FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

In-Place Interaction in Dashboards

BACHELOR'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science

in

Media Informatics and Visual Computing

by

David Pfahler
Registration Number 1126287

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Ao.Univ.Prof. Univ.-Doz. Dipl.-Ing. Dr.techn. Eduard Gröller
Assistance: Dipl.-Ing. Dr.techn. Harald Piringer
Dipl.-Ing. Thomas Mühlbacher

Vienna, 21st October, 2015

David Pfahler	Eduard Gröller

Erklärung zur Verfassung der Arbeit

David Pfahler
Palffygasse 30 2500 Baden

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 21. Oktober 2015

David Pfahler

Abstract

Interaction techniques for information visualization systems require control elements to interact with the data. These elements consume visual space and this is one of the major limiting factors of a visual analytics dashboard. This work introduces interaction methods to control the visualizations of a dashboard without the need of additional visual space. These in-place control elements give the dashboard designer the possibility to design more compact dashboards, while maintaining the same functionality without using space filling controls. This is achieved by hovering areas in the visualizations which opens controls or triggers events.

Contents

List of Figures

CHAPTER 1

Introduction

The speed data is generated increases every year. This results in a huge amount of raw unfiltered data, which is useless for most applications, because the user is not able to analyse the data fast enough. This *information overload* is typically caused by parts of the data being irrelevant, or inappropriately processed or presented for a particular task at hand. [KAF$^+$08]

It takes a lot of time to analyse these masses of data and to query them efficiently. Information visualization is one approach of dealing with the information overload problem. Thereby the data is presented in a visual form, which can be interpreted by a human fast and efficient. By interacting with the visualization, the user is able to gain new insight into the data. [Kei02]

Interactive information visualization systems provide a large set of different types of visualizations, like scatter plots, histograms or parallel coordinates. It can be difficult for novice users to choose the diagrams which are most efficient and suitable for their task, because the system provides a too large range of functions. [MAP15]

To support the users, their tasks are analysed and a task-tailored dashboard is designed [MP14]. With this designed interface the information is more structured and in an easy to read manner [EB11].

An implication of this simplicity is that dashboards typically provide less space for widgets to interact with the data. Additionally to the limitation of the visual space too many control elements in an application create extensive visual clutter, poor information readability and information overload, which distracts the screen user. [Gal07, p.110f]

But the interaction is essential to complete the user tasks to reach their goals. This work introduces interaction methods to control the parameters of visualizations without the need of additional visual space. These in-place interaction techniques give the dashboard designer the possibility to design more compact, orderly, clean, clutter-free dashboards with the same functionality, while avoiding permanent, space-consuming controls.

The main idea is to define dedicated event-triggering areas within the visualizations. Hovering these areas with the input device (typically the mouse) triggers interactions or displays controls on-demand. A well known example for a hover triggered control is the ToolTip, which is a small pop-up window that appears when the mouse pointer is moved over an area. It contains additional information of the area. It is often used to identify controls which do not have a caption because it has been omitted for lack of space.

The disadvantages of the ToolTip are very similar to the key-challenges of the later presented in-place interaction methods. The user does not know where the area is, so he has to discover them and if they appear unwanted they can be distracting for the user. [Gal07, p.111f]

Chapter 2 introduces the used definitions: Direct Interaction (2.1), the visual analytics and information visualization system Visplore (2.2) and Dashboards (2.3).

In Chapter 3 the tasks and requirements of control elements in Dashboards are preseneted:

- Adjusting mappings of numerical scales 3.1
- Reordering columns in tabular layouts 3.2
- Adjusting visual parameters 3.3

As the core contribution of this work, Chapter 4 introduces control elements for in-place interaction in dashboards which are implemented in Visplore.

Chapter 5 contains the discussion of the implemented methods and the conclusion.

CHAPTER 2

Related Work

The visual analytics process is an approach with multiple different definitions. This chapter defines the terms that are used in this work.

Section 2.1 defines the term of direct interaction and the used methods. Section 2.2 presents the application Visplore, which is used as the application to implement the direct interaction methods and in Section 2.3 the method of dashboards is described by the state of the art example of Visplore.

2.1 Direct Interaction

[Dix09, p.4] defines direct interaction as the involvement of "a dialog with feedback and control throughout performance of the task". Indirect interaction in contrast contains batch-processing, where the user enters the data to the system and waits for the results for hours or days.

Direct interaction uses visual representations of the action the user wants to achieve and he interacts with it by pointing on it. Furthermore this actions have to be rapid, incremental and reversible and the results of the generated query need to be displayed in real-time. [AS94]

Typically the user can interact with the data through a control panel, which contains visual widgets. These widgets could be sliders, buttons or text fields. By interacting with them the visual representation of the data changes in the views. [EBN13]

In the case of a control panel these widgets are constantly visible in the interface. In contrast to this approach it is also possible to display them only on demand. Well known examples for on demand interaction methods are the Popup Window and the Popup Menu. This work focuses on this type of direct interaction.

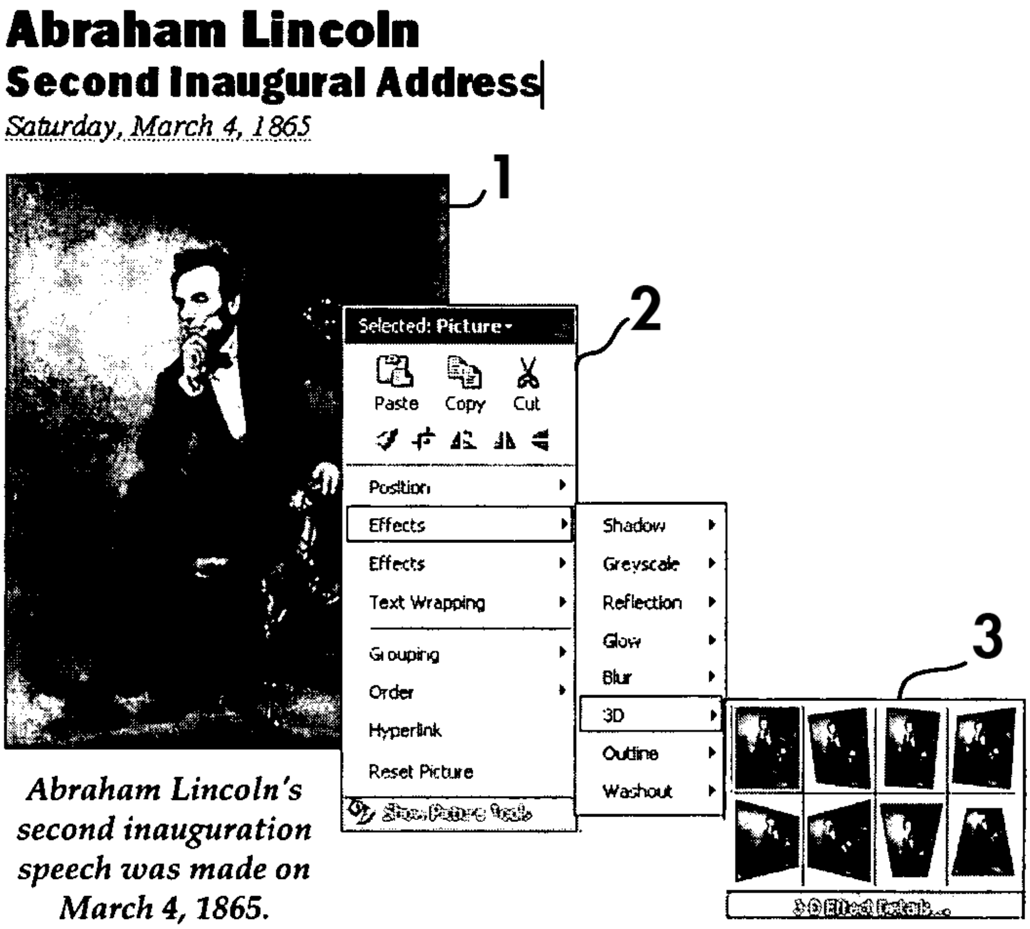


Figure 2.1: 1) A figure inside the text editing program "Microsoft Word" 2) A context specific Pop-Up Menu 3) Control elements for direct interaction [RBH+12]

2.1.1 Popup Windows and Popup Menus

A Popup Window is used similar to a ToolTip which was presented in the previous Chapter. A new window with additional information is only displayed when the user triggers a specific event like hovering an icon. This window does typically not contain the standard window components, such as the title bar or the close button. Additional to textual information, this window can contain controls for direct interaction. [Gal07, p.364ff]

A popup Menu is a special implementation of a popup Window and used to "present alternatives or choices within the context of the task" [Gal07, p.318f]. The presented options in the Popup Menu are often also available from other control elements, which are usually hidden in a complex menu structure. Therefore, the Popup Menu enables the user to achieve his task faster, because the needed commands appear in the working area.

