Scattering Points in Parallel Coordinates

Xiaoru Yuan, *Member, IEEE*, Peihong Guo, He Xiao, Hong Zhou, and Huamin Qu, *Member, IEEE*

Abstract—In this paper, we present a novel parallel coordinates design integrated with points (Scattering Points in Parallel Coordinates, SPPC), by taking advantage of both parallel coordinates and scatterplots. Different from most multiple views visualization frameworks involving parallel coordinates where each visualization type occupies an individual window, we convert two selected neighboring coordinate axes into a scatterplot directly. Multidimensional scaling is adopted to allow converting multiple axes into a single subplot. The transition between two visual types is designed in a seamless way. In our work, a series of interaction tools has been developed. Uniform brushing functionality is implemented to allow the user to perform data selection on both points and parallel coordinate polylines without explicitly switching tools. A GPU accelerated Dimensional Incremental Multidimensional Scaling (DIMDS) has been developed to significantly improve the system performance. Our case study shows that our scheme is more efficient than traditional multi-view methods in performing visual analysis tasks.

Index Terms—Parallel Coordinates, Scatterplots, Information Visualization, Multidimensional Scaling.

1 Introduction

Recent advances in computing science and technology have witnessed an accelerating information explosion. Data with unprecedentedly large size and high dimensionality poses a major challenge to visualization researchers, demanding the provision of effective algorithms and tools. Many techniques have been proposed for exploratory visualization of multidimensional data. Parallel coordinates scheme, introduced by Inselberg and Dimsdale [24, 25], represents an N-dimensional data tuple as one polyline crossing parallel axes. For a large multidimensional data set, parallel coordinates can turn the tuples into a compact two-dimensional visual representation. However, data cluttering in parallel coordinates is almost unavoidable due to line overdrawing on limited screen space. Compared with point representation, each data item in parallel coordinates is drawn as one polyline crossing all dimensional axes, which occupies many more pixels. Due to the cluttering effect and interference with crossing lines, operation of data selection or clustering on parallel coordinates is not trivial when the data density is high.

Scatterplot matrix [12] is another frequently applied multidimensional visualization method. In a scatterplot, two datums of an individual in a data set are used to plot a point in two dimensional space, usually in a Cartesian coordinate system defined by two perpendicular axes, resulting in a scattering of points. The positions of the data points represent the corresponding dimension values. Scatterplots are useful for visually determining the correlation between two selected variables of a multidimensional data set, or finding distinct clusters of individuals in the data set. One single scatterplot can only depict the correlation between two dimensions. Additional limited dimensions can be mapped to the color, size or shape of the plotting points. For visualizing multidimensional data, a matrix consisting of N^2 scatterplots arranged in N rows and N columns is developed.

Compared with the scenario of parallel coordinates, points in scatterplots, instead of polylines in parallel coordinates, are the visual representation of the data. Point clouds consume fewer pixels compared with the same number of lines and it is easier for the user to detect clusters and perform selection by brushing. However, in contrast from parallel coordinates which can visualize all dimensions simultaneously, scatterplots can only display a very limited number of dimensions reliably. For higher dimensional tasks, multiple plots have to be drawn in the arrangement of a scatterplot matrix [12] with the data dimensions on the rows and columns. While a scatterplot matrix can give an overview of the structure of the whole data set, individual scatterplots in the matrix only appear as small image sets that are difficult to explore. Multidimensional scaling [40] can project high dimensional points directly into 2D at the expense of losing individual dimensional information and at high computational cost.

Besides the clutter problem, the interpretation of parallel coordinates requires expert knowledge. Due to the data entries being represented as line segments, visually clustered data has to be interpreted based on both the slope and intercept simultaneously, which is not trivial even to an experienced analyst. When it is required to check the data correlation between multiple dimensions, the task is even more challenging. In the contrast, when visually checking the spatial distribution of the points on scatterplots or multidimensional scaling plots, clusters of points are intuitively observed. As shown in Figure 1, data can have dual forms of representation in lines (left images) or points (right images) with different comprehensibility. While the data in Figure 1(a) can be easily understood, data correlation shown in the parallel coordinates form (Figure 1(b)) can not be trivially comprehended. In the scattered point form of Figure 1(b), the information of four clusters and their distribution patterns is visually clear. It would be difficult if not impossible to discern the few data points away from the four major clusters. Figure 1(c) is another example of data with similar characteristics.

Based on the above observation, a combination of parallel coordinates and scatterplots/multidimensional scaling could utilize the advantages of both. We note that efforts have been taken already to integrate multiple views of different visualization methods into a single system to facilitate data compressibility and exploration. In the majority of existing systems, the integration is done by linking the visual effects of different visual representations, while each representation takes its own window. For example, a 2D visualization system developed by Wong et al. [41] contains parallel coordinates and scatterplots, each presented in their own regions of the window. With linking and merging, interactive changes in one representation can be reflected in the others for better support during data exploration. In their system, users need to switch back and forth from one window to another and use their mental memory for data exploration and analysis.

In this work, we present a more radical design, Scattering Points in Parallel Coordinates (SPPC), that seamlessly integrates point representation into parallel coordinates. One example visualization on analyzing DNA microarray data with SPPC is shown in Figure 9. In the

- *Xiaoru Yuan, Peihong Guo and He Xiao are with the Key Laboratory of Machine Perception (Ministry of Education) and School of EECS, Peking University, Beijing, P.R. China. E-mail: {xiaoru.yuan, peihong.guo, xiaohe.pku}@pku.edu.cn*
- *Hong Zhou and Huamin Qu are with the Department of Computer Science and Engineering at Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong. E-mail: {huamin, hongzhou}@cse.ust.hk*

Manuscript received 31 March 2009; accepted 27 July 2009; posted online 11 October 2009; mailed on 5 October 2009.
For information on obtaining reprints of this article, please send email to: tvcg@computer.org .

design we propose, two or more selected coordinate dimensions are converted into point plots through multidimensional scaling as shown in Fig 2 (b). Scatterplots can be considered as a special case of multidimensional scaling, in which the protection domain and data domain are identical.

To avoid the context jump between polyline and point regions, curves connecting two neighboring polyline regions are drawn in the middle scattered point region. The curved lines are displayed in a consistent color scheme with neighboring polyline segments when the data trend through all dimension is examined. Further equipped with a specially designed uniform brushing tool, the user can freely explore and cluster high dimensional data without tool or context switching.

The specific benefits of SPPC and the contribution of this research are as follows:

- Unified Line/Point Representation: Any two or more dimensions in the parallel coordinate plots can be conveniently converted to scattered points through multidimensional scaling, or converted back in reverse.
- Uniform Brushing Tool: A brushing tool allows the user to make selection on both points and line segments conveniently.
- Dimensional Incremental Multidimensional Scaling (DIMDS): A GPU accelerated multidimensional scaling algorithm.

The remainder of this paper is organized as follows. Section 2 provides a review of related works. An overview of our proposed visualization system is presented in Section 3, followed by system details in Section 4. After implementation details are revealed in Section 5 and case experiments described in Section 6, conclusions and future work are presented in Section 7.

