DataMeadow: A Visual Canvas for Analysis of Large-Scale Multivariate Data

Niklas Elmqvist
INRIA
elm@lri.fr

John Stasko
Georgia Institute of Technology
stasko@cc.gatech.edu

Philippas Tsigas
Chalmers University of Technology
tsigas@chalmers.se

Abstract

Supporting visual analytics of multiple large-scale multidimensional datasets requires a high degree of interactivity and user control beyond the conventional challenges of visualizing such datasets. We present the DataMeadow, a visual canvas providing rich interaction for constructing visual queries using graphical set representations called DataRoses. A DataRose is essentially a starplot of selected columns in a dataset displayed as multivariate visualizations with dynamic query sliders integrated into each axis. The purpose of the DataMeadow is to allow users to create advanced visual queries by iteratively selecting and filtering into the multidimensional data. Furthermore, the canvas provides a clear history of the analysis that can be annotated to facilitate dissemination of analytical results to stakeholders. A powerful direct manipulation interface allows for selection, filtering, and creation of sets, subsets, and data dependencies. We have evaluated our system using a qualitative expert review involving two visualization researchers. Results from this review are favorable for the new method.

Keywords: Multivariate data, visual analytics, parallel coordinates, dynamic queries, progressive analysis, starplots, small multiples.

1 Introduction

Managing and presenting large, high-dimensional datasets is one of the core problems in information visualization, and the vast number of different approaches to solving this problem attests to its difficulty [22]. However, allowing for efficient analysis of such datasets also requires smooth and powerful interaction techniques for selecting, filtering, and combining the data. In fact, realistic multivariate analysis tasks often involve correlation and comparison of data from several sources, requiring that these techniques be capable of operating on multiple large-scale datasets instead of just one. Finally, insights are meaningless if they are not communicated to others, so our tools should support the dissemination of results of the analysis to an outside audience [39].

In this paper, we present a method called the DataMeadow that was designed to meet these requirements. The DataMeadow (Figure 1) provides users with a canvas for exploring multidimensional data sets using advanced visual queries. The data itself is represented by a DataRose, a color-coded, parallel coordinate starplot displaying selected variables of the set. Each displayed variable can be filtered using dynamic query bars [33, 46] attached to each rose axis. Individual DataRoses are connected in a data flow fashion; these connections are illustrated by arrows exiting the center of one DataRose and entering the center of another, as illustrated in the figure. In this way, the user can progressively build more and more complex queries with varying subsets of the data being passed along.

Furthermore, the incrementally-refined queries can be annotated with various visual representations in order to communicate the results to stakeholders (i.e. communication-minded visualization [41]). For added flexibility, the roses can be freely moved around, resized, and manipulated on the meadow canvas to allow for easy comparison to other datasets. To provide for more complex comparisons, DataRoses come in different types, either representing a data source or a specific set operation such as union, intersection, or uniqueness. This allows roses to be connected to other roses using dependencies, forming visual query chains. In essence, the DataMeadow provides a form of visual pivot table, enabling the user to refine and examine selected portions of a large multivariate data set in parallel.

In order to assess the utility and interaction efficiency of the method, we performed an expert review using a think-aloud protocol involving two visualization researchers. Our observations from this study indicate that the DataMeadow is a useful

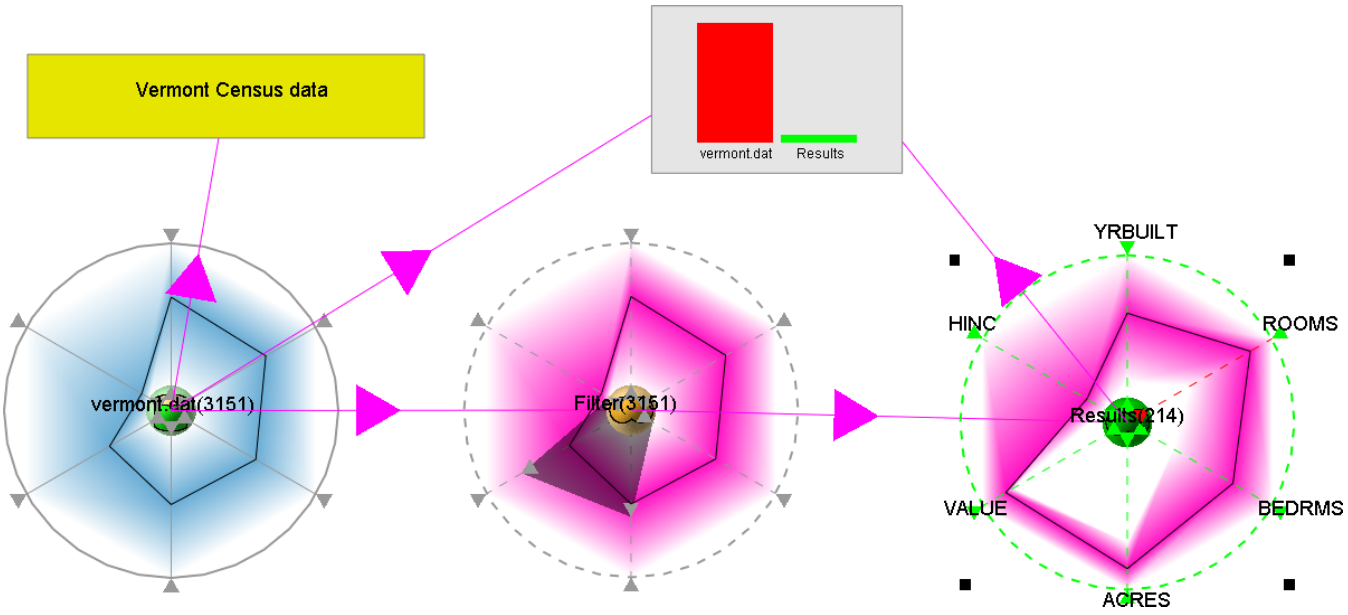


Figure 1: Sample house value and acreage versus number of rooms and owner income query in the DataMeadow.

way of thinking and interacting with multivariate data. The participants both remarked on the ease of creating queries and the power of being able to play with the data using the interaction techniques and getting immediate feedback.

The rest of this paper is organized as follows: We begin with a tour of the existing work on visualization and exploration of multivariate data. We then formulate the requirements for an analysis tool intended for such data, including identifying the user group and the main user tasks. We describe the DataMeadow visual canvas in detail and describe two typical scenarios using the tool. This is followed by our user evaluation and the results we gained from it. We finish the paper with a discussion and our conclusions.

2 Related Work

The work presented in this paper builds on ideas and inspiration both from techniques for visualizing multivariate data, as well as the application of these techniques to visual exploration, visual analytics, and progressive analysis. We describe these areas in turn in the following sections.

2.1 Multivariate Visualization

Much work has been conducted on visually presenting multidimensional data in a form suitable for understanding; Keim [22] gives an overview and taxonomy of such techniques. For large sets of multidimensional data, standard 2D or 3D symbolic displays such as plots, diagrams, and charts are generally insufficient due to scalability reasons, and more advanced methods are needed. Examples of such methods include geometrically-transformed displays [9], iconic displays [8], dense pixel displays [21, 23, 24], and stacked displays [20, 26, 32].

One prolific geometrically-transformed display technique is parallel coordinates [18, 19], which abandons the standard practice of orthogonal dimension axes and instead stacks up the axes in parallel, tracing a line instead of a point through the axes for each data case. The diagram is then easily extended with new parallel axes for each new dimension that is to be visualized. To avoid a linear extension of the diagram, an alternative representation called a *starplot* is constructed by transforming the diagram into polar space, mapping each axis on the radius of a circle. The DataRose component presented in this work is a direct descendant of the starplot and has indeed a parallel coordinate mode, but also other visual representations showing data distribution.

Fua et al. [13] introduce hierarchical parallel coordinates that are rendered in clusters using *opacity bands* instead of drawing each individual data point. The approach was later extended to starplot displays. The DataRose uses opacity bands as one visual representation, but they are manually clustered by the analyst. The system also supports *histogram bands* that show the distribution of the underlying data.

The parallel sets technique [5] is another approach to representing distribution for categorical data in a parallel coordinate diagram. It uses proportional scales and color paths to show how different categories divide among adjacent dimensions. Sifer [35] extends the idea by removing the color paths and instead relying on implict color coding. The DataRose also makes data distribution in the parallel coordinate display explicit, but our approach does not require categorical or hierarchical data.

The DataMeadow presented here can support a very large number of data cases, but if the number of variables to visualize grows too large, the scalability of the technique is affected. In such cases, we must employ techniques for very high dimensional representation, such as the dense pixel displays [21, 23, 24] and stacked displays [20, 26, 32] mentioned above. While our current datasets are not of this magnitude, we can easily foresee integrating visual elements based on these visualizations onto our canvas as well.

2.2 General Visual Exploration

Visual exploration enables gaining insights from (possibly unknown) large datasets using visualization. Its precursor, data exploration, was performed primarily using various statistical analyses [9, 10] combined with static diagrams. For example, Trellis displays [4] combine several diagrams—typically 2D scatterplots or line diagrams—into one panel, enabling comparison of related views with small parameter changes.

Visual exploration is traditionally employed for multidimensional data where the analyst will typically have little or no a priori intuition about the relationship and correlation between the dimensions. Thus, the initial part of the exploration task often becomes one of getting an overview of the data before looking at details [34]. What primarily distinguishes visualization from statistical approaches to data exploration is interaction [50]. Dynamic queries [1, 46] were one of the early approaches to interactive multidimensional exploration.