Another advantage of Popup Menus and Popup Windows is that they do not need additional window space, which is important for the use of Dashboards (see Section 2.3). The disadvantages of these control elements are: [Gal07, p.318f]

- They have a shallow learning curve, because the user has to remember the existence and the location of these elements, as well as how to trigger them (e.g, by a right-click)

- They may be displayed by accident and/or occlude the screen working area.

Figure 2.1 shows an implementation of a context specific Popup Menu in the word processor "Microsoft Word". By clicking on an element, or hovering it, a context specific control panel appears, which features more than just a menu. It includes other widgets, such as buttons, icons or sliders, that can be used to interact with the clicked or hovered element. [RBH+12]

2.1.2 Science of Interaction

"We need to develop a 'science of interaction' rooted in a deep understanding of the different forms of interaction and their respective benefits." [TC05]

Thomas and Cook made this recommendation in the year 2005 and since then several improvements were made.

[YaKSJ07] created general categories of interactions based on the notion of user intent:

Select Users want to select data points that are interesting to them to highlight them as important. For example by colouring data points in a visualization which fulfill the selection criteria with a different colour. A selection is often used before other interaction techniques are executed.

Explore It is not possible to visualize all data points of a large data set at the same time, because of screen- and fundamental perceptual- and cognitive- limitations. Therefore the user needs an interaction technique to change the visualized subset of data. For example by panning the viewed subset by dragging the visualization with the mouse. (see Section 3.1 for an implementation of this concept)

Reconfigure The spatial position of the data items is the most accurate of all visual variables [Mac87]. But a static visualization can present only one perspective onto the data set. By changing the spatial arrangement the user can analyse the data further. For example, by rearranging columns of a table. (see Section 3.2 for an implementation of this concept)

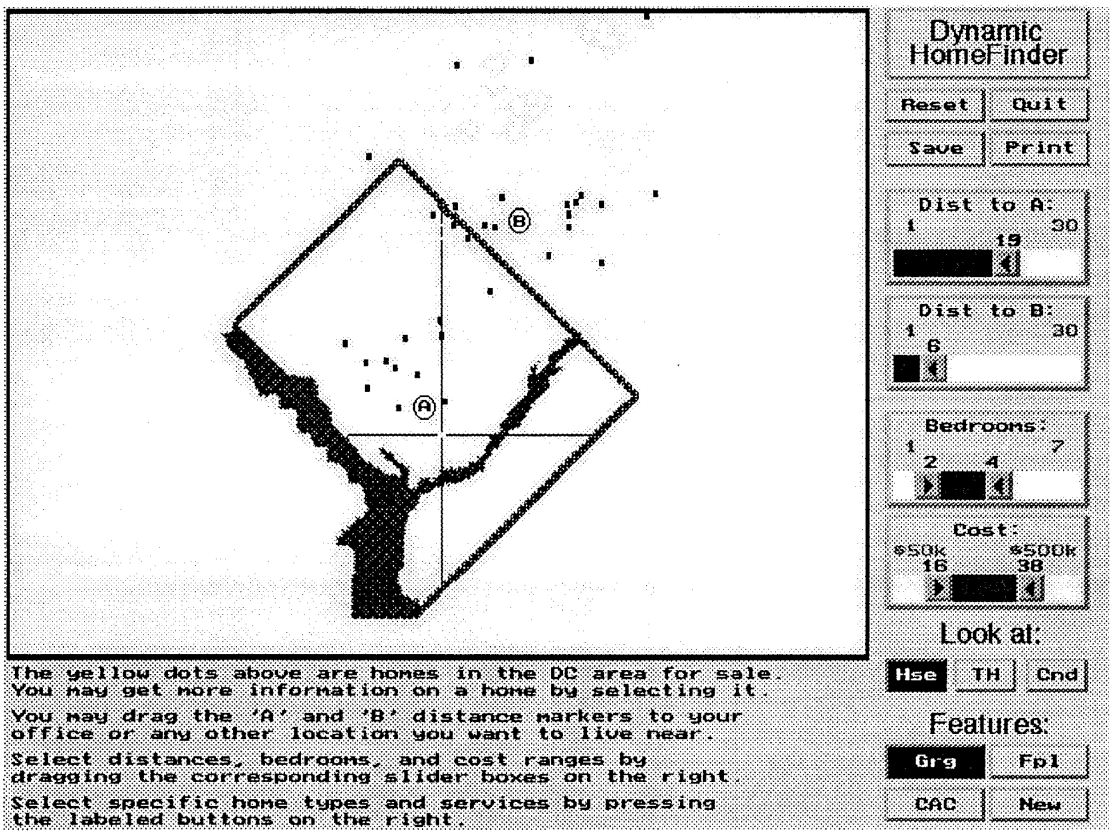


Figure 2.2: The Dynamic Home Finder is a query system to find a home which suites the needs. It uses direct interaction sliders to query the filter. [AS94]

Encode By showing a different representation to the user, he is able to understand new relationships and distributions of the data items. Changing a pie chart to a histogram is a simple example of encoding the data with two different types of presentation. (see Section 3.3 for an implementation of this concept)

Abstract/Elaborate This interaction technique allows the user to adjust the level of abstraction of the data to show him more or less detail. The ToolTip, a very simple and previously mentioned interaction technique, belongs to this category.

Filter To restrict the displayed data items the user defines a condition or a range. Only the data items that meet this condition are presented. An example of this would be direct interaction sliders. In Figure 2.2 the application Dynamic Home Finder uses this direct interaction sliders to query a filter. [AS94]

