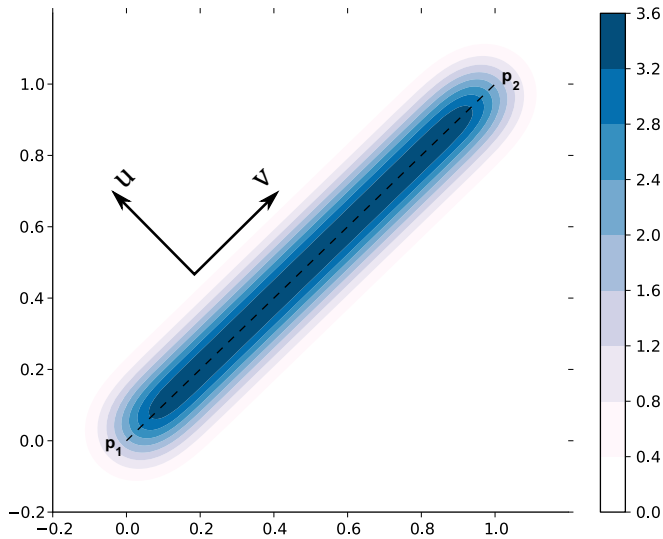

Figure 3.4: A 2D line kernel, as defined in Eq. 3.7 with $\mathbf{p}_0 = [0, 0]$, $\mathbf{p}_1 = [1, 1]$ and $c_i = 1$. The integral of this line is equal to c_i, i.e., 1.

1.3 Curve Density Estimates

If we graph a sine curve from zero to, e.g., a thousand π, depending on our screen resolution, the result will resemble an opaque square spanning the full range, as shown on the top of figure 3.5. A common solution for coping with such situations of massive overdraw is to apply transparency. The second graph in figure 3.5b shows the same graph where semi-transparency is applied. In this graph, figure 3.5b, it seems like there is a higher density at $y = 0$. If we, however, take regular samples and plot them using a scatterplot, the result looks quite different, as shown in the third figure from the top (c). In fact, if we take regular samples at x from this sine curve and insert their y value in a histogram, we see the direct resemblance towards this scatterplot, as illustrated in Fig. 3.6. To reintroduce connectivity, but keep this visualization of density we introduce curve density estimates.

The curve density estimate (CDE) is the continuous probability density estimate of sampled points along a curve. As shown in figure 3.5e the CDE displays the continuous density over y while also representing a continous curve, as opposed to figure 3.5c. On low frequency curves, or if the samples are positioned far apart (along x), the CDE is similar to an antialiased line. When the frequency of the curve increases, or when the curve is sampled multiple times per horizontal pixel, the CDE will resemble a 2D kernel density estimate providing the density of the samples' y position given a x or *time* position. Figure 3.7 shows this

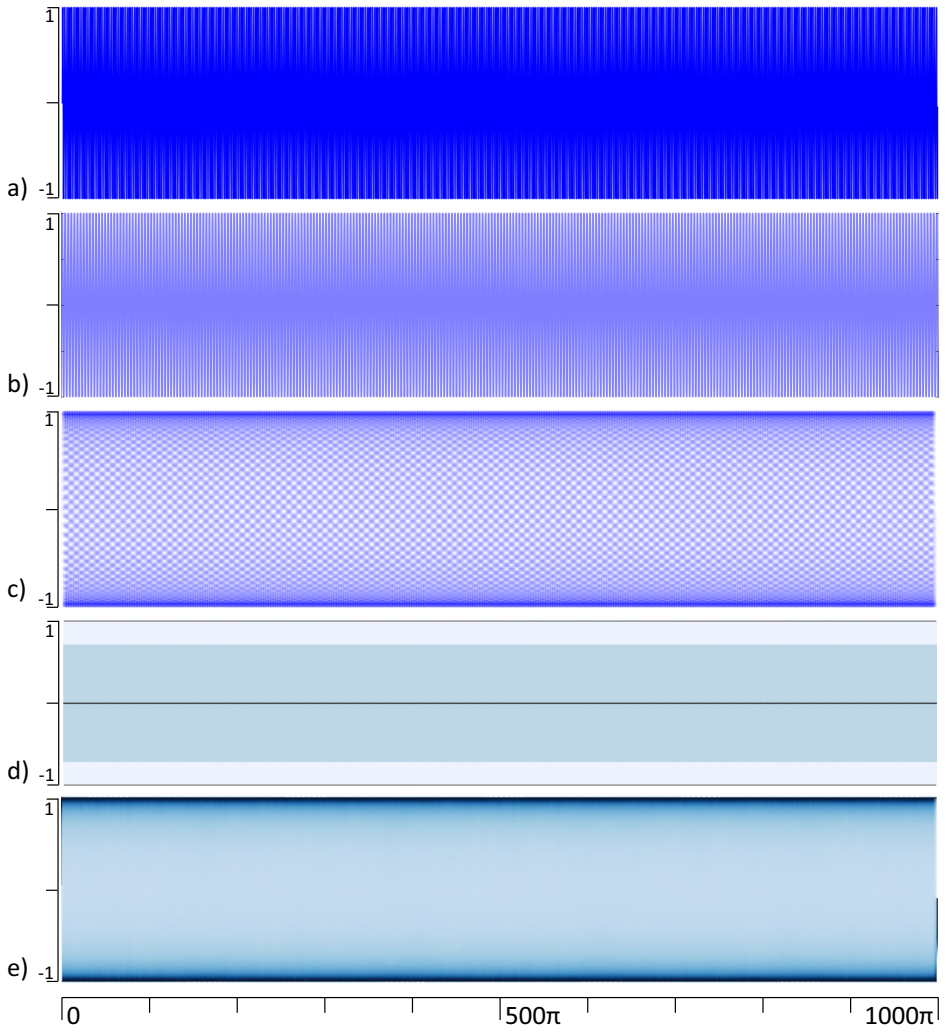

Figure 3.5: Five visualizations displaying the sine curve from zero to 1000π. In the top figure, a, an opaque line is used, and, because of overdraw, only the extent of the function is visible. In the second figure, b, a semi-transparent line is used. The third figure, c, is a scatter-plot of the samples drawn semi-transparently, and it shows the same distribution as the histogram. The fourth figure, d, is aggregated using a moving mean, a moving standard deviation and a moving extent. In the bottom figure, our technique, the Curve Density Estimate, is applied, and the distribution corresponds with that found in the histogram in Figure 3.6.