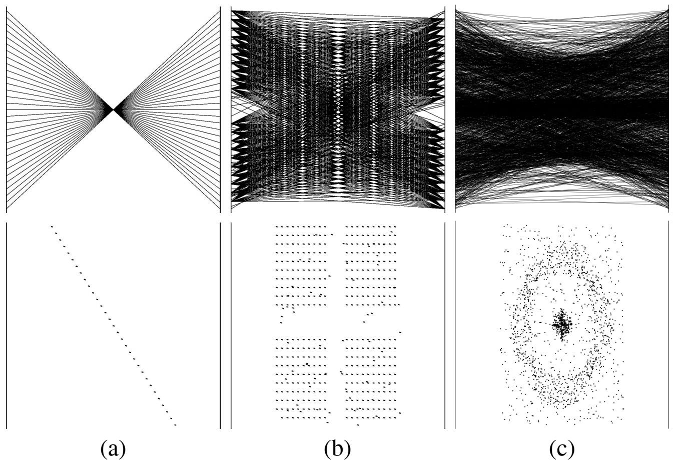
Fig. 1. Data displayed in the top row: line representation (parallel coordinates); bottom row: point representation (Scatterplots).

2 Related Works

Information visualization systems with multiple coordinated views have been considered to be effective for exploratory visualization of complex multidimensional data sets. Visualization techniques in combination can complement each other and help solve challenging problems. A set of guidelines on when and how multiple view systems should be used has been provided by Baldonado et al. [5]. In our work we have developed a system that integrates the point (e.g. multidimensional scaling [41] and scatterplots [12]) and line representations (e.g. parallel coordinates [24, 25]) of multidimensional data sets.

Parallel Coordinates In the design of parallel coordinates, a direct manipulation method developed by Siirtola [34] can dynamically summarize a set of polylines through averaging and visualize correlation coefficients between subsets. Wong and Bergeron [40] used wavelet approximation to create a brushing tool which displays the brushed and non-brushed data at different resolutions. Angular brushing [20] is effective in selecting data subsets which exhibit correlation along two axes. EdgeLens [39] can interactively curve graph edges away from the focus center while keeping the nodes intact. Zhou et al. [44] adjusted the shape of edges based on visual clustering. Animation can

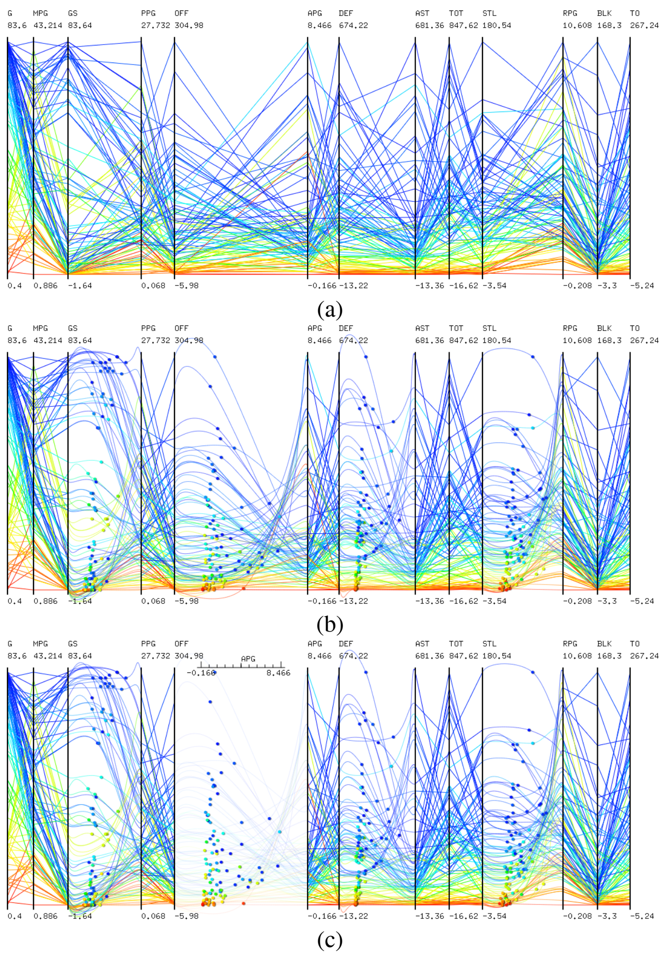

Fig. 2. (a) Traditional Parallel Coordinates; (b) Our proposed Scattering Points in Parallel Coordinates, (c) SPPC with background curves faded.

also be applied to reduce the clutter [43]. Theisel [36] replaced line segments with free-form curves to encode extra information. Curves can also be employed to enable crossed axis tracing [19] in parallel coordinates. By modifying the axes or the line segment representations, fuzzy data and categorical data can also be visualized by parallel coordinates [7, 8, 28]. Parallel coordinates can further be extended into 3D through extrusion to visualize trajectories of higher dimensional dynamical systems [37], or novel axes arrangement can be used to allow the simultaneous examination of the relationships of a single dimension with many others in the data [26]. Artistic rendering techniques can augment parallel coordinates to increase comprehensibility for non-experts [29].

Clutter Reduction In parallel coordinates, patterns are very often difficult to detect due to the visual clutter caused by too many drawn lines. Many efforts have been proposed to reduce the clutter and facilitate user exploration. Dimension reordering based on similarity helps visual clutter minimization [1, 32, 42]. Clustering is another type of approach to reduce clutter. Multiresolutional view of the data through hierarchical clustering [18] assisted with proximity-based coloring has been developed to show aggregation information. Visual abstraction, in the form of texture stripes with various opacity, has been used to distinguish different clusters [30]. Transfer functions, either pre-defined or customized, are provided to highlight different aspects of the cluster data characteristics [27]. Artero et al. [2] filtered out information by constructing frequency and density plots from the parallel coordinate plots. Clutter reduction can also be performed in a focus+context manner [15, 31].

Coordinated Views As one of the most widely used multidimensional data visualization techniques, parallel coordinates have been extensively studied as far as how to integrate with other visualization methods to overcome shortcomings and improve efficiency [9, 13, 17, 33, 35, 41]. SpringView [9] integrates parallel coordinates with Radviz [21] to handle multidimensional datasets. Siirtola [35] combined parallel coordinates with the Reorderable Matrix. In Parallel Glyphs [17], dimension axes of a parallel coordinate plot are extended into star glyphs in 3D by unfolding them around a pivot axis to facilitate data comparison and provide capabilities for interactive exploration.

Scatterplots As an alternative to parallel coordinates, scatterplots [12], more frequently in a form of a scatterplot matrix, depict discrete data values with two data variables as a collection of discrete points. This form can support better interactive navigation in multidimensional spaces [16] through displaying and considering transitions between different scatterplots as animated rotations in 3D space. Continuous scatterplots [4] have also been developed to visualize large scientific data. In the direction of coordinated view visualization, Schmid and Hinterberger [33] combined scatterplot matrix, parallel coordinates plot, permutation matrix, and Addrew's curve view together. A 2D visualization system developed by Wong et al. [41] contains parallel coordinates and scatterplots. Craig and Kennedy [13] studied the combination of a traditional time series graph representation with a complementary scatterplot representation. In the above efforts, each visualization metaphor is presented in their own region and the visualization exploration is performed through linking and merging so that the interactive changes of one representation can be reflected in the other.

3 Overview

The design of our proposed Scattering Points in Parallel Coordinates (SPPC) integrates the point representation into parallel coordinates closely.

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

$$\mathbf{X} = \{\mathbf{x}_m = (x_{m,1}, x_{m,2}, \cdots, x_{m,N})^T \,|\, 1 \le m \le M\} \tag{1}$$

where M is the number of data items, and N is the dimension of the data.

