Hierarchical Parallel Coordinates for Exploration of Large Datasets

Ying-Huey Fua, Matthew O. Ward and Elke A. Rundensteiner
Computer Science Department
Worcester Polytechnic Institute
Worcester, MA 01609
{yingfua,matt,rundenst}@cs.wpi.edu *

Abstract

Our ability to accumulate large, complex (multivariate) data sets has far exceeded our ability to effectively process them in search of patterns, anomalies, and other interesting features. Conventional multivariate visualization techniques generally do not scale well with respect to the size of the data set. The focus of this paper is on the interactive visualization of large multivariate data sets based on a number of novel extensions to the parallel coordinates display technique. We develop a multiresolutional view of the data via hierarchical clustering, and use a variation on parallel coordinates to convey aggregation information for the resulting clusters. Users can then navigate the resulting structure until the desired focus region and level of detail is reached, using our suite of navigational and filtering tools. We describe the design and implementation of our hierarchical parallel coordinates system which is based on extending the XmdvTool system. Lastly, we show examples of the tools and techniques applied to large (hundreds of thousands of records) multivariate data sets.

Keywords: Large-scale multivariate data visualization, hierarchical data exploration, parallel coordinates.

1 Introduction

As data sets become increasingly large and complex we require more effective ways to display, analyze, filter and interpret the information contained within them. Continuously increasing data set sizes challenges fundamental methods that have been designed and conceptually verified on small or moderate sized sets. This challenge manifests itself in methods across many fields, from computational complexity to database organization to the visual presentation and exploration of data. The latter is the subject of this paper.

A multivariate data set consists of a collection of N-tuples, where each entry of an N-tuple is a nominal or ordinal value corresponding to an independent or dependent variable. Several techniques have been proposed to display multivariate data. We broadly categorize them as:

- Axis reconfiguration techniques, such as parallel coordinates [10, 27] and glyphs [2, 4, 23].

- Dimensional embedding techniques, such as dimensional stacking [16] and worlds within worlds [6].

- Dimensional subsetting, such as scatterplots [5].

- Dimensional reduction techniques, such as multidimensional scaling [20, 15, 29], principal component analysis [12] and self-organizing maps [14].

Most of these techniques do not scale well with respect to the size of the data set. As a generalization, we postulate that any method that displays a single entity per data point invariably results in overlapped elements and a convoluted display that is not suited for the visualization of large data sets. The quantification of the term "large" varies and is subject to revision in sync with the state of computing power. For our present application, we define a large data set to contain 10^6 to 10^9 data elements or more.

Our research focus extends beyond just data display, incorporating the process of data exploration, with the goal of interactively uncovering patterns or anomalies not immediately obvious or comprehensible. Our goal is thus to support an active process of discovery as opposed to passive display. We believe that it is only through data exploration that meaningful ideas, relations, and subsequent inferences may be extracted from the data. The major hurdles we need to overcome are the problems of display density/clutter (too much data at once tends to confuse viewers) and intuitive navigation (what tasks comprise a typical exploration process, and how they can be made intuitive).

In this paper, we focus on the interactive visualization of large multivariate data sets using the parallel coordinates display technique. We propose a hierarchical approach that presents a multiresolutional view of the data, with navigation and filtering tools to facilitate the systematic discovery of data trends and hidden patterns. Our implementation is based on XmdvTool [25, 19], a public-domain visualization system that integrates multiple techniques for displaying and visually exploring multivariate data.

2 Related Work

In recent years several research efforts have been directed at the display of large multivariate data sets.

One approach is to use compression techniques to reduce the data set size while preserving significant features. For example, Wong and Bergeron [31] describe the construction of a multiresolution display using wavelet approximations, where the data size is reduced through repeated merging of neighboring points. The wavelet transform identifies averages and details present at each level of compression. However, the transform requires the data to be ordered, making it useful only for data sets with a natural ordering, such as time-series data.

Another approach is to let the characteristics of the data set reveal itself. For example, Wegman and Luo [28] suggest over-plotting translucent data points/lines so that sparse areas fade away while

dense areas appear emphasized. The disadvantage of this method is that it relies on overlapping points/lines to identify clusters. Clusters without overlapping elements will not be visually emphasized.

Keim et al. [13] studied pixel-level visualization schemes which permit the display of a large number of records on a typical workstation screen based on recursive layout patterns. However, the number of displayable records is dependent on the size of the display area. This limitation restricts the scalability of their method. Moreover, since each pixel only represents one variable, it is difficult to convey the interactions among variables.

Wills [30] describes a visualization technique for hierarchical clusters. His approach expands upon the tree-map idea [24] by recursively subdividing the tree based on a dissimilarity measure. However, the main purpose is to display the clustering results, and in particular, the data partitions at a given dissimilarity value.