In more recent work, Theron presents the concept of interactive parallel coordinate plots (IPCPs) [38] as an interactive tool for analysis. IPCPs provide interaction techniques such as brushing [3] and axis filtering [30] similar to the DataRose approach in this paper. However, the DataMeadow allows for linking several DataRoses together to construct composite queries that are dynamically updated as the analyst interacts with the visual elements.

Finally, other work on visual exploration has tackled the problem of multivariate data: Xie et al. [48] consider two approaches to incorporating quality information in multivariate visualization. Pivot graphs [42] extend the concept of pivot tables (found in applications such as Microsoft Excel) by visually aggregating data cases on values for selected dimensions. The Dust & Magnets [51] technique is an example of the power of interaction for exploration, and shows how a simple interaction can provide important insights into a complex dataset through animation. Another example is the parallel coordinate tree [7] introduced by Brodbeck and Girardin for presenting hierarchical and multidimensional data using a tree representation. Their use of focus+context distortion for interacting with the visualization fulfills an integral role in the exploration of the data.

2.3 Visual Exploration Environments

Several visual environments have been proposed and built for explicitly supporting multidimensional visual exploration. Roughly speaking, they are divided into *free-form* and *structured* platforms, where the former provides a sketchboard for constructing analyses, and the latter uses a more rigid data format to allow for processing and visualizing structured information. The two main systems that characterize these approaches are the Sandbox and Polaris, respectively.

The Sandbox [47] system is an example of a free-form visual analytics platform. The tool emphasizes fluid interaction on a 2D canvas using direct manipulation to promote visual thinking. Towards this end, the Sandbox even supports a gesture detection component. Analyses are iteratively built and refined, bringing together many different types of media ranging from documents and text, to images and annotations.

Polaris [36] and Tableau (*http://www.tableausoftware.com/*), its commercial successor, are structured visual analysis platforms that allow for exploring large multidimensional databases based on data cubes (basically the pivot table mechanism popularized by Microsoft Excel). Users explore the data by visually constructing a specification for the graphical representation, mapping

dimensions to fields and columns in a table and aggregating the others. The results can then be visualized using standard techniques such as bar charts, scatterplots, and parallel coordinate displays.

The VITE [44, 45] system represents a mix of the two above extremes by maintaining a two-way mapping between structured information in a database and the informal, spatial arrangements of information that is most natural to a human analyst. Similar to the DataMeadow, the system provides a visual canvas for interacting and manipulating information. Instead of enforcing a strict formalization on the information created by the user, the system stores both informal and formal representations. This allows for retaining intermediate and semi-structured information that is important for problem solving [44].

Similar to VITE, the DataMeadow system presented in this paper is a combination of free-form and structured visual exploration. We provide a semi-structured visual canvas for effortlessly constructing and refining visual queries, just like the Sandbox. We even support the same kind of gesture detection support to further streamline the interaction. However, visual components on the DataMeadow canvas all obey the same multivariate data format, supporting data flow and automatic processing of the data, akin to more structured systems like Polaris. The DataMeadow canvas also allows for visual pivot table-like queries. On the other hand, the data flow model in the DataMeadow is not nearly as strict as the one employed by Polaris and Tableau.

Several other systems beyond these exist in the literature; a few representative examples follow here. Brennan et al. [6] present a framework for exploration of multidimensional data and employ a visual canvas, but focus on collaborative aspects of the platform. Yang et al. [49] present an analysis-guided exploration system that supports the user by automatically identifying important data—"nuggets"—based on the interests of the users. ClusterSculptor [28] is a visual environment for high-dimensional cluster analysis consisting of several visual and analytical components.

2.4 Progressive Analysis

Gotz et al. [14] define *progressive analysis* as an interative process where newly synthesized knowledge becomes the foundation for future discovery. Their HARVEST system allows for marshaling both existing analytical knowledge as well as new knowledge discovered during the analysis process. This concept of progressive analysis is core to many visual analytics environments, such as the Sandbox [47] and Polaris [36].

Spreadsheet applications, such as Microsoft Excel, are classic examples of tools supporting progressive analysis, where users perform calculation by chaining together cells using dependencies and mathematical operations. However, the visualization aspects of most spreadsheet applications are primitive and typically lack interaction. Another problem is that dependencies are virtually invisible on the spreadsheet; Shiozawa et al. [31] address this by lifting dependencies into 3D, but their work is still concerned with traditional spreadsheets.

Chi et al. [17] present an approach to unifying visualization of multidimensional datasets into a spreadsheet framework. The implicit data dependency model in their work resembles the more explicit data flow model in the DataMeadow, and similarly allows for the visualization and modification of (possibly several different branches of) intermediate results in a larger analysis. Furthermore, the tabular layout is familiar to users, allows for comparison between different cells, and supports customized layout depending on the task. Since single cells can hold entire datasets, their framework also provides compound operations such as set subtraction and addition.

However, Chi et al. also note that one of the features of a spreadsheet approach is that it hides the operators and instead show the operands, unlike data flow environments that typically hide operands and focus on the operators. Our objective with the DataMeadow data flow model was to make *both* operands and operators visible on the analysis canvas.

3 Requirements

A number of both functional (task-centered) and non-functional (general) requirements apply for a visual analytics application designed for multivariate data. These requirements have been derived from theoretical treatments [22, 34, 39] on as well as existing cognitive task analyses [47] of visual exploration and the analysis process.

The primary users of the DataMeadow tool are expert analysts familiar with multidimensional data management and representation. Some of the operations, such as filter and set operations, may be too complex for a novice user to easily grasp, yet are necessary to satisfy the requirements of the target user group.

3.1 General

Below are the main non-functional guiding requirements necessary to fulfill analyst goals:

(R1) Interaction: Interaction with the system must be smooth and effortless in order to not slow down or distract the user. It should also cater to expert-level users who are experienced in using the system.

(R2) Exploration: Support and encourage data exploration by providing easy access to analysis tools such as filtering, sorting, correlation, etc [34].

(R3) Iterative refinement: The approach should lend itself to progressive analysis [14] of the data in small multiples [40].

(R4) Communication: Support the production, presentation and dissemination of analytical results [39, 41].

3.2 User Tasks

Visual exploration [22, 25] often follows the "information-seeking mantra" [34]: overview first, zoom and filter, and provide details on demand. Any visual analytics application should support these basic tasks, and this is the case for the system discussed in this paper.

More specifically, in this work we are targeting simultaneous visualization of multiple large-scale data sets. The main user task the application needs to support is *comparison*; either comparison between different datasets, such as data for different states in the United States, or between subsets of the same or different sets, such as data for different cities or counties in the same state.

In the task taxonomy of Amar et al. [2], comparison is classified as a higher-level meta-operation. In our model of the DataMeadow, this is certainly true: in order to support this broad comparison operation, we must provide for a wide range of lower-level user tasks such as (using the terminology of Amar et al.) retrieve value, filter, correlate, characterize distribution, etc. Wehrend and Lewis [43] refer to this operation as compare within and between relations—this also applies to the DataMeadow, where we support both comparison between datasets as well as between subsets within the same dataset.

4 The DataMeadow Method

The DataMeadow method is a free-form visual analytics environment based on a visual canvas with a structured data flow model designed for visualization of multiple high-dimensional datasets. Users build visual queries by connecting visual components into dependency chains that allow for iterative refining. The main driving user task behind the design of the technique is comparison between different sets or subsets of data.

In this section, we describe the visualization method, including the data flow model, the visual elements, and the interaction techniques.

4.1 DataMeadow Canvas

The actual DataMeadow component is an infinite 2D *canvas* populated with a collection of *visual elements* used for multivariate visual exploration. A visual element is a graphical entity with an appearance, a number of integrated user controls, an operation on its input data cases (potentially the identity function), and input and output *dependencies*. Elements can be created, modified, and destroyed as needed. Individual elements are chained together using dependencies, forming a *data flow* that propagates data cases between the elements.

- **Canvas:** The infinite 2D plane on which all components are anchored. Supports sort and layout operations of elements. Has an associated *data format* that describes the meta information about the available dimensions and their data type.
- **Visual elements:** A graphical entity used for data analysis. Different element types perform different operations. Example types include DataRoses, textual annotations, data viewers, etc.
- **Dependencies:** Directed connections linking one visual element to another. Data cases are propagated through dependencies from the source to the destination element.

Taken together, these components can be assembled to construct complex visual queries. In the following sections, we will describe this process in more detail.

4.2 Data Format

Each DataMeadow conforms to a specific *data format* that describes the format of the datasets, i.e. the columns and their metadata (name, data type, domain, labels, operations, etc). The format specifies what information is stored in the visual elements as well as what is passed through the dependencies connecting them. The meadow can contain several different datasets as long as they all conform to the same data format, allowing for comparison of multiple related datasets (such as baseball statistics for different seasons, teams, or players, or US Census data from different years or states).

The data types for each dimension may be one of $\{$*literal*, *nominal*, *ordinal*, *quantitative*$\}$, and govern permissible operations on the dimension (i.e. ordinal data can be sorted, nominal data can be reordered to optimize readability, quantitative data supports mean, median and min/max computation, etc). Data labels for nominal or ordinal data may be used as labels in the visual elements on the DataMeadow canvas.