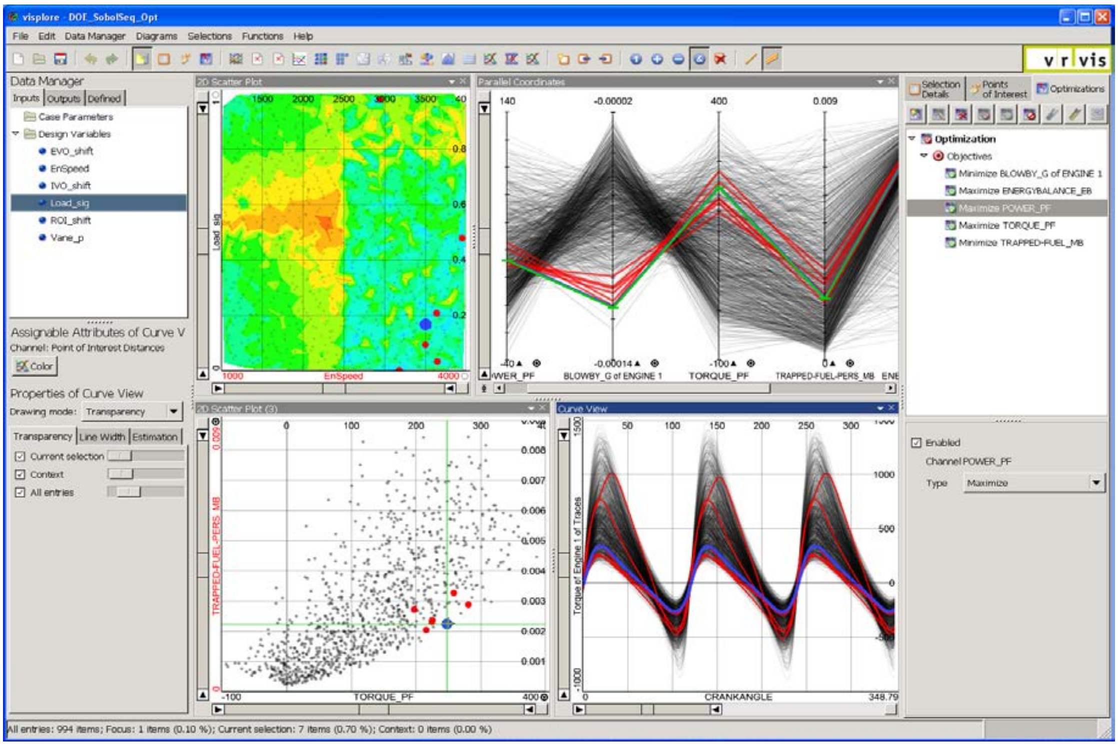


Figure 2.3: Screenshot of the visual analytics and information visualization system Visplore. It shows a car engine design from AVL List GmbH.

Connect This interaction technique enables the user to get insight into the associations and relations of data items. This includes hidden data, which is relevant for the specified data. Brushing is an often used interaction technique to visualize the selection in all displayed views. In Figure 2.3 the selected data entries are displayed as red lines and dots and the currently hovered data entries are rendered with a blue colour in all views.

Furthermore the goal of the science of interaction is the guidance of the analysis and design of visual displays and to define and recognize best practices of interactions. [PSCO09]

2.2 Visplore

As already mentioned Visplore is a visual analytics and information visualization system which features:

- A copious amount of diagrams, which are called views, for example: scatter plots, histograms, parallel coordinates, etc...
- direct interaction with more than one million entries and with thousands of functions [PTMB09, PPBT12]
- Brushing and linking between the different views [PBH08]

- the creation of computational analysis models, like regression models ([PBK10, MP13])
- The design of dashboards by visual programming of existing modules, such as views, selections, or derived data.

Like mentioned in Section 2.1 it is important that the response time of the exploration task is under 100 milliseconds to ensure that the change is perceived as a smooth animation. If the response to an user interaction needs longer than 1 second the user performance might reduce. [TC05, S. 73-81, Chapter 3] To ensure that the user receives real-time feedback for his actions the Visplore system uses a multi thread architecture and an approach called early thread termination, where it is possible to interrupt the rendering within the standard visualization pipeline [Chi00] because changes, usually done by the user, invalidated the final image. This makes it possible to render only partial results of the visualization, which may not be as computationally expensive to compute as the final image or the partial results can be cached. [PTMB09]

The system supports various different visualizations [PPBT12]. This work introduces in-place interaction techniques to some of this views.

2D scatter plot Two data attributes can be assigned to the X- and the Y-axis and their relation is visualized as a collection of points, where the position of a point is determined by the data value of the horizontal and the vertical axis. In Visplore an additional data attribute can be assigned to the visualization, which is shown by color. Additionally it is possible to use the scatter plot as a time series view by connecting the data points with a line-strip provided by an data attribute. Figure 2.3 presents an example of a scatter plot in the bottom left- and top left- visualization.

Parallel coordinates are used to relate multiple data dimensions. In Visplore a user is able to add remove and move data dimensions to the axes and add an additional data attribute to the color of the lines. [Ins09]

1D Overview This view provides a statistical overview of a set of user-defined data dimensions. Additionally a small histogram for every data dimension is displayed. This view is capable of interactively building regression models. [PBH08]

2D Overview This view is used to analyze pair-wise relationships of data attributes. The data attributes are displayed as a matrix and are ranked by statistical properties. This ranking allows focusing on the statistical relevant attribute pairs. To get an overview it is possible to use this view as a 2D scatter plot matrix.

The Visplore framework is used in different technical areas. It is used for optimizing and designing complex engines to improve efficiency, in the area of facility management to optimize maintenance cycles and to analyse the gas consumption of cities.

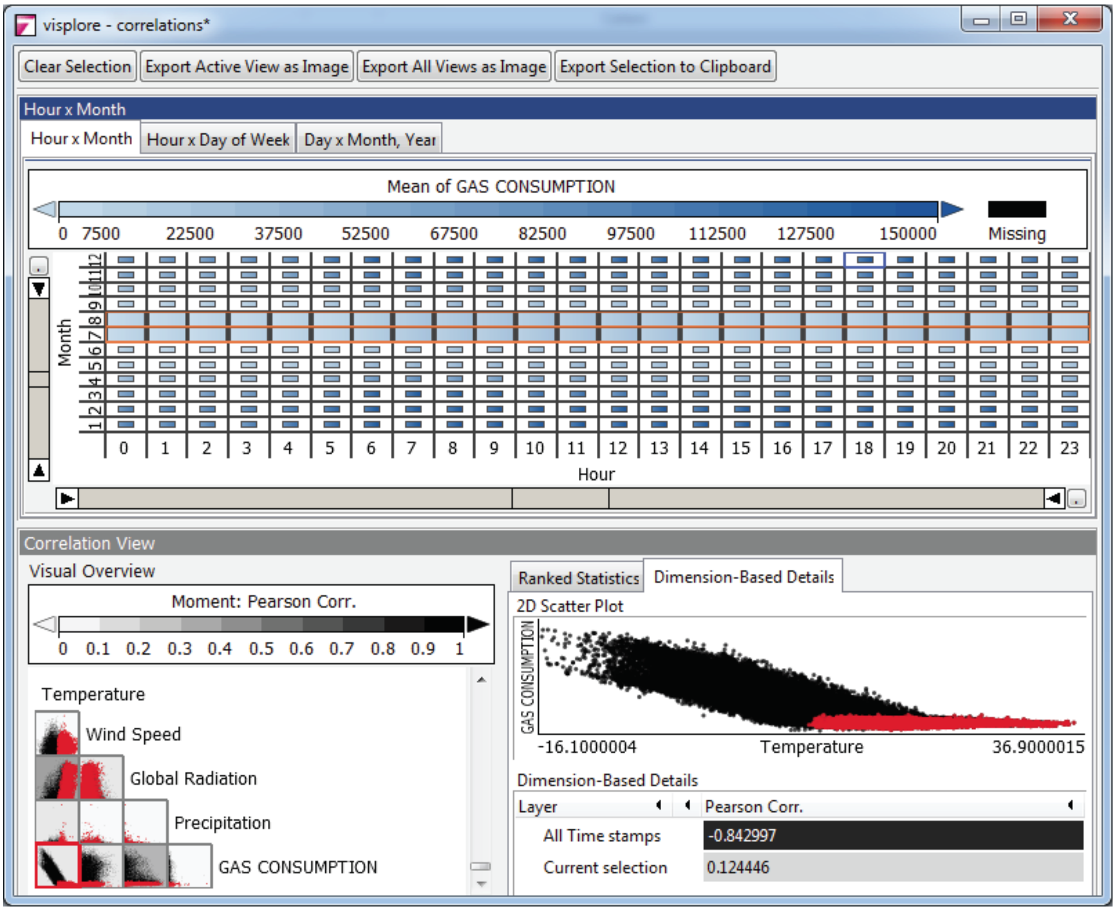


Figure 2.4: Screenshot of a dashboard build with Visplore for the analysis of the correlation of natural gas consumption and meteorological factors. [MP14]

To enable interaction in the exported dashboards this work defines the tasks and the goals of the user and introduces direct in place interaction solutions.

2.3 Dashboards

The range of functions and the area of applications of Visplore enable expert users to solve a variety of tasks in analysis and statistical modelling. However inexperienced and novice users are often overwhelmed by the broad range of functions of the system. [MP14] [EB11] Task-tailored dashboards are one approach of helping a new user to solve a given task.