Our research draws on several of the ideas found in the above work. As in [31], we store and present our data at multiple resolutions. However, to overcome the data ordering limitation of wavelets, we use clustering and partitioning techniques. We also use the opacity of lines as in [28] to reduce clutter. However, rather than conveying data density with overlapping lines, we use data aggregation techniques to collapse data into clusters, and show the population and extents of clusters with bands of varying translucency.
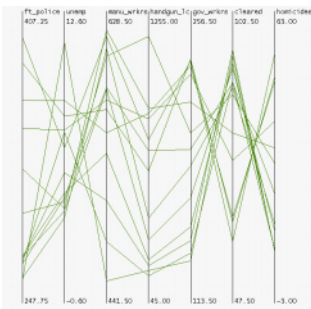
3 Parallel Coordinates

Figure 1: Parallel coordinates of Detroit homicide data set: a 7-dimensional data set with 13 records. Notice that there are inverse correlations between the number of cleared homicides and both the number of government workers and the total number of homicides.

We have chosen parallel coordinates as the visualization technique to extend upon to support large scale data. Parallel coordinates is a technique pioneered in the 1980's which has been applied to a diverse set of multidimensional problems [10, 27]. It has since been incorporated into many commercial and public-domain systems, such as WinViz [17], XmdvTool [25, 19], and SPSS Diamond (http://www.spss.com/software/diamond).

In this technique, each data dimension is represented as a (horizontal or) vertical axis, and the N axes are organized as uniformly spaced lines. A data element in an N-dimensional space is mapped to a polyline that traverses across all of the axes crossing each axis at a position proportional to its value for that dimension.

Parallel coordinates have a distinct advantage over conventional orthogonal coordinates. By laying out the vertical axes horizontally across the screen, the number of dimensions that can be visualized is restricted only by the horizontal resolution of the screen. This is in contrast to multivariate visualization in orthogonal coordinates where previous work [21] has attempted to augment each spatial point with a vector of values, usually with some visual icon that encodes the values. It is clear that with such an encoding scheme, one

soon runs out of encoding possibilities as the number of dimensions increases. The issue of dimensionality never arises in parallel coordinates, though as the axes get closer it may become more difficult to perceive structures or data relations. Moreover, using parallel coordinates, we can easily spot correlations between variables in the data set (see Figure 1).

The main difficulty of directly applying parallel coordinates to large data sets is that the level of clutter present in the visualization reduces the amount of useful information one can perceive. For example, the display of a mass of overlapping lines precludes the perception of relative densities present in the data set (see Figure 2). Our approach reduces the amount of clutter by imposing a hier-
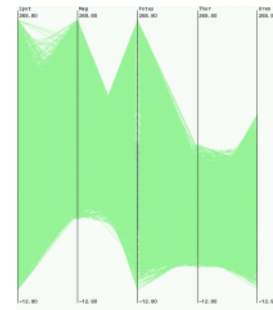

Figure 2: Parallel coordinates display of a Remote Sensing data set: a 5-dimensional data set with 16,384 records. Note the amount of over-plotting precludes the perception of any data trends, correlations or anomalies.

archical organization on the data set. We then display aggregations of the data at different levels of abstraction and provide tools for dynamically navigating and filtering the hierarchy, as described in detail in the following sections.

4 Overview of Hierarchical Parallel Coordinates

Exploratory data analysis is the summarization, display and manipulation of data to make it more comprehensible to human minds, thus uncovering underlying structure in the data and detecting important departures from that structure [3]. A complete data exploratory system thus has three major ingredients. Table 1 lists the three major components to our approach and their corresponding sub-components. The three basic components are: the summariza-

Summarization	Display	Manipulation/ Filtering
Hierarchical organization with statistical aggregation	Proximity-Based Coloring Translucency	Structure-based Brush Drill-Down/Roll-Up Dimension Zooming Extent Scaling Dynamic Masking

Table 1: Basic Components of the proposed Hierarchical Parallel Coordinates Approach

tion of the data by imposing a hierarchical structure on the data set, a scheme for displaying N-dimensional aggregate information and a set of tools for navigating, manipulating and filtering the hierarchical structure. We shall describe each of these components in the following sections.

5 Hierarchical Clustering

Our primary purpose for building a cluster hierarchy is to structure and present data at different levels of abstraction. A clustering algorithm groups objects or data items based on measures of proximity between pairs of objects [11]. In particular, a hierarchical clustering algorithm constructs a tree of nested clusters based on proximity information.

Let $\mathbf{E}$ be the a set of k N-dimensional objects, i.e.,

$$\mathbf{E} = \{e_1, e_2, e_3, ..., e_k\}$$

where e_i is an N-vector:

$$e_i = \{x_{i1}, x_{i2}, x_{i3}, ..., x_{iN}\}.$$

An m-partition $\mathbf{P}$ of $\mathbf{E}$ breaks $\mathbf{E}$ into m subsets $\{P_1, P_2, ..., P_m\}$ satisfying the following two criteria:

$$P_i \cap P_j = \emptyset \quad \text{for all } 1 \le i, j, \le m, i \ne j \text{ , and}$$
$$\bigcup_{i=1}^{m} P_i = \mathbf{E}$$

A partition $\mathbf{Q}$ is nested into a partition $\mathbf{P}$ if every component of $\mathbf{Q}$ is a proper subset of a component of $\mathbf{P}$. That is, $\mathbf{P}$ is formed by merging components of $\mathbf{Q}$. A hierarchical clustering is a sequence of partitions in which each partition is nested into the next partition in the sequence.

A hierarchical clustering may be organized as a tree structure: Let P_i be a component of $\mathbf{P}$, and $\mathbf{Q}$ be the m partitions of P_i. Let P_i be instantiated by a tree node T_i. Then, the components of $\mathbf{Q}$ form the children nodes of T_i.