4.3 Visual Elements

The basic building block of the DataMeadow method is the visual element, a component consisting of a visual appearance, a variable number of user controls (none for some elements), and input and output dependencies. Furthermore, elements have a transformation operation that is applied to all of the input data cases (potentially the identity function for basic elements).

Each element follows a strict multivariate data model based on the currently active data format for the canvas. As discussed above, the data format governs how information flows through the system through the dependencies and how it can be transformed by the elements.

There are three types of visual elements in the DataMeadow method (examples of each type are given in brackets):

- **Sources:** Producer elements from where data originates. The data is propagated to other elements using outgoing dependencies. [database readers, noise generators, number generators, etc]
- **Sinks:** Consumer elements that accept incoming data and consume it, potentially changing its visual appearance to reflect the nature of the data. [viewers, annotations, flags]
- **Transformers:** Input/output elements that transform incoming data using some operation and output it to outgoing dependencies. [DataRoses]

In the following sections we will be describing some of the elements in greater detail.

4.4 Dependencies and Data Flow

A dependency is a directed connection between two visual elements on the same DataMeadow. Data cases from the source flow along the dependency to the destination element using the data format of the meadow. This is the basic principle supporting the iterative refinement (R3) requirement from Section 3.1.

Dependencies are never filtered or constrained (filtering is performed in the visual elements). Mutual or circular dependencies are not possible on the DataMeadow canvas due to the flow-directed nature of the underlying data model. Attempting to create a circular dependency will be detected and prevented by the system.

Dependencies ensure that changes in source data are properly propagated to all dependees. Thus, when an analyst changes the parameters of a visual element in a chain, all elements further down in the chain are immediately updated to provide feedback to the user. This way, the user can directly see the effect of a parameter change to the visual query.

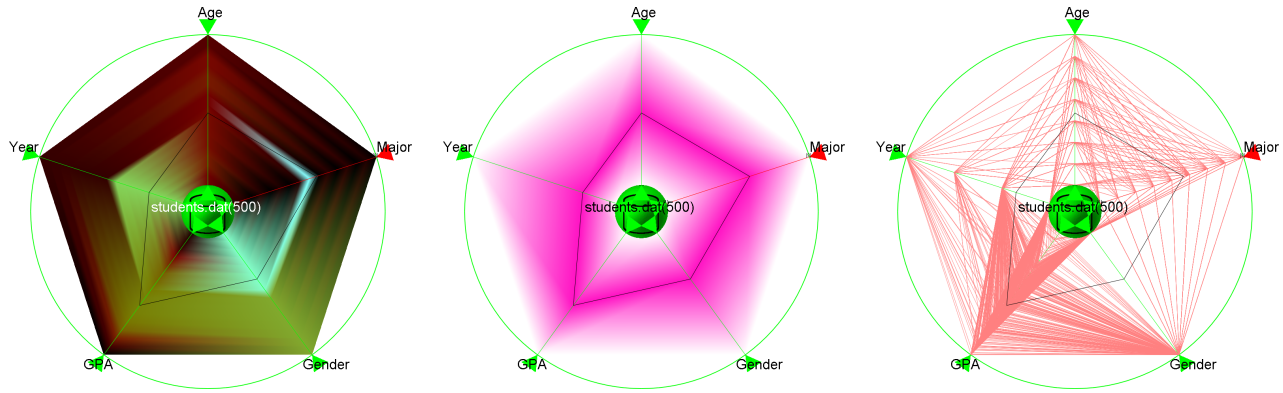


Figure 2: Sample DataRose visualization for a university student database of a computer science department. (a) Color histogram mode (high brightness equals high density). (b) Opacity bands mode. (c) Parallel coordinate mode.

4.5 DataRose

The core visual elements in the DataMeadow method are called *DataRoses*: 2D starplots displaying multivariate data of the currently selected dimensions of the dataset. The data can have different visual representations depending on the task; examples include color histogram mode, opacity band mode [13], and standard parallel coordinates mode. The design intention of the DataRose is to provide a self-contained visual entity that lends itself to side-by-side comparison to other datasets.

A DataRose represents one specific dataset, and can be derived either from a database or noise generator source, or be the result of a set operation (see below for more on this). More specifically, a DataRose is a mathematical set, i.e. all entities contained in a rose appear only once.

4.5.1 Visual Representation

Figure 2 shows the three visual rose representations for a fictitious university student database. The database contains 500 student entries and maintains five value dimensions (beyond the student name): the age (quantitative), major (nominal), gender (nominal), GPA (quantitative), and graduation year (ordinal) of each student (Figure 6 gives a sample of the data format specification).

For all three visual representations, a single black polyline is used to show the average for each dimension. Low values are close to the origin, high values reside on the outer radius. Data labels can be visualized on the axes, but rendering them all may cause high visual clutter. Instead, hovering the cursor over an axis or moving the dynamic query filter handles (see below) will show the currently highlighted data value.

In **color histogram mode**, the data distribution for each dimension is shown on the surface of the rose using a continuous color scale. The color transitions between color values of adjacent axes are rendered using smooth interpolation.

Figure 2(a) shows a color histogram using the OCS [27] color scale. High brightness indicates high density in the underlying distribution, so it appears that age (the 12 o'clock dimension) is fairly evenly distributed across the dataset. Going clockwise around the rose, for major there is a brighter cyan area around the mid-term mark of the dimension, indicating a high concentration for this value. As it turns out, this is a database of students attending courses in a computer science department, and looking at the value legend reveals that this particular value is for students who have computer science as their major. The gender dimension reinforces this fact, as there appears to be a skewed gender balance in the dataset (i.e. the inner half of the gender dimension, representing males, in the DataRose is brighter than the outer part, representing females, indicating a higher number of male than female students) that is characteristic of computer science courses. The students have an above-average, bell-shaped GPA (there is a gradual "hump" of brighter values centered in the upper part of this dimension), and most of them seem to be freshmen or sophomores (the lower half (years one and two) of the year dimension is brighter than the higher half (years three and four).

In **opacity band mode**, the underlying data is abstracted using opacity bands that smoothly go from full opacity at the average

to full transparency at the extremes (minima and maxima). Transitions between adjacent axes are again rendered using smooth interpolation. Figure 2(b) shows an opacity band where the amount of purple color indicates the data density. The same trends we noted from the color histogram representation are visible here as well, albeit at a higher abstraction level. Furthermore, the density of the data for different values is less obvious, and the observation about most students being computer science majors is hard to make here.

Finally, the **parallel coordinate mode** uses traditional parallel coordinate rendering, where all cases of the underlying dataset are rendered using polylines that connect the values for each dimension. However, the disadvantage is that data distribution is more difficult to see in this visual representation.

Different representations are suitable for different tasks; while parallel coordinates certainly display the most information, it is sometimes useful to be able to abstract away some of the details when trying to get an overview of the dataset. Opacity bands are suitable for getting an idea of the average and extreme values of the underlying dataset. For some analysis tasks, it is important to be able to see the data distribution, something which can be very difficult in parallel coordinate mode where a lot of data cases might map to the same position on the axis (especially for nominal dimensions). Color histogram mode shows a detailed breakdown of how the data cases divide among the values along each dimension.

4.5.2 Starplot Layout

Starplots are constructed by splitting a full 360° circle into n parts, one for each of the dimensions $D = \{d_1, d_2, \ldots, d_n\}$ to be visualized. This assigns each dimension $360°/n$ of the circle. For each data field, an axis is drawn radially from the center of the circle to its perimeter. The center part of the plot is reserved for interaction, such as dragging the plot around the canvas and creating dependencies, and this part is also used for the visual icon for the specific plot type. The remaining part of the axis is normalized to the range of the associated data field and is used for plotting individual data cases.

Note that in all visual modes, we use the starplot axes as continuous dimensions, even for nominal data. This is perhaps counter-intuitive and imposes an artificial ordering between these values. For future iterations of the technique, it would be useful to employ the DQC [29] reordering approach to impose an optimal ordering of coordinate mappings of nominal variables.

4.5.3 Axis Filtering

In the DataMeadow, as shown in Figure 1, each DataRose starplot axis also has a dynamic query [1, 46] slider to allow for axis filtering [30]. The handles for each slider are shown as two small opposing triangles on the axis plotted at the extremes of the current filter selection. In addition, a semi-transparent area is drawn over the areas of the DataRose falling outside of the current filter selection. The user can grab the query handles and move them, dynamically changing the filter selection and causing the visual elements further down in the chain of connected elements to be updated. This allows the analyst to go back and make upstream filter changes that affect a whole query.

This iterative refinement using dynamic queries is an important distinction to software systems that are based on dynamic queries (DQ), such as Spotfire. In these systems, the sliders are typically global in scope, whereas they are local for the data flow chain in the DataMeadow.

Figure 3 shows an example of axis filtering where the analyst has filtered the student database example from above to only include students of 25 or above with a certain range of graduation year, major, and GPA. Any outgoing dependencies from this rose will only propagate the filtered data. Furthermore, the data flow model shows interactive feedback at all times as the analyst is changing the dynamic queries, promoting visual exploration of the data. Moving the slider handles also shows the labels of the current range of the filter.

In Figure 1, the analyst has constructed a complete visual query from a house database for the state of Vermont. By changing the DQ filter settings in the middle rose, the analyst is able to study the correlation of high value and acreage on the number of rooms and bedrooms of a house by looking at the average and extreme values in the result rose to the right. The leftmost database rose and the result rose have also been connected to a barchart viewer (see Section 4.6) to show the relative sizes of the two roses.