In parallel coordinates, data $\mathbf{x}_m$ is drawn as a series of line segments:

$$\mathbf{l}_m = \{l_{m,(0,1)}, l_{m,(1,2)}, \cdots, l_{m,(N-1,N)}\} \tag{2}$$

Each line segment $l_{m,(n,n+1)}$ connects points $x_{m,n}$ and $x_{m,n+1}$ on axes of A_n and A_{n+1} respectively, as shown in Figure 3(a). The slope of line segment $l_{m,(n,n+1)}$ is defined as $k_{m,(n,n+1)}$. Each two neighboring axes A_n and A_{n+1} define a region $R(n,n+1)$. In traditional parallel coordinates, M line segments, $\{l_{m,(n,n+1)}, |1 \le m \le M\}$ are displayed in $R(n,n+1)$. Figure 2 (a) is an example of traditional parallel coordinates. Figure 2 (b) is the result of the same data set visualized with SPPC. For the simplicity of our description, we assume no dimension reordering occurs in our following discussion unless explained explicitly.

In our design of SPPC, line segments of two or more selected coordinate dimensions can be converted into point plots in the form of scatterplots or a multidimensional scaling plot. In the simplest representation of two dimensional data, data on dimension n and $n+1$, e.g. line segments in region $R(n,n+1)$, are converted to M scattered points $\{p_{m,(n,n+1)} | 1 \le m \le M\}$. In our default setting, the dimension axis $n+1$ is rotated 90 degrees, forming a Cartesian region with dimension axis n. Note in our implementation, it is not necessary for the two dimensions of a point plot to be consecutive in the original dimension order. In a more general case, k dimensions, where $k > 1$, can be converted through multidimensional scaling, generating a point cloud distributed on a 2D plane, consisting of points $\{p_{m,(n,n+1,\ldots,n+k-1)} | 1 \le m \le M\}$.

To avoid the context jumps between polyline and point regions, curves connecting two neighboring axes are drawn in the point region, illustrated in Figure 3 as red lines. The curves can be displayed in a consistent color scheme with neighboring polyline segments so that the data trend can be continuously tracked throughout all dimensions. Further equipped with our proposed uniform brushing tool, the user can freely explore and cluster high dimensional data without switching between different tools and contexts. In the following sections, more details on the SPPC will be discussed.

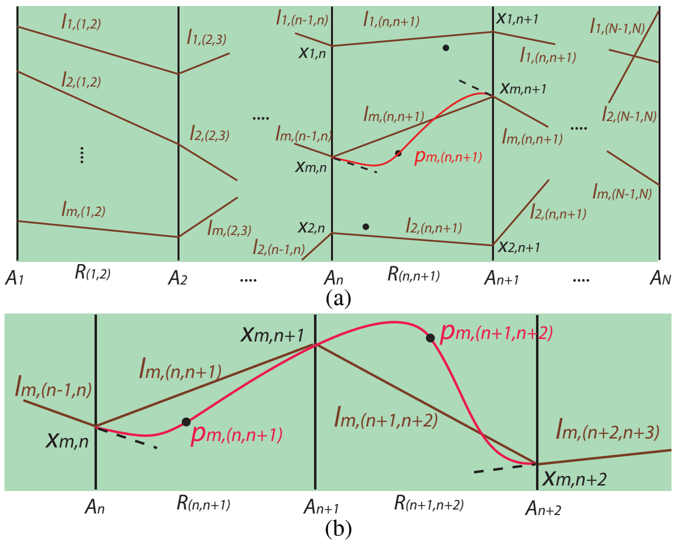

Fig. 3. Illustrations of Scattering Points in Parallel Coordinates. (a) Lines in $R_{(n,n+1)}$ are converted to points; (b) Lines in two consecutive regions $R_{(n,n+1)}$ and $R_{(n+1,n+2)}$ are converted.

4 Scattering Points in Parallel Coordinates

The Scattering Points in Parallel Coordinates (SPPC) is very flexible. In the following subsections, more details on our work will be given.

4.1 Converting Parallel Coordinates Segments to Point Plots

The simplest case of our work is to select a parallel coordinates region $R(n,n+1)$ and convert the line segments between $\{l_{m,(n,n+1)}, |1 \le m \le M\}$ into points $\{p_{m,(n,n+1)}, |1 \le m \le M\}$. A naive approach of such conversion can result in the original parallel coordinates being transformed into two parallel coordinates plots with a single scatterplot in between.

It is also possible to represent multiple dimensions in a single scattered point plot using multidimensional scaling [41].

The first step is to determine the dissimilarities between all pairs of data items. Euclidean distance in k-dimensional space is the most commonly used metric, although other metrics including weighted Euclidean, Minkowski, and Manhattan (a.k.a. city blocks) can be used to measure data dissimilarity of quantitative datasets. Using Euclidean distance, the dissimilarity, $\delta_{p_m,p_{m'}}$ between data m and m', on the dimension of $(n,n+1,\ldots,n+k-1)$ is given by

$$\delta_{p_{m,(n,n+1,\ldots,n+k-1)},p_{m',(n,n+1,\ldots,n+k-1)}} = \sqrt{\sum_{i=0}^{k-1}(x_{m,n+i} - x_{m',n+i})^2} \tag{3}$$

A dataset with N records generates an $N \times N$ real symmetric dissimilarity matrix $D(\delta_{p_m,p_{m'}})$. Each element of this matrix contains the dissimilarity, $\delta_{p_{m,(n,n+1,\ldots,n+k-1)},p_{m',(n,n+1,\ldots,n+k-1)}}$, between data item m and m' of the original data.

In our initial configuration, points are chosen as in the scatter plot of neighboring dimensions. A dissimilarity matrix, $D'(\delta'_{p_m,p_{m'}})$, of the initial configuration will be constructed for the evaluation of the configuration. The difference of the dissimilarity between the original data set and the configuration is defined as a difference matrix:

$$\Delta(\Delta_{p_m,p_{m'}}) = D(\delta_{p_m,p_{m'}}) - D'(\delta'_{p_m,p_{m'}}) \tag{4}$$

For a fast construction of a configuration with sufficient confidence of accuracy, a spring model [11] is adopted which searches for an answer by iteratively updating the positions of points in the configuration. In the spring model, a spring with a relaxed length of $\delta_{p_m,p_{m'}}$ connects point p and p'. All points in the configuration form a multibody system, which reaches an equilibrium after iterating for a sufficiently long period. A simple stress function is used to determine the terminating condition:

$$Stress = \sqrt{(\sum_{m=1}^{M} (\delta'^2_{p_m,p_{m'}} - \delta^2_{p_m,p_{m'}}))/(\sum_{m=1}^{M} \delta^2_{p_m,p_{m'}})} \quad (5)$$

Our method of constructing a 2D configuration to represent k-dimensional data directly applies the spring model to selected dimensions of the original data.

4.2 Dimensional Incremental Multidimensional Scaling

However, the above spring model based MDS is inefficient for large datasets owing to its $O(n^3)$ computational complexity. To alleviate the computation burden and enable dynamic adding or subtracting of dimensions in the k-dimensional sub dataset for configuration construction, Dimensional Incremental Multidimensional Scaling (DIMDS) method is introduced.

To reduce the computation load of MDS for large datasets, Basalaj [6] proposed an incremental algorithm. Large datasets are divided into several parts and added into the MDS configuration step by step. In other words, Basalaj's method focuses on the increment of the size of the dataset.