We broadly categorize approaches that impose a hierarchical structure to a data set as either *explicit* or *implicit* clustering. In *explicit clustering*, hierarchical levels correspond to dimensions and the branches correspond to distinct values or ranges of values for the dimension. Hence, a different order of the dimensions give different hierarchical views. On the other hand, *implicit clustering* tries to group similar objects based on a certain metric, for instance the Euclidean distance.

There is a large body of literature on algorithms for the computation of implicit clusters [11]. The particular method used for the tree construction is however not relevant to this paper. Any method that builds a tree which abides by the above definitions could in principle be used as the tree construction scheme in our system.

However, most clustering algorithms are not appropriate for large data sets because of large storage and computation requirements. In recent years, a number of algorithms for clustering large data sets have been proposed [1, 9, 32]. We adopt one of these, namely the Birch algorithm [32], as our primary clustering technique, although our visualization techniques would work equally well with data clustered by other methods.

6 Visualizing Clusters

Each node T_i in a hierarchical cluster tree $\mathbf{T}$ represents a nested collection of enclosed data points or sub-clusters. At each node, we maintain summary information of all points and sub-clusters rooted from it. The following information may be directly obtained from T_i.

- n_i : the number of data points enclosed.

- m_i : the mean of the data points.

- B_i : the extents, i.e. the minimum and maximum bounds of the cluster for each dimension.

- v_i : a measure of the size of cluster T_i

- l_i : the tree depth at node T_i

v_i is a computed measure of a cluster size and satisfies the following criteria: If T_i is an ancestor of T_j, then

$$v_i > v_j.$$

The value of v_i is directly dependent on the shape of the clusters produced by the clustering algorithm. For spherical clusters, v_i may be the radius of a cluster. For rectangular clusters, v_i may be the N-dimensional volume of the cluster.

We propose to represent the information at a node by making use of *variable-width opacity bands*. Figure 3 shows a graduated band faded from a dense middle to transparent edges that visually encodes the information for a cluster. The mean stretches across the middle of the band and is encoded with the deepest opacity, which is a function of the density of a cluster, defined as the ratio $\frac{n_i}{v_i}$. This allows us to differentiate sparse, broad clusters and narrow, dense clusters. The top and bottom edges of the band have full transparency. The opacity across the rest of the band is linearly interpolated. The thickness of the band across each axis section represents the extents of the cluster in that dimension.
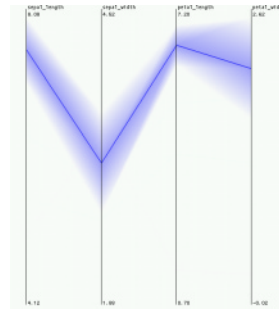

Figure 3: A single multi-dimensional graduated band that visually encodes information at a cluster node.

6.1 Multiresolutional Cluster Display

We define a horizontal cut $\mathbf{S}$ across a tree $\mathbf{T}$ as a boundary that divides $\mathbf{T}$ into a top half and a bottom half and satisfies the following criteria: for each path $\mathbf{R}$ from the root to a leaf, $\mathbf{S}$ intersects $\mathbf{R}$ at exactly one point.

Clearly $\mathbf{S}$ defines a partition of the data set $\mathbf{E}$. We may then vary the level-of-detail (LOD) in our data display by changing the parameters that control the location of $\mathbf{S}$.

Any variable that varies $\mathbf{S}$ is a candidate for the LOD control parameter. For instance, the tree depth is one conceivable discrete control parameter. However, it is a poor choice in some cases because the number of nodes may increase dramatically with depth. This would manifest itself as abrupt screen changes as the LOD switches values at higher depths of the tree.

We desire a continuous LOD control parameter that provides smooth transitions on our data display. We define:

$$v_{max} = \max_{T_i \in \mathbf{T}} \{v_i\}$$
$$v_{min} = \min_{T_i \in \mathbf{T}} \{v_i\}$$

We then choose $w \in [v_{min}, v_{max}]$ as the LOD control parameter. Define $S(w)$ as the collection of clusters whose size v_i is less than or equal to w but whose parent's size is greater than w. Then $S(w)$
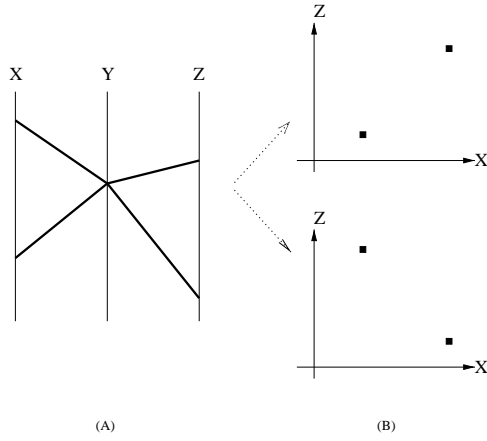
Figure 5: Ambiguous case in monochromatic parallel coordinates. (A) Two data points plotted in parallel coordinates. (B) Differing interpretation of the two points shown on the X-Z plane.

is a partition of **E** that satisfies our criteria for a continuous level-of-detail control parameter. Formally, we define $S(w)$ as:

$$S(w) \quad = \quad \{ \ T_i \ | \ (\ v_i \leq w \text{ or } T_i \text{ is a leaf node }) \text{ and}$$
$$v_{parent(i)} > w\}$$

Note that $S(v_{max})$ is a single partition comprising the entire **E**, while $S(v_{min})$ is a partition consisting of all the leaf nodes of **T**.