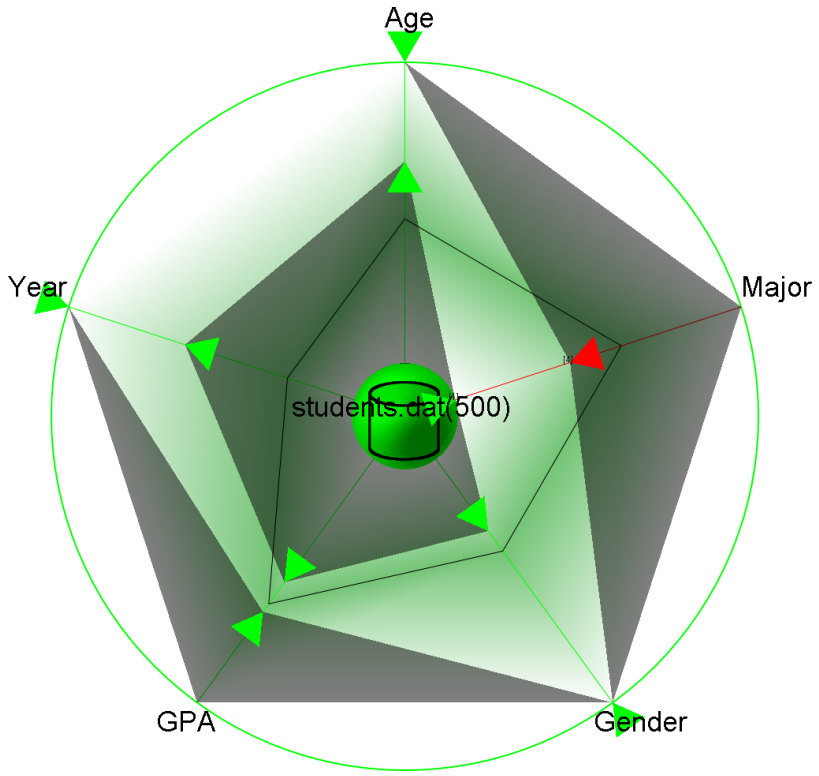
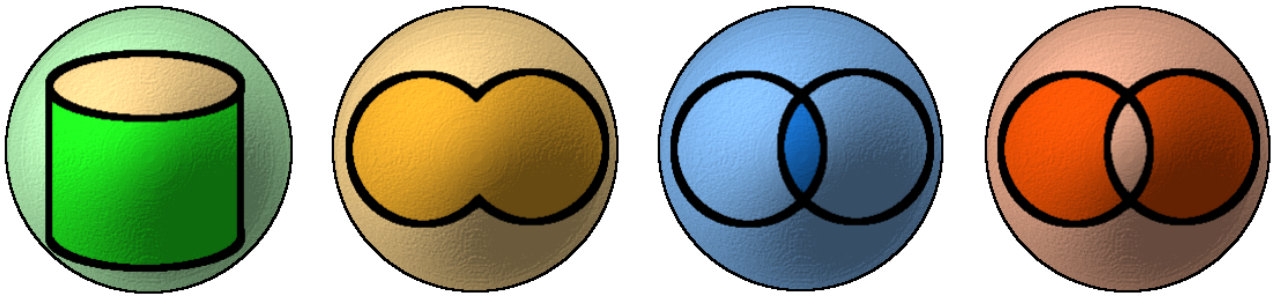


Figure 3: Dynamic query axis filtering for the student database.

4.5.4 Rose Types

To support complex user tasks such as correlation and characterization, we introduce additional DataRose types other than the standard database **source**, which represents an external database loaded from a file or connected to over the network. We define the rose types as set operations, allowing us to construct advanced visual queries through constrained and unconstrained dependencies. All rose types accept variable input dependencies, i.e. they have been generalized from standard binary set theory operations.

- **Source:** External database loaded from a file or connected to over the network (Figure 4(a)).
- **Union:** Set representing the union of all input dependencies, i.e. the combination of all input cases (Figure 4(b)).
- **Intersection:** Set representing the intersection of all input cases, i.e. only cases that are present in all input dependencies (Figure 4(c)).



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

Figure 4: DataRose type icons. (a) Database (source). (b) Union. (c) Intersection. (d) Uniqueness.

- **Uniqueness:** Set representing unique inputs, i.e. only cases that exist in only one input dependency (Figure 4(d)).

Set operation rose types are useful for advanced correlations, such as between different visual query branches. For example, in the case study below, the analyst uses an intersection rose to see whether any of the high value houses he has identified in one visual query also are present in the high acreage subset he derives in another (Figure 7).

Additional rose types representing other, more complex operations can added to the DataMeadow framework.

4.6 Viewer Elements

Viewers are sinks that accept input and have no output dependencies, typically changing their visual representation to reflect the incoming data. They are useful for studying the results of more complex queries involving DataRoses. Lacking the complexity of transformer elements such as the DataRose, they are also suitable for inclusion in reports and presentations.

The following viewer elements are supported by the DataMeadow canvas:

- **Quantity barchart:** Shows the relative amount of cases coming in from the different dependencies as a barchart.
- **Quantity piechart:** Same as the above, but using a piechart representation.
- **Linear histogram:** Data distribution of each dimension shown as a standard linear histogram.

Examples of viewer elements can be seen in Figures 5 (barchart and piechart).

4.7 Annotation Elements

Annotations are sink elements whose primary purpose is to support the *communication* requirement by providing a way for the analyst to incrementally annotate findings using free-text messages and media. Because they are sinks, an annotation object typically has inbound dependencies, and can thus present reports on the data. The following annotation element types are supported in the DataMeadow:

- **Labels:** Names and labels to denote a specific element or analysis result.
- **Notes:** Longer textual descriptions (more than a single line).
- **Images:** Bitmap images to illustrate particular elements or analysis results.
- **Reports:** Textual reports of the incoming data, such as average, minima and maxima, etc. Automatically updated as the data changes.

5 Implementation

The DATAMEADOW application was implemented using the C# programming language and the Microsoft .NET framework. The application uses the Tao (*http://taoframework.com/*) C# bindings for OpenGL to get access to both 2D and 3D accelerated graphics functionality but no special visualization toolkit was used. The interface components were realized using the Windows Forms toolkit.

The prototype implementation has been optimized to deliver interactive framerates even for very large datasets (more than 500,000 data cases). On the data management side, we achieve this using cascaded data tables, data columns, and filters inspired by the InfoVis toolkit [12] and software design patterns for information visualization [16]. On the rendering side, we render the color histogram and opacity band modes using hardware-accelerated OpenGL triangle rendering; parallel coordinate rendering has a much larger performance overhead and is discouraged for datasets of this size (more than 100,000 entities).

The DATAMEADOW application supports loading of comma-separated spreadsheet files (commonly having the `.csv` suffix) based on an XML-encoded data format. A special loader has been added to the application to allow for loading the Public Use Microdata Sample (PUMS) dataset from the US Census data.

As can be seen from Figure 5, the prototype implementation has three distinct interface parts: (i) a main visualization window, (ii) a dimension selection part (upper right), and (iii) a currently visible dimension part (lower right). The main visualization