The approach we propose in this paper is adding or subtracting dimensions gradually into MDS configuration, given an MDS configuration with limited difference among the dimensions included. The DIMDS method is introduced for cases where an MDS configuration already exists, some dimensions are then eliminated or added to the current configuration. Based on our observation, in the system we constructed the user is more likely to add or remove dimensions from the current multidimensional projection of the point region, e.g. in an incremental way. Our proposed DIMDS takes advantage of this and updates the MDS configuration accordingly to obtain a performance gain.

Our approach shows great flexibility of creating MDS configurations as well as effectively reducing computation demands. Moreover, this method provides an overview of the full dataset at any time during the construction of an MDS configuration since all data points are taken into consideration, while the method proposed by Basalaj and Ingram et al. shows only part of the whole dataset before finishing the construction. Since the configuration may change dramatically when new dimensions are added, users may get confused by points moving swiftly. By properly choosing the initial condition, we are able to have good continuity between MDS configurations.

DIMDS constructs new configurations by adopting the existing configuration as the initial configuration and incrementally updating the dissimilarity matrix $D(\delta_{p_m,p_{m'}})$ with data from specified dimensions. Assuming that all data is normalized, in the case of changing one more dimension, $\mathbf{b} = (b_1, b_2, ..., b_N)$, a new dissimilarity matrix D_{new} can be constructed as:

$$\begin{aligned} D_{new} &= D + \delta D_{\mathbf{b}} \\ \delta D_{\mathbf{b}} &= \pm(\delta d_{ij}) \end{aligned} \quad (6)$$

where $\delta d_{ij} = (b_i - b_j)^2$ is the element of matrix D_{new} and the sign of $\delta D_{\mathbf{b}}$ is positive for additive cases and negative for subtractive cases. A normalization operation may be needed after adding dimensions. After the construction of a new dissimilarity matrix, the same iterative method is applied to build the new configuration.

In this incremental approach, the existing configuration serves as the initial configuration and the effect of dimensional modification is loaded into the configuration gradually in the iterative process. This guarantees a smooth transition from the previous configuration. Different from 2D scatter plots, theoretically, MDS projections into two dimensions are free of any particular orientation and may not even converge to a unique solution. DIMDS with different incremental orders may produce different final projections. However, according to our observation, in most cases the differences are small and do not affect the data exploration. As shown in Figure 4, DIMDS generates very similar point distribution patterns as MDS does. In addition, in SPPC, MDS is employed for clustering. As long as the generated graph gives similar clustering results, differences in orientation or distribution are not fatally critical.

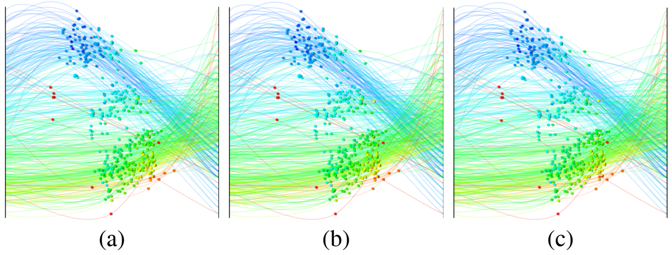
Fig. 4. Projection results of data with 7 dimensions by (a) MDS and (b), (c) DIMDS with two different dimension incremental orders.

4.3 Background Curves in Point Regions

Merely aligning the two visual forms of point and line regions together breaks the integration of the whole multidimensional dataset representation.

In the point region, in addition to the conversion of line segments between two neighboring axes to scattered points, curved lines are drawn to connect ployline segments in the neighboring regions and pass the plotted points in the middle. By introducing curved lines into the point region, our visualization unites two different visual forms into a single organic one.

We first consider the scatterplot case. As illustrated in Figure 3(a), each curved line $l_{m,(n,n+1)}$ corresponds to one point $p_{m,(n,n+1)}$. The following constrains are used for determining the shape of the curved line $l_{m,(n,n+1)}$:

1. line $l_{m,(n,n+1)}$ passes through the point $p_{m,(n,n+1)}$;
2. line $l_{m,(n,n+1)}$ connects points $x_{m,n}$ and $x_{m,n+1}$ on two neighboring coordinate axes correspondingly;
3. line $l_{m,(n,n+1)}$ connects smoothly with the line segments $l_{m,(n-1,n)}$ and $l_{m,(n+1,n+2)}$, e.g. $l_{m,(n-1,n)}$ and $l_{m,(n+1,n+2)}$ are the tangent lines to the curve $l_{m,(n,n+1)}$ at point $x_{m,n}$ and $x_{m,n+1}$ respectively.

A Catmull-Rom spline [10] is frequently used in computer graphics for curves or smooth interpolation. The spline passes through all of the control points. It has C^1 and G^1 continuity but not C^2 continuity. The second derivative is linearly interpolated within each segment so that the curvature varies linearly over the length of the segment. With the above desirable characteristics, a Catmull-Rom spline fits our requirement well. For each spline $l_{m,(n,n+1)}$, 5 control points $\{P_i, i \in [0,4]\}$ are defined: $P_0 = x_{m,n-1} + p_{m,(n,n+1)} - x_{m,n}$, $P_1 = x_{m,n}$, $P_2 = p_{m,(n,n+1)}$, $P_3 = x_{m,n+1}$ and $P_4 = x_{m,n+2} - p_{m,(n,n+1)} + x_{m,n+1}$.

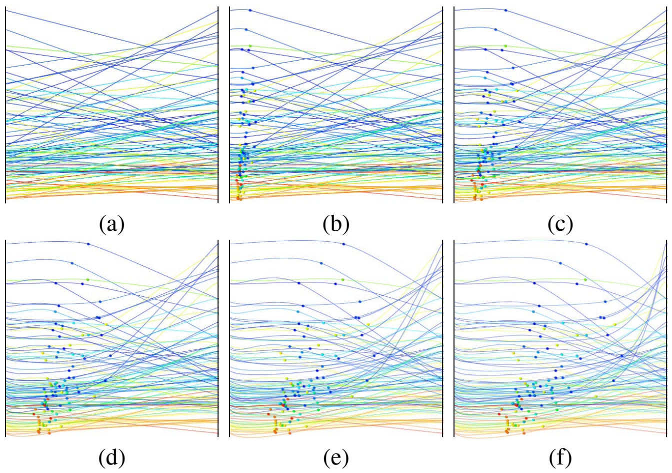
Fig. 5. Animating Points and Curves in SPPC. From (a) to (f), points are moving from the left axis to reach their final destination. The curves are animated accompanying the movement of the scattered points.

The setting of P_0 and P_4 are chosen to meet the tangent requirement. The tangent T_k of the constructed curve at each control point

Data	Dimensions	Data Size	CPU MDS	CPU DIMDS	GPU DIMDS	CPU DIMDS+	GPU DIMDS+
Car	8	406	5671ms	719ms (7.88)	503ms (11.3)	110ms (51.6)	92ms (61.6)
AAUP	14	1161	64312ms	1671ms (38.5)	862ms (74.6)	828ms (77.7)	593ms (108.5)
DNA	21378	68	1485ms	1453ms (1.02)	1043ms (1.42)	32ms (46.3)	81ms (18.3)

Table 1. Performance comparison of our GPU accelerated DIMDS with previous methods. The timing is measured for direct CPU MDS, DIMDS without and with GPU acceleration respectively. DIMDS+ refers to the time for computing one dimensional increment using DIMDS. The acceleration rate over CPU MDS is indicated in the parenthesis.