Figure 4 shows a series of images captured at six varying levels of abstraction (only 3 are shown in the Color Plates). The data set used is approximately 230,000 records of Fatal Accident Reports (8 variables shown) compiled by the National Highway Traffic Safety Administration National Center for Statistics and Analysis Accident Investigation Division.

6.2 Proximity-Based Coloring

Monochromatic line drawings present an inherent difficulty in parallel coordinates. Figure 5 shows a simple case where two three-dimensional data points give an ambiguous interpretation. This ambiguity arises whenever it is visually difficult to trace the topology of a data point as it traverses across the coordinate axes. This commonly occurs where data lines meet at axis lines.

One way to discriminate between such cases is through the use of color. It is easy to distinguish two intersecting data lines that have different colors. Ideally we wish to adopt a coloring scheme that assigns colors via a similarity measure. Data lines that are similar with respect to some measure should be in similar colors, whereas dissimilar data lines should be shown in contrasting colors.

Our method maps colors by cluster proximity, hence the name *proximity-based coloring*. This proximity is based on the structure of the hierarchical tree, that is sibling nodes are considered closer than non-sibling nodes. We first impose a linear order on the data clusters gathered for display at a given LOD value, w. These clusters are simply the partition elements $S(w)$ as described in the previous section. The elements of $S(w)$ are gathered in a recursive top-down manner using an in-order tree traversal.

Finally, we assign colors to each cluster by looking up a linear colormap table. Colors are assigned to clusters based on the following recursive formula:

$$C_0 \quad = \quad 0.5$$
$$C_i \quad = \quad C_{parent(i)} + \frac{\pi(i)}{K^{l_i+1}} \qquad (1)$$

where $C_i \in [0, 1]$ is the normalized color value of node T_i, and C_0 is the color of the root. We currently use C_i as the hue component of an HSV colormap. K is the branching-factor of the cluster tree, l_i is the tree depth at node T_i, and $\pi(i)$ is the sign function defined as:

$$\pi(i) \quad = \quad \begin{cases} +1 & \text{if i is odd} \\ -1 & \text{if i is even} \end{cases} \qquad (2)$$

Equation (1) colors clusters based on the cluster order derived during the tree traversal. The color ranges assigned by Equation (1) are nested just like clusters are nested, meaning larger clusters are assigned a broader range of color values and smaller clusters are assigned narrower ranges. Since small clusters imply that elements are closer to each other, they are assigned closer color values on the narrower color range. Equation (1) satisfies our definition of proximity-based coloring.

The equation however does not differentiate between adjacent elements (with respect to the linear order) belonging to different subtrees. It is important to distinguish between such elements because such adjacent elements are deemed "significantly separated" according to our proximity measure. For this, we revise Equation (1) by introducing a "buffer" between subtrees. The buffer acts as an unused color interval between subtrees so that elements at the proximal ends of subtrees are not assigned colors that are indistinguishable. Clearly the buffer should be larger between large subtrees and smaller otherwise.

Let b, where $b < 1$, be the desired buffer interval. Let the revised definition be:

$$C_i \quad = \quad C_{parent(i)} + \pi(i) \left(b^{l_i} + \frac{1}{K^{l_i+1}} \right) \qquad (3)$$

Equation (3) achieves our desired purpose. We typically choose b to be small with values around 10^{-1}.

Proximity-based coloring highlights the relationships among clusters. Consider the first image on Figure 6 (see Color Plates) which shows the Iris data set [7] without proximity coloring, and the second image which shows the same data set with proximity coloring. By comparing the two images, it is clear that coloring aids immensely in discerning meaningful patterns. In this example, three distinct clusters are apparent, as concentrations of blue, green, and pink cluster trends.

It is however not always possible to impose a linear order on the data clusters. For instance, a cluster chain forming a circular loop is not amenable to any linear order. In this case, an arbitrary break must be made at some point in the loop. Data elements at the break point, though similar according to our proximity measure, may be assigned contrasting colors.

7   Navigation and Filtering Tools

So far, we have structured the data by imposing a hierarchy upon it and have described a technique for displaying the data at a given level-of-detail. In this section, we describe the set of manipulation and filtering tools that allow us to interactively modify the display in order to discover new or hidden relationships in the data set.

7.1   Structure-Based Brushing

Brushing, in the context of multivariate visualization, refers to an interactive process for localizing a subset of a data set [19, 31, 28]. Many useful operations, such as highlighting, deleting, masking, or aggregation, may then be performed on elements that lie within the brushed subspace.

