Examensarbete
LITH-ITN-MT-EX--05/030--SE

Visual Data Analysis using Tracked Statistical Measures within Parallel Coordinate Representations

Daniel Ericson

2005-04-08

Linköpings universitet
TEKNISKA HÖGSKOLAN

LITH-ITN-MT-EX--05/030--SE

Visual Data Analysis using Tracked Statistical Measures within Parallel Coordinate Representations

Examensarbete utfört i medieteknik
vid Linköpings Tekniska Högskola, Campus
Norrköping

Daniel Ericson

Handledare Jimmy Johansson
Examinator Matt Cooper

Norrköping 2005-04-08

Avdelning, Institution

Division, Department

Institutionen för teknik och naturvetenskap

Department of Science and Technology

Datum

Date

2005-04-08**Språk**

Language

- ☐ Svenska/Swedish
☒ Engelska/English

☐ _____**Rapporttyp**

Report category

Examensarbete

- ☐ B-uppsats
☐ C-uppsats
☒ D-uppsats

☐ _____**ISBN****ISRN LITH-ITN-MT-EX--05/030--SE****Serietitel och serienummer****ISSN**

Title of series, numbering

URL för elektronisk version<http://www.ep.liu.se/exjobb/itn/2005/mt/030/>**Titel**

Title

Visual Data Analysis using Tracked Statistical Measures within Parallel Coordinate Representations

Författare

Author

Daniel Ericson

Sammanfattning

Abstract

With our increasing ability to capture and store large multivariate data, these data sets are increasing in size and complexity. Traditionally, data sets from various areas of the society are examined using sophisticated mathematical techniques in order to discover strategic information hidden in the large amount of data. In addition to these automatic methods, a number of advanced techniques have been developed for the purpose of visualizing multivariate data, and to give the user a visual understanding of the data. Many of these techniques encounter problems like cluttered displays, as they are not designed to handle the amounts of entries that are stored in today's databases and data warehouses. This report investigates the current research situation of methods that address the problem of overplotted displays. A novel method called Visual Data Mining Display (VDMD) is presented, to overcome the stated problem by interactively selecting and displaying statistics of the data in a separate view. Changes in the display are visually tracked by animation and vector plotting for easy comparison of statistical values and subsets of the data. The method has proved helpful in providing an overview of large data sets, as well as in observing changes of the distribution in each dimension of the data.

Nyckelord

Keyword

visual data analysis, parallel coordinates, vector animation, tracked statistical measures

Upphovsrätt

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida <http://www.ep.liu.se/>

Copyright

The publishers will keep this document online on the Internet - or its possible replacement - for a considerable time from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: <http://www.ep.liu.se/>

Visual Data Analysis using Tracked Statistical Measures within Parallel Coordinate Representations

Daniel Ericson
NVIS - Norrköping Visualization and Interaction Studio
Linköping University
daner292@student.liu.se

April 20, 2005

Abstract

With our increasing ability to capture and store large multivariate data, these data sets are increasing in size and complexity. Traditionally, data sets from various areas of the society are examined using sophisticated mathematical techniques in order to discover strategic information hidden in the large amount of data. In addition to these automatic methods, a number of advanced techniques have been developed for the purpose of visualizing multivariate data, and to give the user a visual understanding of the data. Many of these techniques encounter problems like cluttered displays, as they are not designed to handle the amounts of entries that are stored in today's databases and data warehouses. This report investigates the current research situation of methods that address the problem of overplotted displays. A novel method called Visual Data Mining Display (VDMD) is presented, to overcome the stated problem by interactively selecting and displaying statistics of the data in a separate view. Changes in the display are visually tracked by animation and vector plotting for easy comparison of statistical values and subsets of the data. The method has proved helpful in providing an overview of large data sets, as well as in observing changes of the distribution in each dimension of the data.

Acknowledgements

This work is done in co-operation with the Department of Science and Technology at Linköping University and NVIS - Norrköping Visualization and Interaction Studio. It is part of the VISIMOD project [38], supported by the Swedish Foundation of Strategic Research.

I would like to thank my advisor Jimmy Johansson, for sharing his ideas to form this diploma work, and for constant and immediate feedback and discussions around the work. I would also like to thank my examiner Matthew Cooper for continuous support and valuable guidelines. Thanks to both of you for co-writing the paper submitted to the conference CMV2005.

Thanks to Margareta Voss at Karolinska Institutet in Stockholm, for providing interesting data sets for testing and evaluation of the application.

Lastly, but not leastly, I would like to give my greatest thanks to all my student colleagues in K5625 for inspiring coffee breaks, and for sharing your interesting projects. My best wishes for the future to all of you.

Daniel Ericson
Norrköping, April 2005

Contents

1	Introduction	1
1.1	Purpose	2
1.2	Methods	2
1.3	Target audience	3
1.4	Structure of report	3
2	Information Visualization	5
2.1	Background	5
2.2	Data type to be visualized	7
2.3	Visualization techniques	8
2.4	Interaction techniques	9
2.4.1	Visual Information Seeking Mantra	10
2.5	Parallel Coordinates	11
2.5.1	Describing geometry with parallel coordinates	12
2.5.2	Advantages of using parallel coordinates in visualizations	14
3	Analysis of multivariate data	16
3.1	Multivariate data	16
3.2	Data Mining	16
3.3	Visual Data Mining	17
3.4	Problem definition (revisited)	18
4	Related work	20
4.1	Brushing methods	20
4.2	Overview and Detail methods	20
4.3	Summarization methods	21
4.4	Re-arrangement methods	22
4.5	Animated visualization methods	22
4.6	Summary of related work	23
5	Implementation	24
5.1	Parallel coordinates	24
5.1.1	Histograms	25
5.1.2	Clustering techniques	27

5.2	Visual Data Mining Display (VDMD)	31
5.2.1	Plotting	31
5.2.2	Animation	34
5.2.3	Tracking	34
5.2.4	Normalization	35
5.2.5	Linking with parallel coordinates	36
5.3	Colour coding	38
5.4	Statistical aggregation functions	39
5.4.1	Mean value	40
5.4.2	Median	40
5.4.3	Geometric mean	40
5.4.4	Mode	40
5.4.5	Quartiles	41
5.4.6	Coefficient of variance	41
5.5	Features implemented, but not included	41
5.5.1	Automatic detection and removal of outliers	41
5.5.2	Brushing within axes	42
5.5.3	Undo/redo possibilities	42
5.5.4	Keyboard commands	42
5.5.5	Histogram for each region	42
6	Application Evaluation	43
6.1	Cars	43
6.2	Pollution	46
6.3	Stocks	48
7	Conclusions and future work	51
7.1	Conclusions	51
7.2	Future work	52
A	Development environment	54
A.1	The .Net concept	54
A.1.1	Visual Studio.Net	55
A.2	Matlab	55
A.2.1	Data structures	56
A.3	OpenViz	57

List of Figures

1.1	Parallel coordinates with cluttered overlapping lines.	4
2.1	Illustration of a cholera outbreak in London 1854.	6
2.2	2D point represented in cartesian and parallel coordinates.	11
2.3	3D point represented in cartesian and parallel coordinates.	11
2.4	Six-dimensional point in parallel coordinates.	12
2.5	Two six-dimensional points in parallel coordinates.	13
2.6	The ambiguous problem of parallel coordinates.	13
2.7	2D Line represented in cartesian coordinates.	14
2.8	2D Line represented in parallel coordinates.	14
2.9	Square represented in cartesian and parallel coordinates.	15
2.10	Cube represented in cartesian and parallel coordinates.	15
2.11	Hypercube represented in parallel coordinates.	15
3.1	Problem with cluttered parallel coordinates.	19
5.1	Graphical user interface of the VDMD.	25
5.2	Selected axis and subregion of parallel coordinates.	26
5.3	Histograms on top of parallel coordinates.	26
5.4	Clusters on top of the parallel coordinates.	27
5.5	2D objects plotted in a coordinate system.	28
5.6	Clusters and dendrogram of 2D objects.	29
5.7	Illustration of the K-means clustering algorithm.	30
5.8	VDMD after brushing interaction.	32
5.9	VDMD with statistics from four dimensions and four subregions. .	33
5.10	Drop-down menu for selecting statistics.	33
5.11	Animation in the VDMD.	34
5.12	Tracking of statistical measures in the VDMD.	36
5.13	Pinpoint interface.	37
5.14	Latest interaction information.	37
5.15	HSV colour space.	38
5.16	RGB colour space.	39
6.1	Parallel coordinate representations of the cars data set.	44
6.2	VDMD representation of statistics from the cars data set.	45

6.3	The zoom display of the VDMD	46
6.4	Parallel coordinate representations of the pollution data set.	47
6.5	VDMD representation of statistics from the pollution data set.	48
6.6	Parallel coordinate representation of the stocks data set.	49
6.7	VDMD representation of statistics from the stocks data set.	50
A.1	Data flow between components in the application.	55
A.2	OpenViz visualization pipeline.	60

Chapter 1

Introduction

Large data sets with many variables are becoming more and more common in many disciplines of society. Data are collected from all kinds of sources, and with dropping hardware prices and faster internet connections, accessibility and storage is becoming less of a problem. Millions of Terabytes of data are generated every year [1], and stored in large *data warehouses*. A data warehouse typically stores data collected from diverse sources, which makes it much more convenient to run queries over data that originally came from different sources. A lot of data are collected automatically, for example, from sources such as credit card transactions, internet traffic or telephony. Data have been continuously stored because people believe in the potential valuable information and patterns that can be hidden in the large amount of multivariate data. Understanding relations and structures of these kinds of data sets can be highly advantageous for anyone with interests in using the data as basis for market analysis and business decisions. Finding this information using traditional statistics or data management systems is difficult, and presenting complete data sets textually is impossible when the number of items to display increases. Sophisticated mathematical methods are being developed and have been used in order to have the computer calculate correlations and find patterns in large data sets. Such methods are frequently and traditionally used in decision support¹ systems to find structures in the data that are important for future development. The drawback of these kinds of tools is the lack of understanding that may be given to the user. The user who has to make the decision may not fully understand what trade-offs are involved, and why the mathematical model has proposed this solution as the best one [3]. From the mid 90's, when it was believed that human data analysts could and would be totally replaced by data mining [6], the trends are now pointing at the importance of letting the human expertise play a role in the investigation. The integration of visualization of data sets helps the user to get more knowledge by interactively exploring the data and having visual feedback from the interaction.

¹Decision support: Use of computers to supply and process information needed to make decisions. The term is typically applied to business applications.

1.1 Purpose

Information visualization and Visual Data Mining are increasing research areas and a number of advanced techniques have been developed for the purpose of visualizing multidimensional data. With more sophisticated acquisition and storage methods, data sets are rapidly increasing in size and complexity. The amount of entries that can be collected from today's databases and data warehouses is enormous. When visualizing these large data sets with any traditional method that displays a single data item per point, we frequently encounter overplotting problems like display cluttering, as they are not designed to handle this amount of data. Single data observations will be impossible to distinguish and trends will no longer be discernible. Figure 1.1 shows an example of overplotted *parallel coordinates* [18], which is a popular visualization technique described below. One part of this diploma work is to investigate previous techniques that have been developed in order to solve the problem of cluttered visualization displays. Another part, and the most accentuated, is to describe a novel method that uses the concept of coordinated views to extract and display statistics from the cluttered displays. The problem of overplotted displays, and the question that is examined within this thesis is explained again in section 3.4, after a background of the area has been given in the following chapters.