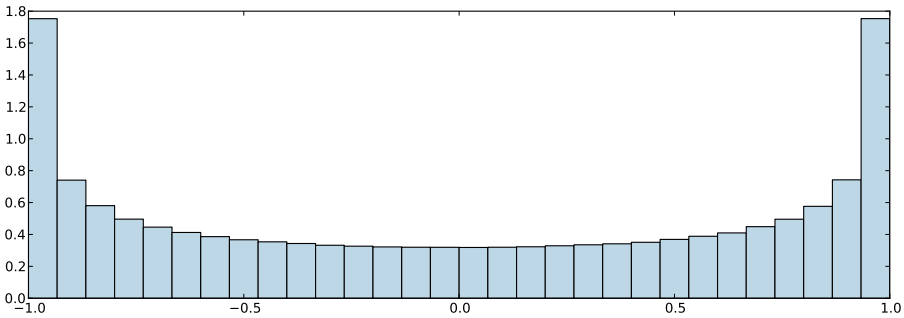

Figure 3.6: 30 bins histogram of $y = \sin(x)$ for regularly sampled values of x.

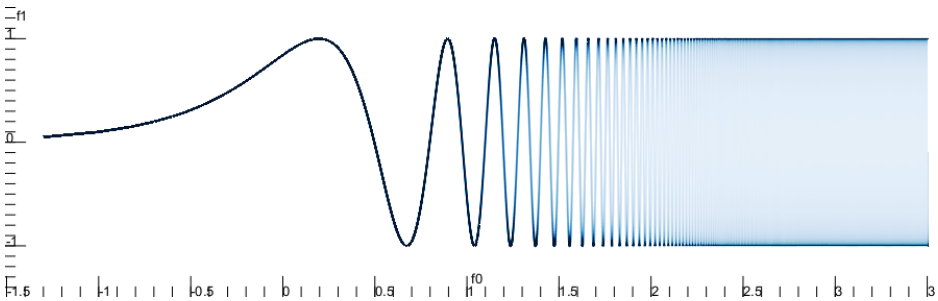

Figure 3.7: A curve density estimate of the sine curve. The x axis uses a logarithmic scale, which shows the smooth transition between a low frequency curve to a high frequency curve. The logarithmic exponent is given on the x axis.

smooth transition between a low frequency curve to a high frequency curve by drawing the sine curve on a logarithmic x axis.

2 Interaction with Kernel Density Estimates

This section describes three novel interaction techniques for the interactive visual analysis of process data. The first technique, proposed in paper A, entails an automaticaly updated bandwidth tied to the screen size, instead of, as usual, in terms of the data space. This automatic bandwidth allows the user to zoom in and out while the KDE automatically updates its level of aggregation, resulting in meaningful densities regardless of zoom-level. The second technique, as proposed in paper D, enables a differential analysis by interactively defining difference views. The third, as proposed in paper E, describes a technique on how to interactivly build statistical models that describe parts of the data.

2.1 Screen-Space Bandwidth Automation

Silverman [108] describes kernel density estimates according to equation 3.1 where h is denoted the bandwidth. This bandwidth, also called kernel size, smoothing parameter or window width, has seen several research works suggesting different techniques to identify the optimal bandwidth. Silverman introduced the normal scale rule [108] as a global bandwidth estimator for distributions that are close to a normal distribution. The normal scale rule leads to over-smoothing, however, if the data does not adhere to a normal distribution [130]. In paper A we recognized that the bandwidth can be expressed in screen space, rather than in data space. The reasoning behind this change is to enable interactivity, e.g., zooming and panning. A fixed bandwidth will, depending on zoom level, either be smaller than a pixel, or larger than the entire viewport. The visual effect of a bandwidth smaller than a pixel is that, either nothing is shown, or that a strongly aliased kernel is shown. The visual effect of a bandwidth larger than the display however will result in a near constant value all over the screen.

Figure 3.8 shows an example, where first the entire coast of Norway is shown, and then, by zooming, the city of Stavanger is brought into the viewport. Our automatic bandwidth adjustment sets the bandwidth to approx. 50 km in the first image, ≈ 10 km in the next, and ≈ 1 km in the last. An interesting aspect with this interaction is the seamless and smooth aggregation of all traffic when zooming out again. In relation to the image showing the largest area (the leftmost in figure 3.8), the final zoom (shown on the very right) makes out approximately 20 by 30 pixels, yet all the traffic from the final zoom is aggregated, and contribute to the larger one.

In a right hand system, a viewport is defined by the two points, in data-space, the lower-left $\mathbf{p}_1$, and the upper-right $\mathbf{p}_2$. Furthermore the viewport is defined by the screen-size in pixels $\mathbf{s}$. We define the size of a pixel in data-space as $\mathbf{q} = \frac{(\mathbf{p}_2 - \mathbf{p}_1)}{\mathbf{s}}$. From the previous deduction we showed that the bandwidth $\mathbf{H}$ must be significantly smaller than the viewport, but larger than the size of a pixel, i.e., $k \cdot \mathbf{r} > \mathbf{H} > \mathbf{q}$, for a constant k. The size of this constant k, and its influence on the visualizations, is explored in paper A, but for simplicity, most of the visualizations shown here use a $2 \leq k \leq 20$.

This technique does not make any assumptions about the data distribution. To utilize it to create an overview, the viewport should be set up first so that it covers all of the data and the bandwidth can be set to around five pixels (further details in paper A). When the user, by interaction, either increases or decreases this bandwidth (to either create a crisper or a smoother/aggregated image) it is the ratio towards the pixel-size which is changed, and not in terms of the data space, since this retains the zoom independence.