Brushing is a direct and data-driven metaphor. The operation may be performed in 2-D screen space, e.g., via methods such
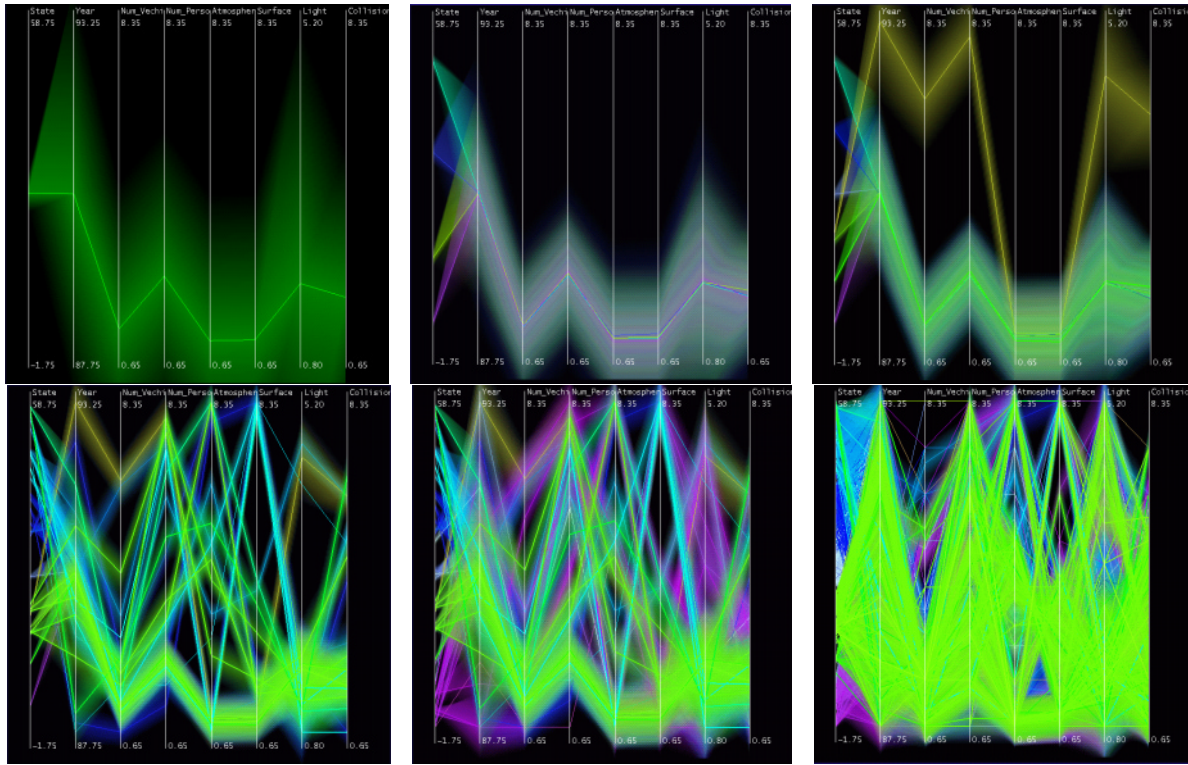
Figure 4: This image sequence shows a Fatal Accident data set of 230,000 data elements at different level of details. The first image shows a cut across the root node. The last image shows the cut chaining all the leaf nodes. The rest of the images show intermediate cuts at varying levels-of-detail. (See Color Plates).
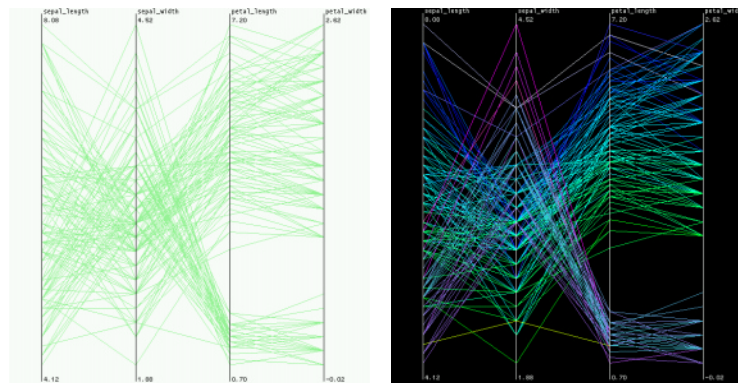

Figure 6: Left image shows Iris data set without proximity-based coloring. Right image shows Iris data set with proximity-based coloring revealing three distinct clusters depicted by concentrations of blue, green and pink lines. (See Color Plates).

as rubber-banding rectangles or mouse lasso operations. Brushing may also be performed in N-D data space by interactive creation of N-D hyperboxes by painting over data points of interest.
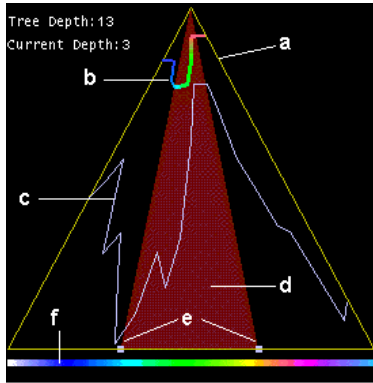

Figure 7: Structure-based brushing tool. (a) Hierarchical tree frame; (b) Contour corresponding to current level-of-detail; (c) Leaf contour approximates shape of hierarchical tree; (d) Structure-based brush; (e) Interactive brush handles; (f) Colormap legend for level-of-detail contour.

We introduce a new variant of brushing that we have developed as a general mechanism for navigating in hierarchical space called *structure-based brushing* (see Figure 7). Details of the structure-based brush can be found in [8].