A dashboard in an auto mobile is a compact and intuitive combination of different visualizations of various data sources. In visual analytics this term is also used for the designed interface, which connects different views, widgets and data sources to organize them. [EB11]

Typically visual analytics experts design a dashboard for a specific task in a way that the user of the dashboard is able to visualize the data in an easy understandable way. Like in the dashboard of a car, the designer knows what kind of data is expected and chooses the best way to visualize it. Figure 2.4 shows a dashboard of the Visplore system which enables the user to easily analyse the correlation of natural gas consumption and meteorological factors. The three views are on fixed positions, but the user is able to load a different data set, select data and export the selection as a new data set.

One of the key challenges of designing a dashboard is choosing the ideal number of views and their arrangement. By adding more views the user is able to solve a broader set of tasks with the same dashboard with the drawback that it increases the complexity. In addition to that, the available screen space per view gets smaller if more views are displayed. [Sof15] Another challenge is choosing which controls should be displayed in the dashboard.

In a business intelligence dashboard the designer is also able to use interactive exploration to enhance the understanding of the information. However, as a specific drawback in the context of this thesis, interaction requires widgets to interact with the data which use visual space. This is one of the major limiting factors of a dashboard. [cKFY11]

Applications that support the design of dashboards are for example Tableau [MHS07] and Visplore (Section 2.2). In this work Visplore was used to implement direct interaction methods for dashboards to save visual space.

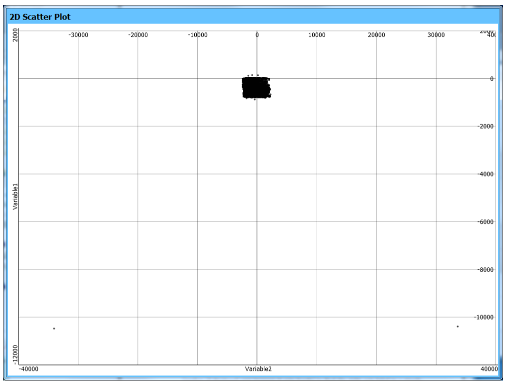
CHAPTER 3

Tasks and Requirements

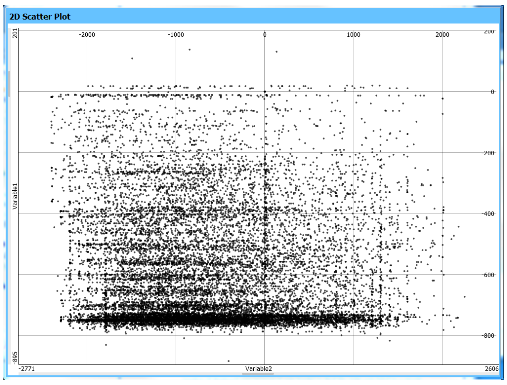
In Section 2.1, diverse categories of interactions were presented. This work focuses on three of these categories, which support the user to solve visual analytics tasks. This chapter presents a more detailed overview of these three tasks and the requirements of interactions within a dashboard.

3.1 Restricting the shown range

This type of interaction enables the user to adjust the scaling of a continuous axis in order to display a certain interval. Common use cases include to "explore" [YaKSJ07] the data step by step or to avoid distorting the scaling by outliers.

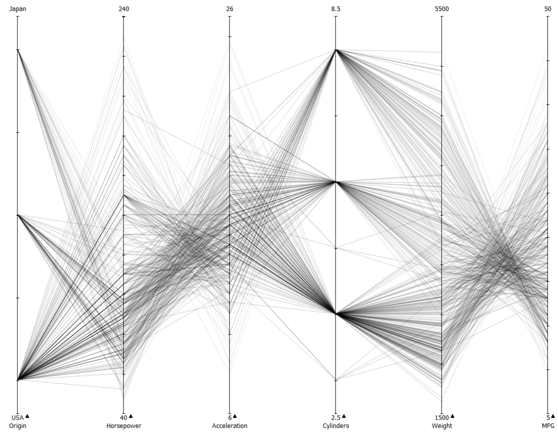


(a)

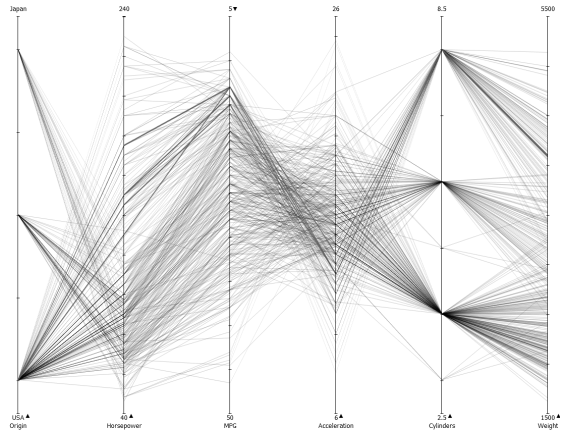


(b)

Figure 3.1: (3.1a) shows a scatter plot with two outliers, which axes are arranged that all data points are visible in the visualization. (3.1b) A more detailed view of the scatter plot by changing the data mapping of the x-and y-axis.



(a)



(b)

Figure 3.2: Two different representations of the same axes of parallel coordinates visualization where different correlations can be observed.

Figure 3.1a shows a scatter plot which displays all data points of a certain dataset. Due to outliers the data points on the left are overlapping and it is not possible to detect any patterns. After a restriction of the shown range, by adjusting the scaling of the x- and y-axis, as shown in 3.1b, the user is able to discover a previously hidden relationship between the two variables.

3.2 Reordering columns in tabular layouts

Tabular layouts are used in various domains to store data in columns and rows to be able to easily compare and look up values. This layout is also used in various visualizations and visualization systems. An example for a tabular layout in visualizations is the 2D overview, where multiple scatter plots are displayed in a grid to compare them. Other examples are the table lens view or parallel coordinates, where only one row or column with multiple entries exists. (see Section 2.2)

Furthermore the design of a whole dashboard can be structured as a table, so that the used views are arranged in rows and columns.

The user needs interaction techniques to change the spatial arrangement of the representations, to enable the user to look at the data at a different perspective. Because it is possible that a single representation is not sufficient enough for gaining the insight into certain characteristics and relationships of the data. [YaKSJ07]

As an example, Figure 3.2 shows two different representations of the same data as parallel coordinates. On the left image it is hard to see the correlation between *Horsepower* and *Miles Per Gallon (MPG)* and after reordering the axes it is easy to see this correlation.

3.3 Adjusting visual parameters

The last interaction technique that was implemented as in-place interaction method by this work is categorized as "encoding" by [YaKSJ07]. Data item attributes are mapped to visual attributes such as position, colour, shape, etc... [Mac87]. The mapped data items are rendered into the screen space of the visualization. This space is the limiting factor of the visualization. For example if the rendered shapes are displayed too big, they overlap and if they are rendered too small, they can not be interpreted correctly by the user. Changing the visual mark attribute enables the user to experiment with different representations.