1.2 Methods

The first part of the project aimed at collecting information about previous related work. Since this work is extending the parallel coordinate technique, numerous articles and scientific papers about this specific technique and about approaches to solve the problem of cluttered visual displays have been read and summarized. The main ideas concerning the implementation were clear when the project started and a lot of papers have been read to be sure that someone else's ideas were not reinvented.

After this thorough investigation, the implementation phase started. The parallel coordinate display with interaction possibilities was implemented, as well as the basic initial ideas of plotting statistics in a separate view. As the implementation proceeded, some new ideas and features were discussed. Some of them were rejected, and some were implemented in the application.

In the evaluation phase, different kinds of data sets were used to test the application, in order to draw conclusions about its advantages and disadvantages when used in conjunction with parallel coordinates.

The application was developed in Visual Basic.Net, with the visualization library OpenViz as a complement. Matlab has been used for all necessary calculations. More about the development environment can be read in appendix A.

1.3 Target audience

The intended target audience of this report is anyone with interest in interactive visualization and visual data mining. Basic knowledge of mathematics and statistics is expected for a good understanding of the advantages of the described methods. Some tips for further reading are proposed when a subject is come across that is interesting and related to this work, but these are not essential for the overall understanding.

1.4 Structure of report

This report is organized as follows: An overview of the area of information visualization will be given in section 2. Chapter 3 discusses present and historical issues of analysis of multivariate data and specifies the problem that is considered within this thesis. Chapter 4 discusses previous work that addresses the problem of over-plotted visual displays and related visualization techniques for multivariate data. Chapter 5 gives a thorough explanation of the features implemented in the developed application called Visual Data Mining Display (VDMD). Chapter 6 evaluates the application and visualization methods. The evaluation is performed with a few case studies with data sets from different potential application areas. In Chapter 7, conclusions about the method are given and suggestions are made about what could be improved as future work. The appendix A describes the specific software and graphics libraries used in the implementation.

The figures of this report are best understood if printed in colour, although the connected text and captions should describe them well enough even for readers with black and white copies.

A shorter version of this report has been accepted for the presentation and for publication in the proceedings of the Third International Conference on Coordinated & Multiple Views in Exploratory Visualization (CMV 2005) to be held 5 July 2005 in London, England.

Figure 1.1: Parallel coordinates with cluttered overlapping lines.

Chapter 2

Information Visualization

2.1 Background

Today it is hard to go through a normal day without noticing any kind of information visualization. Just turn on your TV or open today's newspaper, and you are likely to find business diagrams, sales charts or even weather forecasts, visualizing a lot of data in a compact and understandable figure. The idea of visualizing information with the purpose of gaining insight into data is not a new issue. The traditional geographic map is an excellent example of how we always have tried to present our data visually in order to share abstract information with each other. Although it has been stated that some kind of maps were drawn on clay tablets as long as 5000 years ago [12], it was not until the middle of the 19th century that statistics were plotted on top of geographical maps, which can be seen as a step toward visualizing information from a combination of different sources.

A famous example of early use of information visualization is the illustration of a cholera outbreak in 1854 by Dr John Snow. He plotted the location of deaths on a map, which could be seen to be concentrated around a certain water pump, see figure 2.1. The pump was found to be contaminated and was taken out of service, which ended the neighborhood epidemic. This procedure was done without prior knowledge about what caused the cholera epidemic. For this reason it is often mentioned in information visualization contexts, where the solution, or even the problem, might not be known before the data set is visualized.

Despite the fact that some kind of visualization has been used for thousands of years, information visualization is still a fairly new area of research. With the vast amount of data that is to be processed within traditional data mining and data exploration systems, there is an increasing need for human interaction for a better utilization of the collected data. A large number of visualization techniques have been developed over the last 10 to 15 years to support the exploration of large data sets. Information visualization today is not only about presenting results in charts, more important is the ability to interact with the data to see how variables are correlated with each other. Many of the visualization and interaction techniques

Figure 2.1: Illustration of a cholera outbreak in London 1854. The plots are location of deaths from cholera and the filled squares are pump sites.

are developed as research projects within computer laboratories. It is the belief of many people that these techniques are ready to be commercially adopted and used as everyday tools for information professionals [39]. One important argument for this is that we have reached a stage where even sophisticated interactive visualizations of large data sets can be produced on a standard desktop computer, which is essential for reaching the crowd of people with expertise within a specific area, but with no exceptional computer experience.

The need for visualizing a data set can have various reasons, and many different types of techniques have been developed for different purposes. The technique to choose depends on the goal to achieve, and the level of prior knowledge about the data. In *exploratory* visualizations, it is not necessarily known from the beginning what the user is looking for. One goal can be to come up with a hypothesis, that can be further examined using other tools or visualization techniques. To reach this goal, it is important to have a dynamic visualization that responds to user interactions. When the user already has a hypothesis that needs to be tested, the visualization is said to be *confirmatory*. The interactivity of the visualization does not have to be as high, since the scenario is more predictable. The task of the visualization is to help the user to confirm or refute a given hypothesis. When the hypothesis is tested and validated, the visualization can work as a *demonstrative* tool, in order to present the result in a meaningful way. The visualization is more

focused on demonstrating the result than for the user to interact with the data.

The classification above is described in [9], which also provides a good overview of specific techniques developed for interactive visualizations, as well as an evaluation of the same. In this report, the focus is mainly on the first two types of visualizations (exploratory and confirmatory), since the goal is to use the visualization to learn more about the data and to test hypotheses. Interaction is an essential part of these types of visualizations.

Another deeper classification of data and techniques used in information visualization was proposed in 2002 by Daniel A. Keim. The classification is based on the *data type to be visualized*, the *visualization technique*, and the *interaction and distortion technique*. Some of these categories with sub-categories are described briefly below to provide an overview of the area of information visualization. For a further explanation of specific techniques, the reader is referred to [1] and to the sources referred to in the Related work chapter. Since the method described in this report is extending the parallel coordinates, this technique will be described in detail in section 2.5.

2.2 Data type to be visualized

In most cases, the data to be visualized consists of a number of *tuples*, each having a number of variables or dimensions. Each tuple, or row in the data, corresponds to an observation, measurement, transaction, etc. whereas each column represents one property of the tuple. The number of dimensions in the data, as well as the variation in complexity, yield the following categories of data.

One- Two- and Three-Dimensional Data Temporal data are examples of one-dimensional data. Temporal data can be thought of as timestamps, where each data item denotes the evolution of the object over time. Two-dimensional data consists of two primary attributes that can be represented with two axes. Such data are often represented as x-y-plots in a two-dimensional coordinate system. Geographical data are typical examples, where the two attributes are latitude and longitude, which can be displayed on a map. Adding a z-axis perpendicular to the x-y-plot, three-dimensional data can be mapped onto such a coordinate system.

Multi-Dimensional Data Data that consist of more than three variables have no natural mapping to the two dimensions of the screen, or to the three dimensions of our space, and hence need more sophisticated visualization methods. Multi-dimensional data are found everywhere where an object or a record is to be described with many attributes. Examples are tables from *relational databases*, where each column represents one dimension.

Text and Hypertext Data does not necessarily have to be described with a number or a value. From sources such as the World Wide Web, a lot of information

is presented in text format. Most of the standard visualization techniques can not handle this type of data without prior transformation into a format that makes the entries comparable with each other. Such a transformation can, for example, be as simple as counting all words in a text, in order to compare how frequently they appear, or to group words within similar contexts. The result can then be visualized using traditional methods or with other metaphors such as Themeriver [17].

Hierarchies and Graphs Data records are often structured in hierarchies and can have relations to each other, as well as to other pieces of information. The data in a hierarchical data set are organized in a tree structure, where each item, or *node*, has a single parent node (except for the top-most root node). Hierarchical structures can be found in many areas, such as business organizations, computer data storage systems, and genealogical trees.

Algorithms and Software Another class of non-numerical data are algorithms and software. Keeping the overview of large software projects can be a challenging task. Visualization can help the developer in understanding the structure, for example, by showing the flow of information in a program. It can also be of value to support the programmer with visual aids when debugging, such as visualizing errors.

2.3 Visualization techniques

The domain of visualization techniques is wide. It ranges from basic standard 2D/3D-techniques, such as x-y(x-y-z) plots, bar charts, pie charts etc. that can easily be created in spreadsheet software such as Microsoft Excel, to more sophisticated solutions that map many dimensions onto the two dimensions of the screen. The following categories correspond to the basic visualization principles proposed by Keim. These are just basic principles, and many techniques use the benefits of a combination of several principles.

Standard 1D/2D/3D displays Standard x-y plots or 3D visualizations fall into this category of visualization techniques, where each value is mapped onto an axis of a coordinate system with two or three dimensions. One-dimensional data can be described using just marks on one single axis.

Geometrically Transformed Displays The aim of geometrically transformed displays is to find a transformation of multidimensional data sets so that they can be described using the two dimensions of the screen. The parallel coordinate technique described in section 2.5 is one example.

Iconic Displays The idea of iconic display techniques is to map the attribute values of a multidimensional data item to the features of an icon. In such a visualization, the appearance of the icons will change according to the values of the data. Different dimensions of the data can, for example, be mapped to the position and colour of the icon, but also to its shape and angle. An early approach of multidimensional iconic displays is the Chernoff face [5], where values are mapped onto the properties of a human face.

Dense Pixel Displays In dense pixel displays, each pixel corresponds to one dimension value. By arranging and colouring the pixels in an appropriate way, the visualization can provide detailed information on correlation and dependencies in the data set.

Stacked Displays When working with hierarchical data, the concept of stacked displays can help to understand the different levels of the hierarchy. The basic idea is to have one coordinate system embedded into another. On a two-dimensional screen, two attributes form the outer coordinate system, and two other attributes are embedded into the outer coordinate system and so on.

Many visualization techniques aim to present details of the data in one part of the screen, while preserving an overview in another. The concept of using two distinct views for this purpose is known as *Overview and Detail*. A related principle is to keep the overview and detail in the same view, but to magnify or focus on the interesting part. This is often referred to as *Focus + Context* techniques.

2.4 Interaction techniques

For a modern visualization system to be effective, and to provide exploration of the data, the interaction possibilities are of crucial importance. The advantage of moving visualizations from paper to the computer screen is that the screen is dynamic. A map drawn on a sheet of paper is static, unless you want to erase and redraw the whole map, whereas a visualization on a computer screen can change its appearance to dynamically visualize new aspects of the data. Traditionally, the computer screen is regarded as output device and the mouse is used for inputs from the user. These distinctions are rarely found in the real world, where we are used to interacting with objects around us, and getting feedback from the same. For example, you can both draw and print your ideas and display them for your colleague using the same sheet of paper. One goal of an interactive visualization, is to blur the difference between input and output devices to resemble the interactions we are used to in real life. Graphical objects do not need to just act as a nice-looking presentation of the data. They can also be used as input from the user, in order to find out more about the data. They can, for example, be clicked on with the mouse to extract details, or act as an interface for changing parameters or selecting

data. The challenge of creating interactive visualizations has led to an extensive development of a large number of techniques that let the user interact directly with the visual objects. Some of the main concepts are described below.

Interactive Filtering When visualizing and exploring large data sets, it is of high value to have the ability to limit the range of data values that are visible, in order to focus on interesting subsets of the data.

Interactive Zooming Zooming into the visualization of the data is a widely used technique when dealing with large data sets. Zooming is not only about making the objects larger, but also about presenting more details at higher zoom levels. It is often an essential feature to provide an overview of the data, and to be able to zoom into interesting subsets.

Interactive Distortion Interactive distortion techniques can be said to extend the zooming feature. Interesting parts of the data are shown with a high level of detail, while the periphery of the same view shows the context in a more compressed representation. The advantage is that both the overview of the data and the focused part can be displayed simultaneously in the same view.