The triangular frame depicts the hierarchical tree. The leaf contour depicts the silhouette of the hierarchical tree. It delineates the approximate shape formed by chaining the leaf nodes. The colored bold contour across the middle of the tree delineates the tree cut $S(w)$ that represents the cluster partition corresponding to a level-of-detail w (Section 6.1). The colors on the contour correspond to the colors used for drawing the nodes on the main parallel coordinates display (Section 6.2). The two movable handles on the base of the triangle, together with the apex of the triangle, form a wedge in the hierarchical space.

The brushing interaction for the user consists of localizing a subspace within the hierarchical space by positioning the two handles at the base of the triangle. The embedded wedge forms a brushed subspace within the hierarchical space. Elements within the brushed subspace may be examined at different level-of-detail (Section 7.2), or magnified and examined in full view (Section 7.3), or masked or emphasized using fading in/out operations (Section 7.5).

7.2   Drill-down and Roll-up

The two basic hierarchical operations when displaying data at multiple levels of aggregation are the "drill-down" and "roll-up" operations. Drill-down refers to the process of viewing data at a level of increased detail, while roll-up refers to the process of viewing data with decreasing detail.

Our system provides smooth and continuous level-of-detail control in all drilling operations. The control parameter is based on a measure of cluster size (Section 6.1). The level-of-detail can be varied indirectly using a slider or directly by adjusting the colored contour across the cluster tree.

We couple our drilling operations with brushing. Our system permits selective drill-down and roll-up of the brushed and non-brushed region independently. This flexibility is important as it allows the viewing of a subset of elements in varying levels of detail in relation to elements outside the subset.

7.3   Dimension Zooming

In parallel coordinates, the brushed subspace appears as a confined strip across the coordinate axes. For a narrow brush, it may be difficult to examine the data within this confined strip. To be able to study elements within a subspace and explore its interesting characteristics, it is essential that we treat this subset of elements as data in its own right, and place them in full view so that they can be examined as appropriate.

The use of distortion techniques [18, 22] has become increasingly common as a means for visually exploring dense information displays. Distortion operations allow the selective enlargement of subsets of the data display while maintaining context with surrounding data. We introduce a distortion operation that we term *dimension zooming*. We scale up each of the dimensions independently with respect to the extents of the brushed subspace, thus filling the display area. The subset of elements may then be examined as an independent data set. This zooming operation may be performed as many times as desired. For a data set occupying a large range of values, this operation is invaluable for examining localized trends.

One common problem with such scaling operations is that it is easy to lose context of the big picture, and be lost wandering in some isolated subspace. To maintain contextual information, we display a mini-map showing the position of the currently zoomed space in relation to the entire data space. Figure 8 shows an instance of this zooming operation and the accompanying mini-map. As an additional mechanism for context preservation, we animate the zooming process, which shows both the differences in scaling across the dimensions as well as the effects on data points neighboring the brushed area.

7.4   Extent Scaling

Where there are overlapping bands, it is often difficult to isolate or tell them apart. Our system overcomes this difficulty by allowing the thickness of bands to be scaled uniformly via a dynamically controlled scale factor. With this feature we can, for example, reveal the relative sizes of the extents while reducing occlusions (see Figure 9).

7.5   Dynamic Masking

Another tool for managing the complexity of a dense display is a process we call *dynamic masking*. This involves controlling the relative opacity between brushed and unbrushed areas. With dynamic masking, the viewer can interactively fade out the unbrushed nodes, thereby obtaining a clearer view of the brushed nodes. Conversely, the brushed nodes can be faded out, thus obtaining a clearer view of the unbrushed region. Hence, context is maintained while reducing clutter (see Figure 10).