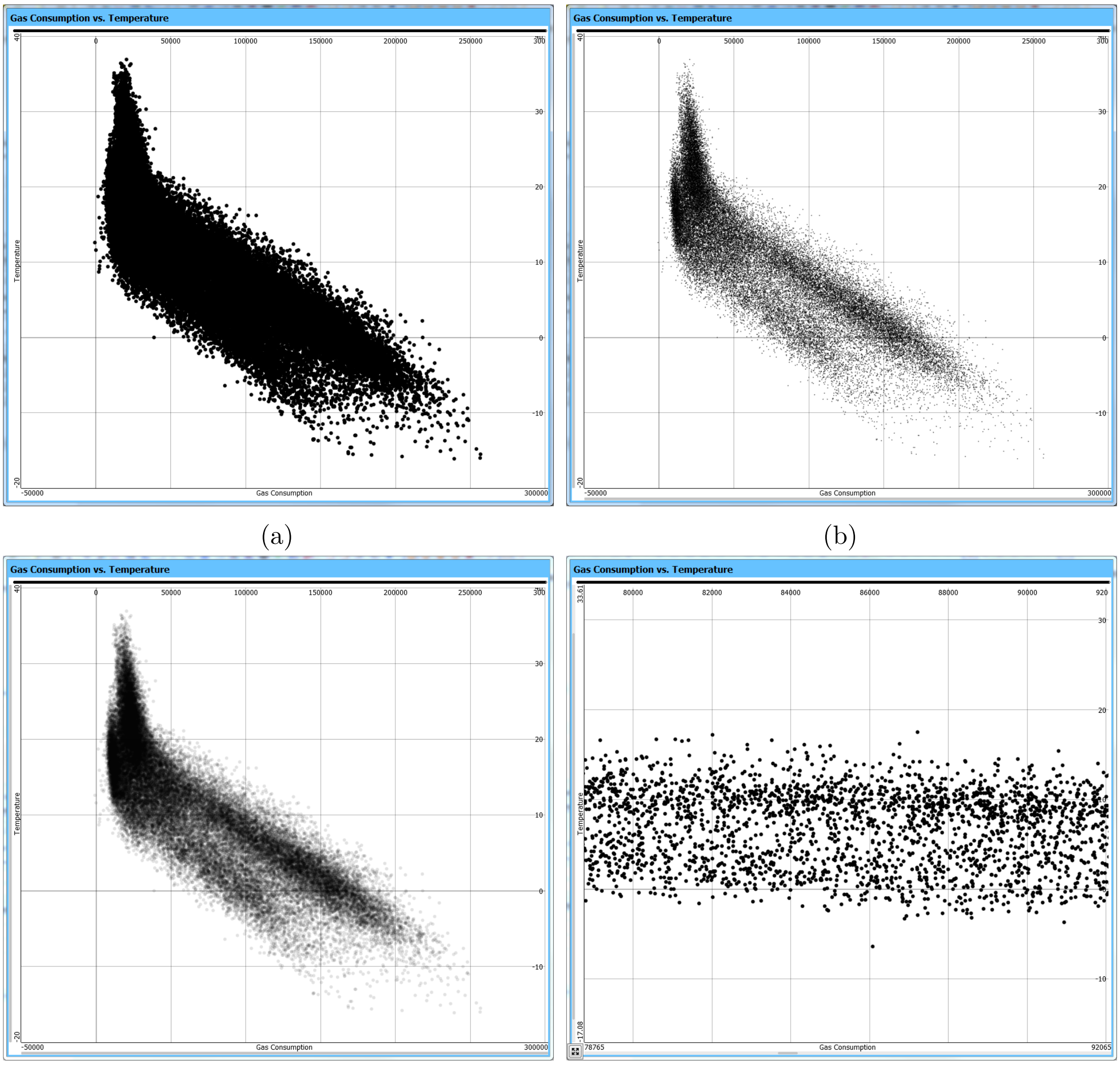
The upper left Figure 3.3a shows a scatter plot with many data points. On the x-axis the data attribute *Gas Consumption* is assigned and on the y-axis *Temperature*. Only the position is encoded, but the true distribution can not be perceived due to the overlapping of the data points, because the size of one point is too big. The user is not able to interpret this visualization and needs to change some visual mark attributes to explore the data further.

In Figure 3.3c the opacity of the points is changed which enhances the perception of clusters in the data. In Figure 3.3b the same effect was reached by reducing the size of the points to 1 pixel. Two clusters can be observed by a user and further analysis shows that these two clusters belong to the data of the day and the night.

In Figure 3.3d the shown range was restricted as described in Section 3.1. Zooming into the visualization enhances the according screen space per data point ratio and the rendered points no longer overlap. In contrast to the previous mentioned techniques, the global overview of the data is lost and it is not possible to gain the same insight into the data.

3.4 Export and Present

After the user was able to gain insight into to data and received new knowledge, he wants to present the gained information to other people. The dashboard should be able to



(a) (b)

(c) (d)

Figure 3.3: Different adjustments of the visual parameters of a visual cluttered scatter plot.(3.3a) initial visualization, (3.3b) reduced point size, (3.3c) reduced transparency and (3.3d) restricting the shown range

support this task or even screen capturing should give good results. To support this the dashboard should not include any usability controls that encode redundant information (e.g. Input fields with values that are also present in the visualization) or no information at all (e.g. buttons). [Few09] Of course that includes the buttons for exporting the visualizations.

Figure 2.4 shows a screenshot of a dashboard build with Visplore. The upper visualization features two control elements to restrict the shown range. These controls encode a redundant information and are not needed to represent the dashboard for presentation purposes.

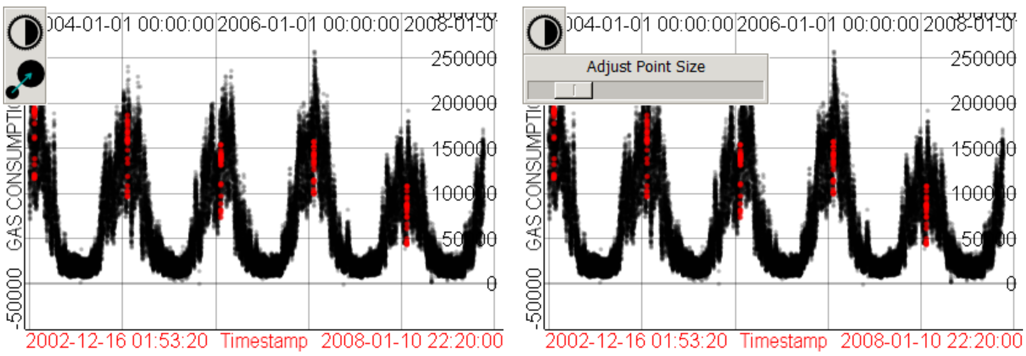
CHAPTER 4

Control elements for In-Place Interaction

This chapter focuses on the implementation of in-place interaction controls in views of the Visplore system to support explorative user tasks in dashboards. Visplore is a system written in C++ uses the GTK+ widget toolkit for creating graphical user interfaces and OpenGL for rendering. In this work an additional class of controls were developed that are integrated in the existing system.

The new class of controls provides the functionality of accessing and hiding the existing control element on demand (Section 4.1). Additionally it performs the positioning of the controls, which is described in Section 4.2.

Multiple in-place interaction controls were implemented and tested in different views. Section 4.3 describes how these controls support the user tasks in dashboards.



(a) (b)

Figure 4.1: "Two-Stage Popup Controls" used to reduce the point size of a time series visualization

4.1 Triggering Controls on Demand

In the previous Section the need for in-place interaction is described. The whole interface of Visplore is composed of so-called “widgets”, which are on one hand the existing control elements and on the other hand the `GL`-areas. To access the controls as a trigger, a so-called “trigger widget” needs to get defined. If necessary it is also possible to restrict the area of interaction of this widget to a certain “trigger area”, which is defined by an upper left and lower right corner point. The `GL`-area of the scatter plot for example, where the visualization is rendered, can be configured as a trigger widget. A rectangle around the name of a rendered axis can be defined as its restricted trigger area.

In Section 2.1.1 the problem of accidental appearing of popup controls was presented. This might happen due to not intended mouse movements by the user or just passing by with the mouse pointer.

To avoid this unintended behaviour, a timeout is provided by the control. The mouse pointer of the users has to stay uninterrupted inside the trigger area for the whole timeout to open the control.

After the user adjusted the visual parameters, changed the arrangement, restricted the shown range, used the displayed popup control in another way or opened it accidentally, the popup should disappear. This is handled similar than accessing it.

If the user exits the popup with the mouse pointer, the popup is not hidden instantly, but a hide timeout is started. If the user does not re-enter the control within this time interval, the popup hides. This is needed especially for small popup controls, to prevent the issue that the control hides itself unwanted, because of accidental small mouse movements by the user.

