Eurographics/ IEEE-VGTC Symposium on Visualization 2009
H.-C. Hege, I. Hotz, and T. Munzner
(Guest Editors)

Volume 28 (2009), Number 3

Splatting the Lines in Parallel Coordinates

Hong Zhou[1], Weiwei Cui[1], Huamin Qu[1], Yingcai Wu[1], Xiaoru Yuan[2], Wei Zhuo[1]

[1]The Hong Kong University of Science and Technology, Hong Kong
[2]School of EECS & Key Laboratory of Machine Perception (Ministry of Education), Peking University, China

Abstract

In this paper, we propose a novel splatting framework for clutter reduction and pattern revealing in parallel coordinates. Our framework consists of two major components: a polyline splatter for cluster detection and a segment splatter for clutter reduction. The cluster detection is performed by splatting the lines one by one into the parallel coordinates plots, and for each splatted line we enhance its neighboring lines and suppress irrelevant ones. To reduce visual clutter caused by line crossings and overlappings in the clustered results, we provide a segment splatter which represents each polyline by one segment and splats these segments with different speeds, colors, and lengths from the leftmost axis to the rightmost axis. Users can interactively control both the polyline splatting and the segment splatting processes to emphasize the features they are interested in. The experimental results demonstrate that our framework can effectively reveal some hidden patterns in parallel coordinates.

1. Introduction

Parallel coordinates [ID90] have been widely used for analyzing high-dimensional datasets. By drawing dimensions as parallel axes and hyperspace data items as polylines connecting their scalar values on the axes, parallel coordinates can represent N-dimensional data in a 2-dimensional space. However, when the sizes of the datasets become so large that millions of lines overwhelm the display, parallel coordinates may become too dense to be interpreted. Therefore, reducing visual clutter caused by excessive line crossings and overlappings is very important for parallel coordinates.

Many methods have been proposed for clutter reduction in parallel coordinates [FWR99, PWR04, ED06]. Among these approaches, clustering is widely used to detect patterns. Many interactive techniques are available to help users explore the clustered results in parallel coordinates. The clustering algorithms can be classified into two major categories: data clustering and visual clustering. High-dimensional data clustering is a very difficult problem. There is no omnipotent clustering method which works well for all kinds of datasets. In addition, it is usually difficult for users without expert knowledge to control the clustering level when using these data clustering methods. As another kind of promising approach, visual clustering [AdL04, NH06, ZYQ*08] exploits the display information in plots to assist the clustering process and allows users to identify clusters by themselves in visually-enhanced results. However, these methods mainly consider the line or point distributions in the 2D plane formed by two adjacent axes instead of the information in N-dimensional space.

In this paper, we propose a splatting framework to generate animated images to reveal underlying patterns in parallel coordinates. It can also be considered as visual clustering in the time domain. Our framework consists of two major components: a polyline splatter for cluster detection and a segment splatter for clutter reduction. Clusters are detected by splatting the polylines into the parallel coordinate plots one by one, and for each splatted polyline we enhance its neighboring lines and suppress irrelevant ones. Our polyline splatter can provide immediate line enhancement results. Users can easily monitor the clustering process by watching the splatting animation. They can also control the clustering results by directly editing on the enhanced images based on their observations, and change the clustering level by stopping the animation and examining any interesting frames. Different clusters can be visualized using different colors. However, even in the plots with color encoded clusters, excessive line crossings and overlappings may still cause visual clutter. To further reduce clutter in the clustered results, we design a segment splatter which represents each polyline by one segment, and splats these segments from the leftmost axis to the rightmost axis along their polyline paths. These segments move at different speeds, colors, and lengths such

that the original overall line crossings and overlappings can be highly reduced and the underlying patterns can be revealed. Users can configure the segment splatter to enhance various patterns they are interested in.

The major contributions of this paper are as follows:

- We propose a splatting framework to reduce visual clutter and enhance patterns in parallel coordinates. Our framework is flexible, efficient, and configurable.
- We introduce a cluster detection technique which can exploit user's domain knowledge to aid the clustering based on their observations on splatting animations.
- We present a clutter reduction method which can use segments to represent polylines, and spread segments in the time domain to reduce line crossings and overlappings.