8   Conclusion and Future Work

This paper describes hierarchical enhancements to the parallel coordinates visualization technique to facilitate the exploration of very large multivariate data sets. There are three main contributions of our general approach to hierarchical visualization on parallel coordinates. First, our cluster-based hierarchical enhancements provide a multiresolution view of the data and aid in revealing data trends at different degrees of summarization. Second, our proximity-based coloring scheme assures that data and clusters from similar parts of the hierarchical structure are shown in similar colors. The color scheme not only has a visual impact, but also aids in direct data selection by color. Third, we augment our system with a set of
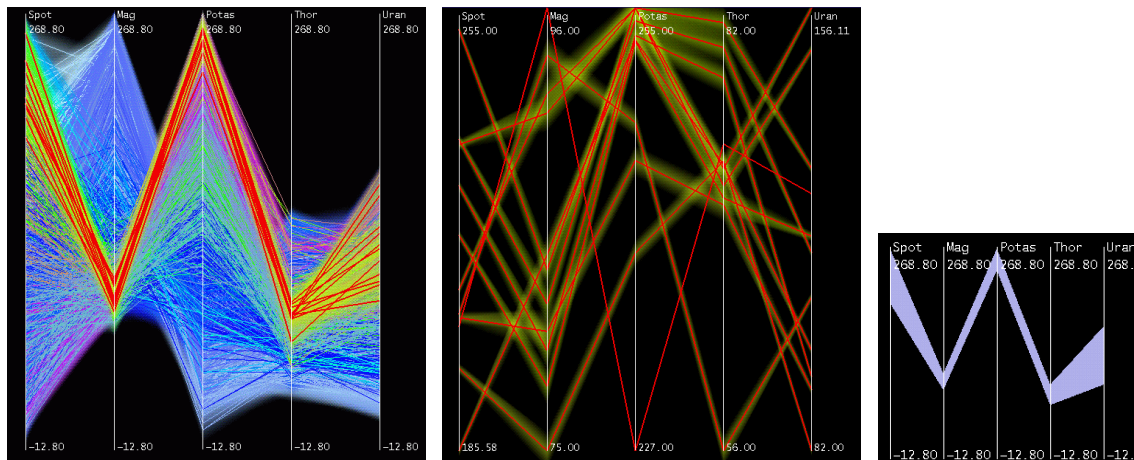
Figure 8: The image in the middle shows a magnified view of the brushed region indicated by the red lines in the leftmost image and an accompanying mini-map that captures the location of the brush with respect to the entire data space. (See Color Plates).
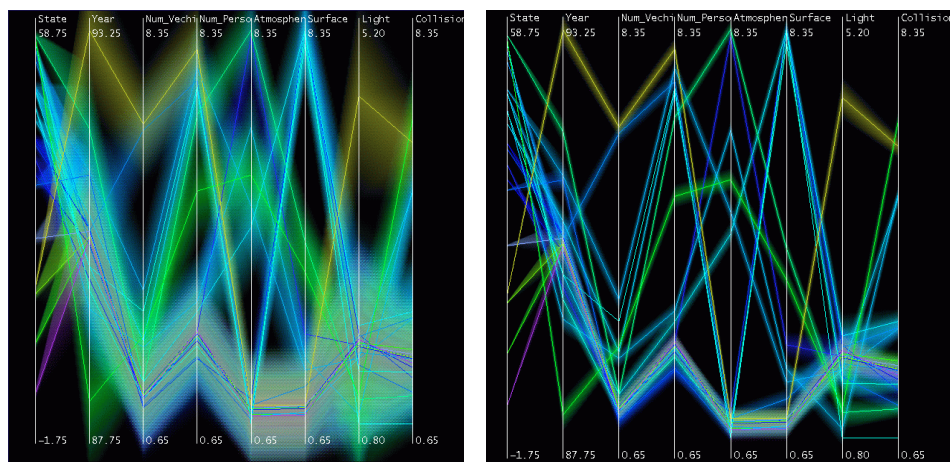

Figure 9: The left image shows a view of the Fatal Accident data set without extent scaling. Notice that the overlapping bands make it difficult to gauge the relative extent of the bands, as opposed to the same data set at the right image but with extent scaling.
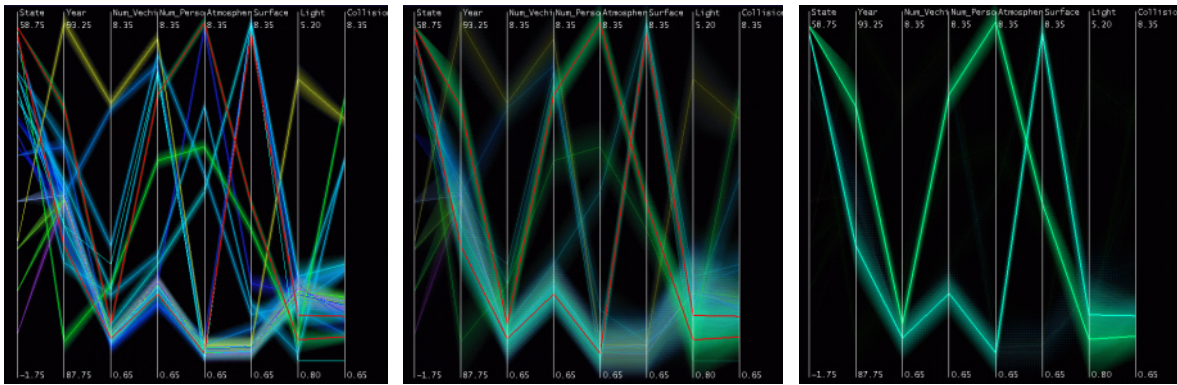
Figure 10: The image on the left shows the Traffic Safety data set displayed without dynamic masking. The middle image shows partial fading. The image on the right shows the effect of complete fading, with the unselected nodes faded out. The nodes are drawn with their actual encoded colors to better distinguish them. Notice the increased clarity in the right image. We observe that both of these clusters have accidents that involved not more than two vehicles and took place during lighted conditions. However, they differ widely in the number of persons involved and the kind of atmospheric conditions under which the accidents occurred.

navigation tools to support data localization and subspace drilling operations while maintaining context within the data space.

The ideas in this paper are implemented as OpenGL extensions to XmdvTool 3.1. The source code and video clips highlighting the operations supported will be made available to the public domain in the near future (see http://davis.wpi.edu/~xmdv).