Interactive Linking and Brushing Combining different visualization methods can be a good way of utilizing the advantages of several techniques. Linking them so that changes in one visualization are reflected in another view has been proved to be of high value for the understanding of the data [35].

2.4.1 Visual Information Seeking Mantra

There are many guidelines about how to design interactive visualization interfaces, but an important one can be summarized as the Visual Information Seeking Mantra [32]:

- Overview first
- Zoom and filter
- Then detail on demand

The core of this *mantra* is that all visualizations should provide an overview of the data for an overall understanding of the structure and the dimensions. The next step is to zoom in on interesting items and filter the data to clear the view from objects that are not of interest for the investigation. Having selected an interesting subset of the data, it should now be possible to extract details from single objects or groups that are currently displayed.

2.5 Parallel Coordinates

The parallel coordinates technique [18] was introduced in the early eighties as a new way of presenting multi-dimensional information. In traditional cartesian coordinates, all axes are perpendicular, and the highest number of dimensions to visualize is three. For data represented in parallel coordinates, the dimension axes are placed equally spaced and parallel to each other. Each data item is represented as a set of line segments connecting the points on each axis that are included in the data.

A point in a two-dimensional cartesian coordinate system is visualized in parallel coordinates as a line connecting two axes at the intersections that correspond to the two values (x and y) of the point. Figure 2.2 shows a point represented in cartesian coordinates (left) and in parallel coordinates (right). Similarly, a point in

Figure 2.2: A single point in 2D represented in cartesian coordinates (left) and in parallel coordinates (right).

three dimensional cartesian space is visualized with two line segments connecting three axes, as shown in figure 2.3. Generally, a point in n -dimensional space becomes a polygonal line laid out across the n parallel axes with $n - 1$ line segments connecting the n data values.

With the fact that the axes in a cartesian coordinate system need to be perpendicular to each other, a six-dimensional point can not be represented in such

Figure 2.3: A single point in 3D represented in cartesian space (left) and in parallel coordinates (right).

Figure 2.4: A six-dimensional point $(-5, 3, -4, -2, 0, 1)$ represented in parallel coordinates.

a system, without varying the appearance of the symbol that is plotted. It would, however, be possible to change the size or the colour of the symbol to include more dimensions. Visualizing the same point in parallel coordinates though, is just a matter of adding more axes to the display. The only limit on the number of dimensions that can be displayed is the resolution of the screen, or the space on the paper. The six-dimensional point $(-5, 3, -4, -2, 0, 1)$ is visualized as shown in figure 2.4. Each value of the point is localized at the corresponding axis and then connected by line segments. The result is a polyline that intersects all axes of the parallel coordinates. To visualize multiple points in the same coordinate system, one polyline is drawn for each point in the n -dimensional space. Figure 2.5 demonstrates how the points $(-5, 3, -4, -2, 0, 1)$ and $(5, 3, 3, 0, 0, 3)$ are visualized in parallel coordinates. Already here, we have encountered one of the problems with parallel coordinates. Consider the parallel coordinate representation in figure 2.6. It is not clear whether the original data corresponds to the upper or the lower of the cartesian coordinate systems. One solution could be to colour the lines differently to separate them from each other.

2.5.1 Describing geometry with parallel coordinates

A line in cartesian coordinates can be considered as a connection of a series of points. Samples from the line $(x_2 = -3x_1 + 20)$ are plotted as points in figure 2.7. The parallel coordinate representation of those point is shown in figure 2.8. Each line in the parallel coordinates represents one point in the cartesian coordinates. Note how all lines intersect in one point between the axes. In general, a two dimensional line, $x_2 = mx_1 + b$, is represented by the point $(1/(1 - m), b/(1 - m))$ in parallel coordinates.

A square in a two-dimensional cartesian coordinate system can be described with four points. As we have seen, points in cartesian coordinates are represented as lines in parallel coordinates. Figure 2.9 shows how a square is represented in

Figure 2.5: Two six-dimensional points $(-5, 3, -4, -2, 0, 1)$ and $(5, 3, 3, 0, 0, 3)$ represented in parallel coordinates.

Figure 2.6: The ambiguous problem of parallel coordinates. It is not clear whether the parallel coordinate representation to the left corresponds to the upper or the lower of the cartesian coordinate systems to the right.

cartesian and in parallel coordinates. Adding another axis to our parallel coordinate representation, we have the opportunity to describe points in three-dimensional cartesian space. Figure 2.10 shows how a cube is represented in cartesian and in parallel coordinates.

The analogy between the cartesian and the parallel coordinate systems stops when we reach three dimensions. A point with four, five or six dimensions can not be described in our three dimensions of space. In parallel coordinates, however, additional axes are added for each dimension. The 256 corners of an eight-

Figure 2.7: Samples from the line ($x_2 = -3x_1 + 20$) represented in cartesian coordinates.

Figure 2.8: Samples from the line ($x_2 = -3x_1 + 20$) represented in parallel coordinates.

dimensional hypercube can be visualized in parallel coordinates as in figure 2.11.

2.5.2 Advantages of using parallel coordinates in visualizations

In a parallel coordinate system, the shape of the line segments can convey information about the values of each dimension for the data item. It is an intuitive way of displaying an overview of all dimensions and all objects in the data set simultaneously. The strength of the parallel coordinates is particularly shown when some interaction is allowed, and observations can be made on subsets of the data. One usual interaction is the ability to select polylines to see all the properties of a single data object. There are usually means of constraining the range of values in one dimension to see how other dimensions are affected.

Figure 2.9: A square represented in cartesian coordinates (left) and in parallel coordinates (right).

Figure 2.10: A cube represented in cartesian coordinates (left) and in parallel coordinates (right).

Figure 2.11: 256 corners of a eight-dimensional hypercube represented in parallel coordinates.

Chapter 3

Analysis of multivariate data

3.1 Multivariate data

Humans, by nature and history, try to keep the number of dimensions to observe as low as possible. We try to simplify and arrange the world in a structure that we can understand, and we want to be able to describe complex structures with some kind of a system. Just think of classical arrangements, like the periodical system and Linne's classification of plants and animals. The truth is often more complex and objects with multivariate properties are found everywhere in our everyday life. In fact, everything that can be described with a set of properties can be said to be multivariate, or more correctly multidimensional, since the term variate implies that the attributes are dependent. Independent attributes should be termed dimensions.

In early types of data analysis, humans manually calculated and represented statistics and summaries from the data using graphs, charts and tables. With the introduction of digital data storage, data sets could easily contain a few hundred or even thousands of dimensions. When the number of items in the data sets was also growing, it became impossible to analyze them manually. There was a need for an automated activity of exploring the data, which is now referred to as data mining.

3.2 Data Mining

Data mining is the activity of exploring databases with the goal of extracting information and discovering patterns and structures in the data that were initially hidden and unknown. There are many definitions of the concept of data mining, but [9] describes it in a short and compact way as:

Data mining is the mechanized process of identifying or discovering useful structure in data.

Sometimes, data mining has been treated as a synonym for *Knowledge Discovery in Databases*, abbreviated KDD, although some researchers claim data mining to be a part of the wider concept of KDD [15]. Besides data mining, KDD actions

would include tasks like acquiring and cleaning the data, integrating multiple data sources and transform the data into a useful format. These steps are extremely important for a satisfying result of the data mining. However, they are not in the scope of this work so no further details are explained here. The interested reader is referred to the book *Exploratory Data Mining and Data Cleaning* [6] by Dasu and Johnson.

Data mining projects are traditionally performed using non-visual techniques, such as statistical methods, rule induction¹ or unsupervised neural network² modeling. The goal can be to try to fit the data into a specific functional form described by mathematical parameters. There was a point in the mid 90s when it was believed that computers could totally replace the human analyst and perform all data mining activities automatically [6]. The data mining would act as a black box, transferring the raw data into satisfying result with interesting patterns. However, the value of having a human expert in the analysis has again risen. It is now the general belief that most analyses require domain-specific knowledge that can never be replaced by a computer.

3.3 Visual Data Mining

Re-integrating the human into data analysis requires a concept of how the user can interact with the data. Many interaction techniques are derived from the area of information visualization to form the branch *Visual Data Mining*. A definition of Visual Data Mining is stated in the foreword of the proceedings from the International Workshop on Visual Data Mining, 2001 [40]:

Visual data mining is a collection of interactive reflective methods that support exploration of data sets by dynamically adjusting parameters to see how they affect the information being presented.

The main difference between the definitions of Data Mining and Visual Data Mining is that the former has the word *mechanical* in it, while the latter focuses on *interactive*. This implies that in Visual Data Mining the user is in charge of the analysis and the knowledge is gained by exploring the data interactively. Just as with traditional data mining, the goal is to find correlations in databases that were initially not known.

Introducing visualization into data mining is clearly needed to provide an experimental environment. Although the benefits are clear and visualization techniques are being developed, there are still not many systems that effectively integrate both visualization and automatic data mining algorithms. Some data mining

¹Rule induction: Classifying the data using logical decision rules.

²Neural network: A form of artificial intelligence in which a computer simulates the way a human brain processes information.

packages include visualization, but most often with just a small number of techniques included, or with poor help in knowing when to best apply the specific visualization technique [9].

3.4 Problem definition (revisited)

Having explained the background of information visualization and data mining in this and the previous chapter, it is time to return to the problem stated in section 1.1. It should now be clear what the strengths are with the parallel coordinates technique. Looking at figure 1.1 on page 4, it can also be understood that the method has its weaknesses when it comes to displaying large data sets. The advantages of having the ability to display the properties of many dimensions on a two-dimensional screen are drowned by the fact that changes in the distribution along the axes are not discernible when the data set is large enough. Looking at figure 3.1, there seems to be no significant difference in the distribution of points along the axis between figure a and b. The only thing we can clearly see is that some objects have been removed from the very bottom of the axis. The truth, which we can not see from these two representations, is that objects have been removed along the whole lower half of the axis. Figure 3.1(c) shows the objects that have been removed from the highlighted data. Even if we can see the range of the removed objects, there are too many objects to determine the density of lines along the range. One major problem that can be stated is that, when the data set is large, the visual information given by parallel coordinates does not always agree with the real distribution of values. The reason is clearly that the number of pixels is not large enough to represent all the visual items in a good way. With the development of higher resolution screen standards, such as HDTV³, we would expect to be able to display more items on the same space. However, as long as we are limited by a finite space on the screen with a finite number of pixels, the fundamental problem will still remain, as data sets will continue to grow. The problem of overplotted visual displays will only increase as data sets get bigger and bigger. There is a need for alternative ways of displaying summaries of large data sets. This leads us to the main question behind this diploma work.

Can we visualize the distribution of values along the axes of the parallel coordinates, in such a way that changes can be understood and variables can be compared, even if the original parallel coordinate view is cluttered with too many data items?

³HDTV: High-definition television. A new display standard with far higher resolution than today's standards

Figure 3.1: The difference between the representations in (a) and (b) is hard to determine because of the cluttered view. Figure (c) shows the actual difference between (a) and (b). The objects in (c) are the ones that are removed from (a), such that (a)-(c)=(b).

The approach of a novel solution, that is described in detail in chapter 5, can be summarized with the three following steps.

- Display statistics of the data as plots in a separate view.
- Animate the plots when the statistics change.
- Leave a visible track in the display that shows how the plots were animated.