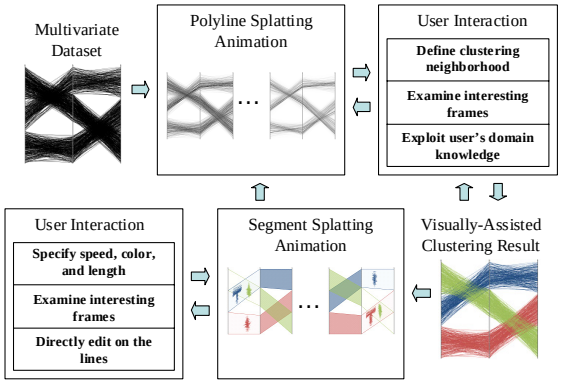


Figure 1: *Splatting framework overview.*

2. Related Work

To address the visual clutter problem in information visualization, many methods have been proposed [ED07]. Some of them have been applied in parallel coordinates. These methods can be classified into the following five categories:

Dimension reordering techniques [ABK98, PWR04] can find a good ordering of dimensions to reduce visual clutter. A hierarchical ordering approach [YPWR03] is designed to improve the interactivity of dimension ordering, spacing, and filtering. Our method can be used together with dimension reordering techniques to further improve their results.

Interactive exploration approaches are designed to navigate data globally and locally to discover any interesting patterns. Brushing techniques, such as wavelet brushing [WB96] and angular brushing [HLD02], are effective tools to select and highlight certain groups of polylines with specific trends between neighboring axes. Other techniques such as scatterplots and graphs can also be used to find patterns in parallel coordinates [STQ08]. Novotny and Hauser [NH06] provided an outlier-preserving focus+context method based on 2-dimensional binning for each pair of adjacent axes. Transfer functions which are widely used in volume graphics have also been applied in parallel coordinates to achieve color enhanced results [JLJC05, ZYQ*08]. Our method is designed to decrease line crossings and overlappings, and thus it can be used together with other interactive techniques.

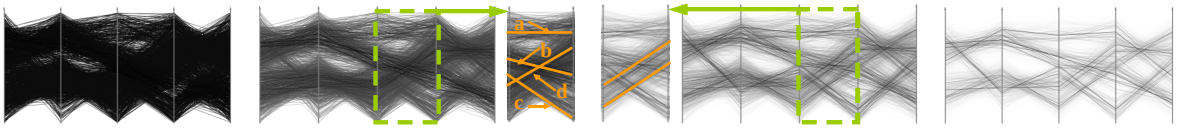
Clustering methods treat each cluster as a whole (*e.g.*, one rectangular) and use different colors or textures to indicate different clusters. Fua *et al.* [FWR99] adopted the Birch algorithm, a hierarchical clustering method, and proposed an interaction tool to convey the multiresolution information of the resulting clusters. Many methods [Nov04, JLJC05] use the k-means algorithm [DH73] to compute the cluster information. However, it will become difficult to set the required input (*i.e.*, *k* value) appropriately if the data are very dense and complex. In our method, the clustering can be facilitated by user's domain knowledge. In another group of methods, binning-based clustering [HLD02] and visual clustering [ZYQ*08] perform the clustering based on the analysis of 2D display information in plots. Our method is more flexible as it can explore the clusters either in a 2-dimensional space or in an N-dimensional space.

Filtering approaches can filter data to show an overview of visually-cluttered areas. Artero *et al.* [AdL04] filtered the information based on 2-dimensional data frequency histograms or density histograms for each pair of adjacent axes. Ellis and Dix [ED06] conducted several comparisons and proved that random sampling is effective to examine particular cluttered subareas. The screen space quality method [JC08] measures and filters data items based on the sampled or clustered result, while still preserving the significant features in the original datasets. Our splatting framework not only detects clusters but also reveals detailed patterns by distributing lines in the time domain.

Animation is good at showing time-varying datasets. It is also useful in interfaces for a variety of purposes in information display. Animation has been widely used in several kinds of data visualization [HR07, EDF08]. Recently, several animation schemes have been developed to aid the data exploration in parallel coordinates. Barlow *et al.* [BS04] used time-based animation on parallel coordinates to visualize the movement of high-dimensional objects. Johansson *et al.* [JLJC05] proposed feature animation to convey statistical properties about clusters in parallel coordinates. Shearer *et al.* [SOMK08] periodically selected groups of lines and faded them in and out to show different patterns. In this paper, we propose splatting as an animation scheme for clutter reduction and pattern detection in parallel coordinates.