This work is part of ongoing research at WPI focusing on multivariate visualization of large data sets. Our future undertakings include extending the hierarchical methods to other visualization modes in XmdvTool, including scatterplots, glyphs, and dimensional stacking. This will be done in a unified manner to maintain consistency across all display modes. In particular, we are interested in studying whether the navigation/interaction tools we have developed for parallel coordinates will apply across other visualization techniques. We are also investigating effective database management strategies within a large-scale multivariate visualization setting, including innovative indexing schemes and query optimizations to maximize performance for interactive data exploration tasks. Finally, we plan to expand our work on perceptual benchmarking for multivariate data visualization [26] to focus on assessing the effectiveness of various techniques when dealing with large data sets.

References

[1] P. Andreae, B. Dawkins, and P. O'Connor. Dysect: An incremental clustering algorithm. *Document included with public-domain version of the software, retrieved from Statlib at CMU*, 1990.

[2] D. Andrews. Plots of high dimensional data. *Biometrics, Vol. 28, p. 125-36*, 1972.

[3] D. Andrews. Exploratory data analysis. *International Encyclopedia of Statistics, p. 97-107*, 1978.

[4] H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association, Vol. 68, p. 361-68*, 1973.

[5] W. Cleveland and M. McGill. *Dynamic Graphics for Statistics*. Wadsworth, Inc., 1988.

[6] S. Feiner and C. Beshers. Worlds within worlds: Metaphors for exploring n-dimensional virtual worlds. *Proc. UIST'90, p. 76-83*, 1990.

[7] R. Fisher. The use of multiple measures in taxonomic problems. *Annals of Eugenics 7, p. 179-88*, 1936.

[8] Y. Fua, M. Ward, and E. Rundensteiner. Navigating hierarchies with structure-based brushes. *Proc. of Information Visualization '99*, Oct. 1999.

[9] S. Guha, R. Rastogi, and K. Shim. Cure: an efficient clustering algorithm for large databases. *SIGMOD Record, vol.27(2), p. 73-84*, June 1998.

[10] A. Inselberg and B. Dimsdale. Parallel coordinates: A tool for visualizing multidimensional geometry. *Proc. of Visualization '90, p. 361-78*, 1990.

[11] K. Jain and C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[12] J. Jolliffe. *Principal of Component Analysis*. Springer Verlag, 1986.

[13] D. Keim, H. Kriegel, and M. Ankerst. Recursive pattern: a technique for visualizing very large amounts of data. *Proc. of Visualization '95, p. 279-86*, 1995.

[14] T. Kohonen. The self-organizing map. *Proc. of IEEE, p. 1464-80*, 1978.

[15] J. Kruskal and M. Wish. *Multidimensional Scaling*. Sage Publications, 1978.

[16] J. LeBlanc, M. Ward, and N. Wittels. Exploring n-dimensional databases. *Proc. of Visualization '90, p. 230-7*, 1990.

[17] H. Lee and H. Ong. Visualization support for data mining. *IEEE Expert Vol. 11(5), p. 69-75*, 1996.

[18] Y. Leung and M. Apperley. A review and taxonomy of distortion-oriented presentation techniques. *ACM Transactions on Computer-Human Interaction Vol. 1(2), June 1994, p. 126-160*, 1994.

[19] A. Martin and M. Ward. High dimensional brushing for interactive exploration of multivariate data. *Proc. of Visualization '95, p. 271-8*, 1995.

[20] A. Mead. Review of the development of multidimensional scaling methods. *The Statistician, Vol. 33, p. 27-35*, 1992.

[21] G. Nielson, B. Shriver, and L. Rosenblum. *Visualization in Scientific Computing*. IEEE Computer Society Press, 1990.

[22] R. Rao and S. Card. Exploring large tables with the table lens. *Proc. of ACM CHI'95 Conference on Human Factors in Computing Systems, Vol. 2, p. 403-4*, 1995.

[23] W. Ribarsky, E. Ayers, J. Eble, and S. Mukherjea. Glyphmaker: Creating customized visualization of complex data. *IEEE Computer, Vol. 27(7), p. 57-64*, 1994.

[24] B. Shneiderman. Tree visualization with tree-maps: A 2d space-filling approach. *ACM Transactions on Graphics, Vol. 11(1), p. 92-99*, Jan. 1992.

[25] M. Ward. Xmdvtool: Integrating multiple methods for visualizing multivariate data. *Proc. of Visualization '94, p. 326-33*, 1994.

[26] M. Ward and K. Theroux. Perceptual benchmarking for multivariate data visualization. *Proc. Dagstuhl Seminar on Scientific Visualization*, 1997.

[27] E. Wegman. Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association, Vol. 411(85), p. 664*, 1990.

[28] E. Wegman and Q. Luo. High dimensional clustering using parallel coordinates and the grand tour. *Computing Science and Statistics, Vol. 28, p. 361-8.*, 1997.

[29] S. Weinberg. An introduction to multidimensional scaling. *Measurement and evaluation in counseling and development, Vol. 24, p. 12-36*, 1991.

[30] G. Wills. An interactive view for hierarchical clustering. *Proc. of Information Visualization '98, p. 26-31*, 1998.

[31] P. Wong and R. Bergeron. Multiresolution multidimensional wavelet brushing. *Proc. of Visualization '96, p. 141-8*, 1996.

[32] T. Zhang, R. Ramakrishnan, and M. Livny. Birch: an efficient data clustering method for very large databases. *SIGMOD Record, vol.25(2), p. 103-14*, June 1996.