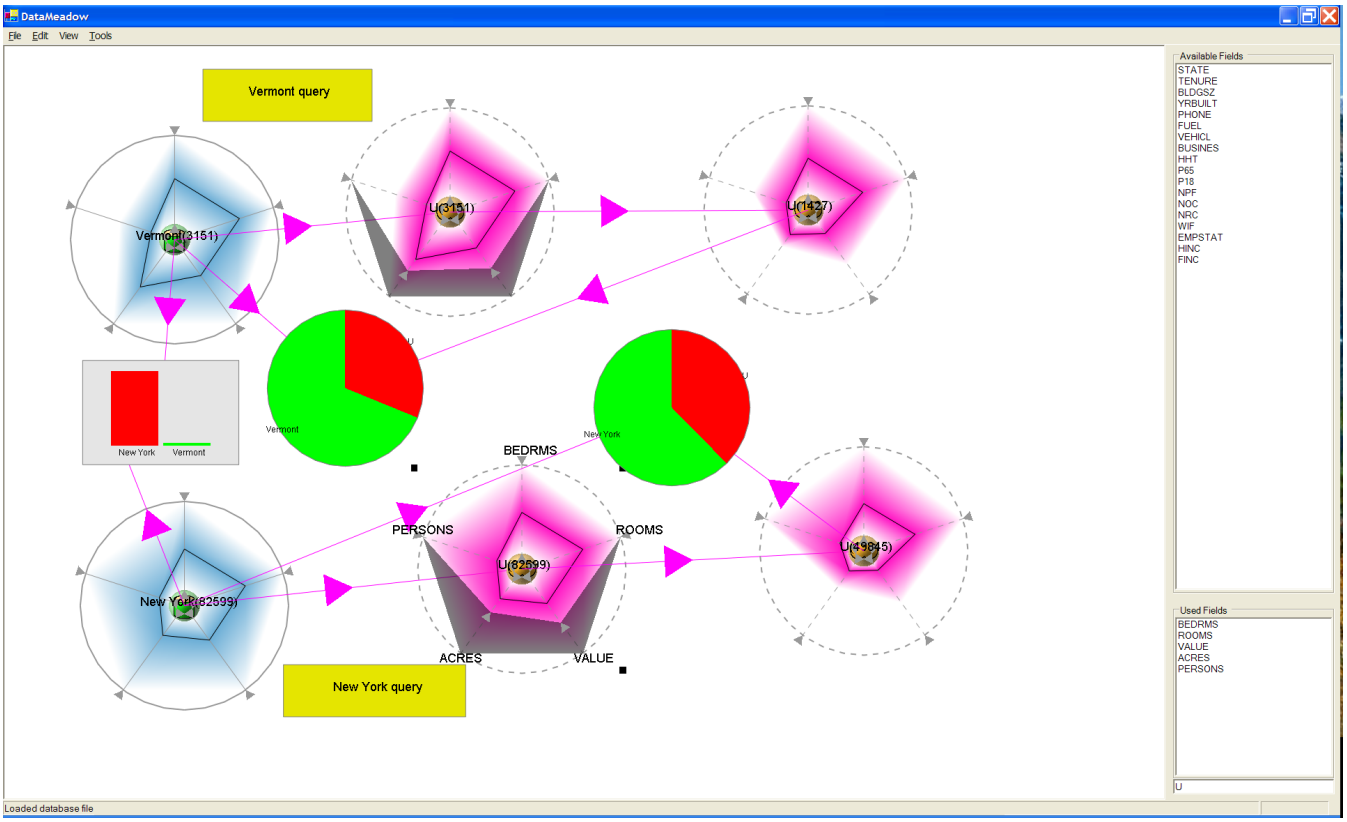


Figure 5: The DATAMEADOW prototype implementation. The main panel shows the visualization canvas and the smaller panels to the right show the available and currently visualized dimensions in the data.

window is a continuously zoomable viewport into the infinite 2D canvas representing the DataMeadow. Users can zoom and pan across the whole canvas using simple mouse interactions. The dimension selection interface boxes allow the user to select which dimensions in the data format to visualize—this can be dynamically changed, hiding or showing dimensions as necessary.

5.1 Dynamic Queries

Since the axis-filtering dynamic queries are so central to the DATAMEADOW application, special attention has been devoted to optimizing these. The mechanism may filter potentially hundreds of thousands of data cases for every update, putting severe requirements on the implementation.

Following the example of the InfoVis Toolkit [12], our implementation assigns one bit for every axis-filtered dimension per data case. If a particular row falls within the query for a particular axis, the bit will be set. Updating the query for one axis will only cause recomputation of the particular bit corresponding to that axis, resulting in performance linear to the number of rows (data cases). When computing its output, the DataRose only propagates cases whose bits are all set.

Tanin et al. [37] describe additional optimization techniques for dynamic queries, but these have so far not been necessary for interactive performance in the DATAMEADOW implementation.

5.2 Interaction Techniques

The DATAMEADOW implementation provides a number of interaction techniques (supporting the interaction requirement of Section 3.1):

- **Mouse navigation:** The viewport can be panned by pressing the center mouse button and dragging, or zoomed in or out by pressing the right mouse button and dragging.
- **Brushing:** Selecting a data case in one DataRose will highlight the case in all of its appearances in other DataRoses (parallel coordinates only).
- **Keyboard accelerators:** Common functionality can be accessed using keyboard hotkeys.
- **Mouse gesture detection:** The user can perform complex mouse gestures on the canvas to create new set operation DataRoses.

The keyboard accelerators and mouse gesture support allows the analyst to construct visual queries without having to leave the visualization window to access menu options or toolbar buttons. For example, drawing a U-shaped gesture on the canvas will create a union rose, and an upside-down U will create an intersection rose.

While interface shortcuts, such as mouse gestures and keyboard accelerators, are hard to learn and lack visibility [15], they are important factors for supporting the interaction requirement (R1) of the DATAMEADOW system.

5.3 Layout Mechanisms

The DataMeadow canvas lends itself nicely to employing a number of layout mechanisms for arranging the roses and their dependencies. A number of simple layouts such as circle, grid, and dependency depth order have been implemented. A more complex physically-based layout scheme using springs and dampers can also be employed. The ambition is to provide semi-automatic layout (akin to [47]) to aid the user in organizing the visual elements.

5.4 Format and Dataset Builder

As mentioned above, the current DATAMEADOW implementation supports loading comma-separated spreadsheet (`.csv`) files as well as the US Census PUMS dataset. However, the application requires a data format to be able to process and visualize datasets correctly. Data formats are stored as XML files (Figure 6 shows an example). Manually building such a format can be difficult as well as tedious, particularly for complex datasets.

```
<?xml ...?>
<dm:format>
  <dm:dimension id="Name" index="0" type="4"/>
  <dm:dimension id="Age" index="1" type="0"/>
  <dm:dimension id="Major" index="2" type="0">
    ...
    <dm:value val="5" id="Computer Science"/>
    ...
  </dm:dimension>
  <dm:dimension id="Gender" index="3" type="0">
    <dm:value val="0" id="Male"/>
    <dm:value val="1" id="Female"/>
  </dm:dimension>
  <dm:dimension id="GPA" index="4" type="0"/>
  <dm:dimension id="Year" index="5" type="1"/>
</dm:format>
```

Figure 6: Sample data format for a university student database.

To remedy this problem, we have implemented a preprocessing format and dataset builder in Java. The application has two different purposes:

1. To build a data format using a large set of raw datasets as input; and

2. To build structured dataset files using a data format.

The first task simply involves loading all (or at least a representative subset) of the datasets (as comma-separated text files) that will be included in the analysis. The application parses all of the dimensions found in the different datasets and automatically determines the type and domain of each dimension. The user can then edit these settings and select which of the available dimensions should be included in the final data format. The application also support elimination of synonyms as well as special transformations, such as splitting a compound dimension into several separate dimensions. Finally, the user can save an XML representation of the computed data format for use in the DATAMEADOW application.

The second task is performed after the user has a complete data format, and takes the raw data (as comma-separated files) and builds structured data tables that conform to the format. In the process, the application can join several input tables into a single output table using a primary key, such as when merging two separate tables for the same entity but with different dimensions into a single, coherent table.

6 Case Studies

We illustrate the use of the DataMeadow using two case studies: a classic demographics analysis and a more everyday task of finding a suitable digital camera.

6.1 US Census Data

Let us follow a fictitious analyst (Alan) who is using the DataMeadow to study the Public Use Microdata Sample (PUMS 1%) of the US Census data from 2000. The prototype implementation has support for loading data formats based on either the person or housing records of the PUMS dataset. This allows Alan to easily select and load the database file for a specific state into the application. Alan is interested in studying the PUMS housing records, so he first loads the housing data format. He then decides to start his analysis in the state of Vermont, so he loads this dataset into the application.

Upon finishing loading, Alan is presented with an empty DataRose representing the Vermont dataset, containing 3,151 entries. First, he selects which of the 18 dimensions in the database he wants to display, opting for build year, number of rooms, number of bedrooms, acreage, value, and owner income. He quickly creates a data flow chain by right-clicking and dragging on the Vermont rose to create a first derived rose, and then again on the new rose to create a second. He will use the first derived rose for filtering, and the second to view the results, so he labels them accordingly. Finally, he creates a barchart viewer and connects the Vermont rose to the result rose so that he can easily observe size ratios as he explores the data. See Figure 1 for his starting setup.

Now Alan is free to get a feeling for the data by changing the filter selection on the filter rose. He does this by clicking and dragging on the DQ handles on this rose and observing the visual results in the results rose as well as the barchart. He is able to quickly confirm some things that he already knows: for instance, that high value and high acreage implies many rooms and bedrooms.

Next, Alan wants to start a new line of reasoning, so he creates a second two-element chain of derived roses from the Vermont database. He is free to leave his first query undisturbed. He decides to remove the number of bedrooms dimension and instead look at the number of persons in the household. Feeling that he may be on to something, he decides to cross the results of the first query with the results of the second. In order to do so, he creates an intersection rose and connects the two queries to it. This rose will now show the houses from the original dataset that are part of **both** results from the two separate queries. See Figure 7 for the state of his DataMeadow canvas.

Alan decides to bring the state of New York into the picture to contrast against Vermont. He gets rid of the second query branch and loads the New York dataset, resulting in a second blue database rose. All dimension axes are automatically rescaled by the application to use the same scaling factor, making it possible to directly compare roses from two different datasets. Alan builds up a new query chain for New York and starts exploring the data using axis filtering. By imposing the same constraints on the chains of both states, he can see differences in the datasets. At one point, he notices that in Vermont state, a high number of persons in a household often implies a large acreage, but that this is not at all the case for New York state. See Figure 8 for his final analysis result.