3. Splatting Framework Overview

Splatting [MSHC99] is a classic volume rendering technique in scientific visualization. This technique considers volume voxels as splats and projects these splats onto the screen. The splats are then rendered with varying properties (*i.e.*, color and transparency). Inspired by this idea, our frame-



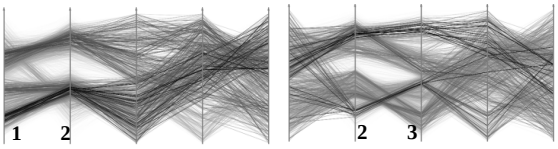
(a) (b) (c) (d)

Figure 2: *Intermediate frames during a polyline splatting animation: (a) original plot; (b)-(d) splatting in N-dimensional space with iteration 200, 800, and 1400 respectively. The cluster d in (b) is further divided into two neighboring clusters highlighted by the orange lines in (c).*

work splats lines into the plot one by one to show displays with enhanced opacities and splats segments from the leftmost axis to the rightmost axis to show hidden patterns. Fig. 1 illustrates the pipeline of our approach. Given a multivariate dataset, the polyline splatter generates a series of animated plots with enhanced cluster information. Users can monitor and control the clustering results interactively. For example, they can examine some representative frames and then provide inputs to our system based on their domain knowledge to guide the clustering process. The clustered information is then sent to the segment splatter to generate an animation with segments moving from the leftmost axis to the rightmost axis. We use one segment to represent one polyline in the parallel coordinates. Each segment moves along its polyline path. Users can change segment attributes (*i.e.*, speed, color, and length) to generate several splatting animations to reveal patterns in the data. They can also go to the polyline splatter again to modify the clustering results based on the detected patterns, and generate new segment splatting animations for further data exploration.

4. Polyline Splatter for Cluster Detection

Our polyline splatter splats the polylines one by one into the parallel coordinates, and for each splatted polyline we enhance its neighboring lines and suppress irrelevant lines in order to reveal any underlying cluster information. This approach is inspired by a daily phenomenon of the ripple effect after throwing straws on a water surface. When a straw is thrown on a quiet water surface, it immediately serves as the trigger source from which a series of ripples spread. The ripple at the place where the straw first hits the water has the largest phase. The swing phases of other ripples decrease with the increasing distances from the trigger source. In addition, all the swing phases decrease as the time goes by. However, if another straw is thrown on the same or nearby place before all the ripples fade out, this hit place will probably pop out with a large swing phase of ripple as this ripple is a combination of the new and existing ripples. After a while, the ripples around most-hit places may always have larger swing phases. In the designed polyline splatter, we want to simulate this ripple effect and highlight cluster centers with "large swing phases" after a period of splatting "straws" into the parallel coordinates.



(a) (b)

Figure 3: *Splatting in 2-dimensional subspaces: (a) formed by axis 1 and 2; (b) formed by axis 2 and 3.*

We abstract this ripple effect by considering each polyline as a "straw" and splatting them into the parallel coordinates "surface" one by one. When a polyline is thrown in, it enhances the opacity values (*i.e.*, "swing phases") of its neighboring polylines. The augmented intensity of the neighbors is determined by their distances from the splatted polyline. Larger distances make smaller enhancements and vice versa. We randomly select polylines and splat them one by one, therefore, each polyline has equal probability of being thrown. When time goes by, we constantly decrease opacities of all the polylines to simulate the weakening of swing phases in the ripple effect. Therefore, after running for a while, outliers and noise gradually become transparent due to low probabilities of being reinforced by neighbors; and it is very likely that cluster centers are then continuously highlighted compared to other places. Finally, the polylines' opacities can indicate how likely they belong to a cluster.

4.1. Polyline Splatting Algorithm Overview

Our polyline splatting algorithm is an iterative process based on the random polyline selection which has been studied and proved to be useful by Ellis and Dix [ED06]. We first set opacity values of all the polylines to be 1.0, and then continuously change their opacities to fade out noise and outliers and highlight cluster centers. Each iteration has the following four steps:

- ***Choose a polyline.*** We randomly choose one polyline as the current straw to be thrown. Since each polyline should have equal probability of being selected, we shuffle the

data items into a circularly-linked list and choose polylines from it one by one.