Chapter 4

Related work

There have been many previous efforts to overcome the problem of overplotted visualization displays. While some focus on reducing the number of items to display simultaneously, other approaches aim to summarize the whole data set and display important characteristics. This chapter presents work that is in some way related to this thesis. These are mainly extensions or combinations of techniques described earlier in the sections 2.3 and 2.4. Most of the methods are just described briefly in this report, since the purpose is to give an overview of the current research situation, not to provide full details of specific techniques. For each method, a reference is given to the corresponding scientific paper, or to a book where the reader can learn more about the specific method.

4.1 Brushing methods

Brushing is the operation of interactively selecting subsets of the data to be highlighted, masked or deleted. Brushing multivariate data requires a concept of how to map all dimensions onto the two dimensions of the screen. Martin and Ward [22] describe the design of several brushes that are defined in data space rather than screen space. Many different types of brushing methods have been developed. Hauser et al. [16] present a method called angular brushing to brush data with respect to the correlation between adjacent axes in parallel coordinates.

4.2 Overview and Detail methods

There are several well-known techniques that take interesting parts of the data into consideration while ignoring or blurring the rest. The advantage of this is the ability to keep the overview of the whole data set, while focusing on a subset of the data. Examples are distortion techniques like the fish-eye lens [11, 31] and the Perspective Wall [21].

4.3 Summarization methods

When the whole data set is of interest, displaying the distribution of values, rather than focusing on each single item of the data, can be beneficial. Histograms are intuitive ways of showing such distributions and have been implemented with parallel coordinates in, for example, [29] and [16].

In the scatter plot matrix, described in [9], all dimensions in a data set are compared pairwise and mapped onto a two-dimensional projection. The projections are arranged in a grid structure to give the user an impression of the overall relations between dimensions in the data.

Another way of summarizing data is to group it into clusters to see if there are any obvious groupings in the data, that can be further manipulated and analyzed. Extensions to traditional clustering techniques have been proposed in order to gain more information about the overall structure and information about single clusters. Fua, Ward and Rundensteiner [10] propose a multiresolution view of parallel coordinates to obtain a level of detail structure via hierarchical clustering. Johansson, Treloar and Jern [19] use an unsupervised learning algorithm to initially classify clusters of data displayed in parallel coordinates. The size of the clusters are visualized by varying the width of the band representing each cluster.

Siirtola [34] describes two novel techniques to manipulate parallel coordinates. The polyline dynamically summarizes a set of polylines by displaying the average line in a selection. The other technique enhances the knowledge of correlation between variables by plotting a bar between the ranges. The bar points upwards or downwards, indicating whether the correlation between adjacent axes is positive or negative.

An approach to interactive data summarization is proposed in [20]. The idea is to have the computer do the exhaustive search rather than the user, and inform the user in advance about which manipulations would change the summary the most. The authors also state the relevant difference between summarization and other tools that allow the user to explore, rather than summarize, the data in order to find interesting and significant patterns. Summarization can be viewed as a kind of lossy compression, keeping the most important observations.

The box plot, described in [24], was introduced in the 1970's and is still a widely used technique to display variable distributions in many statistical software packages. It encodes minimum, maximum, mean, median and quartile information in a compact representation. The Mondrian tool [37] uses multiple boxplots on top of the axes in a parallel coordinate plot to display statistics of a given subset of the data. In [4], box plots are extended to "ellipse plots" to compare subsets of the data with the whole data set.

Zhao et al. [42] introduce trend figures as an extension to parallel coordinates. The horizontal axis of the figure represents the sequence of the data record, and the vertical axis shows its value in each data record. Extending each axis of the parallel coordinates with these trend figures enables the user to quickly observe the variables that change in similar ways. Their work was specifically used to observe

changes made in the design and test cycle of mobile phones.

In [2], axes in parallel coordinates are scaled based on normalization by statistics in the data set, which can be helpful in showing the distribution of data points on the axes.

Miller and Wegman [23] describe how to replace the raw data in parallel coordinates with a density plot, in order to better be able follow structures in the data when lines overlap each other.

4.4 Re-arrangement methods

Axis re-arrangement is an old and now obvious extension to parallel coordinates. Comparison of two dimensions is best performed when the axes are placed next to each other. The XmdvTool [41] is a public-domain software package for interactive multivariate data exploration where this feature is implemented. In [19] the re-arrangement of axes can be done with mouse interaction in the user interface.

Deletion and addition of axes [16] can be useful to clear up the view, and to compare multiple variables with one specific dimension of interest.

Peng et al. [28] discuss in more detail the advantages of re-arranging dimensions in multi-dimensional visualizations, and propose algorithms to automatically find the optimal axis arrangement.

The Reorderable Matrix [33, 30] presents multivariate data graphically in a table that has objects in columns and properties of those objects as rows. Each crossing between rows and columns has a rectangle whose size is relative to the corresponding data value at that point.

Siirtola [35] examines the benefits of combining two conceptually different information visualization techniques. The parallel coordinates plot and the reorderable matrix were used to view the same data, with positive results. He concludes that linking different kinds of displays can enable users to see different things in their data, as well as reducing the cognitive load when they switch between the views.

When two lines share a point on an axis in parallel coordinates, an ambiguity problem occurs which makes it impossible to distinguish which line segments are connected, as described in section 2.5. Graham and Kennedy [14] present a method that replaces line segments with smooth curves, allowing individual data elements to be traced. They also propose spreading of points to further separate line segments or curves from each other.

4.5 Animated visualization methods

Animation has been widely used in visualizations to enhance the understanding of time-varying variables. Typically, animation has been used to demonstrate complex physical simulations, for example, to show particle traces that change over time in a scientific visualization.

“The Animator” [26] was introduced by Barlow and Stuart with the purpose of illustrating how animation can be used in parallel coordinates. The line segments of the parallel coordinates are animated to enhance the understanding of how objects within the multidimensional space are changed over time.

Elmqvist and Tsigas [8] present a technique called “Growing Squares” where they use the metaphor of colour pools spreading over time on a piece of paper, to visualize causal relations in a system.

4.6 Summary of related work

The sections above describe previous work that is related to the ideas of this thesis. These are extensions to parallel coordinates or other methods of displaying multivariate data. Solutions exist, that summarize the data displayed in a parallel coordinate plot using aggregated information, for example in [37, 4, 29]. However, most of them are focused on displaying statistics for a given state of the data, without letting the user follow changes of the statistics that may have occurred as a consequence of interaction. The novelty of the VDMD application presented next is the ability to follow and track changes of the data set statistics via animation and vector plotting.

Chapter 5

Implementation

This chapter describes the implementation of the Visual Data Mining Display (VDMD), which is a complement to parallel coordinates. It combines the concepts of describing data with statistics and the concept of using visualization for gaining insight into data. The purpose of the VDMD is to display statistical analyses of the data in such a way that they enhance the user's ability to understand correlations and structures in large data sets. The VDMD is specifically designed for data sets where the number of items is too big for them to be visualized in a meaningful way by traditional multivariate visualization techniques.

The data to be loaded into the application can be any complete multivariate data set, structured as rows and columns in a spreadsheet. The steps associated with pure KDD (see chapter 3.2), are performed manually before the data set is loaded. That means that a specific data set is selected and objects with missing data values have been removed to avoid holes in the data.

Several of the standard interaction and visualization extensions to the parallel coordinates technique described in chapter 4 are implemented, such as axis re-arrangement, brushing and clustering. The purpose of this is twofold. First, for the examination of how this method can be compared with features already implemented in previous work, and second for the exploration of how it functions as a replacement or as an extension to these.

The GUI¹ of the application (figure 5.1) consists of three parts. The upper central part is the parallel coordinate representation of the data set. When axes in the parallel coordinates are selected, their corresponding statistics are plotted in the VDMD in the central lower part of the interface. Menus for interaction with the VDMD and the parallel coordinates are located to the left of these displays.

5.1 Parallel coordinates

When the application starts, the data are visualized in parallel coordinates. Each axis is split into four parts that divide the range into four sub-ranges of equal size.

¹GUI: graphical user interface

Figure 5.1: Graphical user interface of the VDMD application.

Whole axes, as well as sub-regions of axes, can be selected, using simple mouse interaction, for further exploration in the VDMD. Figure 5.2 shows a zoom of three axes in the parallel coordinates, where one axis and one region have been selected. Brushing is performed by moving ‘handles’, attached to each axis, to crop the data in this dimension. In figure 5.2, objects from the lower part of the leftmost axis have been deselected with the handle. The cropping is propagated to all axes and complete polylines will be removed. Objects that are removed from the selection are shaded in grey, but still visible in order to retain the visual information about the complete data set. To enable comparison of arbitrary dimensions side by side, axes can be re-arranged by simple drag-and-drop mouse interaction.

Classifications of the currently selected data, as clusters and histograms, can be turned on and off as layers on top of the parallel coordinates.

5.1.1 Histograms

The addition of histograms (figure 5.3) on each axis helps the user to get a quick overview of the distribution of data points in an initial stage, as well as to observe changes of the distribution after brushing or interaction. The drawbacks of this representation are its static properties and the inability to convey changes of multiple dimensions simultaneously. Even though changes may occur on many of the

Figure 5.2: Three axes from the parallel coordinates. The rightmost axis and the third region on the middle axis are selected.

Figure 5.3: Histograms are placed on top of each axis to show the distribution of objects.

Figure 5.4: Six clusters on top of the parallel coordinates, drawn at the corresponding centroids of the clusters.

histograms, it can be hard for the user to focus on more than one or two dimensions at a time.

The histogram algorithm divides the data of each axis into a number of equally spaced containers and returns the number of elements in each container. The data can then be visualized with bars representing each container. The lengths of the bars correlate to the number of elements in each container.

The number of containers, or bins, to divide the data into is by default the square root of the initial number of objects, but can be selected in the interface to fit the user's needs. There is no optimal way of selecting the number of bins so that it fits all data sets, but [24] claims that the number of bins should increase with the number of observations (n) and suggests the square root of n or $1 + \log_2 n$ to be appropriate values.

5.1.2 Clustering techniques

Clustering is known as the operation of dividing a data set into subsets or groups of data items with similar characteristics. The data should be divided, such that all objects within each cluster are more closely related to one another than to objects in other clusters. The two major algorithms used for this purpose are *hierarchical clustering* and *K-means clustering*. Both techniques are described below, to state the difference between them, although the K-means is the one used in the application. The reason for choosing K-means before hierarchical clustering is mainly that it operates much faster on large data sets.

Hierarchical clustering

In hierarchical clustering, all objects are compared pairwise, to compute the distance between them all. There are many ways to calculate distances between ob-

jects, the most common used and the default algorithm in Matlab² is the *euclidean* distance. It is basically the length of a straight line between two points in multidimensional space. Using the Pythagorean theorem, the euclidean distance between all points is the sum of the squared distances between the vector values, $d_E = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$. Another popular distance measurement is the *City Block* distance, also known as the *Manhattan* distance. The name inherits from the fact that the distance can only be measured in right angles, as would be the case if we had to walk from one place to another within a city block system. It makes sense to use the City Block distance when one wants the result to be discrete rather than continuous, since the distance can only be whole multiples of the unit that is used.

The result of the distance calculating operation is a matrix containing the distances between all objects. For example, consider a data set made up of five two-dimensional objects. Each object has a position registered as an x-y coordinate.