Besides from mouse movements, the displayed controls should also hide themselves when the focus of its trigger widget changes. For example when the user switches between top-level windows without using the mouse while a popup is open, it should hide when its underlying widget is not visible anymore. Another example would be, that the controls from the last active view should hide, when the user clicks onto another view.

The used timeouts can be configured to match the requirements of different controls. Obtrusive controls that possibly occlude other interesting parts, typically benefit from longer timeouts. A short or even no timeout at all can make sense for subtle visual cues indicating a possible action over the currently hovered area, e.g., by highlighting.

Another way to prevent the unwanted appearance of the controls is to introduce a “two-stage” popup interaction: If the user hovers the trigger area a small representation of the control in form of an icon appears, instead of the actual control. This icon is much smaller than the control itself and thereby doesn’t disturb the user as much as the original popup, when displayed by accident or as a visual cue.

In Figure 4.1a this approach is shown. The user hovers the scatter plot and the two icons for adjusting its visual parameters are displayed. Hovering one of the icons then displays the actual control, i.e., a slider for changing the transparency or the point size (see Figure 4.1b).

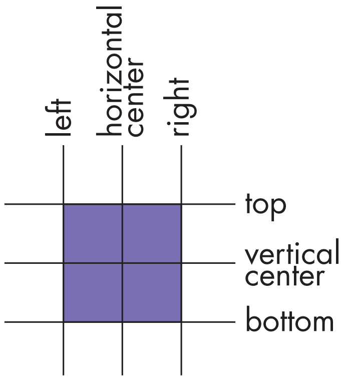


Figure 4.2: Anchor lines to specify the position of the popup controls

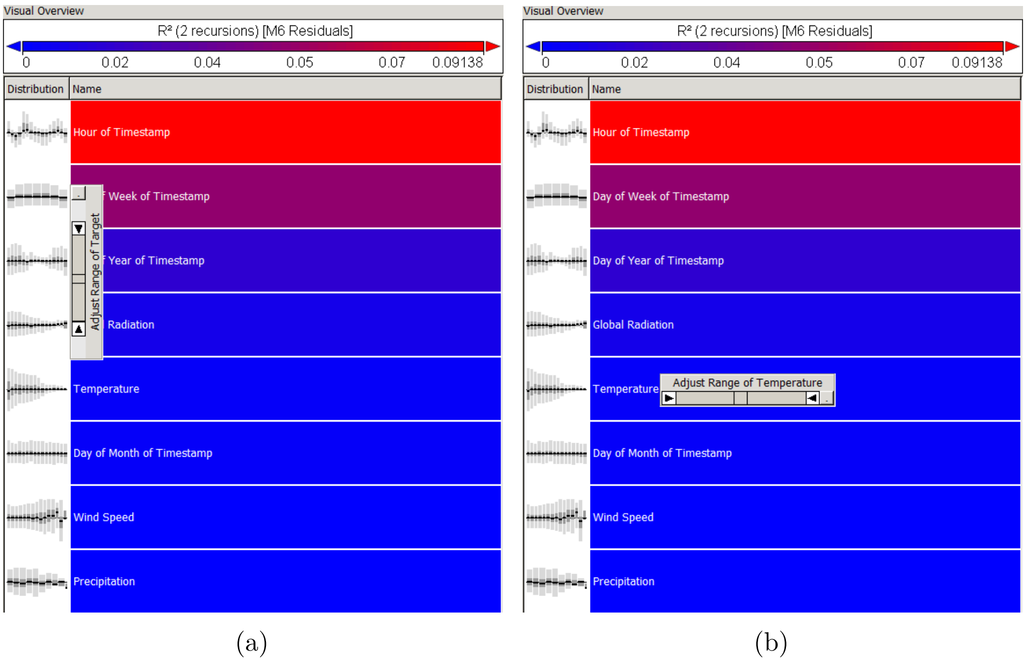


Figure 4.3: Two examples of combined positioning of the popup in a 1D Overview [PBH08]. (4.3a) right of the trigger widget and relative to the mouse position and (4.3b) right of the trigger area

4.2 Positioning of Controls

A second design question refers to the positioning of the displayed popup control. When the initial on-demand action is done, like hovering a trigger area and the control is about to be displayed the position where it should be displayed has to be defined.

Therefore a popup control is able to specify a set of “anchor lines”. Figure 4.2 shows the different lines: left, horizontal center, right, top, vertical center and bottom. This anchor lines can be set in relation to either the current mouse position or the trigger area. A relation to the trigger area additionally has to contain a set of anchor lines to specify where the anchor of the control should be displayed. For example bounding a popup control to the right edge of a trigger area and center it vertically, it is necessary to specify the relation from the popup anchor lines *left* and *vertical center* to the trigger area achnor lines *right* and *vertical center*. (see Figure 4.3b)

Additionally it is possible to specify an offset in the x and y direction. This is necessary to move the displayed control in a way that it does not hide informative elements such as legends or labels.

Figure 4.3 shows two examples of combined positioning of in-place controls. The diagram shows small-multiple plots visualizing the relationship of one target variable with different independent variables [MP13]. In-place controls are used to adjust the scaling of the common target variable on the y-axis, as well as for the x-axis scaling of the individual independent variables of each row.

In Figure 4.3a the trigger area of the common vertical scaling control is configured as the whole left column of the table, i.e., the union of all small-multiple-plots. The positioning is defined as relative to the right edge of the trigger area for the x coordinate and relative to the mouse position for the y coordinate. For the horizontal axis slider in Figure 4.3b the trigger area is only the label in the cell of the second column stating the name of the respective independent variable. In this case, the relative positioning is chosen vertically centered at the right border of the trigger area.

4.3 Controls to support exploratory tasks

This section introduces controls and interaction techniques to support the exploratory tasks of the user within dashboards of the Visplore system.