- ***Find the neighbors.*** We find the neighbors of the selected polyline by a distance threshold d.
- ***Enhance the neighbors.*** We increase the opacities of the found neighbors using a given enhancement function.
- ***Reduce all opacities.*** For each polyline in the data, we decrease its opacity by a given ratio.

The iteration is repeated until users stop it or all the opacities become almost zero.

4.2. Neighborhood Definition

In the step of *"Find the neighbors"*, the neighboring polylines are determined based on the distances of corresponding data items. The data distance can be measured in the N-dimensional space or in any preferred 2-dimensional subspace formed by adjacent axes. Fig. 2(a) to 2(d) show a series of frames during the splatting based on the N-dimensional Euclidean distance of data items. Fig. 3(a) and 3(b) show two splatting results with neighbors measured in different 2-dimensional subspaces. One is specified in the 2-dimensional subspace formed by axis 1 and 2, and the other one is formed by axis 2 and 3.

4.3. Enhancement Function

In the step of *"Enhance the neighbors"*, for each neighboring polyline j of the selected polyline i, its new opacity $\widetilde{O_j}$ is defined as follows:

$$\widetilde{O_j} = O_j + O_j G(D_{ij}) \tag{1}$$

where O_j is line j's opacity in the previous iteration step; D_{ij} is the distance between polyline j and i; $G(D_{ij})$ is the defined Gaussian distribution function which is used to describe how enhanced intensity changes. A larger distance results in a lower enhanced intensity and vice versa. The Gaussian distributor effectively diffuses the enhanced intensity as follows:

$$G(D_{ij}) = w_{max} e^{-\frac{2D_{ij}}{d^2}} \tag{2}$$

where d is the distance threshold and w_{max} is the weight. Basically, all non-increasing functions can be used here as a distribution function. We tested some other distribution functions, and found that our defined Gaussian distributor is good enough to show the desired results.

4.4. Visually-Assisted Clustering

In our polyline splatting process, polylines representing outliers and noise become more and more transparent, and polylines around cluster centers become gradually enhanced with high opacities (See Fig. 2). Therefore, after using our splatting for a while, the major clusters will automatically pop out with larger opacities. Users can stop the splatting if the patterns of major clusters have been clearly revealed. Then our automatic clustering algorithm computes the clusters based on the opacity-enhanced result. It examines all the data items starting from polylines with the highest opacities in the image and considers other polylines with lower opacities by iteratively decreasing an opacity threshold. The data items with the highest opacity values are taken to be the cluster centers first. Then data items with lower opacities are classified into existing clusters or into new clusters based on a distance threshold. The cluster centers are also updated after new lines are classified into them, and become merged if some of them are too close. Thus, each cluster is gradually grown by including nearby data items, until all the data items are assigned to a certain cluster or no data item exists within the opacity threshold.

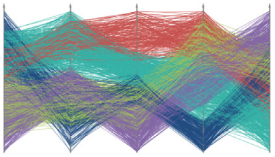


Figure 4: *Highlighted clusters with different colors.*

Users can use their domain knowledge to assist and control the clustering through several interaction tools. They can stop the splatting animation at any time and examine any interesting frames which they think are good enough to reveal the overall patterns in the data. They can specify the number of clusters or directly brush on the selected frame to specify the cluster centers based on their observations, and our system can use the K-means algorithm to cluster the data. Our system can also use the proposed automatic clustering algorithm described above to cluster the data.

5. Segment Splatter for Clutter Reduction

The proposed polyline splatter can detect clusters in cluttered data and use colors to indicate different clusters. However, visual clutter may still be a problem. Fig. 4 illustrates such an example of visually-cluttered images where colors are used to show multiple clusters in parallel coordinates. After assigning a unique color to each cluster, some patterns can be revealed. However, when lines in different colors highly intersect and overlap one another, the resulting clutter makes the relationships among clusters and detailed structures within clusters unclear. Although drawing each cluster as one polygon can better show relationships among clusters, the details in clusters are still missing. To show the details, we have to draw all the lines simultaneously on the same display. However, if too many lines are blended together, the effectiveness of information revealing will be highly reduced by the visual clutter caused by blended lines.