- Object 1: 1, 2
- Object 2: 2.5, 4.5
- Object 3: 2, 2
- Object 4: 4, 1.5
- Object 5: 4, 2.5

Plotting these objects in a two-dimensional coordinate system would look like

Figure 5.5: The five two-dimensional objects presented above.

figure 5.5. When all distances are calculated, the result can be presented as a matrix of euclidian distances as follows:

²Matlab is a tool for doing numerical computations, that is used for all calculations in the application. Section A.2 describes how Matlab is integrated in the application.

$$\text{distances} = \begin{pmatrix} 0 & 2.9155 & 1 & 3.0414 & 3.0414 \\ 2.9155 & 0 & 2.5495 & 3.3541 & 2.5 \\ 1 & 2.5495 & 0 & 2.0616 & 2.0616 \\ 3.0414 & 3.3541 & 2.0616 & 0 & 1 \\ 3.0414 & 2.5 & 2.0616 & 1 & 0 \end{pmatrix}$$

The diagonal of the matrix is obviously zero, since it represents the distances between each object and itself.

Using this distance information, objects with the shortest distance between each other are grouped together into binary clusters (clusters made up of two objects). The next step is to group these created clusters into new clusters, each containing four objects. The procedure continues until all objects are linked together in a hierarchical tree. Following the example above, it can be seen that objects 1 and 3 are grouped together, as well as objects 4 and 5 are. They create new clusters, which in the next step are grouped together to form a bigger cluster. The last step is to join this cluster with object 2 to complete the hierarchical tree. The clusters are shown in figure 5.6(a). This tree representation of hierarchical clusters are also known as a *dendrogram*, and is shown in figure 5.6(b). The information that this graphical representation carries is twofold. The end of the lines point at the objects or clusters that are grouped together, and the length of the lines represent the distance between objects or clusters centroids. The numbers 1 to 5 represent the actual objects, while the numbers 6 to 8 signify the newly created clusters.

Figure 5.6: Clusters and dendrogram of the objects in figure 5.5.

K-means clustering

The K-means clustering is an iterative algorithm that uses optimization methods to minimize the sum of distances between all objects and their corresponding cluster centroids. The algorithm follows the steps below, illustrated by figure 5.7.

1. Assign the number of clusters to be calculated, (e.g. $k = 5$).
2. Randomly position k initial cluster center locations.
3. Each data item finds its closest cluster center.
4. Each center finds the centroid of all items that it owns.
5. Each center moves to its calculated centroid.

Figure 5.7: Illustration of the iterative steps of the K-means algorithm (with permission from Andrew W. Moore [25]).

After each iteration, the sum of all distances between objects and the centroids of their clusters is calculated. Step 3 to 5 are repeated until the centroids no longer move. Just as with hierarchical clustering, there are several ways of measuring the distance between objects in multidimensional space.

The lines in figure 5.4 represent the centroids of the clusters. The number of clusters to be calculated and displayed can be selected in the interface. The least number is two, and it can not exceed 100. The reason for the upper limit is that the clusters themselves make the display cluttered when too many lines are plotted.

The reason that the parallel coordinates technique has been chosen to be built upon is its advantage of easy and intuitive selection and brushing of the data. The approach might, however, be usefully applied in conjunction with other multivariate visualization techniques.

5.2 Visual Data Mining Display (VDMD)

The VDMD consists of several two-dimensional coordinate systems, that are linked with the parallel coordinate view to display statistics from selected dimensions and subranges of the data. It is divided into three parts that represent different levels of detail within the selected axes. The leftmost part of figure 5.9 shows statistics for a whole axis, while the middle part is divided into four regions, each displaying statistics of the corresponding sub-region of the axis. Each subregion is considered independently, in order to gain more information, at a local level, about the axes. Observations of changes at this level can be valuable in understanding what causes the statistical value for a whole axis to change. The rightmost part of the VDMD further divides a single region of an axis, selected by clicking on that region in the parallel coordinates axis display, into four sub-regions of equal size, and statistics of these sub-regions are displayed in the same manner. This level-of-detail feature could be extended indefinitely by dividing sub-regions into even smaller regions but, at present, experiments have only been carried out with three levels since, in the data sets used so far, the number of items would then probably be too small to calculate any useful statistics from.

The plots in the VDMD represent statistical values of selected dimensions. To provide comparison between selected dimensions, multiple symbols can be plotted side by side in the same display. All statistical values are normalized to keep a value between 0 and 1, therefore even dimensions with different ranges can be compared. The symbols representing the statistical values are, when changed, animated to their new positions, and leave a visible vector as a track of the last movement. Each of these features are described in detail in the following sections.

5.2.1 Plotting

Selected axes of the parallel coordinates will be highlighted with predefined colours that are associated with the specific dimensions even after re-ordering of the axes.

For each axis selected, graphical symbols will be plotted in the VDMD to display statistics from the specific dimension. One symbol is plotted in the left display and the middle and right displays will be populated if the corresponding subregion of the axis contains any data item.

Figure 5.8: The appearance of the VDMD before and after the left axis in figure 5.2 has been brushed. The median value of the right axis of figure 5.2 is plotted on the y-axis. The x-axis represents the mean value.

The graphics in the VDMD follows the same colour coding as the parallel coordinates, and can thus easily be connected with the corresponding axis. Each symbol in the VDMD is positioned to display two statistical values for the associated axis in a two-dimensional coordinate system. When axes in the parallel coordinates are brushed, so that the plotted statistical value changes, the symbol will be moved to represent the current value. The symbol will also be moved if it is set to represent another aggregation function, whose value is not equal to the value of the previous one. Figure 5.8 demonstrates how the median and mean values of a selected axis change when the brushing of another axis is performed.

The statistical aggregation functions that are mapped to the x- and y-axes respectively can be changed by the user in drop-down menus, see figure 5.10. The default graphical glyph in the VDMD is a filled square. Additional symbols in the shape of filled circles can be added to display other statistics from the same dimension. If no aggregation function is selected, the symbols will be lined up according to the internal positions of the parallel coordinate axes. The changes in the VDMD will then only occur in one dimension, which can be advantageous if only one aggregation function is of interest.

The default graphical glyph in the VDMD is a filled square. Additional symbols in the shape of filled circles can be added to display other statistics from the same dimension.

Figure 5.9: Plotting of statistical values from four dimensions and four subregions of the parallel coordinates. Coefficient of variance is plotted on the x-axis and the mean value is plotted on the y-axis.

Figure 5.10: Aggregation functions are selected in a drop-down menu.

Figure 5.11: Sequence of six frames during the animation after one variable of the parallel coordinates has been brushed. The sequence demonstrates how the median and the coefficient of variance of four axes are changed when objects on fifth axis are brushed.

5.2.2 Animation

Once a change is made that affects the position of a symbol, it will be moved to its new position in the VDMD. To better be able to follow the changes, all movements of the symbols are animated along a vector to their new positions. This feature gives the user a good sense of how different variables are affected by a certain interaction, and hence which variables are correlated with each other. Since animation of multiple symbols start and stop at the same times, the speed of the movement will correspond to the size of the change of the statistical value. A faster movement means a bigger change, and will tend to attract the user more. Figure 5.11 demonstrates how animation is used to enhance the visual cues about changes in the display. The animation in the example is run for 100 frames and takes two seconds.

5.2.3 Tracking

The user has the option to track the latest change in the display by having a vector drawn, that represents the movement of the glyph. The endpoint of the vector is always at the current position of the selected statistical value, that is, the position where the glyph is plotted according to section 5.2.1. It also follows the animation scheme described in section 5.2.2. By default, the origin of the vector is set to be the previous position of the glyph. The vector starts as a zero vector and is elongated until it reaches its endpoint. Consequently, the result is a vector representing the effect of the change that occurred in the last interaction, whether it occurred as a change of the selected data in the parallel coordinates or as change of aggregation

function in the menu.

In addition to this, the origin of the vector can be locked at the current position by clicking the button “Set Pinpoint” shown in figure 5.13. The current position will be set as a visible ‘pinpoint’, and future vectors can be chosen to have this point as its origin. If the checkbox “Draw from pinpoint” is checked and a change occurs in the VDMD, the vector will visualize the difference between the current value and the value represented by the pinpoint. In this case, the magnitude of the vector might increase or decrease depending on how the symbol is moved.

The purpose of the pinpoints is to help the user remember specific statistical values from specific subsets of the data. If multiple axes are selected in the parallel coordinates, vectors are drawn for each axis to provide comparison between variables. A great advantage of the vector plotting is the ability to track changes of arbitrary dimensions, even if the corresponding axes were not selected at the time of interaction. When pinpoints are set, they are calculated for all dimensions in the data set, but only visible for the ones selected in the parallel coordinates. If an axis or a region is selected after a change is made, the vector will still represent the effect of the last interaction, and can be compared with vectors already visible. This means that the user does not have to keep track of changes of all dimensions simultaneously to see how they are affected by an interaction. Figure 5.12 demonstrates a sequence where the median is set as a pinpoint, and later compared with the 25th and 75th percentile respectively.

Figure 5.13 shows the pin-point control interface, the bottom half of which shows the current pin-points that have been set. The max and min values are useful for determining how the selected data was restricted when the pinpoints were set. The values that are changed by the user to brush the data are marked with an asterisk. The colour legend to the right of the dimension names is an additional help in connecting the symbols with the correct axes. There are also notations about which statistical variables the pinpoints represent.

Information about the latest change of the VDMD is also given in text format (figure 5.14) as a reminder of what last affected the VDMD. Using this information it is clear how the vectors should be interpreted, and what the origin and the endpoint of each vector represent.

5.2.4 Normalization

In order to be able to compare statistical values from dimensions with different ranges, all values must be normalized before being used in the system. All values in the dimension x are locally normalized to keep a value between 0 and 1 using the following formula:

$$(x_n = x_i - \min(x)) / (\max(x) - \min(x))$$

For the plotting in the zoom displays (middle and right displays in figure 5.9), the subregions are locally normalized with respect to the boundaries of each subregion.

Figure 5.12: Vectors are plotted in the VDMD to show the difference between statistical measures. (a): Median is set as pinpoint for the y-axis. (b): The current aggregation function is changed to 25th percentile. (c): The aggregation function is changed to 75th percentile, while the pinpoint is still at median.

5.2.5 Linking with parallel coordinates

One of the main purposes of the VDMD is to see how statistical values are affected when the selected subset of the data is changed. Therefore, interactions in the parallel coordinates view are directly reflected in the VDMD. There is also a need for linking in the opposite direction to display the real statistics from the normalized values of the VDMD. The real values of a symbol's position can be obtained by clicking anywhere in the VDMD. Labels on each axis of the parallel coordinates will then be visible, showing the real value that corresponds to the y-position of the click. In figure 5.9, the lines on both sides of the leftmost display indicate which normalized value is being processed.

The aggregation functions available in the menu are: mean, median, mode, 25th and 75th percentiles, geographical mean and coefficient of variance. All but the last are applicable to be mapped along an axis with real values. The coefficient

Pinpoints

☒ show pinpoints

☒ draw from pinpoint

set pinpoint

x-axis:

y-axis:

Hide pinpoint information

Number of items at pinpoint:

Current number of items:

dim		min	max
1: mpg		-48.5	-9
2: cyl		* 4.2	8
3: disp		68	473.4
4: hp		* 81.2	239.3
5: weight		* 2798.3	* 4978.5
6: acc		8	25.8
7: model		70	* 80
8: origin		1	3

Figure 5.13: Information about the current pinpoints is given in a separate view.

Latest change

Brushing

origin (upper) : 3 => 1.7

(a) The latest change was a brushing of the parallel coordinates.

Latest change

Statistics

y-axis: mean => median

(b) The latest change was a change of the aggregation function to plot.

Figure 5.14: The latest change that affected the VDMD is displayed in the menu.

of variance does not have a natural mapping onto the real values of the axes, and is, for that reason, not applicable to be displayed on the labels on the axes.