P_1 to P_3 is equal to $(P_{k+1} - P_{k-1})/2$, where $k = 1,2,3$. Note that for a Catmull-Rom spline, points on a line segment may lie outside the original parallel coordinate region. In our practise, only a few lines exhibit such a problem. It does not harm the visualization. The Catmull-Rom spline also works in the situation where two point regions are next to each other as shown in Figure 3(b).

The visualization we propose also has a smooth transition during the process of converting a line region into a point region by employing an animation. During the transition animation, each point $p_{m,(n,n+1)}$ moves from the left axis at position $x_{m,n}$ to its final location. At the beginning stage of the animation, P_1 and P_2 are very close to each other, resulting in a self-crossing spline. To remedy this problem, we use a Cardinal spline, which is a general form of a Catmull-Rom spline. The tangent m_k is defined as $(1-c)(P_{k+1}+P_{k-1})/2$. In our animation, the parameter c is linearly interpolated from 1 to 0 as the point moves from the left axis to its destination. When the point reaches its destination, $c = 0$, the Cardinal spline degenerates into a Catmull-Rom spline. Figure 5 shows a sequence of snapshots depicting the animation described above. Drawing too many background curves can cause the same clutter problem as the parallel coordinate lines do. Curves may also interfere the interpolation of the scattered points. To solve the above mentioned problems, we implemented one function that the user can optionally choose automatic fading out of the background curves when the mouse hovers on one point region as shown in Figure 2(c). The fading function is only active when the user performs operations on the scattered point regions. When the point regions shows only two dimensions, e.g. the 2D scatter plot case, our system can also display a horizonal x-axis with scale of the corresponding dimension at the same time when the fading function is effective.

5 Implementation Details

We implemented all visualization algorithms for our experiments on a Dell Precision T3400 desktop with Intel Core 2 Duo E7400 CPUs and 1GB Memory. The graphics card used is an NVIDIA Quadro NVS290 with 256MB of DDR2 memory. The software environment is based on Windows XP SP2 with Visual Studio 2005, and NVIDIA CUDA 2.0. An NVIDIA Tesla C870 PCI-E Card with 1.5GB memory is installed for GPU accelerated computation. In the following subsections, implementation details about GPU accelerated DIMDS and more specification on user interface design will be discussed.

5.1 GPU Accelerated DIMDS

We implement the DIMDS by taking advantage of GPU acceleration with CUDA [22]. Ingram et al. proposed a multi-level GPU MDS algorithm [23] based on a parallel force-based subsystem simulation. Their approach organizes datasets into a hierarchy of levels, and recursively constructs an MDS configuration. The multi-level procedure together with GPU acceleration shows great performance improvement over previous methods. However, their approach focuses on the whole dataset with all dimensions included. In the case of handling large datasets with very high dimensionality, it still suffers from extremely heavy computational costs.

Our implementation begins by constructing the MDS configuration from the scatter plot between neighboring axes. As the user drags more axes into the plot, more new dimensions are added into the dissimilarity matrix. The change in the dissimilarity matrix will then result in a change of MDS configuration. Since our construction of the new configuration uses the existing plot as the initial configuration and with the adoption of the spring model, points in the plot move smoothly starting from their previous balance positions until they finally reach an equilibrium.

Note that in each iteration of the spring model, the computation mainly focuses on creating an updated dissimilarity matrix and retrieving new position vectors from the matrices. As the computation for each point in the plot is relatively independent from other points, a GPU algorithm can achieve a significant speed increase by parallel processing the computation task of each point. In our implementation, each point is treated individually in the following procedures:

1. Calculate the current dissimilarity matrix $D'(\delta'_{p_m,p_{m'}})$ with all points' coordinates;
2. Calculate the difference matrix $\Delta(\Delta_{p_m,p_{m'}})$;
3. Calculate force matrices $F_x(f_{x,(p_m,p_{m'})})$ in x direction and $F_y(f_{y,(p_m,p_{m'})})$ in y direction, then merge the force matrices into two force vectors $\overrightarrow{F}_{x,p_m}$ and $\overrightarrow{F}_{y,p_m}$ for each point;
4. For all points, update their velocities (v_{x,p_m}, v_{y,p_m}) and coordinates (x_{p_m}, y_{p_m}) with force vectors $\overrightarrow{F}_{x,p_m}$ and $\overrightarrow{F}_{y,p_m}$.

Our algorithm can be summarized as follows:

1. Determine the initial configuration and initial dissimilarity matrix;
2. For all points, update their position with the above procedure;
3. Repeat step 2 until the convergence condition is reached or dimensional change is made. In the case of adding or subtracting dimensions, return to step 1.

We compare the performance between regular CPU based DMS, DIMDS, and GPU accelerated DIMDS by testing each visualization algorithm on three data sets with different sizes. All dimensions are included in the MDS or DIMDS computation. Our experiments show that our modification of MDS procedures yields remarkable speed improvements and provides satisfactory interactivity for users. As indicated in Table 1, DIMDS improves the performance over previous MDS methods when the data size is large.

In our SPPC, the most common situation is that the user adds dimensions into the point region one by one. Then there is only one dimension difference that needs to be computed. We name this situation as DIMDS+. The speedup rates range from 40-80 times, depending on the data size. The GPU accelerated DIMDS can give a further 1-3 times improvement. The total acceleration rate can reach as high as 108 times in our experiment. From the timing data, we notice that the data with larger sizes receive a high acceleration rate. Note that for the DNA data with very small data size but large dimension number, the GPU do not improve the performance too much, and even give lower frame rates for the DIMDS+ situation.

5.2 User Interface and Interaction Design

In addition to acceleration on the computational side, the SPPC visualization system has a carefully designed interface to facilitate usability and improve data exploration performance.

Point/Line Transition: In SPPC, a region can be switched between line and point forms back and forth simply by double clicking the mouse in the target area. During the transition, points emerge from the left axis of the specified region and move to their destination in the middle region through an animation. Corresponding curves are also drawn and animated accordingly. In the reverse process, the scattered points travel from the middle region to the left axis.

Dimension Operations: In SPPC, several operations on data dimensions can be performed. Users can reorder the coordinates by dragging the axes. The colored radial buttons at the bottom of the screen shown in Figure 6 indicate the type of the corresponding region. If two regions next to an axis are all in line form, the color of the radial button is blue. If a dimensional axis separates a point region

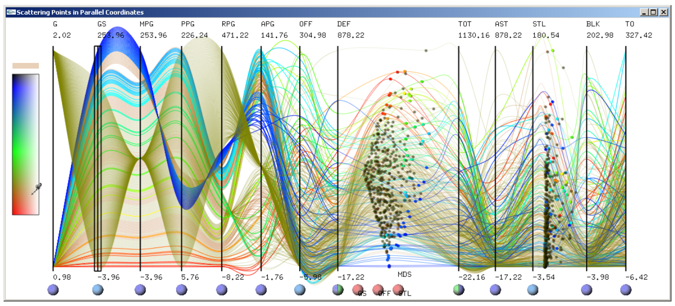
Scattering Points in Parallel Coordinates
G GS MPG PPG RPG APG OFF DEF TOT AST STL BLK TO
MDS

Fig. 6. Interface of Scattering Points in Parallel Coordinates.