In order to show detailed line information and correla-

tions between clusters, we propose a segment splatter which draws the cluster polygons as the context and moves shorter lines as the focus. Each polyline is displayed as a segment. These segments move along their original polyline paths at different speeds from the leftmost axis to the rightmost axis. The speed function encodes cluster information and separates different clusters in the time domain. Segments will have dynamically-changed lengths representing the local information in each cluster. We also design a color function in order to compensate the missing spatial relations between different lines. Combining the speed, length, and color functions together, our segment splatter method can visually reveal both the global and local cluster structures.

5.1. Speed Function

Segments in different clusters are moving from one axis to the adjacent axis at different speeds. Segments within the same cluster will have the same speed. Therefore, at any time, the front-ends of segments in the same cluster will form a compact group. For each cluster g, the speed $\widetilde{v_g}$ is defined as:

$$\widetilde{v_g} = \frac{N_g}{N_{total}} + \Delta v'_{g_t} \tag{3}$$

where N_g and N_{total} represent the number of items in cluster g and the total number of items in the dataset respectively. Ware and Bobrow [WB06] found that pattern detection in motion is most effective when all the glyphs are moving at the same frequency but in different phases. Based on their suggestions, we use the dynamically-changed speed $\Delta v'_{g_t}$ to control the phase differences between clusters and reduce crossings between lines from different clusters. At time t, for each cluster g, we compute the center position P_g of all its segments. If any other cluster's center position (*e.g.*, P_k) falls within the g's force circle area A_g, cluster g will give a force F_{gk} to cluster k at P_k (See Fig. 5). F_{gk} is the universal gravitation computed by $Gm_g m_k / r_{gk}^2$, where r_{gk} is the distance between the two cluster centers, and m_g and m_k are determined by the cluster size. Larger clusters have larger m and vice versa. Therefore, if at time t, P_g falls totally into n clusters' force circle areas, the Δv_{g_t} is computed as follows:

$$\Delta v_{g_t} = \frac{\sum_{i=1}^{n} F_{ig}}{m_g} \Delta t \tag{4}$$

Note that $\Delta v'_{g_t}$ used in equation 3 is the propagated Δv_{g_t} in equation 4 on the cluster g's moving direction which is fixed and cannot be changed. If cluster g already arrives at the next axis but other clusters are still behind it, we will set $\widetilde{v_g}$ to be zero to make cluster g wait for other clusters. After all the other clusters arrive, all the segments will start to move to the next axis together.

5.2. Length Function

We can use the length of segments to encode some local structures or statistical information. Segments within the same clusters will be of different lengths. For each polyline i in cluster g, we define the segment length $\widetilde{l_i}$ based on its front-end position (x,y) as follows:

$$\widetilde{l_i} = w_1 |P_i - \overline{P_g}| + w_2 |\sin(\alpha_{i_{x,y}} - \varphi_{x,y})| \tag{5}$$

where

- P_i is the corresponding hyperspace point of polyline i;
- $\overline{P_g}$ is the gravity center of all the points in cluster g;
- $\alpha_{i_{x,y}}$ is the angle of polyline segment i at (x,y);
- $\varphi_{x,y}$, calculated by Kernel Density Estimator [DH73], is the primary angle of all polyline segments which fall in the screen space around point (x,y);
- w_1 and w_2 are the corresponding weights for each term.

We use $\widetilde{l_i}$ to model the local structures. The first term assigns long lengths to segments with large Euclidean distances to the cluster center; and the second term assigns long lengths to segments with large angular distances to the primary angles around its current location area. Therefore, our length function tends to use longer lengths to highlight outlier or cluster boundary segments. Cluster centers will still be discernable because of the dense segments moving around.

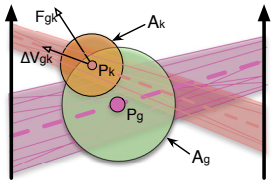


Figure 5: *An example when cluster k's center falls into cluster g's force circle area.*

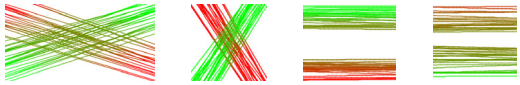


Figure 6: *Color diffusion examples between two clusters: (a) with small interaction angle; (b) with large interaction angle; (c) with long distance; (d) with short distance.*