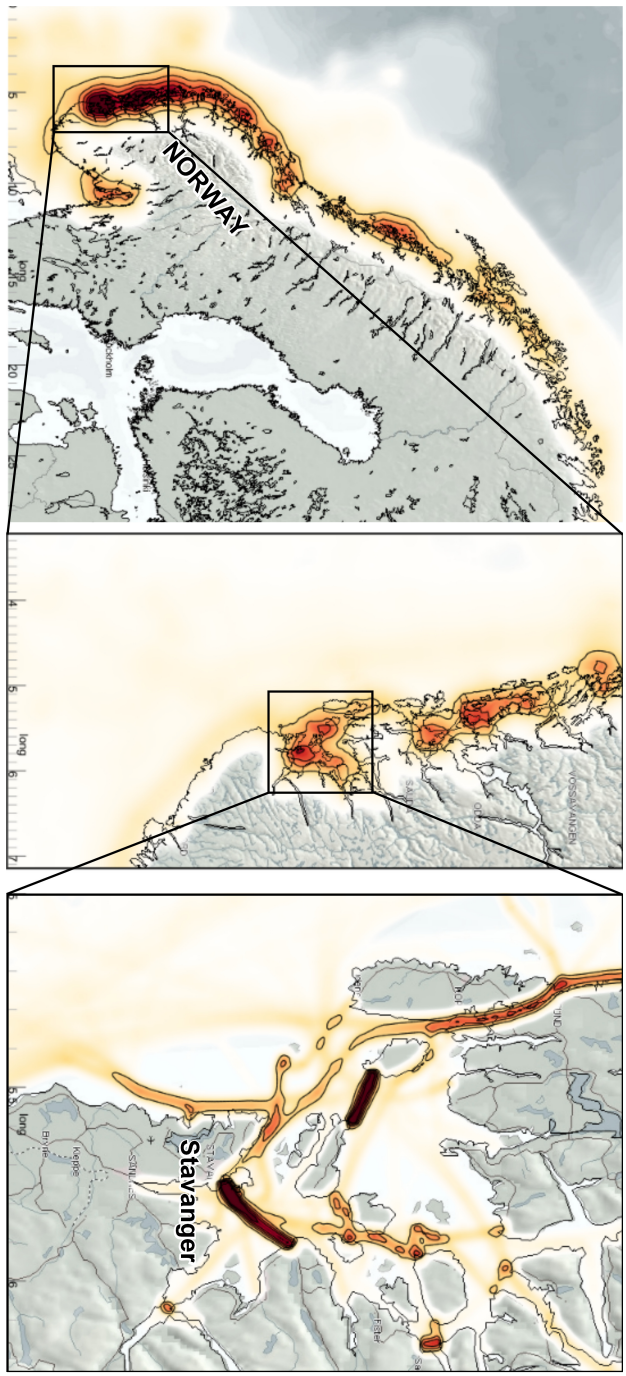

Figure 3.8: A density estimate of non-stationary vessel traffic along the coast of Norway. The bandwidth is automatically updated when zooming in (left to right) towards the city of Stavanger.

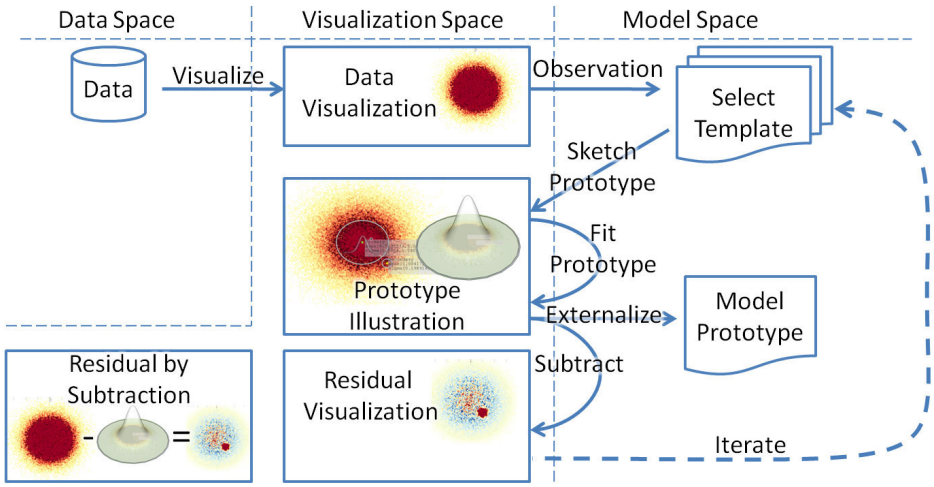

Figure 3.9: Our proposed workflow: visualize and observe, sketch and fit, externalize and subtract, then iterate.

2.2 Model Sketching

In most fields dealing with natural phenomena, e.g., physics or statistics, modeling is an important part of making sense out of data and understanding the underlying structure. Modeling serves multiple purposes, e.g., to develop a basic understanding of how a phenomenon works, prediction, or simplification (parameterization). Modeling is often performed as trial and error, based on both observation and intuition. Modeling is also often considered a global property, such as tests for normality, or calculations of skew. When multiple processes are interacting, it is important to be able to both specify data locally (select a subset) and perform the modeling on this selection. We propose a workflow where the user is enabled to quickly sketch models locally, aided by automated fitting and feedback algorithms. The workflow is detailed in figure 3.9 as *visualize and observe, sketch and fit, externalize and subtract, then iterate.*

As an example for this workflow, consider a simple dataset, as shown in figure 3.10, consisting of two features, one large 2D normal distribution with parameters $\mu = [0, 0]$ and $\sigma = [1, 1]$ and one small artefact represented by a normal distribution scaled by 0.05 with parameters $\mu = [1, 1]$ and $\sigma = [0.2, 0.2]$. The first step of the workflow is to visualize the data and, by observation, decide upon a suitable model. In the middle of figure 3.10 the normal distribution seems like a good candidate model.

The second step of the workflow, *sketch and fit*, starts with interactive sketching. The interface to sketch the normal distribution is to simply click near the observed mean of the data and modify the variance by rolling the mouse wheel. When satisfied with the sketching, the user lets go of the mouse button, and a

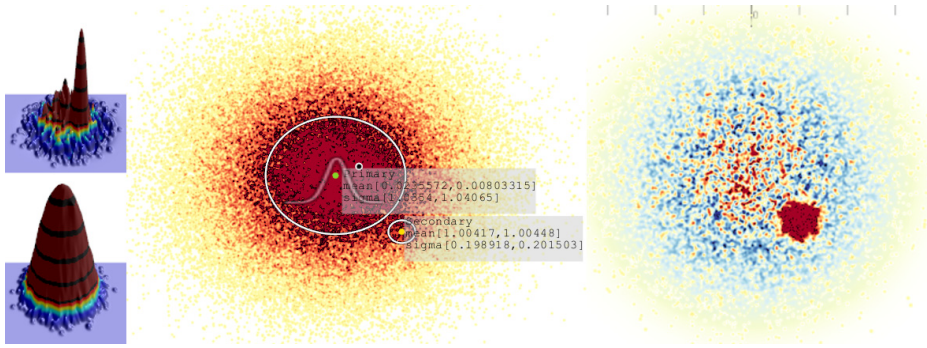

Figure 3.10: Several KDE plots of a normal distribution with a small artifact. The lower left and the middle figures show the (logarithmic) KDE of the data. The images on the right, and the upper left, shows the KDE after abstracting the primary feature and thus clearly revealing the secondary feature (the artifact).

slowly iterated optimization starts automatically. This iteration is intentionally slow to enable the user to stop it if it would diverge from the users intended local optimum. However, if the iteration converges, and remains un-interrupted, iterations of Newton's method is applied until convergence.

The third step of the workflow, *externalize and subtract*, starts when the user accepts the output from the automatic fitting algorithm. The result from the second step and the example, in figure 3.10, is the parameters for the selected normal distribution. In this case the output parameters are: a) the selected model, a normal distribution, b) its mean, $\mu = [.02, .01]$, c), variance, $\sigma = [1.035, 1.04]$, and d) the error measurement sum of squared differences. Together these parameters represent a model instance. This model instance either represent an externalization, where the result is understood, and stored, or represent a "simplification", where the data can be replaced by the models parameters. An important contribution of this workflow is the residual visualization which is made by subtracting the fitted model from the original KDE. This residual is shown to the right of figure 3.10, and since the large feature is removed from view, the smaller feature is clearly visible.

The fourth step of the workflow, *iterate*, starts the procedure over again. The smaller feature which is now visible can be modeled in the same way as the first, and then extracted/externalized, yielding $\mu = [1.004, 1.004]$ and $\sigma = [0.1989, 0.2015]$ vs. the reference $[1, 1]$ and $[0.2, 0.2]$ in this example.