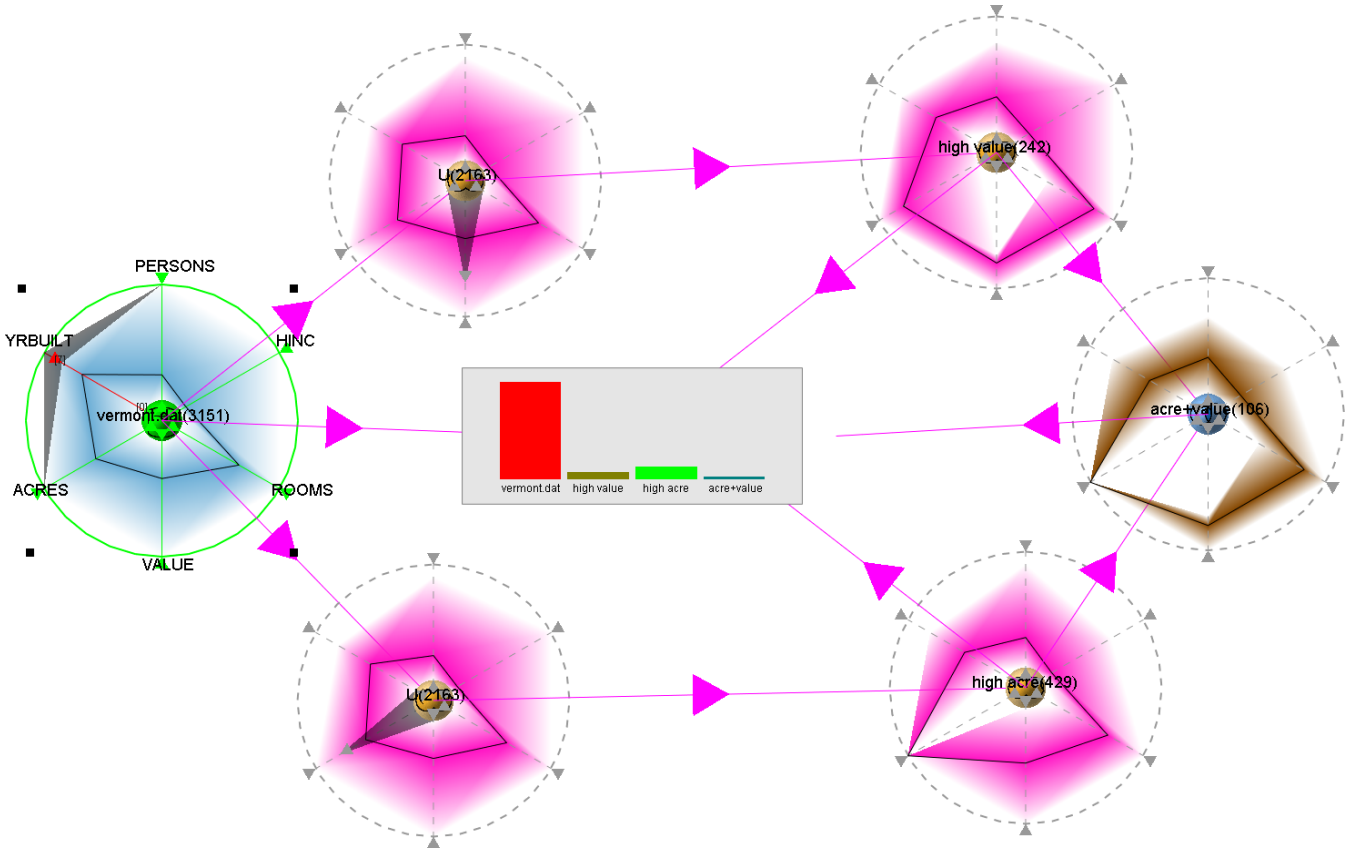


Figure 7: Two visual query branches (value and acreage) crossed using an intersection rose (brown).

6.2 Digital Cameras

To show how the DataMeadow tool might also be employed by a normal (i.e. non-analyst) user, we will follow Anna, who is trying to figure out which digital camera to buy. Anna has a number of different criteria for her new camera that she knows are conflicting and that she has to weigh against each other, but feels that neither vendors nor online review sites give her sufficient and reliable support in doing this. Therefore, she resolves to use the DataMeadow application to explore her options.

Lacking a coherent dataset of digital cameras, Anna's first step will be to prepare one for her analysis. Speaking to her vendor, she manages to get a collection of Excel spreadsheet files, one for each camera manufacturer. In total, these files contain data for 1,088 cameras. However, the files are in an idiosyncratic format on both the available dimensions and their domains. Furthermore, the vendor provides camera prices as separate Excel files that will have to be paired with the rest of the camera data.

Anna turns to the DataMeadow format builder (Section 5.4) to transform her idiosyncratic digital camera database into a coherent set of datafiles obeying a unified data format. Loading all of the manufacturer Excel files into the application, she is able to select the dimensions she wants to preserve for her analysis as well as edit their domains to get rid of aliases. Finally, she uses the builder application to join the price files with the main camera datasets—using the camera model names as the key—to produce a single comma-separated dataset file for each camera manufacturer.

She is now ready begin her analysis. She starts the DataMeadow application, loads the newly derived data format XML file, and then loads all of the datasets into the application. They appear as DataRoses on the visual canvas, one per manufacturer. Since Anna is interested in looking at all of the cameras in the dataset, she creates a union DataRose and connects all of the manufacturer roses to it to create a rose representing the whole dataset. See Figure 9 for an example of this.

Collapsing all of the 11 database roses into a union rose representing them all, she starts to build a visual query. She first derives a filter rose and a result rose by right-dragging on the "all cameras" rose. Anna is an amateur ornithologist, so she primarily wants a camera that has a high zoom factor. She moves the filters for the "Zoom tele" dimension to a high range (from 500 mm

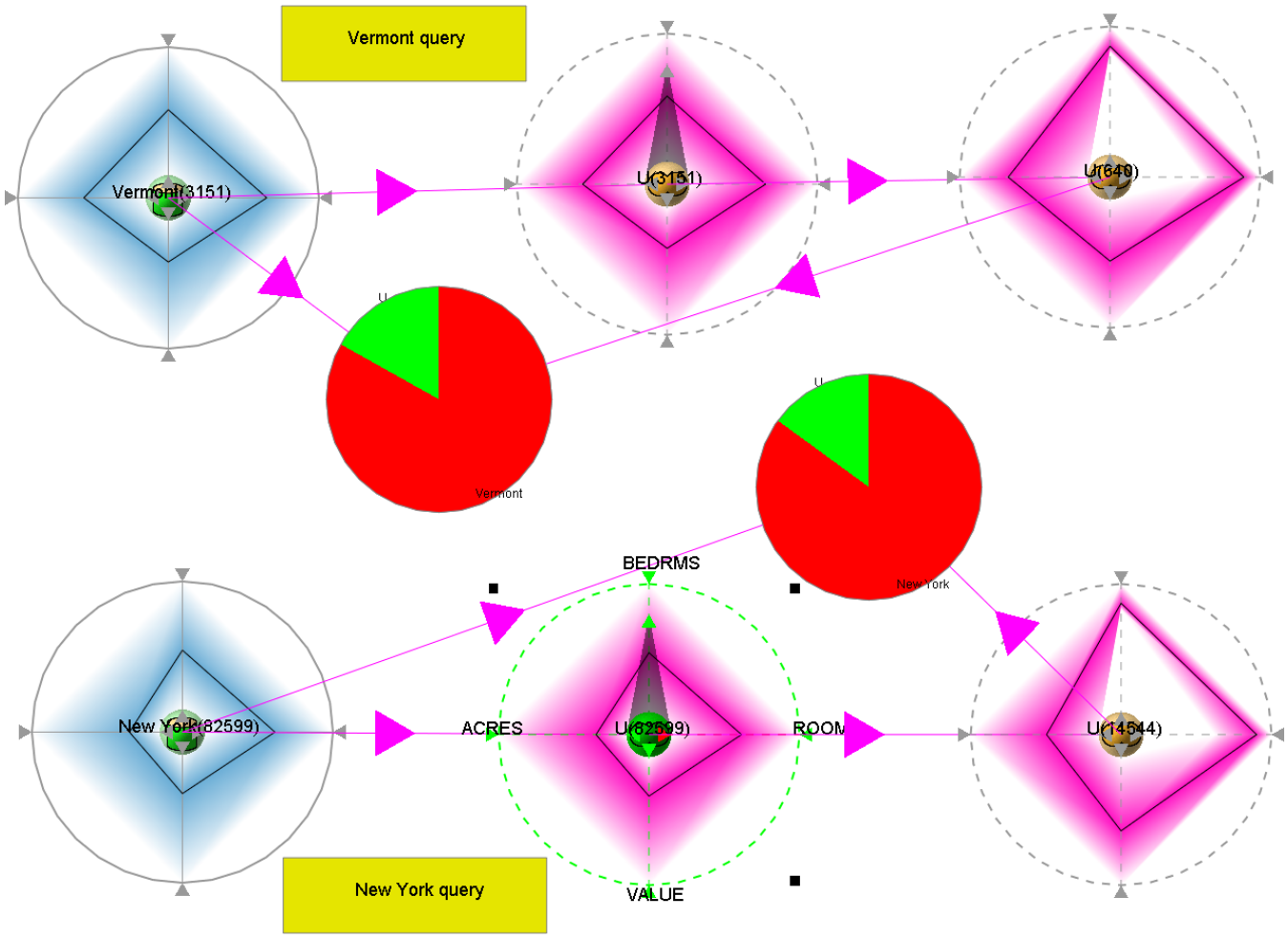


Figure 8: Comparing person data for houses of Vermont (upper branch) and New York (lower branch).

and up). Birdwatching is only a pastime for Anna, so she decides that she does not want to spend more than $ 500. Finally, since she is not interested in lugging around a large camera in the forest, she puts the upper limit on weight to be 500 grams.

Figure 10 shows Anna's visual query. The resulting DataRose yields 14 different cameras that satisfy her demands, and drilling down into the data gives her a table of these cameras. She finally settles for an Olympus SP-550 Ultra Zoom, a 7.1 megapixel compact SLR-like camera with 18x (504 mm) optical zoom weighing in at 460 grams and a price around $ 449.