5.3 Colour coding

When colours are used in a visualization, it is of very importance that their meaning can not be misinterpreted by the user. In the case of plotting symbols in the VDMD, it needs to be clear which dimension, or which subregion, in the parallel coordinates the symbol corresponds to. The best way of achieving this is by having the colours of the plotted symbols as widely separated from each other as possible. To achieve this, the HSV colour space can be used. Figure 5.15 shows how the hue, saturation and value (HSV) components are mapped. The hue component is an approximation to the visible spectrum and is determined by an angle starting at 0 degrees with red, and varies along the circle through yellow, green and blue before it returns to red at 360 degrees (or 1 using normalized values). The saturation component denotes how *pure* is the colour perceived by the viewer. No saturation means that only the grey scale is used, and a high-saturation colour is vivid and intense. The value component gives the colour a brightness, and is denoted by a point on the black-white axis. By setting the saturation and value of each used colour to a maximum, bright and separated colours can be obtained by varying the hue component. The hue component of each colour is set so that the colours are equally spaced on the edge of the colour circle. The colours are calculated in HSV colour space and converted to RGB colour space, to fit the colour conversion of the visualization software. In the additive RGB colour space, various amounts of the three components red(R), green(G) and blue(B) are mixed to produce new colours.

A slightly different way of achieving separated colours is to use the RGB colour cube. Figure 5.16(a) shows how the colours are arranged in a colour cube. Figure 5.16(b) shows the same information, but is intended for black/white copies of this report. The colour black (0,0,0) is located as an origin in one of the corners

Figure 5.15: HSV colour space represented as a conical object.

of the cube. From this corner, the three primaries red, green and blue are mapped on each of the perpendicular axes of the cube. The algorithm used for separating colours uses the three dimensional space that is spanned by the colour cube, and arranges the colours so that they are as much spaced from each other as possible. The enhanced colour cube algorithm used in Matlab also attempts to provide more steps of grey, pure red, pure green, and pure blue.

In the VDMD application, the user can choose which algorithm to use for colouring the axes in the parallel coordinates. The subregions of the axes are always coloured using the colour cube algorithm, since the number of subregions are most often too big to have the HSV algorithm return colours not too similar to each other.

There are plenty of other ways of describing and mapping colours, that will not be further mentioned here. The interested reader is referred to [13].

Figure 5.16: RGB colour space represented as a cube.

5.4 Statistical aggregation functions

Analyzing summaries of the data instead of the raw data is a way of speeding up the task of the exploration of large data sets. Computing typical values and comparing them with others has been proved to help us find structures and patterns in the data [6]. Each statistical aggregation function has its advantages and disadvantages, and is used for different purposes. Combining several functions can often reveal more information about the structure of the data than if they were used individually.

All statistical values are calculated with respect to the normalized values of each dimension or subregion of an axis. Each time a change of the current data selection occurs, all statistics are re-calculated using the Matlab engine described in section A.2. The following aggregation functions are implemented.

5.4.1 Mean value

The mean value is simply calculated by adding all values and dividing the sum by the number of elements. The mean value is widely used and gives a good overview about the data. However, if the distribution is highly skewed, or if it contains extreme outliers, the mean value is no longer as efficient.

5.4.2 Median

The middle value of a distribution is called the median. If the number of observations is odd, the median is simply the middle number. For example, the median of the sequence 2, 4, 7 is 4. When there is an even number of observations, the median is the mean of the two middle numbers. Thus, the median of the sequence 2, 4, 7, 12 is $(4 + 7)/2 = 5.5$.

5.4.3 Geometric mean

The mean value, in its daily use, is also known as the *arithmetic* mean. The *geometric* mean answers the question, "if all the quantities had the same value, what would that value have to be in order to achieve the same product?" Mathematically, the geometric mean of the numbers $(a_1, a_2, a_3, \dots, a_n)$ is the n th root of the product $(a_1 \cdot a_2 \cdot a_3 \cdot \dots \cdot a_n)$. Among other areas, it is often used in financial contexts. For example, consider an investment that increases 10% the first year, 60% the second year and 20% the third year. What is the average rate of return per year? The geometric mean will give the answer $(1.1 \cdot 1.6 \cdot 1.2)^{1/3} \approx 1.283$.

5.4.4 Mode

The most frequent value of a distribution is known as the mode. In a distribution with discrete values, that is, only a set of predefined values can be used, the mode can give an understanding of how the values are distributed. In a normal or near-normal distribution, the mode will be close to the mean and the median. In a symmetric distribution with one single local maximum, the mode, mean and median may have the same value. In a distribution with continuous values, the mode may be a very insecure way of analyzing statistics. The reason is that between any two numbers of a continuous distribution, there are an infinite number of other possible values. Two values that are almost identical will then be considered to be different when calculating the mode. The calculated mode may appear where a few values happen to be exactly identical, even if the actual peak of the distribution is somewhere else. Such distributions are better visualized with histograms, where adjacent values are binned together and considered equal.

5.4.5 Quartiles

The p^{th} percentile of a set of data is defined as the value which is greater or equal to at least p percent of the data and which is less than $(100 - p)$ percent of the data. The upper and lower quartiles, as well as the median described above, are special cases of the percentiles of the data. The upper quartile is the same as the 75th percentile, meaning that 25% of the data have values that are higher than this value. In the same way, the lower quartile is the value where 25% of the data have values that are lower than this value. In an example of 12 sorted values, (3, 4, 7, 9, 9, 10, 12, 13, 14, 16, 17, 18) the lower quartile is 8 and the upper quartile is 15, meaning that 25% of the values are lower than 8 and 25% of the values are higher than 15.

5.4.6 Coefficient of variance

It is in many cases not enough just knowing the actual placement of statistical values. Two dimensions might have exactly the same mean and median values, but have different distributions. There is a need to show how much the data varies along the dimension. Usual measures for this purpose are variance and standard deviation. The variance $V(X)$ is defined as $V(X) = E(X - m)^2$, where E denotes the average, or mean value. If the distribution is concentrated around m , $(X - m)^2$ will be small and hence the variation will be small. The standard deviation $D(X)$ is the square root of the variance, $D(X) = \sqrt{V(X)}$. The ratio $R(X) = D(X)/E(X)$ is called the coefficient of variance. The coefficient of variance is a measure of how much the data varies along the dimension.

5.5 Features implemented, but not included

Since this project has been done with the purpose of researching alternative novel ways of displaying large data sets, a lot of features have been tested in order to find the best solutions. Some features were implemented in the research phase, but not included in the final version due to various reasons. They are described briefly below, with an explanation of why they have been rejected. They are rejected from this version of the application, but can be thought of in future versions or integrations with the VDMD. Possible future work of the VDMD specifically is further discussed in chapter 7.

5.5.1 Automatic detection and removal of outliers

Outliers are data objects that show an abnormal value compared to the rest of the data set. Detecting outliers can be important from two perspectives. They can be the objects that contribute the most to the characteristics of the data. A good example is the informal “80-20” rule, where 20 percent of the customer base generates 80 percent of the profits [6]. Another type of outlier can arise simply with bad

quality of the data. Typos, like a missing separator comma or an extra zero can generate fatal errors in the data. The automatic detection and removal of outliers was removed because of the potential danger of letting mathematical algorithms hide objects that should contribute to the statistics plotted in the VDMD.

5.5.2 Brushing within axes

The first brushing feature that was implemented, allowed the user to brush arbitrary sections within the parallel coordinate axes. That would let the user more specifically choose which data items should be highlighted and included in the statistics of the VDMD. The feature is good, but it did not integrate well with the more important ‘handles’ described in section 5.1. For them to work together, the interface needs to better clarify how data have been removed. The feature with the handles was kept, because of the more intuitive way of setting the max and min values for each dimension.

5.5.3 Undo/redo possibilities

Actions that resulted from brushing within axes could easily be reversed using undo and redo features in the menu. This feature did not make any sense when the brushing feature described above was removed, but is certainly essential as a combination with the same.

5.5.4 Keyboard commands

In early stages of the implementation, keyboard commands were used to alter the actions of mouse click and drag interactions. With less actions to be performed with the mouse, the keyboard commands were brought down to the menu of the GUI for a more visual understanding of possible interactions.

5.5.5 Histogram for each region

At one point, histograms were implemented for each of the subregions of an axis. They were removed because they did not convey more information than was available from the histograms of whole axes.

Chapter 6

Application Evaluation

The application and ideas have been evaluated using data sets from three different areas. The first two were originally of moderate size (up to 500 items in 8 dimensions), but one has been synthetically extended to hold more objects. These have been collected from [36]. The third one is a data set of stocks information of approximately 12000 items in 6 dimensions.

6.1 Cars

The first data set is a modified and extended version of a classic data set of a collection of cars. The data consists of 3136 observations and each entity has 8 variables: miles per gallon, number of cylinders, displacement, horsepower, weight, acceleration, model and origin. All of the data items are numeric. The origin variable is a number between 1 and 3 where 1 is America, 2 is Europe and 3 is Japan.

Our task is to examine how the *weight* variable is affected by brushing the *origin* axis in the parallel coordinates. Are American cars in general heavier than European or Japanese cars?

If we keep just the American cars, we can see in figure 6.1(b) that some objects are removed from the very bottom of the weight axis. This would indicate that the cars with the lowest weight are not produced in America, but we still know little about how the distribution along the axis has changed after the brushing. Figure 6.1 demonstrates how the parallel coordinate representation is changed when the *origin* axis is brushed.

Even if the parallel coordinates display is too cluttered for us to see changes in density along the axis, using the VDMD will still help us to draw conclusions about the data. We let the square symbol display the median of the *weight* axis, taking the whole data set into consideration. If we then keep just the American cars by brushing the *origin* axis, we will see how the median changes to a higher value, see figure 6.2(b). This indicates that American cars are, in general, heavier than European and Japanese cars. Looking at the zoom display of the VDMD in figure 6.3, we can see that it is mainly the two lower regions that have caused the change

(a) Entire data set. The weight axis is selected.

(b) American cars are highlighted. The weight axis is selected.

(c) American cars are highlighted. Multiple axes are selected.

Figure 6.1: Parallel coordinate representations of the cars data set, showing the effect of brushing the *origin* axis and selecting multiple axes.

of the median. These observations would be an indication that most American cars have a higher weight than average. If the pinpoints were set to represent all cars in the data set, we can now select any other axis to see how it was affected by

Figure 6.2: VMD representation of statistics from the cars data set. X-axis: Coefficient of variance. Y-axis: Median. (a): Statistics of the *weight* axis when the entire data set is included. (b): Statistics of the *weight* axis after the brushing in figure 6.1(b). (c): Statistics of the selected axes in figure 6.1(c).

Figure 6.3: The zoom display of the VDMD shows which parts of the weight axis has been affected the most by the brushing in figure 6.1(b).

the brushing of the *origin* axis, see figure 6.2(c). This is one great advantage of the VDMD. Even if we did not initially intend to concentrate on changes on other axes, we can now easily select arbitrary axes to see how they were affected. We can, for example, draw the additional conclusions that engines of American cars seem to have more horsepower and bigger displacements than the average car. From figure 6.1 it is hard to determine how many cars with three, four or five cylinders have been shaded. Figure 6.2(c) shows how the VDMD can help us with this task. The longest vector, representing the number of cylinders shows that the median has increased significantly after the brushing, indicating that American cars generally have engines with more cylinders than average.

6.2 Pollution

This data set originates in a norwegian study where air pollution is measured and compared with traffic volume and meteorological data. It contains 500 items in 8 dimensions. The first column represents the logarithm of the concentration of NO_2 at each observation, and column 2 is the logarithm of the number of cars per hour. Columns 3 to 8 represent temperature, wind speed, temperature difference between 25 and 2 meters above ground, wind direction (degrees between 0 and 360), hour

(a) Entire data set. The cars axis is selected.