2.3 Differential Analysis

Analyzing differences, or comparative visualization, between two results can either be performed by placing them side by side, or by animation from one to the other [126]. These two techniques provide the images for comparison (either

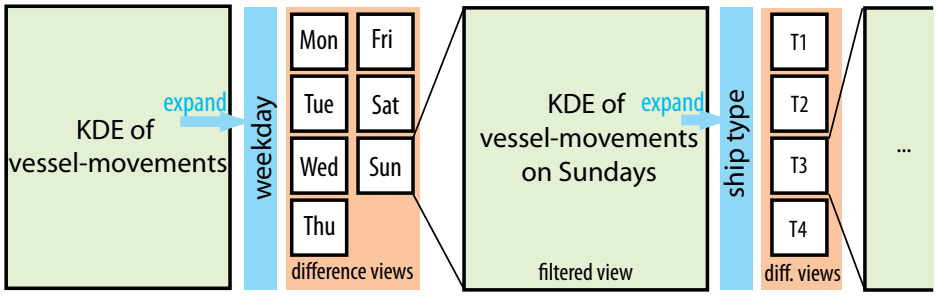

Figure 3.11: Iterative data exploration via difference views.

spatially or temporally), and the extraction of the exact differences is left to the user. When the results are provided as a scalar image/field, however, the differences can be presented utilizing a direct difference operator as one single image called direct comparative visualization by Pagendarm and Post [92]. This image provides exact difference measures and a quantitative result. This exact nature of the difference views are a strength that should be utilized instead of side by side views when possible and desirable. One major problem with the facilitation of difference views into an analysis framework is how to define *what* to take the difference between.

In paper D we propose a workflow on how to specify and analyze difference views. Our basis for a difference view is the KDE plot, as shown in section 1.1, which supports the difference operator. The workflow is described around a concept we call *expand over*.

1. Initialize by creating a KDE plot, which entails selecting two attributes/ dimensions and optionally a third attribute as the weight (compare to section 1.1), and view parameters.
2. Select a parameter to *expand over*.
3. If the expand parameter is categorical, the number of categories defines the variable N.
4. If the expand parameter is continuous, an arbitrary[3] number of bins is chosen, e.g., four, six or nine, and this number defines the variable N.
5. Create N views each containing only the samples filtered by either the corresponding category or bin, and subtract this density field from the average[4].

[3]The number of bins is arbitrary in principle. However, often an application-dependent categorization exist, e.g., the Beaufort scale for wind speed, that would lend itself as the basis for the binning. This number should not exceed the practical limitations wrt. screen space and optimally be a $n \cdot m$ multiple.

[4]The average field is defined by all the samples, unfiltered, but normalized

6. The N views show diverging[5] values compared to the average. By inspection the user selects one view.
7. Remove all the introduced views, and replace them with one view, similar to the original one, but with the filter from the selected view applied.
8. Repeat from step 2.

Figure 3.11 details this proposed workflow (paper D) on how to facilitate difference views in a particular visual analysis context.

Paper E also describes differential analysis, but in a different context. When sketching models, both the automatic fitting internally, and the visual feedback on the quality of the fit is provided by differential analysis. In the automatic fitting, the difference between the model, and the data (for the L1 norm), or the squared difference (for the L2 norm) is used internally for the optimization. As a visual feedback, both for determining the quality of the fit, and for residual analysis, the difference is shown where the models are subtracted from the view of the data.

3 Curve-centric Volume Reformation

Up to this point the visualization and interaction techniques described apply first and foremost to the 2D domain. In this section we investigate how we can combine and compare traditional 2D parameterized visualizations in an exact and quantitative way to 3D volumetric data. This comparison is achieved by extracting the volumetric neighborhood of the curve. In this way measurements along this curve are put into a direct context with the volumetric data, in which there might be a correlation. The proposed technique is designed to exploit a logical 1D parameterization within the 3D volume, e.g., a sensor moving in a volumetric space. The sensor collects data and its movement will define a curve in space that has a 1D parameterization either in time or in terms of arc-length.

On the left of figure 3.12 a synthetic volume with a curve within it is shown. We first consider this volume to be geology, and the curve as a well. The reformed volume, to the right of figure 3.12, contains only the neighborhood of the well. This extracted neighborhood can influence either the measurements or drill procedure and is therefore often an important context to provide.

The general idea is to reform (or deform[6]) the volumetric data in such a way that the 1D parameter-space from the curve within the volume is aligned and parallel with one of the two dimensions on the 2D screen. This alignment is then used to compare across multiple modalities of data. Figure 3.13 shows such a

[5] A color scheme that emphasizes the extremes on both sides of a middle range, or here positive and negative values, compare to Brewer [54]

[6] Definition by Merriam-Webster: Reform – to put or change into an improved form or condition. Deform – to spoil the form of or to alter the shape of by stress

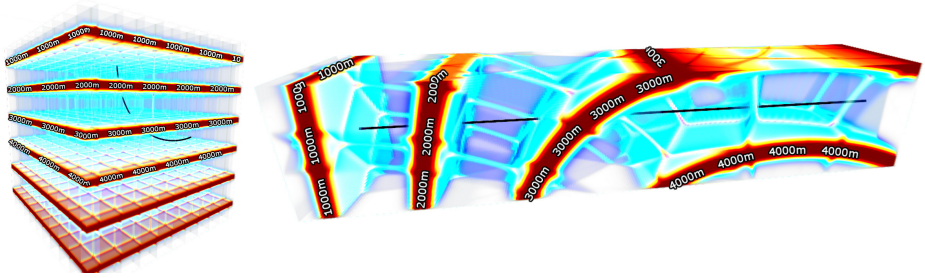

Figure 3.12: Left, a test volume with a curve in it, and right the result of Curve-Centric reformation. The curve is, after the reformation, the straight line shown in the middle of the volume to the right.

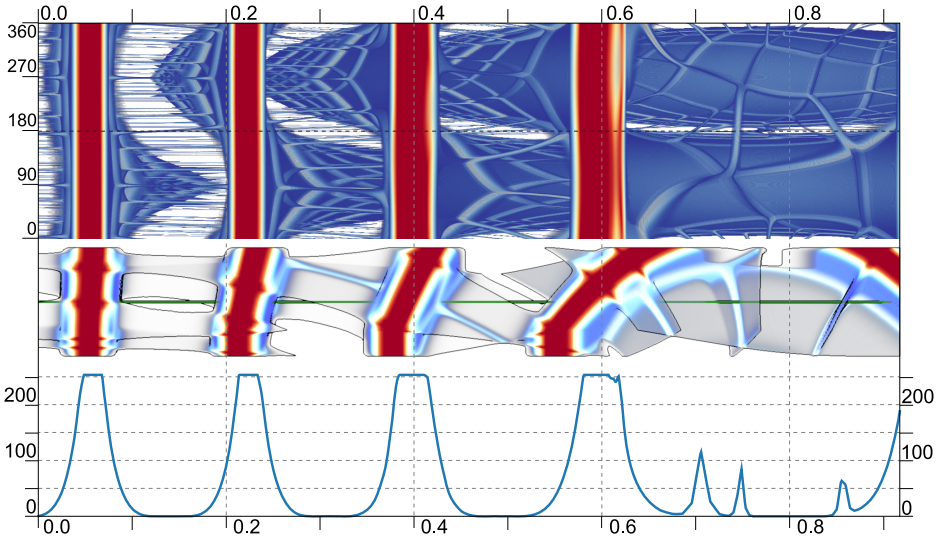

Figure 3.13: Three different techniques, applied to the same data: on top a reformed volume, in the middle radial raycasting, and below, sampled values from the volume as a function graph. This comparative visualization allows the accurate comparison of intensity values to their spatial neighborhood.

comparison where three different views on the same data highlight the vertical alignment.