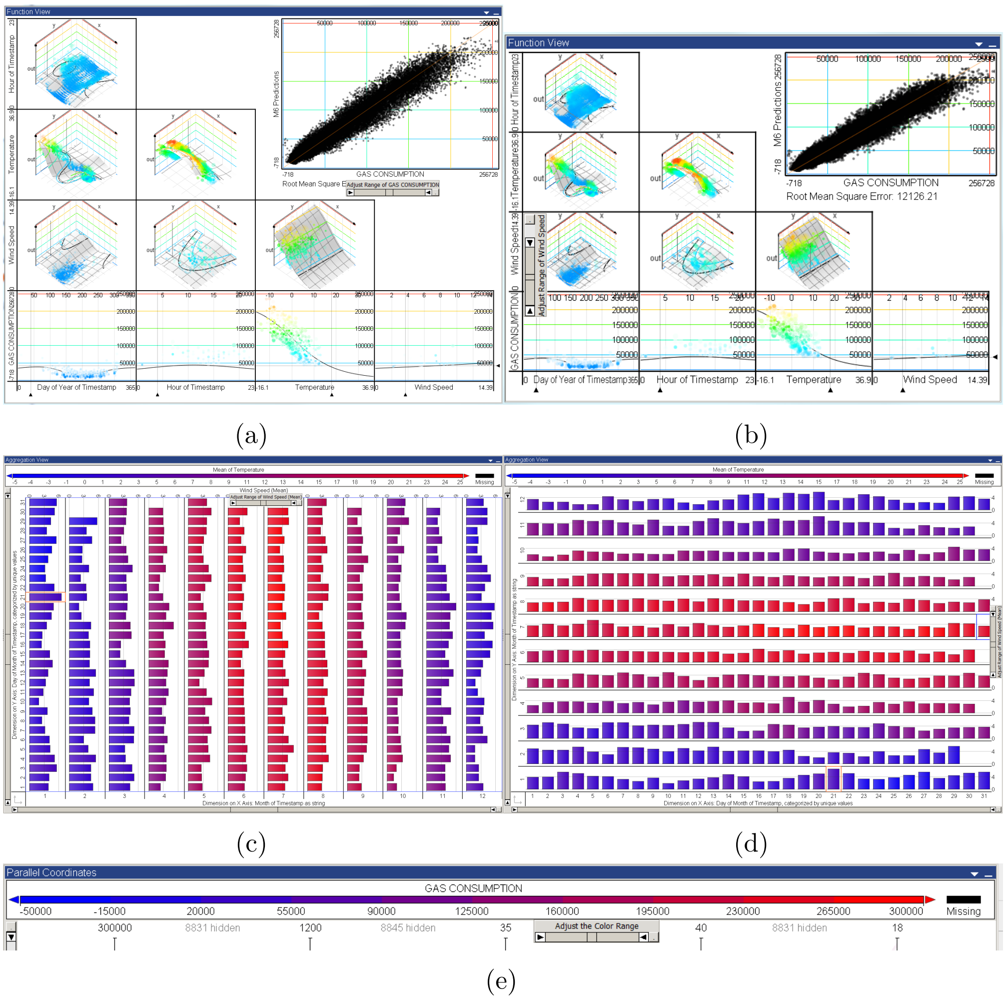
4.3.1 Restricting the shown range

In Section 3.1 the motivation for the user task “restricting the shown range” was presented. For this user interaction, a widget enables the control of the visual mapping.

In Visplore this control element is called the “data mapping widget”. This widget enables the user to set the visible data range as well as the location, the extend and the degree of distortion of the data range [ARP14]. It is designed as a range-slider, which is a well-known concept for dynamic data filtering [AS94].

For visualizations with only a small number of displayed data attributes, which visual mappings need to be changed, it is possible to permanently display the data mapping widgets of this attributes in the visualization (e.g. a scatter plot). But when the number of attributes increases, the visual space is not sufficient to display all controls permanently (e.g.: in parallel coordinates).

One common approach is to access this widget via a control panel, which contains visual widgets depending on the currently selected view and attribute. The disadvantage



(a) (b) (c) (d) (e)

Figure 4.4: Positioning of the trigger area of the popup controls is by convention on labels. This figures show examples of popup controls in different visualizations.

of using a control panel is that it is not available in a dashboard and additionally it takes time finding the right control for the current task.

Instead of permanently displaying the widget, popup controls are used to access the data mapping widget.

As convention, the label of the data attribute whose displayed range is about to get changed, is used as the trigger area. Examples are presented in Figure 4.4. The two Figures on top (4.4a,4.4b) are screenshots showing the *function view*, which is used to validate multivariate regression models with respect to their input variables [PBK10]. Data mapping widgets for adjusting the displayed range of each model input variable are displayed vertically for vertical labels and horizontally for horizontal labels.

Figures 4.4c,4.4d,4.4e show the position of the data mapping widget relative to the labels of aggregated measures in the "aggregation view", a diagram for pivotizing and

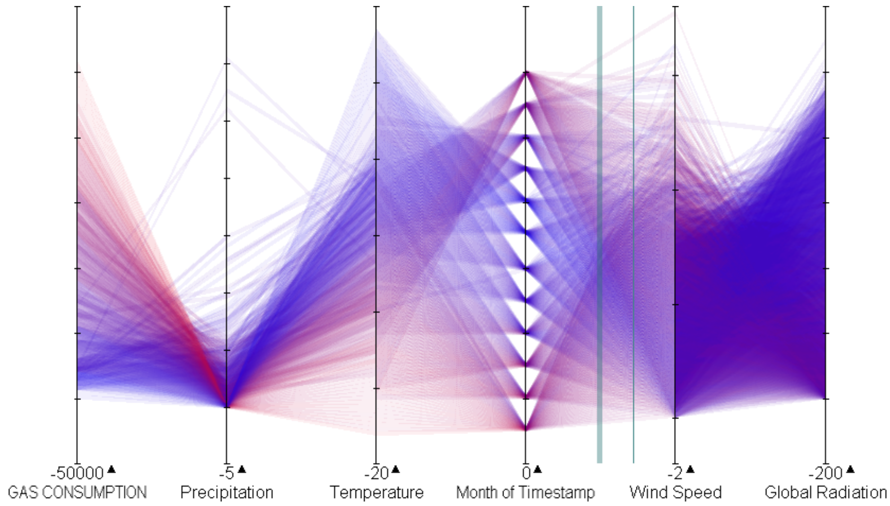


Figure 4.5: Dragging an axis from the parallel coordinates creates a thin cyan line as visual feedback and the changed representation if the axes is dropped at the current position is indicated by a thick cyan line between two other axes.

aggregating data tables. In these examples, data mapping widgets can be used in-place to adjust the displayed range of measures mapped to the height or width, as well as the color of bars. Further views featuring in-place data mapping widgets for adjusting ranges of numerical data attributes include:

- 1D Overview (Figure 4.3)[PBH08]
- Scatter Plot Matrix or 2D Overview [PBH08]
- Parallel Coordinates
- Radial View: a 2D- Scatter Plot, but one axis is mapped to the angle and the other to the distant from a center point.

4.3.2 Reordering columns in tabular layouts

In Section 3.2 the motivation for the user task “reordering columns in tabular layouts” was presented. Previously, Visplore enabled to re-order the axes of the parallel coordinates view and the columns of the Table Lens view by common GUI elements like lists.

As like in the previous section this implies the need of a control panel, which is on one hand not available in the dashboard version of Visplore and on the other hand not an intuitive way of reordering a tabular layout.

The in-place interaction technique which replaces the need for an additional control is *Drag and Drop*. By pointing onto an area which is close to the visual representation of the data attribute, the representation changes to give the user a visual feedback that it is possible to interact with it. By clicking onto the area and holding the mouse button down another feedback signals the user that the dragging of the data attribute has started. This feedback is a simplified representation of what the visualization would look like if the user would release the mouse button (drop) at the current position.

Figure 3.2 shows the implementation in the parallel coordinates view of Visplore. The labels underneath the axes are already a trigger area for the data mapping widget, which makes them unusable as a trigger area for the changing of the arrangement of the axes. In this case, the area above the label is used as the trigger area. Figure 4.5 shows the dragging of the axis “Global Radiation” to the left. The thin cyan line indicates the current mouse position and the thick cyan line represents the position where the axis would be if the user dropped it at this moment. In this example, the dropped axes would be inserted between the axes “Month of Timestamp” and “Wind Speed”. Furthermore, drag and drop was implemented for the columns of the table lens view [RC94]. This is handled similar to the parallel coordinates.

The advantage of this technique is that it does not need any further controls or widgets and it is faster to use and to learn. [Bux86] [HS12]

4.3.3 Adjusting visual parameters

In Section 3.3 the motivation for the user task “Adjusting visual parameters” was presented. Previously, Visplore enabled the adjusting of these parameters with sliders and input fields in the control panel.

To be able to adjust the visual parameters of the view with in-place interaction controls, a “two-stage” popup interaction technique was implemented (see Section 4.1). In Figure 4.1a a menu like structure is used to group different visual parameters, that share the same trigger area. However, this grouping is limited, because too many icons would occlude a big part of the view.

To reduce the number of icons, the original used controls were simplified into one control. For example the point size slider in Figure 4.1b controls in fact not one, but three related properties that are linked to this slider:

- A basic size of all data points displayed in the scatter plot.
- The size of currently selected data points as defined by brushing, which are drawn slightly larger in red color on top of the non-selected points
- The size of all currently hovered data items [BP10], which are highlighted in blue color, and drawn on top of the aforementioned point subsets.

This in place interaction techniques were implemented for:

- the 2D scatter plot: For transparency and point size
- the curve overview: For resizing glyphs representing curve data [PPBT12]
- the curve view: For line width and transparency of line graphs

CHAPTER 5

Discussion

The problems that occurred during the implementation and while testing it with an expert group are discussed in Section 5.1. Section 5.2 presents a conclusion of the work.