(b) Observations with high concentration of NO_2 are highlighted. The cars axis is selected.

(c) Observations with high concentration of NO_2 are highlighted. Multiple axes are selected.

Figure 6.4: Parallel coordinate representations of the pollution data set, showing the effect of brushing the *response* axis and selecting multiple axes.

of the day and a continuous day number.

Using this data set, we first want to know if there is any correlation between

Figure 6.5: VDMD representation of statistics from the pollution data set. X-axis: Coefficient of variance. Y-axis: Median. (a): Statistics of the *cars* axis when the entire data set is included. (b): Statistics of the *cars* axis after the brushing in figure 6.4(b). (c): Statistics of the selected axes in figure 6.4(c).

high response value of NO_2 and the amount of cars passing. Figures 6.4(a) and 6.4(b) demonstrate how the parallel coordinates representation changes when we highlight the observations on the upper half of the *response* axis. We can observe that the density of observations on the lower half of the *cars* axis appears to decrease. This can be confirmed by looking at the changes of the VDMD. Figure 6.5(b), shows how the mean value of the *cars* axis is increased. From figure 6.5(c) we can see that high concentration of NO_2 also seem to be correlated with lower wind speeds and lower temperatures than the average observation.

6.3 Stocks

The last data set to be tested contains approximately 12000 entries of stocks information, from which we have no initial knowledge. The first two columns represent the company and the date. Column 3 to 6 show the opening, highest, lowest and

the closing rates, respectively. The last column represents the volume of the stock. Figure 6.6(a) shows the entire data set represented with parallel coordinates. We want to know if the overall closing prices have been higher or lower than average over the last time period. In figure 6.6(b), the entries with dates from the upper quartile are highlighted. It can be observed that entries with the highest closing prices have been removed. This would tempt us to believe that the overall closing prices have decreased. Looking closer though, we can see that many of the removed objects have had lower closing prices too. From these observations, we are not able to draw any conclusions at all. Extending the parallel coordinates with the VDMD, we can observe that the overall median is in fact higher during this period, see figure 6.7.

(a) Entire data set.

(b) Stock entries from the latest time period are highlighted.

Figure 6.6: Parallel coordinate representation of the stocks data set.

Figure 6.7: VDMD representation of the closing rate axis. The y-axis represents the median.

Chapter 7

Conclusions and future work

7.1 Conclusions

It can be concluded that information visualization and visual data mining is a big and active area of research. A lot of visualization and interaction techniques have been developed to be used in exploration of large data sets. With many of these techniques, the problem of cluttered views is getting bigger and bigger, as the size of data sets continues to increase. There is a need for alternative ways of visualizing objects or summaries of these large data sets, in order to find out more about the data.

In this report, the Visual Data Mining Display (VDMD) has been introduced, which is a method to extract and display statistics from data presented in parallel coordinates. The method is suitable for analysis of correlations and patterns in multivariate data, especially when the number of data objects is too large to be visualized in parallel coordinates without clutter. It has proved helpful in giving an overview of large data sets, as well as in observing changes in the distributions, making use of animation and vector plotting. The method has advantages both with continuous and discrete data. Large sets of continuous data may appear very cluttered in parallel coordinates, whereas multiple line segments of discrete data sets may appear as one single data item.

The method can be used both by experienced and novice users of visual data mining, although the user needs to be aware of the nature of the statistical aggregation functions that are used. As an example, the mode works well with discrete data to display the most frequent value. With continuous data it can be very misleading, however, since the mode can appear anywhere where a few objects happen to have exactly the same value. The experienced user is assumed to be aware of such properties. Non-numeric discrete variables may not provide useful statistical information but can be included in the data to give the user the ability to select subsets of the data (such as the *origin* axes of the cars data set described in section 6.1).

After some testing with numerous data sets, the feature of having the x-axis

mapped to a statistical value had not been used very much. For that reason, the default setting was changed to just use the y-axis to plot one variable, and have the symbols separated from each other along the x-axis. The use of the x-axis to plot another statistical measure is still possible, but could possibly be hidden in an *experienced* mode.

The software used for the implementation is described in appendix A. The combination of the three environments Visual Basic.Net, OpenViz Particles and Matlab works sufficiently for this demo version of the VDMD. In a commercial application, however, a faster language like C or C++ would be preferable to be able to handle larger data sets without losing too much of the performance efficiency. One drawback with the OpenViz visualization components is that many of them do not allow any interaction in their basic execution. For example, all interactions associated with the parallel coordinate display needed to be added separately, which was time consuming and not in the focus of the actual diploma project.

7.2 Future work

Since the project has been time-limited, there are features that have been discussed as potentially advantageous, but have not been implemented due to the restriction of time and resources. The application should be seen as a software for demonstrating the ideas described above, and as inspiration for future possible extensions. Some interesting future features are discussed below.

In the current implementation, the sub-regions of the axes are predefined and fixed so that each region corresponds to a quarter of the whole range. One future extension would be to let the user interactively set the ranges, for a more flexible level-of-detail feature.

There are currently only two kinds of glyphs available for displaying statistics in the VDMD: filled squares and circles. These can be arranged in a two-dimensional coordinate system to display two values. For experienced users, more variations in the glyph appearance could be used to enhance the amount of information in the display. Additional features of the glyphs could, for example, include variations in size and orientation, or simply further types of graphical symbols.

There are three available levels of detail in the VDMD. It might be useful to extend the number of levels, to explore data more locally or even to explore single observations.

With more computer power, the interactions could be reflected in the VDMD in a more dynamic way than in the present version. One feature would be to have the symbols moved in real-time when handles are moved.

The VDMD could be extended to display multiple vectors for each dimension. The different vectors would then correspond to the various selections and interactions that have been made in the parallel coordinate plot. This would be useful as a journaling tool, especially if clicking the origins of the vectors resulted in the related selection being selected in the parallel coordinate plot.

The system has been developed as a research project, and has not yet been tested commercially, or with external professionals within the areas of the used data sets. The reason is that no efforts have been made to customize the interface for specific needs. The evaluation phase has proved that the method itself is helpful in exploring large data sets from various areas. However, the question still remains about how it should best be integrated with existing solutions. The parallel coordinates technique was chosen as basis because of the intuitive ways of brushing and selecting subsets of the data. An interesting future development would be to tailor a visual data mining system that includes the VDMD, but also fulfills the specific needs and requirements from professionals of the specified business area.

Appendix A

Development environment

This appendix describes the environment in which the application has been developed. It is more focused on technical and programming details than previous chapters, and the reader is expected to be familiar with some basic concepts and terminology of object-oriented programming.

The application is developed in Visual Basic.Net with the visualization library OpenViz [27] as a complement to handle interactions and graphical components of the parallel coordinates. The remaining part of the interface, as well as the VDMD, is developed with COM components in Visual Studio.Net. The concept of .Net will be described briefly below. For a full understanding of the .Net framework, the reader is referred to the documentation by Microsoft at [7]. The use of Matlab to handle calculation and OpenViz to handle visualizations can be read about in the sections A.2 and A.3. An overview of the data flow and interactions between these components is shown in figure A.1.

A.1 The .Net concept

The .NET Framework is an initiative from Microsoft that allows programming languages and libraries to work together to create Windows-based applications. The .Net environment has a large number of COM (Component Object Model) objects that can be used independently of the programming language. When a program written in the .Net environment is compiled, the runnable file is written in Microsoft Intermediate Language (MSIL), which is independent of the language the program was originally written in. This makes it possible to combine classes written in different languages that are supported and compatible by the .Net framework. For example, part of the program can be written in Visual Basic, while another part can be written in C#. Another advantage of the .Net framework is its ability to connect servers and clients from different platforms through the use of *web services*. The web services are small applications written in XML, which is a universal language for data exchange.

Figure A.1: Data flow. The solid lines illustrate the path through which all data are processed and normalized in the Matlab engine before being passed to the graphical components. The dashed lines illustrate how interactions in the graphical components affect each other. Interactions in the parallel coordinates cause the Matlab engine to re-calculate statistics for the entire data set.

A.1.1 Visual Studio.Net

Visual Studio.Net is a convenient development environment in which code for .Net languages can be written and managed. With the use of pre-defined Windows objects, such as buttons and drop-down menus, graphical user interfaces are easy to build.

For the development of the VDMD application, Visual Basic.Net was chosen as programming language, mainly because a great part of the documentation of Open-Viz components is written for this language. Although many examples are written in the earlier Visual Basic 6, they can easily be converted to Visual Basic.Net by slightly changing the syntax.

A.2 Matlab

Matlab is an advanced tool for doing numerical computations with matrices and vectors. Besides a large number of toolboxes designed for specific areas, it includes a lot of functions suitable for mathematical areas such as linear algebra and statistics. It also includes functions for integrating algorithms written in Matlab with external applications and languages, such as C, C++ or .Net languages like Visual Basic.

Once the VDMD application is run, a Matlab engine is started as a server to take care of all calculations that need to be performed. Matlab-written functions are called from the Visual Basic environment whenever a calculation or change in

the data is to be performed. The greatest advantage of using Matlab with large data sets is its ability to treat and manipulate data as matrices. Operations like splitting matrices, finding specific rows and columns and calculating statistics can be done using pre-defined Matlab commands rather than having to be performed internally in Visual Basic. The Matlab engine saves a lot of computing time, which is essential for providing an interactive interface of the application. Each time a change of the data in the parallel coordinates occurs, the Matlab engine re-calculates statistics of the whole data set. The advantage of this approach is that when a new axis is selected, statistics are already calculated, which makes a quick visual response possible. The drawback is that brushing of axes takes slightly longer time to perform. Matlab has a lot of pre-written functions for calculating statistics, that operate much faster than if we had to perform the same action using for-loops in the Visual Basic environment.

A.2.1 Data structures

The data set is initially read from a tabularized text file into the Matlab format as an $m \times n$ matrix, where m is the number of rows (objects) in the data and n is the number of columns (dimensions). All data are normalized and stored in a new matrix, which always contains the objects that are currently selected in the parallel coordinates. Removed objects are stored in a separate matrix, with the additional information of which axis it was removed from. When data are to be replaced to the parallel coordinates from the matrix of removed data, this information makes it easy to find the right rows to be replaced. The following is an example of how the matrices would look like after some interaction. When the data set has been locally normalized, each value is between 0 and 1. A data set with five objects and five dimensions would look like this:

$$data = \begin{bmatrix} 0.2 & 0.4 & 0.2 & 0.3 & 0.3 \\ 0.1 & 0.4 & 0.4 & 0.2 & 0.7 \\ 0.3 & 0.5 & 0.1 & 0.5 & 0.5 \\ 0.2 & 0.6 & 0.4 & 0.6 & 0.5 \\ 0.2 & 0.5 & 0.1 & 0.3 & 0.3 \end{bmatrix}$$

Brushing the fourth axis to remove objects with values ≥ 0.5 and the fifth axis to remove objects with values ≥ 0.7 will result in the following matrix:

$$data = \begin{bmatrix} 0.2 & 0.4 & 0.2 & 0.3 & 0.3 \\ 0.2 & 0.5 & 0.1 & 0.3 & 0.3 \end{bmatrix}$$

The removed objects are stored in a separate matrix where the first column indicates which axis the object were removed from:

$$removed = \begin{bmatrix} 4 & 0.3 & 0.5 & 0.1 & 0.5 & 0.5 \\ 4 & 0.2 & 0.6 & 0.4 & 0.6 & 0.5 \\ 5 & 0.1 & 0.4 & 0.4 & 0.2 & 0.7 \end{bmatrix}$$