In order to perform a curve centric volume reformation we first define a moving coordinate frame. The moving frame is a local coordinate system, or a tensor containing orthonormal vectors for every point on a curve $\mathbf{r}(t)$. In our work we provide a moving coordinate frame that is both smooth and that adheres to a

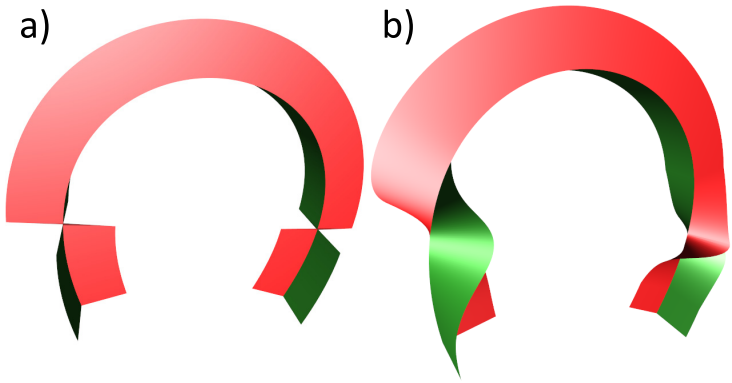

Figure 3.14: Surfaces following the normal $\mathbf{N}(t)$ as red, and the binormal $\mathbf{B}(t)$ as green. Figure a) shows the modified Frenet frame, experiencing a sign change that our smoothed version, b), handles gracefully.

user specified up direction. This moving frame is similar to a Frenet frame [43] but it does not require the second derivative, and we apply smoothing using quaternions to the tensor. This smoothing accomplishes two important features. The first feature is the solution to the problem where the Frenet frame collapses and potentially switches sign at the point of inflection. The second feature occurs when dealing with either noisy or high frequency movements, which without smoothing will lead to an overly distorted neighborhood. As an example for the first solution, where smoothing works around discontinuity problems at a point of inflection, two sample moving frames are shown in figure 3.14. In this figure the Frenet frame is compared to our technique and two of the three orthonormal vectors are shown using a colored surface. Note that the discontinuities at the points of inflection are removed in our coordinate frame. The second issue, caused by enforcing the tangent vector of the curve into the tensor, is also solved by loosening this definition by smoothing. Figure 3.15 shows how different results are given in two different cases, one where the tangent is smoothed/kept constant, and the other where the tangent vector is equal to the derivative of the curve.

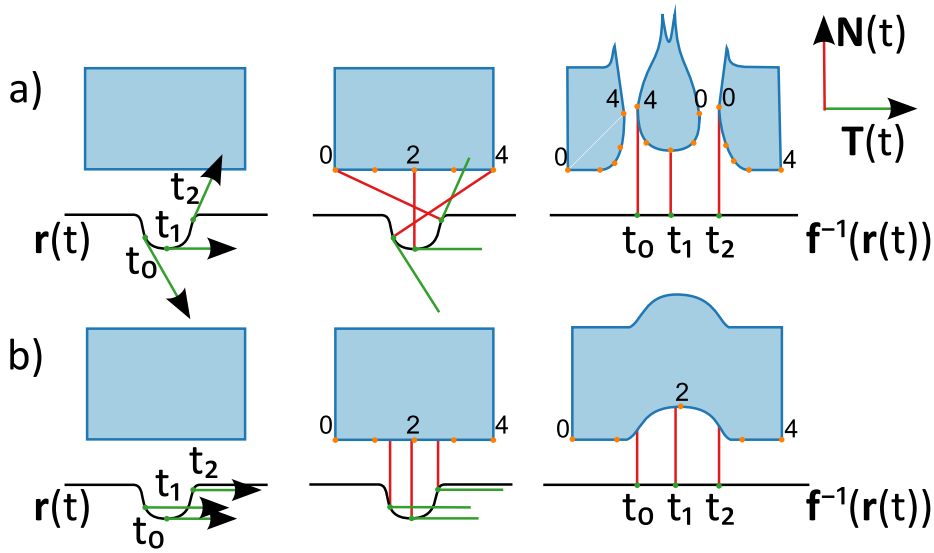

Figure 3.15: TThe blue box represents a volume, and the black line the curve to straighten. Two rows showing a) curve-linear reformation with tangential $\mathbf{T}(t)$ and b) the same curve-linear reformation, but here with constant tangents. Columns left to right show, tangents, normals, and lastly the deformed box.

Chapter 4

Demonstration

This chapter demonstrates the applicability of the techniques as described in chapter 3. It is divided according to the two industries that we collaborated most with, i.e., the petroleum industry, and maritime vessel traffic monitoring.

1 Drilling for Oil and Gas

This section provides three examples. Two exemplify the interactive visual analysis of streaming data using kernel density estimation. The third addresses the positional uncertainty associated with drilling.

1.1 Streaming Drilling Data

Streaming data, as introduced in section 2, are made from data-samples arriving in a sequential manner. Two important aspects of streaming data visualization were, first, the unknown *dataset size*, and second, that the stream in its entirety cannot be stored, only viewed *in situ* with a limited history. In the case of data from drilling operations, however, all the data is usually stored in massive databases. Still, for visualization and monitoring applications with limited resources, the practical scenario is that of an in situ setting.

The most prominent visualization of streaming data in the industry today is that of windowed time graphs, showing anything from the last hour to several hours. Such a graph, a typical drilling data visualization, or a dashboard, is shown in figure 4.1. In a typical operation center this would be placed on a large wall-display and connected with one or several drilling operations. This dashboard would then be used as a "background" to get a glimpse of the current status of the running operations, and a more thorough investigation would be done through other tools.