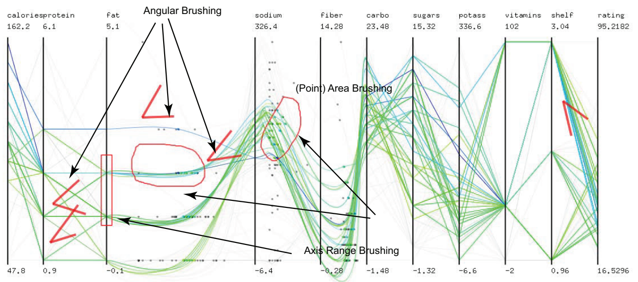

Fig. 7. Uniform Brushing in SPPC. Three types of brushes are enabled: Angular, Axis Range, and Point Region.

from a line region, its corresponding button is half in blue and half in red. For the MDS region, additional dimensions are indicated by a corresponding number of buttons. In Figure 6, the MDS region visualizes 9 dimensions. Users can add or remove dimensions of an MDS region by dragging the colored buttons into or away from the MDS region. By such an intuitive way of dimension manipulation, our system provides a very convenient way for the user to explore data correlation when the data involves large numbers of dimensions.

Focus+Context Operation: In our design, a user-customized region width between any two dimensions is allowed. When the mouse is placed over the focused region, scrolling the mouse wheel can interactively widen or narrow the region width, and the sizes of other regions are changed accordingly.

Uniform Brushing: To facilitate user interaction, we designed a uniform brushing tool that can be directly applied to both point and line regions in SPPC.

In the line regions, similar to general parallel coordinates, axis range and angular brushing are supported. Here we allow the user to directly apply raw sketching on the displayed visualization to take advantage of the flexibility of free-hand sketching. To select a range of axis values, the user can either directly brush on the targeted area of an axis with a line, or circle a desirable range. The system can automatically fit the drawing to the corresponding region and compute the range. Users can also apply angular brushing with a V-shape sketch gesture. Two lines are applied to fit the sketching input from the user. The formed angle defines the lines with the corresponding slope range to be selected. If sketching is applied in the point region, points in the area bounded by the input stroke are selected. The above scenario is illustrated in Figure 7.

Note that in our approach, there is no action needed for switching between range, angular, and point selection brushing. With our designed tool, the user can focus more on visual exploration with higher efficiency.

6 Case Study

In this section, we demonstrate the effectiveness of our SPPC visualization system through experiments on several datasets.

6.1 Car Data

We first tested SPPC on a car information dataset with 7 variables and 392 data items, which has been used in many publications. The relationship between "acceleration", "model year", "origin" and "number of cylinders" are explored in our system.

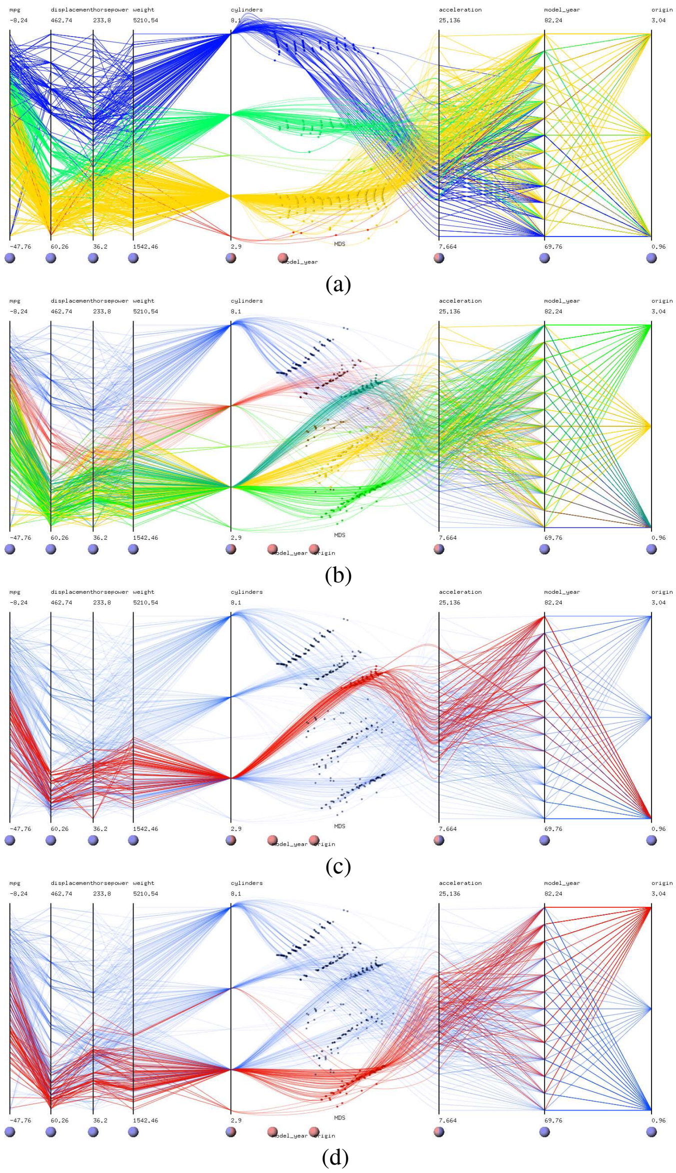

Fig. 8. Experiments on a dataset with 7 variables and 392 data items. Data from http://lib.stat.cmu.edu/datasets/cars.data.

Naturally, "number of cylinders" has an inverse correlation with the dimension "acceleration". When the dimension "model year" is introduced, as shown in Figure 8 (a), three major clusters can be clearly observed in the MDS point region. In Figure 8(b), after adding one more dimension "origin", the previous three clusters in Figure 8(a) split into five portions. With the assistance of the animated transition in our system, we can observe that the lowest cluster is split into three new clusters, and one new small cluster is formed by combining some data items from the originally blue and yellow clusters in Figure 8(a). Without the animation, such kinds of information are very difficult to obtain. This exploration process also indicates a hierarchical classification structure of this car data.

By further performing sketching on the resulting point plot, we can clearly view the properties of each group as shown in Figure 8. For example, the cluster shown in Figure 8(c) contains cars which are relatively new and all come from the USA. This indicates a trend of manufacturing cars with small numbers of cylinders in the USA, while other countries already have a longer history of making more economical cars. The highlighted clusters in Figure 8(d) provide a depiction of the car production in Europe in contrast of that in the USA as illustrated in Figure 8(c).

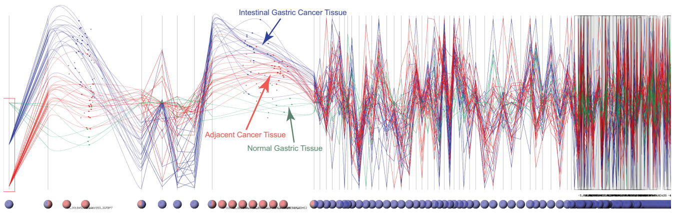

Fig. 9. Visualization of DNA Microarray data with Scattering Points in Parallel Coordinates (SPPC). Gene expression data of intestinal gastric cancer tissues (in blue) and adjacent cancer tissues (in red) from cancer patients, and normal gastric tissue (in green) from healthy people are separated.