5.3. Color Function

Our segment color function is designed to compensate for the missing spatial relations between different clusters during the segment splatting process. For example, two clusters may intersect with each other at a certain place. However, because they are assigned different speeds, they may pass through that place at different times. Although users can reconstruct the information by extending segments in their mind, this important pattern may still not be obvious. To address this issue, we propose a color formulation which

is inspired by the diffusion effect in water color paintings. When two clusters intersect, their colors infuse each other. The bigger cluster can inject more color to the smaller one. The injected intensity decreases when the intersected angle increases, and reaches the minimum when two clusters are perpendicular to each other (See Fig. 6(a) and 6(b)). In addition, if two clusters are parallel at a certain axis interval, they also infuse their colors to each other. The longer the distance between them, the smaller the diffusion level, and vice versa (See Fig. 6(c) and 6(d)). For each polyline i, its color C_i is determined by its cluster information and will not be changed. We use one segment to represent one polyline and move the segment along its polyline path. Therefore, when segment i proceeds at position (x,y) with n polyline paths in the screen space around (x,y), the color of segment i at position (x,y) is defined as follows:

$$C_{i_{x,y}} = w_3 C_i + w_4 \frac{\sum_{j=1}^{n} |\cos(angle(\alpha_{i_{x,y}}, \beta_{j_{x,y}}))| C_j}{\sum_{j=1}^{n} |\cos(angle(\alpha_{i_{x,y}}, \beta_{j_{x,y}}))|} \quad (6)$$

where

- $\alpha_{i_{x,y}}$ is the direction of the polyline i at (x,y);
- $\beta_{j_{x,y}}$ is the direction of polyline j at (x,y);
- $angle(\alpha_{i_{x,y}}, \beta_{j_{x,y}})$ is the angular difference between $\alpha_{i_{x,y}}$ and $\beta_{j_{x,y}}$;
- C_j and C_i are the assigned colors of polyline j and polyline i;
- w_3 and w_4 are the corresponding weights for each term.

Although the segment colors may change during the splatting, users will not be confused about which clusters the segments belong to, because the segments within the same cluster move together at similar speeds during the animation.

5.4. User Editing

Our segment splatting generates animations of segments running from the leftmost axis to the rightmost axis. Users can first get an overview, then use the provided sliding bar to fast forward or backward to explore the animation process. Since static depictions of motion may be more effective than the motion itself for the analysis purpose [RFF*08], users can freely choose any frames to analyze various interesting patterns. The segment splatting results can be customized by setting the weight parameters in the length and color functions. For example, users can change the weights of different length terms to focus on their interested outlier features, or change the weights of different color terms to modify the diffusion intensity.

6. Experimental Results

We applied our splatting method to three datasets. We first tested our method with a synthesized dataset of 7736 five-dimensional items with clear cluster distributions. The dataset is the same one used by Hong *et al.* [ZYQ*08]. It consists of 4800 noisy items and other 2936 items distributed into four clusters with 876, 752, 608, and 700 items respectively. Fig. 7(b) and 7(c) show two intermediate frames during the polyline splatting animation. We can see that the pattern of four clusters starts to appear after the first 800 iterations and they are stably highlighted during the remaining animation process. Based on our observations of the splatting animation, we noticed that there is a lot of noise in the dataset, because the light grey background does not have clear patterns. Therefore, we used the system to remove the noise first and then cluster the data into four clusters. With the help of user's domain knowledge, our system correctly detected clusters and visualized them with different colors (See Fig. 7(d)). Fig. 7(e) shows the result of hierarchical parallel coordinates [FWR99]. This algorithm could not detect clusters very well because of noise. Fig. 7(f) shows the result by visual clustering [ZYQ*08]. After comparing Fig. 7(d) with 7(f) and checking with the data, we found that our approach can also remove noise and differentiate clusters successfully. In addition, our splatting-based clustering does not contain any time-consuming optimization process and thus is more efficient. In this experiment, we can see that our splatting algorithm can filter out noise and reveal underlying clusters.