Progress Overview

Where understanding the recent history with the use of windowed data visualizations is common, getting a larger overview of longer processes / time-series is often problematic. In paper A we introduced a visualization where streaming

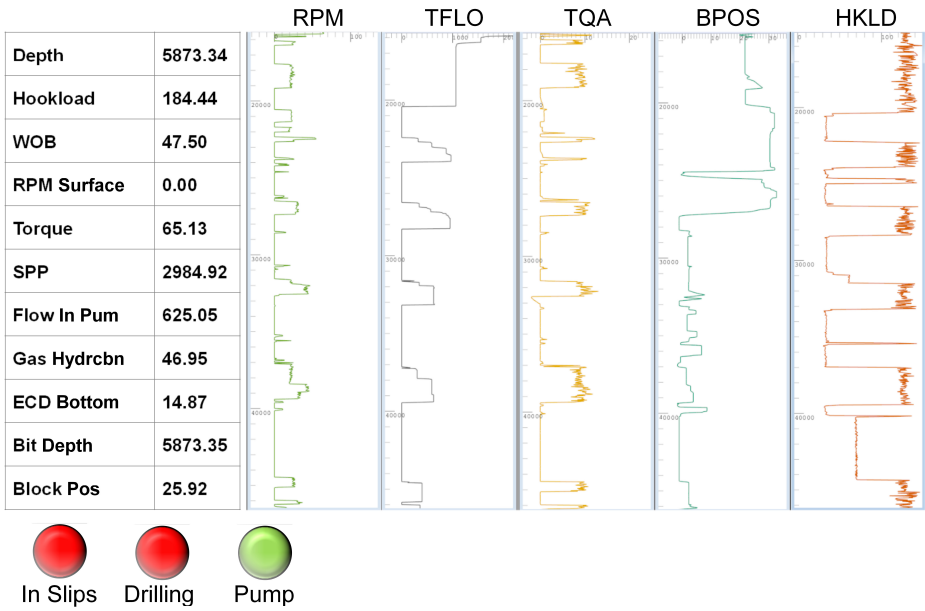

Figure 4.1: A typical dashboard visualization showing the recent history of a drilling operation with one graph per measured attribute (time displayed vertically) and the most recent values on the bottom. This figure also displays the exact values and some derived binary status indicators.

data are added to a kernel density estimation, eliminating the storage problem of the stream. Figure 4.2 shows streaming data added to a kernel density estimate showing the "time spent" distribution over *hook-load* and *measured depth*, two common parameters to visualize the progress in drilling operations. Using this visualization the user can observe where the majority of time has been spent in the drilling operation. Furthermore, since this visualization is quantitative, it enables user to mark an area and have the sum (or integral) interactively, and instantly, calculated, and display its result in (the unit) minutes; also shown in figure 4.2. This particular drilling operation, as shown in figure 4.2, stalled at just over a thousand feet, which was indicated by the large density there. The user specified box contains 76 minutes of non-productive time, which would need to be accounted for by further inspection or inquiry.

Investigating Friction

In paper E we introduced an additional interaction technique to KDE where the data can be modeled through interaction, without access to the data (as a requirement of the streaming access). A task often performed during drilling operations

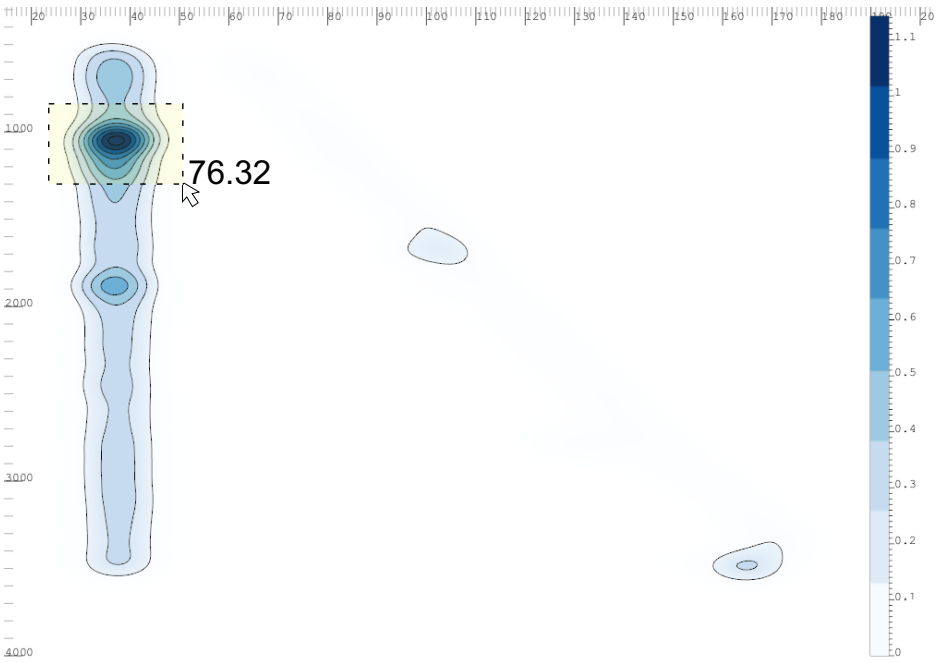

Figure 4.2: A kernel density estimate showing the distribution of time in a drilling operation. Highlighted in the top left is a large buildup of nonproductive time, which would need further inspection. Here, a large bandwidth (to smooth the data) is used to de-emphasize the details, leaving a high level overview.

is that of creating a *road map*. The road map is a chart of the expected/simulated frictions that will be encountered on the way down. The friction is an important parameter to monitor since it is an indicator of several potential problems. E.g., when there is a buildup of cuttings (loose materials), this can, in the worst case, lead to a stuck pipe. More frequently, however, but also potentially dangerously, this can lead to a *pack off*. A pack off occurs when the buildup of material gathers behind the wellbore and plugs the hole around the drill string. This plug leads to an increase in pressure, due to the loss of circulation. This pressure spike will often fracture into the formation, and lead to loss of mud, or influx.

The problem is, however, that friction cannot be measured directly, but needs to be derived from either the modified weight during movement, or derived from the measured torque while rotating. A common operation to perform at certain depths is called a *rotation off bottom* or a ROB. The drill bit is lifted off the bottom and a given rotation or RPM is maintained and the torque is measured. A road map will contain the expected torques on these given depths and is used to compare against these measured torques. Figure 4.4 displays the KDE of torque over depth, and is the result of accumulated streaming data. Using our interactive