7 User Study

We conducted a qualitative expert review on our prototype implementation. Our goal was to explore the capabilities of the method and get an idea of its utility. The study involved two visualization researchers from the field. Neither of the two had prior knowledge of the tool.

7.1 Procedure

We structured our expert review based on the US Census 2000 Public Use Microdata Sample (PUMS) dataset and a set of questions to drive the visual exploration. Only four out of fifty available states in the PUMS dataset were included in the study. In total, there were nine open-ended questions divided into three different groups (inspired by the conceptual levels for

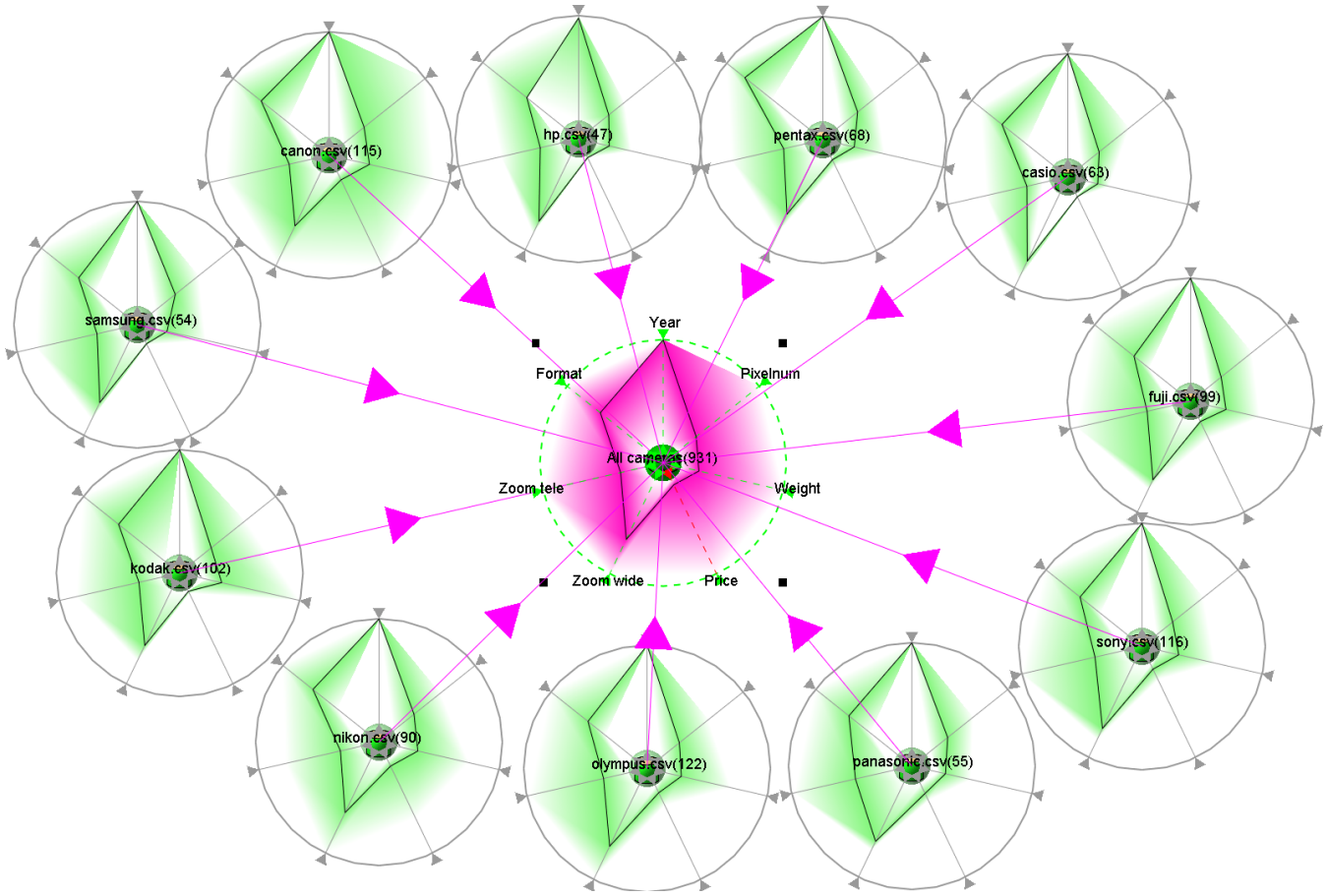


Figure 9: Union DataRose (center) of 931 digital cameras from 11 different manufacturers.

situation awareness [11]): direct facts, comprehension, and extrapolation (Table 1). In this way, we hoped to be able to evaluate all aspects of our method.

Conceptual level	**Example**
direct facts	What is the average house value in Georgia?
comprehension	Which state has the highest ratio of small and expensive houses?
extrapolation	Is there a relation between fuel type and building size in Alaska?

Table 1: Categories and examples of open-ended questions about the US Census 2000 data employed in the expert review.

Our two experts were introduced to the DataMeadow using a tour of the system in which the experimenter showed its main features and analysis methods. This tour lasted ten minutes. After that, the participants were allowed to familiarize themselves with the application. This session typically lasted ten minutes as well.

During the solving of the nine questions on the US Census dataset, the participants were instructed to follow a think-aloud protocol. The experimenter collected observations and comments by taking notes. Each evaluation session lasted around one hour in total. At the end, we conducted an informal interview about their experience using the tool.

7.2 Results and Discussion

We intentionally designed our nine questions to be of an open-ended nature—we were not interested in quantitatively recording the performance of our experts, but rather to have them exercise all parts of the system and get their feedback on its utility. Still, both participants were able to arrive at answers that they were satisfied with to all questions.

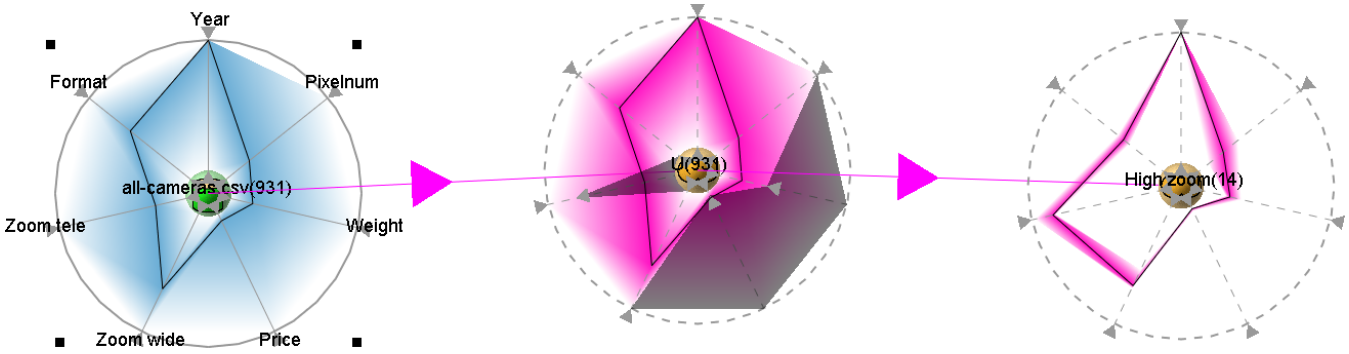


Figure 10: Visual query for the digital camera dataset for a high-zoom and inexpensive camera weighing less than 500 grams.

The participants liked the free-form type of interaction and both remarked it was a good match to how one might think about the analysis process. Being able to filter in situ on the dimension axes themselves corresponded well to how the participants perceived multidimensional filtering. The ability to "play" with the filter settings at different levels in a dependency chain was often used to both form hypotheses and to inform the next line of reasoning. This was also one of the key strengths of the platform that both participants emphasized, both while performing the analysis as well as in the debriefing interview.

Both participants thought that the opacity bands representation was the most efficient for general analysis. In some cases involving the distribution of mostly nominal data (e.g. fuel type), the color histogram was used. Participants remarked that this representation was often too dark because the data was often distributed rather evenly across the dimensions, resulting in only the lower half of most color scales to be used. None of the participants really liked the parallel coordinate representation, remarking that it "showed too much" for the analysis task they were doing.

Some improvements that were pointed out were to include viewers with logarithmic scales to avoid one dataset dwarfing another, to be able to copy query filter settings from one rose to another, and to be able to set color scales for individual data roses.

None of our participants during the session used mouse gestures or keyboard shortcuts despite having seen them demonstrated during the training session. We believe this is due to the low visibility of this kind of shortcuts, and that this might change for an expert-level user.

8 Conclusions and Future Work

This paper presents a visual analytics method called the DataMeadow for reasoning about multiple large-scale sets of multidimensional data. The primary user task supported by the method is comparison, a high-level meta-task that requires a considerable number of low-level user tasks such as retrieve value, correlation, and filtering. The method consists of an exploratory 2D canvas and individual datasets called DataRoses. DataRoses are variable-dimension starplots that employ a visual multivariate data representations to visualize the data distribution along the coordinate axes being displayed. To summarize, the contributions of this paper are the following:

- a highly interactive canvas (the DataMeadow) for multivariate data analysis;
- a visual representation (the DataRose) based on axis-filtered parallel coordinate starplots that can be linked together to form complex and dynamically-updated visual queries; and
- results from a user study indicating that our method is a useful way to reason about and query multivariate data.