5.1 Discussion

This section discusses the implemented work. In Subsection 5.1.1 the improvements of this work for the visual analysis tasks are presented and in Subsection 5.1.2 the problems while implementing the in-place interaction controls and while using them are discussed.

5.1.1 Improvements

The previously presented changes to the Visplore system enables the user to fulfill tasks in a dashboard which were not possible before. Additionally the popup controls support the user in non dashboard contexts. The essential controls for interacting with the visualizations are easier to find and to access, because they appear at convenient positions, while they don't require any additional visual space.

This new class of in-place interaction controls enables the designer of the dashboard to easily add and configure the needed interaction tools.

5.1.2 Problems

Several problems occurred while implementing the popup controls. One major problem was to find the right timeout for accessing the popups on demand (see Section 4.1).

If the timeout is too short, it is possible that the popup controls appear unintended, when the mouse pointer of the user only passes by the trigger area. In that case the emerging controls are likely to occlude relevant information and additionally need some time to hide themselves after the mouse is no longer over the trigger area, because it is not possible to manually hide them.

On the other hand if the timeout is too long it is hard for users to find the trigger area without guidance. Furthermore if the user is familiar with the trigger area and wants to access the control, but has to wait for a noticeable long time, his workflow is disrupted.

A possible solution to this problem was presented in Section 4.1: The two-stage popup interaction, where a small, unobtrusive icon is displayed immediately and the actual control is shown only when hovering this icon. Preliminary experience feedback of users suggested a general preference of this two-stage approach. Adopting it for all of the implemented in-place controls might be considered in future work.

The visual space is also a restricting factor to the number of trigger areas for in-place interaction controls. Too many trigger areas might confuse the user and have a flat learning curve, because it is hard to remember the different functionalities. A solution introduced by this work could be to simplify the original used controls into one control (see Section 4.3.3).

5.2 Conclusion

This work introduced different techniques to control the views of a dashboard without reserving a permanent, dedicated visual space for them. This is achieved by hovering areas in the view which opens controls or triggers events.

For three user tasks in-place interaction methods were implemented for the visual analytics system Visplore. This work enables users to fulfill the exploratory tasks of restricting displayed ranges, reordering columns in tabular layouts, and adjusting visual parameters using the described methods.

These interaction techniques give the dashboard designer the possibility to design more compact dashboards with the same functionality without using space filling controls. Additionally to the reduction of the needed visual space the dashboard gets less cluttered, easier to understand and it is possible to export it as an image without removing the controls first.

A problem with this approach is that the user needs to know where the trigger areas of the in-place interaction are located in the visualization and additionally too many areas are confusing the user.

Bibliography

[ARP14] C Arbesser, O Rafelsberger, and H Piringer. The focus-filter widget: A versatile control for defining spatial focus + context in 1d. In *Proceedings of IEEE InfoVis*, 2014.

[AS94] Christopher Ahlberg and Ben Shneiderman. Visual information seeking: tight coupling of dynamic query filters with starfield displays. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 313–317. ACM, 1994.

[BP10] W Berger and H Piringer. Peek brush: A high-speed lightweight ad-hoc selection for multiple coordinated views. *Proceedings of Intern. Conference Information Visualisation (IV2010)*, pages pp. 140–145, 2010.

[Bux86] William Buxton. Chunking and phrasing and the design of human-computer dialogues. In *IFIP Congress*, pages 475–480, 1986.

[Chi00] Ed H Chi. A taxonomy of visualization techniques using the data state reference model. In *IEEE Symposium on Information Visualization*, pages 69–75. IEEE, 2000.

[cKFY11] Bum chul Kwon, Brian Fisher, and Ji Soo Yi. Visual analytic roadblocks for novice investigators. In *IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 3–11. IEEE, 2011.

[Dix09] Alan Dix. *Human-computer interaction.* Springer, 2009.

[EB11] Micheline Elias and Anastasia Bezerianos. Exploration views: understanding dashboard creation and customization for visualization novices. In *Human-Computer Interaction–INTERACT*, pages 274–291. Springer, 2011.

[EBN13] Alex Endert, Lauren Bradel, and Chris North. Beyond control panels: Direct manipulation for visual analytics. *Computer Graphics and Applications, IEEE*, 33(4):6–13, 2013.

[Few09] Stephen Few. *Now you see it: simple visualization techniques for quantitative analysis.* Analytics Press, 2009.

[Gal07] Wilbert O Galitz. *The essential guide to user interface design: an introduction to GUI design principles and techniques.* John Wiley & Sons, 2007.

[HS12] Jeffrey Heer and Ben Shneiderman. Interactive dynamics for visual analysis. *Queue*, 10(2):30, 2012.

[Ins09] Alfred Inselberg. *Parallel coordinates.* Springer, 2009.

[KAF+08] Daniel Keim, Gennady Andrienko, Jean-Daniel Fekete, Carsten Görg, Jörn Kohlhammer, and Guy Melançon. *Visual analytics: Definition, process, and challenges.* Springer, 2008.

[Kei02] Daniel A Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8, 2002.

[Mac87] Jock D Mackinlay. Automatic design of graphical presentations. Technical report, Stanford Univ., CA (USA), 1987.

[MAP15] T Mühlbacher, C Arbesser, and H Piringer. Statistical forecasting in the energy sector: Task analysis and lessons learned from deploying a dashboard solution. In *Short Paper in Proceedings of IEEE VIS 2015 (Visualization in Practice Track, to appear)*, 2015.

[MHS07] Jock Mackinlay, Pat Hanrahan, and Chris Stolte. Show me: Automatic presentation for visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1137–1144, 2007.

[MP13] Thomas Muhlbacher and Harald Piringer. A partition-based framework for building and validating regression models. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):1962–1971, 2013.

[MP14] T Mühlbacher and H Piringer. Task-tailored lightweight setups for interactive visual analysis of time series. In *Proceedings of IEEE Vis*, 2014.

[PBH08] Harald Piringer, Wolfgang Berger, and Helwig Hauser. Quantifying and comparing features in high-dimensional datasets. In *Information Visualisation, 2008. IV'08. 12th International Conference*, pages 240–245. IEEE, 2008.

[PBK10] Harald Piringer, Wolfgang Berger, and Jürgen Krasser. Hypermoval: Interactive visual validation of regression models for real-time simulation. In *Computer Graphics Forum*, volume 29, pages 983–992. Wiley Online Library, 2010.

[PPBT12] Harald Piringer, Stephan Pajer, Wolfgang Berger, and Heike Teichmann. Comparative visual analysis of 2d function ensembles. In *Computer Graphics Forum*, volume 31, pages 1195–1204. Wiley Online Library, 2012.

[PSCO09] William A Pike, John Stasko, Remco Chang, and Theresa A O'connell. The science of interaction. *Information Visualization*, 8(4):263–274, 2009.

[PTMB09] Harald Piringer, Christian Tominski, Philipp Muigg, and Wolfgang Berger. A multi-threading architecture to support interactive visual exploration. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1113–1120, 2009.

[RBH+12] Roland Radtke, Aaron M Butcher, Jensen M Harris, Catherine R Morrow, and Jesse Clay Satterfield. User interface for displaying selectable software functionality controls that are contextually relevant to a selected object, February 14 2012. US Patent 8,117,542.

[RC94] Ramana Rao and Stuart K Card. The table lens: merging graphical and symbolic representations in an interactive focus+ context visualization for tabular information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 318–322. ACM, 1994.

[Sof15] Tableau Software. Visual analysis best practices: A guidebook. `http://www.tableau.com/learn/whitepapers/tableau-visual-guidebook`, 2015. Last visited 2015-10-10, 2015.

[TC05] James J Thomas and Kristin A Cook. *Illuminating the path: The research and development agenda for visual analytics.* IEEE Computer Society Press, 2005.

[YaKSJ07] Ji Soo Yi, Youn ah Kang, John T Stasko, and Julie A Jacko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1224–1231, 2007.