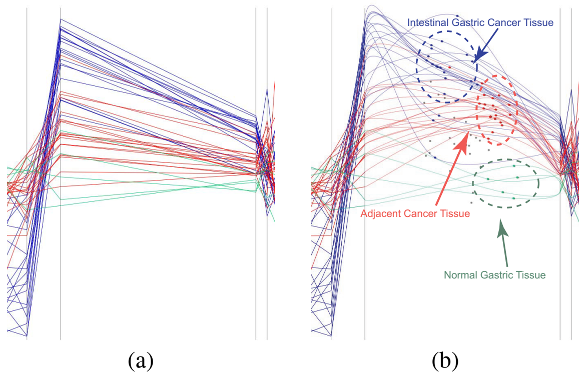

Fig. 10. DNA microarray data visualized by (a) parallel coordinates, (b) SPPC.

6.2 DNA Microarray Data Analysis with SPPC

We also applied our SPPC visualization scheme on microarray analysis. Identifying genes whose expressions are specifically altered in cancer cells is essential for early cancer diagnoses. Due to the enormous amount of data generated from DNA microarray experiments, the use of certain statistical analysis is very difficult. We analyze a typical DNA microarray data with our system.

The original dataset consists of expression profile data from two types of gastric cancer versus common reference using a 21,329-oligonucleotide microarray chip. The samples had been histologically confirmed and documented to be compatible age and gender. In our experiment, we focused on the comparison of data from the intestinal gastric cancer tissue against normal gastric tissue. The profile we studied includes the data of the intestinal gastric cancer tissues and adjacent cancer tissues from 20 patients, and normal tissues from 5 healthy persons.

The expression intensity of each gene is equal to the readout intensity value from the original microarray chip of the corresponding gene spot normalized by a common reference value. Expression intensities larger than 2 or smaller than 0.5 are considered to be significant in terms of difference against the common reference. Note expression intensities between the range of 0.5 and 2 are important. Only picking up genes with the highest expression intensities (value larger than 2 or smaller than 0.5) may not generate best classification of cancer and normal issues.

In Figure 9, a visualization result with SPPC is shown. Measurement of expression intensity of each gene can be treated as one dimension. Selected 100 dimensions (with high gene expression level) of the raw data are visualized simultaneously. In the middle point region, the three types of tissues, marked with blue, red and green color respectively are separated into three clusters with DIMDS. This DIMDS involves nine genes as the classifiers, including COL18A1, VSIG2, RGS1, BDKRB2, PLEKHG2, RDHE2, etc.

With our system, the user can quickly find out the best gene combination for distinguishing different types of tissues, e.g. identify the gene classifiers for possible early cancer diagnosis. Without the integrated environment of MDS with parallel coordinates and the enabled animation during dimension addition and substraction, the identification would take much longer if using traditional approaches. Comparing the enlarged image sets in Figure 10, with only two dimensions involved in Figure 10(a), it is difficult to discern the different tissue type, while SPPC works effectively in Figure 10(b). Note due to the ambiguous nature of the data, there are no sharp boundaries between each class. However, with the power of interactive exploration and the integrated visualization environment provided by SPPC, the task of identifying genes for cancer diagnosis becomes easier.

6.3 Discussion

From the experiments above we can see that our SPPC visualization scheme has some clear advantages. In some multidimensional data analysis tasks, showing a point distribution plot can improve data comprehensibility. It is also easier for a user to detect the clusters directly. Integration of point/line visualization is the main feature of our scheme that differentiates our system. This marks a difference from previous methods. Comparing with other multiple view systems, in which the user performs an operation and observes the consequence in different windows, our integrated system removes the memory boundary between two different visualization algorithms. With continuous curved lines, data trends can be tracked easily.

Animation is implemented in both processes of SPPC. First, in the point/line transition process, and second in the DIMDS dimension addition/subtraction. In both processes of SPPC, animation has been implemented, e.g. in the point/line transition process and in the DIMDS dimension addition/substraction process. The animation plays a critical role. By checking the dynamics during the transition, the user can understand the underlying relationship by interactively modifying the visualization conditions. Based on the feedback from a preliminary testing by several selected users, animation is one of the most important features that helps the users understand data.

Note that many techniques have been developed for parallel coordinates to help the user in exploring clusters and correlations in the data, (e.g. clustering techniques, aggregation techniques). They many not be compatible with the the point representation part of the SPPC approach. However, such techniques can still be applied on the line representation portion of the SPPC to facilitate data comprehension. Methods such as grand tour [3, 38] which examines structure of high dimensional data from all possible angles, and projection pursuit [14] which only shows important aspects of high dimensional space, can be applied to our system.

DIMDS creates new configurations starting from a scatterplot of neighboring axis whose *Stress* value is zero. The incremental process only increases *Stress* a little bit each time. This allows us to create configurations with very low *Stress* values within only a very few steps of iteration. Together with the improvement through the introduction of the GPU acceleration, the interactive DIMDS greatly helps the user with data exploration and leads to new discoveries in the data domain, which is very difficult to achieve with a non-interactive MDS system. The spring model in DIMDS, although having $O(n^3)$ complexity, presents a reversible feature in the process of incrementally creating an MDS configuration. That is, by subtracting the most recently added dimensions, we are able to come back to the previous state.

As mentioned earlier, scatterplots linking to parallel coordinates has been well established. Based on feedback, users prefer our approach, mainly because of SPPC reduces content switching which is unavoidable in the multiple coordinated windows approaches. Animation of the point/line transition and the automatic background curved line fading in/out also receives positive comments for their contribution to the integration of two visual metaphors.

7 Conclusions and Future Work

The visualization approach as presented in this paper shows that Scattering Points in Parallel Coordinates is promising to improve usability in visualizations of large multidimensional data. By closely integrating scatterplots or multidimensional scaling plots into the drawing of parallel coordinates, an efficient method of interacting with visualization systems is possible. With our seamless design of transition between parallel coordinate representation and scattered point representation, switching between the two visual forms requires very little cost. Unform brushing enables user navigation and selection throughout the multidimensional data visualization domain.

There are many algorithms that can project data from *n*-D to 2-D. We would like to investigate other projection methods, including PCA, in the future. In line regions, existing parallel coordinate algorithms can still be applied to improve the system. We would like to apply several parallel coordinate techniques in our system, e.g. edge clustering [44], illustrative parallel coordinates [29]. We would also like to further investigate how other multidimensional visualization forms can be closely integrated with parallel coordinates. Finally, a more thorough user study will be conducted to further investigate the usability of our system, especially the comparison between SPPC and traditional multiple coordinated windows methods.

Acknowledgments

This work is supported by Beijing Municipal Natural Science Foundation (No. 4092021), MOST 2009CB320903, RFDP 200800011004, Key Project of Chinese Ministry of Education (No. 109001) and HK RGC CERG 618706. We thank Dr. Youyong Lu from Peking University School of Oncology for providing the DNA microarray data. We thank Finlay Mungall for proofreading. We are grateful to the anonymous reviewers, whose valuable suggestions greatly improved this paper.

References

[1] M. Ankerst, S. Berchtold, and D. A. Keim. Similarity clustering of dimensions for an enhanced visualization of multidimensional data. In *Proceedings of the IEEE InfoVis'98*, pages 52–60, 1998.

[2] A. O. Artero, M. C. F. de Oliveira, and H. Levkowitz. Uncovering clusters in crowded parallel coordinates visualizations. In *Proceedings of the IEEE InfoVis'04*, pages 81–88, 2004.

[3] D. Asimov. The grand tour: a tool for viewing multidimensional data. *SIAM J. Sci. Stat. Comput.*, 6(1):128–143, 1985.