In the future, we expect to integrate additional visual representations into the DataMeadow. Another interesting approach would be the use of both non-standard input devices (e.g. stylii and pen-based interfaces) and output devices (large displays) for the application. We are also currently planning an in-depth longitudinal field study using the tool with real analysts attempting to solve real problems.

References

[1] Christopher Ahlberg, Christopher Williamson, and Ben Shneiderman. Dynamic queries for information exploration: An implementation and evaluation. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 619–626. ACM Press, 1992.

[2] Robert A. Amar, James Eagan, and John T. Stasko. Low-level components of analytic activity in information visualization. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 111–117. IEEE Computer Society Press, 2005.

[3] Richard A. Becker and William S. Cleveland. Brushing scatterplots. *Technometrics*, 29(2):127–142, 1987.

[4] Richard A. Becker, William S. Cleveland, and Ming-Jen Shyu. The visual design and control of trellis display. *Journal of Computational and Graphical Statistics*, 5(2):123–155, 1996.

[5] Fabian Bendix, Robert Kosara, and Helwig Hauser. Parallel sets: A visual analysis of categorical data. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 133–140. IEEE Computer Society Press, 2005.

[6] Susan E. Brennan, Klaus Mueller, Greg Zelinsky, IV Ramakrishnan, David S. Warren, and Arie Kaufman. Toward a multi-analyst, collaborative framework for visual analytics. In *Proceedings of the IEEE Symposium on Visual Analytics Science & Technology*, pages 129–136. IEEE Computer Society Press, 2006.

[7] Dominique Brodbeck and Luc Girardin. Visualization of large-scale customer satisfaction surveys using a parallel coordinate tree. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 197–201. IEEE Computer Society Press, 2003.

[8] Herman Chernoff. Using faces to represent points in *k*–dimensional space graphically. *Journal of the American Statistical Association*, 68:361–368, 1973.

[9] William S. Cleveland. *Visualizing Data*. Hobart Press, Summit, NJ, USA, 1993.

[10] William S. Cleveland and Marylyn E. McGill, editors. *Dynamic Graphics for Statistics*. Statistics/Probability Series. Wadsworth & Brooks/Cole, Pacific Grove, CA, USA, 1988.

[11] Mica R. Endsley, Betty Bolté, and Debra G. Jones. *Designing for Situation Awareness: An Approach to User-Centered Design*. CRC Press, 2003.

[12] Jean-Daniel Fekete. The InfoVis Toolkit. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 167–174. IEEE Computer Society Press, 2004.

[13] Ying-Huey Fua, Matthew O. Ward, and Elke A. Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. In *Proceedings of the IEEE Conference on Visualization*, pages 43–50. IEEE Computer Society Press, 1999.

[14] David Gotz, Michelle X. Zhou, and Vikram Aggarwal. Interactive visual synthesis of analytic knowledge. In *Proceedings of the IEEE Symposium on Visual Analytics Science & Technology*, pages 51–58. IEEE Computer Society Press, 2006.

[15] Tovi Grossman, Pierre Dragicevic, and Ravin Balakrishnan. Strategies for accelerating on-line learning of hotkeys. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 1591–1600. ACM Press, 2007.

[16] Jeffrey Heer and Maneesh Agrawala. Software design patterns for information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):853–860, 2006.

[17] Ed Huai hsin Chi, Phillip Barry, John Riedl, and Joseph Konstan. A spreadsheet approach to information visualization. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 17–24. IEEE Computer Society Press, 1997.

[18] Alfred Inselberg. The plane with parallel coordinates. *The Visual Computer*, 1(2):69–91, 1985.

[19] Alfred Inselberg. Multidimensional detective. In *IEEE Symposium on Information Visualization*, pages 100–107. IEEE Computer Society Press, 1997.

[20] Brian Johnson and Ben Shneiderman. Tree maps: A space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the IEEE Conference on Visualization*, pages 284–291. IEEE Computer Society Press, 1991.

[21] Daniel A. Keim. Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):59–78, January/March 2000.

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

[23] Daniel A. Keim, Ming C. Hao, Umeshwar Dayal, and Meichun Hsu. Pixel bar charts: a visualization technique for very large multi-attribute data sets? *Information Visualization*, 1(1):20–34, 2002.

[24] Daniel A. Keim and Hans-Peter Kriegel. VisDB: Database exploration using multidimensional visualization. *IEEE Computer Graphics and Applications*, 14(5):40–49, September 1994.

[25] Daniel A. Keim, Florian Mansmann, Jorn Schneidewind, and Hartmut Ziegler. Challenges in visual data analysis. In *Proceedings of the Tenth International Conference on Information Visualization*, pages 9–16, 2006.

[26] Jeffrey LeBlanc, Matthew O. Ward, and Norman Wittels. Exploring N-dimensional databases. In *Proceedings of the IEEE Conference on Visualization*, pages 230–237. IEEE Computer Society Press, 1990.

[27] Haim Levkowitz and Gabor T. Herman. Color scales for image data. *IEEE Computer Graphics and Applications*, 12(1):72–80, January 1992.

[28] Eun Ju Nam, Yiping Han, Klaus Mueller, Alla Zelenyuk, and Dan Imre. ClusterSculptor: A visual analytics tool for high-dimensional data. In *Proceedings of the IEEE Symposium on Visual Analytics Science & Technology*, pages 75–82. IEEE Computer Society Press, 2007.

[29] Geraldine E. Rosario, Elke A. Rundensteiner, David C. Brown, Matthew O. Ward, and Shiping Huang. Mapping nominal values to numbers for effective visualization. *Information Visualization*, 3(2):80–95, 2004.

[30] Jinwook Seo and Ben Shneiderman. Interactively exploring hierarchical clustering results. *IEEE Computer*, 35(7):80–86, 2002.

[31] Hidekazu Shiozawa, Ken ichi Okada, and Yutaka Matsushita. 3D interactive visualization for inter-cell dependencies of spreadsheets. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 79–82. IEEE Computer Society Press, 1999.

[32] Ben Shneiderman. Tree visualization with treemaps: a 2-D space-filling approach. *ACM Transactions on Graphics*, 11(1):92–99, January 1992.

[33] Ben Shneiderman. Dynamic queries for visual information seeking. *IEEE Software*, 11(6):70–77, November 1994.

[34] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Symposium on Visual Languages*, pages 336–343. IEEE Computer Society Press, 1996.

[35] Mark Sifer. User interfaces for the exploration of hierarchical multi-dimensional data. In *Proceedings of the IEEE Symposium on Visual Analytics Science & Technology*, pages 175–182. IEEE Computer Society Press, 2006.

[36] Chris Stolte, Diane Tang, and Pat Hanrahan. Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transcations on Visualization and Computer Graphics*, 8(1):52–65, 2002.

[37] Egemen Tanin, Richard Beigel, and Ben Shneiderman. Incremental data structures and algorithms for dynamic query interfaces. *SIGMOD Record*, 25(4):21–24, December 1996.

[38] Roberto Theron. Visual analytics of paleoceanographic conditions. In *Proceedings of the IEEE Symposium on Visual Analytics Science & Technology*, pages 19–26. IEEE Computer Society Press, 2006.

[39] James J. Thomas and Kristin A. Cook, editors. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. IEEE Computer Society Press, 2005.

[40] Edward R. Tufte. *Envisioning Information*. Graphics Press, 1990.

[41] Fernanda B. Viégas and Martin Wattenberg. Communication-minded visualization: A call to action. *IBM Systems Journal*, 45(4):801–812, April 2006.

[42] Martin Wattenberg. Visual exploration of multivariate graphs. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 811–819. ACM Press, 2006.

[43] Stephen Wehrend and Clayton Lewis. A problem-oriented classification of visualization techniques. In *Proceedings of the IEEE Conference on Visualization*, pages 139–143. IEEE Computer Society Press, 1990.

[44] Hao wei Hsieh and Frank M. Shipman III. Manipulating structured information in a visual workspace. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 217–226. ACM Press, 2002.

[45] Hao wei Hsieh and Frank M. Shipman, III. VITE: A visual interface supporting the direct manipulation of structured data using two-way mappings. In *Proceedings of the International Conference on Intelligent User Interfaces*, pages 141–148. ACM Press, 2000.

[46] Christopher Williamson and Ben Shneiderman. The dynamic HomeFinder: Evaluating dynamic queries in a real-estate information exploration system. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM Press, pages 338–346, 1992.

[47] William Wright, David Schroh, Pascale Proulx, Alex Skaburskis, and Brian Cort. The sandbox for analysis: Concepts and evaluation. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, ACM Press, pages 801–810, 2006.

[48] Zaixian Xie, Shiping Huang, Matthew O. Ward, and Elke A. Rundensteiner. Exploratory visualization of multivariate data with variable quality. In *Proceedings of the IEEE Symposium on Visual Analytics Science & Technology*, pages 183–190. IEEE Computer Society Press, 2006.

[49] Di Yang, Elke A. Rundensteiner, and Matthew O. Ward. Analysis guided visual exploration of multivariate data. In *Proceedings of the IEEE Symposium on Visual Analytics Science & Technology*, pages 83–90. IEEE Computer Society Press, 2007.

[50] 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 of Visualization and Computer Graphics*, 13(6), 2007.

[51] Ji Soo Yi, Rachel Melton, John Stasko, and Julie Jacko. Dust & Magnet: Multivariate information visualization using a magnet metaphor. *Information Visualization*, 4(4):239–256, 2005.