If the user decides to replace objects that were removed from the fourth axis and have values ≤ 0.5 , the matrices would look as follows:

$$data = \begin{bmatrix} 0.2 & 0.4 & 0.2 & 0.3 & 0.3 \\ 0.2 & 0.5 & 0.1 & 0.3 & 0.3 \\ 0.3 & 0.5 & 0.1 & 0.5 & 0.5 \end{bmatrix}$$

$$removed = \begin{bmatrix} 4 & 0.2 & 0.6 & 0.4 & 0.6 & 0.5 \\ 5 & 0.1 & 0.4 & 0.4 & 0.2 & 0.7 \end{bmatrix}$$

If axes in the parallel coordinates are re-arranged, the order of the columns in the Matlab matrices need to be changed. For example, if the fourth axis is placed to the left of the first axis, the two matrices would look as follows:

$$data = \begin{bmatrix} 0.3 & 0.2 & 0.4 & 0.2 & 0.3 \\ 0.3 & 0.2 & 0.5 & 0.1 & 0.3 \\ 0.5 & 0.3 & 0.5 & 0.1 & 0.5 \end{bmatrix}$$

$$removed = \begin{bmatrix} 4 & 0.6 & 0.2 & 0.6 & 0.4 & 0.5 \\ 5 & 0.2 & 0.1 & 0.4 & 0.4 & 0.7 \end{bmatrix}$$

A.3 OpenViz

OpenViz Particles from AVS is an API¹ for creating powerful visualization on desktop computers. It consists of a number of graphical components with built-in interaction capabilities, as well as components for management of the data structure. OpenViz Particles uses an architecture that supports both the Microsoft Common Object Model (COM) and JavaBean component models, which makes them easy to integrate into commercial .Net or Java-solutions. Figure A.2 shows how the OpenViz components in the VDMD application are connected.

Starting from the top of the figure, it is shown that all data go through the Matlab engine before they are processed in the visualization. Once the data set is loaded or when changes have occurred, Matlab functions calculate new statistics for the whole data set. The data are imported to the Visual Basic environment and stored in temporary two-dimensional arrays before they are passed on to the OpenViz data structure. At the same time statistics of the data set are sent to the VDMD for a fast access when axes are selected for plotting of statistics.

The first step of the OpenViz pipeline is the *TableMapper*, where the data are mapped onto the internal table structure of OpenViz. This structure is used by various visualization chart components which takes data as rows and columns as input.

¹ API: Application Programming Interface

Three instances of the *ParallelCoordinatesChart* are used, which is a pre-defined and ready-to-use component. One chart is used for the data that are currently selected, and the other two visualize the removed data and the clusters in separate colours. Separating the data into different charts enables the user to choose which data should be visible or not.

Everything that is connected with the parallel coordinates and share the same coordinate system are collected in a *domain*. In addition to the parallel coordinates charts, the domain also contains labels, scrollbars and graphical bars indicating the selectable regions of the axes. The graphical bars are instances of the class *VzUnstructuredFieldbuilder*, which is used for drawing graphics of arbitrary shape in the visualization. The task of the domain is to translate the coordinate system used by its child components into the coordinate system defined by the enclosing *workbox*. The *workbox* can then be moved or resized without any knowledge of the local coordinate systems of the components it contains. Following the right-hand arrows of figure A.2, we see that the histograms are building its own path in the flowchart. Each histogram is built with the *BarChart* component as basis, and is placed in its own domain and *workbox* since they have separate (but similar) coordinate systems.

For an easy management of the layout of all components, there is a wide range of *Containers* available in OpenViz. The *workboxes* containing the histograms, are put together in a *GridContainer* that simply places its children in a grid structure. The parallel coordinates *workbox* and the *gridcontainer* are then placed as layers in a *CardContainer*.

The last part in the visualization pipeline is to connect the container to a *Viewer*. The viewer is treated as a COM object, and can be placed in the Windows form just as any other component of the interface. To the viewer, there are three types of *Interactors* connected, to handle interactions with the parallel coordinates. The *PointProbeInteractor* catches pick events in the viewer and is used to position the mouse pointer, in order to determine which axis has been selected. The *Drag-ProbeInteractor* determines the mouse positions when the mouse button is clicked, the mouse is moved and the button is released. It is therefore used to determine the new positions of the axes when they have been re-arranged by dragging the mouse. The last interactor is the *ManipulatorInteractor*, which is used to catch events from the handles of the scrollbars attached to each axes. The positions of the scrollbars determine how the data set is currently brushed.

In order to draw the graphics on the screen, a *renderer* must be connected to the viewer. To take advantage of the graphics hardware acceleration, a renderer that supports *OpenGL*² is used to render the scene.

The leftmost line in the figure illustrates that all interactions in the viewer are passed on to the Matlab engine to update the current data matrices. The *tablemappers* are then cleared and fed with the new data, to keep the visualization compo-

²Open Graphics Language (OpenGL) is a software standard interface and graphics library for developing portable presentation of 2D and 3D visual data.

nents updated.

The components inside the dashed squares are not included in the OpenViz pipeline, but they illustrate how the OpenViz components are connected to the other components of the application. The viewer is part of the form, that is developed in Visual Basic environment. The form also include the VDMD and the menu.

Figure A.2: Visualization pipeline of the data flow through the OpenViz components.

Bibliography

- [1] Daniel A. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, vol.8, no.1, 2002.
- [2] Gennado Andrienko and Natalia Andrienko. Constructing parallel coordinates plot for problem solving. In *Proc. 1st International Symposium on Smart Graphics*, pages 9–14, 2001.
- [3] Gennado Andrienko and Natalia Andrienko. Coordinated views for informed spatial decision making. In *Proc. Of Coordinated and Multiple Views in Exploratory Visualization*, pages 44–54, 2003.
- [4] Gennado Andrienko and Natalia Andrienko. Parallel coordinates for exploring properties of subsets. In *Proc. Of SmartGraphics*, 2004.
- [5] H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 1973.
- [6] Tamraparni Dasu and Theodore Johnson. *Exploratory Data Mining and Data Cleaning*. John Wiley and Sons, Inc., 2003.
- [7] Microsoft.net framework. <http://msdn.microsoft.com/netframework/>.
- [8] Niklas Elmqvist and Philippos Tsigas. Growing squares: Animated visualization of causal relations. In *ACM Symposium on Software Visualization*, 2003.
- [9] Usama Fayyad, Georges G. Grinstein, and Andreas Wierse. *Information Visualization in Data Mining and Knowledge Discovery*. Morgan Kaufmann Publishers, 2002.
- [10] Ying-Huey Fua, Matthew O Ward, and Elke A Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. Technical report, Worcester Polytechnic Institute, 1999.
- [11] G.W. Furnas. Generalized fisheye views. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology*, pages 16–23, 1986.

- [12] Gary Geisler. Making information more accessible: A survey of information visualization applications and techniques. <http://www.ils.unc.edu/geisg/info/infovis/paper.html>.
- [13] Rafael C. Gonzalez and Richard E. Woods. *Digital image processing*. Addison-Wesley Publishing Company Inc., 1992.
- [14] Martin Graham and Jessie Kennedy. Using curves to enhance parallel coordinate visualisations. In *Proceedings of IV 2003 - 7th International Conference on Information Visualization*. London, UK, pages 10–16, 2003.
- [15] J Han. Data mining. In *Encyclopedia of Distributed Computing*. Kluwer Academic Publishers, 1999.
- [16] Helwig Hauser, Florian Ledermann, and Helmut Doleisch. Angular brushing of extended parallel coordinates. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02)*, 2002.
- [17] Susan Havre, Elizabeth Hetzler, Paul Whitney, and Lucy Nowell. Themeriver: Visualizing thematic changes in large document collections. *IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS*, 32:9–20, 2002.
- [18] A Inselberg and B Dimsdale. Parallel coordinates: A tool for visualizing multidimensional geometry. In *Proc. Of IEEE Conference on vis'90*, pages 361–378, 1990.
- [19] Jimmy Johansson, Robert Treloar, and Mikael Jern. Integration of unsupervised clustering, interaction and parallel coordinates for the exploration of large multivariate data. In *Proceedings of the 8th IEEE International Conference on Information Visualisation*, pages 52–57, 2004.
- [20] Neal Lesh and Michael Mitzenmacher. Interactive data summarization: An example application. In *AVI'04, May 25-28, Gallipoli(LE), Italy*, 2004.
- [21] Jock D. Mackinlay, George G. Robertson, and Stuart K. Card. The perspective wall: detail and context smoothly integrated. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology*, pages 173–176, 1991.
- [22] A.R. Martin and M.O. Ward. High dimensional brushing for interactive exploration of multivariate data. In *Proceedings of IEEE Conference on Visualisation'95*, pages 271–278, 1995.
- [23] John J Miller and Edward J Wegman. Construction of line densities for parallel coordinate plots. In *Computing and Graphics in Statistics*, pages 107–124. A. Buja and P. Tukey, eds., 1991.

- [24] Douglas C. Montgomery. *Introduction to Statistical Quality Control*. John Wiley and Sons, Inc., 2005.
- [25] Andrew W. Moore. K-means and hierarchical clustering (02/28/2005). <http://www.cs.uta.fi/hs/iv99/>.
- [26] L. J. Stuart N. Barlow. Animator: A tool for the animation of parallel coordinates. In *Proceedings of the Eighth International Conference on Information Visualisation (IV04)*, 2004.
- [27] Openviz homepage (03/16/2005). <http://www.openviz.com>.
- [28] Wei Peng, Matthew O Ward, and Elke A Rundensteiner. Clutter reduction in multi-dimensional data visualization using dimension reordering. In *IEEE Symposium on Information Visualization, Austin, Texas, USA*, pages 89–96, 2004.
- [29] Kevin B. Pratt and Gleb Tschapek. Visualizing concept drift. In *SIGKDD’03*, 2003.
- [30] The reorderable matrix (01/26/2005). <http://www.cs.uta.fi/hs/iv99/>.
- [31] Manojit Sarkar and Marc H. Brown. Graphical fisheye views of graphs. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology*, pages 83–91, 1992.
- [32] Ben Schneiderman. The eyes have it: a task by data type taxonomy for information visualizations. In *IEEE Symposium on Visual Languages*, pages 336–343, 1996.
- [33] Harri Siirtola. Interaction with the reorderable matrix. In *Proceedings of the Information Visualization ’99, London*, 1999.
- [34] Harri Siirtola. Direct manipulation of parallel coordinates. In *IEEE International Conference on Information Visualization*, pages 373–378, 2000.
- [35] Harri Siirtola. Combining parallel coordinates with the reorderable matrix. In *Proceedings of The Coordinated and Multiple Views in Exploratory Visualization*, 2003.
- [36] Statlib datasets archive homepage. <http://lib.stat.cmu.edu/datasets/>.
- [37] Martin Theus. Interactive data visualization using mondrian. Technical report, Department of Computeroriented Statistics and Data Analysis, Augsburg, Germany.
- [38] Visimod homepage. <http://www.visimod.com>.
- [39] Colin Ware. *Information Visualization, Perception for Design*. Morgan Kaufmann Publishers, 2000.

- [40] *Proceedings of the International Workshop on Visual Data Mining: VDMP-KDD 2001*, 2001.
- [41] Xmdvtool homepage. <http://davis.wpi.edu/xmdv/>.
- [42] Kaidi Zhao, Bing Liu, Thomas M Tirpak, and Andreas Schaller. Detecting patterns of change using enhanced parallel coordinates visualization. In *Proceedings of the Third IEEE International Conference on Data Mining (ICDM'03)*, 2003.