[4] S. Bachthaler and D. Weiskopf. Continuous scatterplots. *IEEE Trans. Vis. Comput. Graph.*, 14(6):1428–1435, 2008.

[5] M. Q. W. Baldonado, A. Woodruff, and A. Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of AVI'00*, pages 110–119. ACM, 2000.

[6] W. Basalaj. Incremental multidimensional scaling method for database visualization. In *Proceedings of Visual Data Exploration and Analysis VI, SPIE*, pages 149–158, 1999.

[7] F. Bendix, R. Kosara, and H. Hauser. Parallel sets: visual analysis of categorical data. In *Proceedings of the IEEE InfoVis'05*, pages 133–140, 2005.

[8] M. R. Berthold and L. O. Hall. Visualizing fuzzy points in parallel coordinates. *IEEE Trans. Fuzzy Sys.*, 11(3):369–374, 2003.

[9] E. Bertini, L. Dell'Aquila, and G. Santucci. SpringView: cooperation of radviz and parallel coordinates for view optimization and clutter reduction. In *Proceedings of CMV'05*, pages 22–29, Jul. 2005.

[10] E. Catmull and R. Rom. A class of local interpolating splines. In R. Barnhill and R. Riesenfe, editors, *Computer Aided Geometric Design*, pages 317–326, New York, 1974. Academic Press.

[11] M. Chalmers. A linear iteration time layout algorithm for visualising high-dimensional data. In *Proceedings of the IEEE Visualization'96*, pages 127–133, 1996.

[12] W. C. Cleveland and M. E. McGill. *Dynamic Graphics for Statistics*. CRC Press, Inc., Boca Raton, FL, USA, 1988.

[13] P. Craig and J. Kennedy. Coordinated graph and scatter-plot views for the visual exploration of microarray time-series data. In *Proceedings of the IEEE InfoVis'03*, pages 197–201, 2003.

[14] S. L. Crawford and T. C. Fall. Projection pursuit techniques for the visualization of high dimensional datasets. *Visualization in Scientific Computing*, pages 94–108, 1990.

[15] G. Ellis and A. Dix. Enabling automatic clutter reduction in parallel coordinate plots. *IEEE Trans. Vis. Comput. Graph.*, 12(5):717–724, 2006.

[16] N. Elmqvist, P. Dragicevic, and J.-D. Fekete. Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation. *IEEE Trans. Vis. Comput. Graph.*, 14(6):1539–1148, 2008.

[17] E. Fanea, S. Carpendale, and T. Isenberg. An interactive 3d integration of parallel coordinates and star glyphs. In *Proceedings of the IEEE InfoVis'05*, pages 149–156, 2005.

[18] Y.-H. Fua, M. O. Ward, and E. A. Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. In *Proceedings of the IEEE Visualization'99*, pages 43–50, 1999.

[19] M. Graham and J. Kennedy. Using curves to enhance parallel coordinate visualisations. In *Proceedings of the Intl. Conf. on Information Visualization*, pages 10–16, Jul. 2003.

[20] H. Hauser, F. Ledermann, and H. Doleisch. Angular brushing of extended parallel coordinates. In *Proceedings of the IEEE InfoVis'02*, pages 127–130, 2002.

[21] P. Hoffman, G. Grinstein, K. Marx, I. Grosse, and E. Stanley. Dna visual and analytic data mining. In *Proceedings of the IEEE Visualization'97*, pages 437–441, 1997.

[22] N. Inc. Nvidia CUDA programming guide. http://www.nvidia.com/object/cuda_home.html, 2007.

[23] S. Ingram, T. Munzner, and M. Olano. Glimmer: Multilevel mds on the gpu. *IEEE Trans. Vis. Comput. Graph.*, 15(2):249–261, 2009.

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

[25] A. Inselberg and B. Dimsdale. Parallel coordinates: a tool for visualizing multi-dimensional geometry. In *Proceedings of the IEEE Visualization'90*, pages 361–378, 1990.

[26] J. Johansson, M. Cooper, and M. Jern. 3-dimensional display for clustered multi-relational parallel coordinates. In *Proceedings of the Intl. Conf. on Information Visualization*, pages 188–193, 2005.

[27] J. Johansson, P. Ljung, M. Jern, and M. Cooper. Revealing structure within clustered parallel coordinates displays. In *Proceedings of the IEEE InfoVis'05*, pages 125–132, 2005.

[28] R. Kosara, F. Bendix, and H. Hauser. Parallel sets: interactive exploration and visual analysis of categorical data. *IEEE Trans. Vis. Comput. Graph.*, 12(4):558–568, 2006.

[29] K. T. McDonnell and K. Mueller. Illustrative parallel coordinates. *Computer Graphics Forum*, 27(3):1031–1038, 2008.

[30] M. Novotny. Visually effective information visualization of large data. In *Proceedings of CESCG'04*, pages 41–48. CRC Press, 2004.

[31] M. Novotny and H. Hauser. Outlier-preserving focus+context visualization in parallel coordinates. *IEEE Trans. Vis. Comput. Graph.*, 12(5):893–900, 2006.

[32] W. Peng, M. O. Ward, and E. A. Rundensteiner. Clutter reduction in multi-dimensional data visualization using dimension reordering. In *Proceedings of the IEEE InfoVis'04*, pages 89–96, 2004.

[33] C. Schmid and H. Hinterberger. Comparative multivariate visualization across conceptually different graphic displays. In *Proceedings of SSDBM'94*, pages 42–51, 1994.

[34] H. Siirtola. Direct manipulation of parallel coordinates. In *Proceedings of the Intl. Conf. on Information Visualization*, pages 373–378, 2000.

[35] H. Siirtola. Combining parallel coordinates with the reorderable matrix. In *Proceedings of CMV'03*, pages 63–74, 2003.

[36] H. Theisel. Higher order parallel coordinates. In *Proceedings of VMV'00*, pages 415–420, 2000.

[37] R. Wegenkittl, H. Löffelmann, , and E. Gröller. Visualizing the behaviour of higher dimensional dynamical systems. In *Proceedings of the IEEE Visualization'97*, pages 119–125, 1997.

[38] E. J. Wegman and Q. Luo. High dimensional clustering using parallel coordinates and the grand tour. *Computing Science and Statistics*, 28:352–360, 1997.

[39] N. Wong, S. Carpendale, and S. Greenberg. Edgelens: An interactive method for managing edge congestion in graphs. In *Proceedings of the IEEE InfoVis'03*, pages 51–58, 2003.

[40] P. C. Wong and R. D. Bergeron. Multiresolution multidimensional wavelet brushing. In *Proceedings of the IEEE Visualization'96*, pages 141–148, 1996.

[41] P. C. Wong and R. D. Bergeron. Multivariate visualization using metric scaling. In *Proceedings of the IEEE Visualization'97*, pages 111–118, 1997.

[42] J. Yang, W. Peng, M. O. Ward, and E. A. Rundensteiner. Interactive hierarchical dimension ordering, spacing and filtering for exploration of high dimensional datasets. In *Proceedings of IEEE InfoVis'09*, pages 105–112, 2003.

[43] H. Zhou, W. Cui, H. Qu, Y. Wu, X. Yuan, and W. Zhou. Splatting the lines in parallel coordinates. *Computer Graphics Forum*, 28(3):759–766, 2009.

[44] H. Zhou, X. Yuan, H. Qu, W. Cui, and B. Chen. Visual clustering in parallel coordinates. *Computer Graphics Forum*, 27(3):1047–1054, 2008.