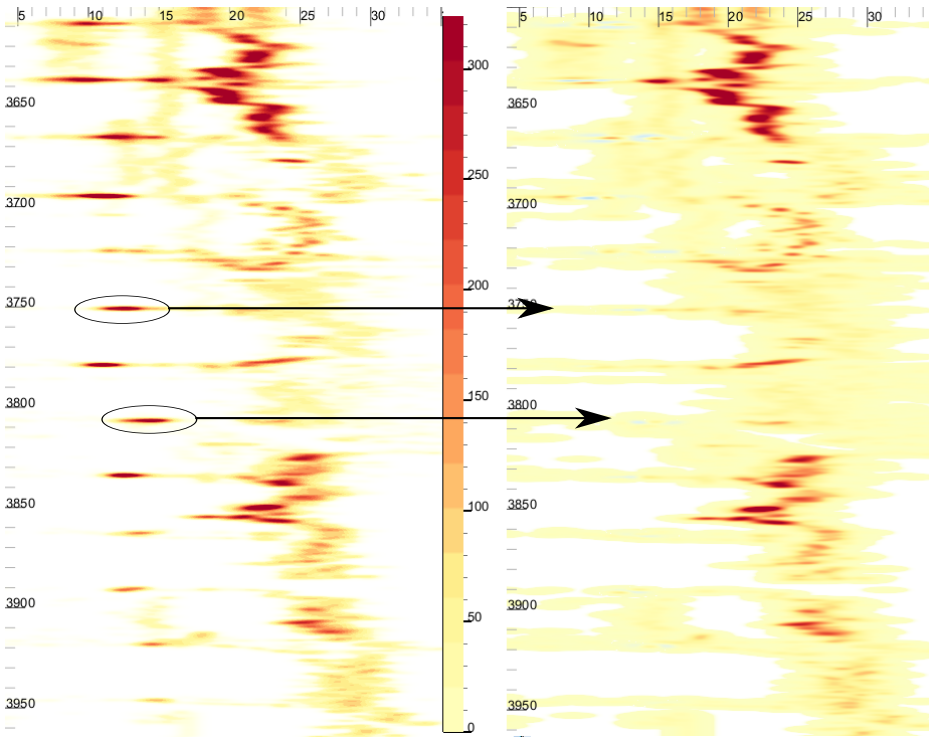

Figure 4.3: Torque in kN.m over depth. The figure on the left shows the original data, containing some ROB tests, which then were modeled and subtracted leading to the residual view on the right.

model building techniques the user sketches each rotation off bottom procedure, and extracts the depth and torque along with their respective variance. These extracted values can then be inserted into the roadmap, as an abstracted model of friction shown in figure 4.4. This figure can be compared against the predicted friction, and the calculated error bars can be used to determine the risk within given probability thresholds.

1.2 Positional Uncertainty

There are several uncertainties associated to the process of drilling. Such uncertainties can result from individual measurements where an error might be included, but these are often well understood and modeled. The larger uncertainties are both in what exactly one will find in the subsurface, but also in understanding exactly where the position of the drill bit is. The positional uncertainty is a cumulative error that can be modeled fairly well. The calculated potential position of the drill bit, at any given time, results in an ellipsoid-shaped

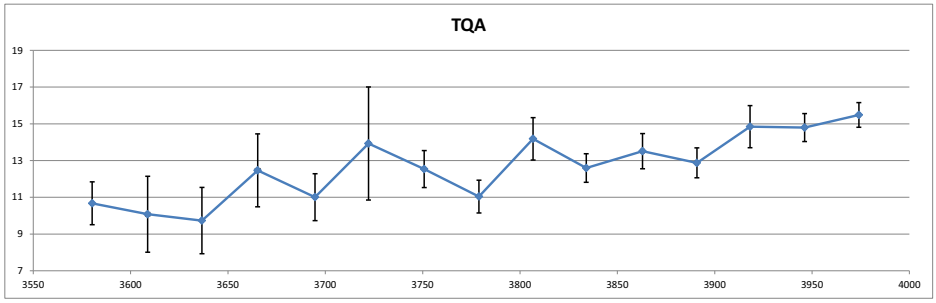

Figure 4.4: Changing torque in kN.m over depth in feet, for a series of ROB tests. The abstracted results from Fig.4.3 are shown in a graph with error bars at 1σ for both depth and torque uncertainty. In this figure, the uncertainty bars for depth are so small they cannot be seen, and are thus negligible.

field of uncertainty, with its mode the highest likely position. In figure 4.5 the ellipsoid, or ellipse in 2D, for 95.4% positional certainty is shown. An important part of the positional uncertainty lies in determining where the drill-string crosses over from one formation to the next. If there is a pressure difference within these, such as often found when entering a reservoir, special precautions must be made. These precautions are needed to make sure that the pressure inside the well does not drop below the pressure in the formation, at the same time as it should not rise above the formation fracture pressure[1]. One such precaution is the setting of a new casing[2], which allows adjusting to a higher pressure further down while protecting the formation above from fracturing.

The predominant way to observe the progress of the drilling process is measured in feet along the *measured length* or arc length, and to compare this to the planned well path. This one-dimensional parameterization of the planned path is shown along with the expected stratigraphy (expected layering) and the planned casings or milestones as a drill plan and overview. While the simplicity of this design serves its purpose well, there are several problems which could be addressed. The first problem is that this 1D stratigraphy is only correct given that the planned path is followed exactly, and the second problem is that the three-dimensional position error is reduced to a one-dimensional measure. In figure 4.5 the positional error is shown both in three dimensions, and in the described one dimensional case. Relying on the 1D stratigraphy combined with the projected error ellipse, this visualization will not indicate that the next formation is possibly entered. However, in the three-dimensional view, the ellipse is touch-

[1] Pressure above which the injection of fluids will cause the rock formation to fracture hydraulically.

[2] A casing is the insertion of a metal pipe almost as large as the well hole. Cement is then used to fill the gap outside the casing, permanently protecting the well from the outside formation.

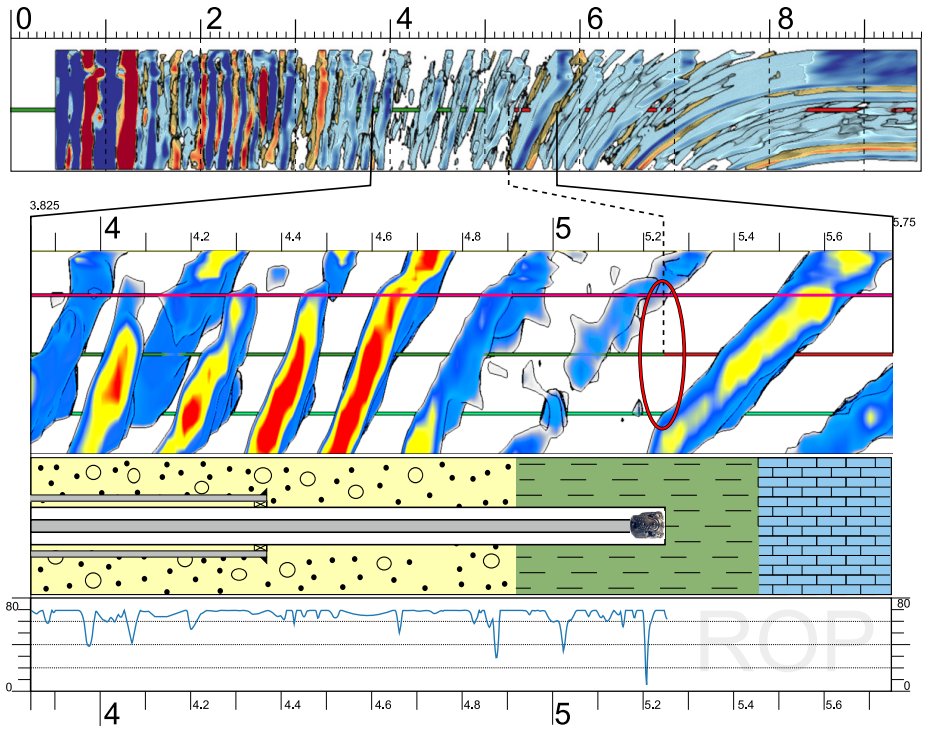

Figure 4.5: A reformed 3D seismic data visualization can provide spatial reference for real time drilling as well as showing uncertainty in the 1D lithology column (the second from the bottom image). The image on top shows the full length reformed wellbore, below a zoom-in of the section currently drilled, which also contains the 1D lithology/stratigraphy with the current drill bit position, and a real time graph showing the rate of penetration (ROP) for the section. The red ellipse in the center shows the positional uncertainty.

ing the next formation, and we can thus not guarantee with a 95.4% probability safety margin that we are still outside. This could provide an alert, cautioning the operator to be wary of other signs or indications of the increased pressure.

2 Differential Analysis of AIS Data

The Automatic Identification System, AIS, has the primary function of supporting collision avoidance, so that two vessels can detect each other's signals, alert and take corrective actions. In Norway the coastal administration has a series of receivers along the coast which collect and store the movement data of all the vessels within range. We were contacted by the Norwegian Coastal Administration

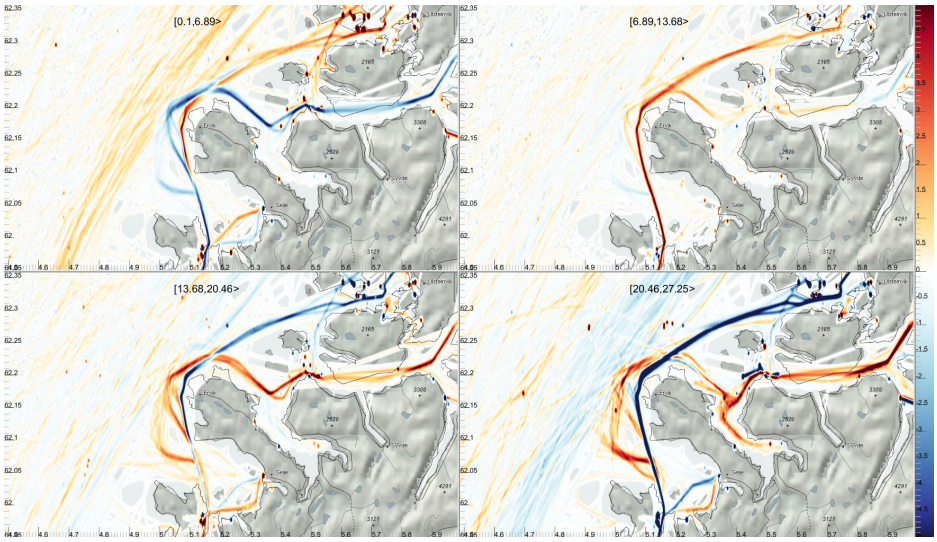

Figure 4.6: Passing vessels outside the Stad peninsula, and their changed movement pattern given stronger winds. From lower winds to stronger winds from the upper left to the lower right. Red colors indicate more than average traffic in that interval of winds, and blue colors indicate less than average.

to analyze AIS data related to questions on a proposed sea tunnel through the peninsula of Stad (detailed in paper D). One of the questions was to investigate how the traffic pattern around Stad changes due to the varying weather. Using the workflow proposed in paper D we first created a KDE of all the traffic around Stad. We then *expanded* this view with the wind-speed attribute, resulting in the visualization shown in figure 4.6. Our expand technique first categorized wind speed into four bins, then visualized the traffic given one wind speed bin subtracted by the average traffic. Investigating these four views (in figure 4.6) reveals that the red paths (more traffic than average) move further out from the coast outside Stad given stronger wind speeds. This indicates that the vessels opt for the safer route as wind speeds increase, something also shown north of Stad, where more than average vessels pull closer to the coast. This visualization (and other statistics presented in paper D) was handed off to the NCA which will potentially use them in their final recommendation to the government.

Chapter 5

Conclusion and Future Work

A starting point for this PhD project was to address the visualization challenges associated with the introduction of, and a possibly more widespread adoption of, technology enabling more advanced sensors during the drilling for oil and gas. We specifically sought to target the wired pipe [16] technology. A technology that increases the potential communication bandwidth with down-hole sensors by several orders of magnitude. This initial problem definition is quite narrow in terms of the application domain, but the visualization techniques that could be used to address it were open and left up to the researcher. From early on we recognized the strengths in creating solutions which were applicable beyond the application domain and to generalize our visualization techniques. We are confident that we have proven, through the application examples given, that we have achieved this.

An interesting research topic that we found highly applicable to process and streaming data was that of KDE based visualization. From our experiences, as documented through our publications, we see the biggest potential applications and usages for KDE-based visualizations being:

- As a frequency view, for dealing with occlusion, clutter, and thus scale beyond the number of samples a scatter-plot can coherently display.
- As a fixed memory representation for large streaming datasets. Independently of stream-size the memory usage of the KDE-based visualization remains constant.
- As a normative and quantitative visualization for samples with a unit that provides meaningful aggregates.
- As a visualization that supports algebraic operations, e.g., difference views, or, divide the KDE, of total contributions to a campaign per square mile, with the density estimate of people per square mile and get an estimate of contribution per person over an area.
- As a scale independent visualization technique, either through zooming, or through the usage of multiple bandwidths.

In paper E we introduced a technique that enables interactive visual analysis on this fixed memory footprint representation. As far as we know, this visual analysis on visual representations (as abstractions for the data) is an unexplored area of research, and is also an area we would like to see explored. Through our research

we addressed several challenges that we encountered during the exploration of KDE in visualization, but several more still exist. In future work we would like to see several challenges related to KDE-based visualizations addressed, including:

- Further explore the visual analysis of visual representations as an intermediary to the data.
- Using the same bandwidth for all samples is often not suitable. E.g., using the same bandwidth for two distinct distribution clusters. A variable kernel density estimate, however, sets an optimal bandwidth per sample, based on its local distribution.
- Exploiting the fixed memory bandwidth of KDE in large streaming data requires additional techniques on how to combine this with zooming and panning, e.g., rendering to a tile based map solution.
- The curve density estimate provides a continuous representation of the distribution of the curve. Modeling this distribution (e.g., detecting modes) could abstract the curve in ways that a moving average could not.

The implementation done for the different papers in this PhD project have been combined into a framework for interactive visual analysis, named Enlighten, as described in paper D. In the later years this application has also seen commercial usage in consultancy for different domains, additionally strengthening this research's applicability.