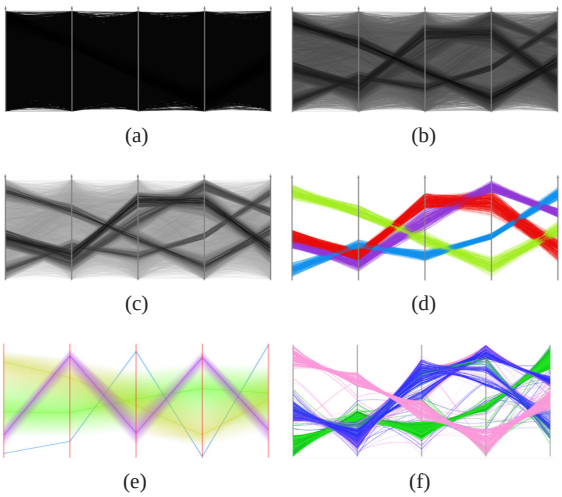


Figure 7: *Experiments on a synthesized dataset. (a) original plot; (b) after 800 iterations; (c) after 2000 iterations; (d) splatting result with colors indicating different clusters; (e) hierarchical parallel coordinates result; (f) visual clustering result. (e) and (f) are Courtesy of Hong* et al. *[ZYQ*08].*

Some datasets do not have much noise, but the clusters are blended together without clear separation. To further test our algorithm, we synthesized a dataset of 1430 five-dimensional data items distributed into five blended clusters with 356, 202, 297, 372, and 203 items respectively. Our polyline splatting can blur outliers and fuzzy cluster boundaries to reveal clearer cluster centers (See Fig. 2). The cluster

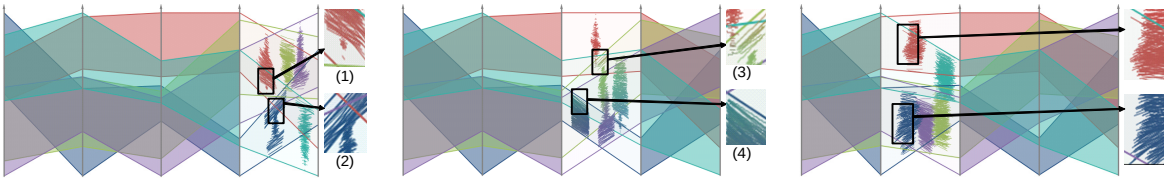


Figure 8: *Segment splatting results on the same dataset used in Fig. 2: (a)* $w_1 = 0, w_2 = 1.0, w_3 = 1.0, w_4 = 0$*; (b)* $w_1 = 0, w_2 = 1.0, w_3 = 1.0, w_4 = 1.0$*; (c)* $w_1 = 1.0, w_2 = 1.0, w_3 = 1.0, w_4 = 0$.

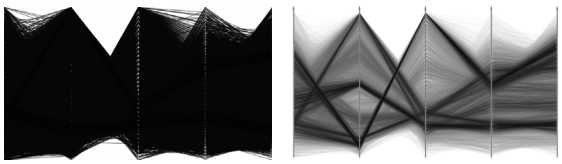


Figure 9: *Splatting experiments on the out5d dataset. (a) original plot; (b) after 12800 iterations.*

labeled with *d* in Fig. 2(b) is further divided into two neighboring clusters (highlighted by the orange lines in Fig. 2(c)), by removing their fuzzy boundaries in the splatting process. The detected clusters can be displayed using different colors, however, it is still difficult to differentiate them and see detailed patterns in the color-encoded results (See Fig 4). Then we applied our segment splatter. In Fig. 8(a), one major frame during the splatting from axis 4 to 5 is shown. The splatting parameters is set with $w_1 = 0, w_2 = 1.0, w_3 = 1.0,$ and $w_4 = 0$, which means that color diffusion is not used and the segment length only indicates the angular distance information. Two interesting subareas are highlighted by the dark rectangles. In region 1, the red cluster is clearly revealed after removing the intersected purple and light green lines. Some short segments are displayed in the lower part of this region, which means that the direction of these red lines is similar to those of the intersected lines from other clusters. The long segments in region 2 indicate that these lines have large angular differences from other lines passing through this region. Fig. 8(b) shows the result with the color injection effect applied to convey the spatial relations between clusters. In region 3, the red cluster accompanies another green cluster in the lower side, because its lower side is infused with green, while the upper side is still pure red. In region 4, the blue cluster hides in the green cluster, because their colors are highly integrated with the green color. Therefore, if the correlations between clusters are partially weakened, they can still be discerned by our color codes. In Fig. 8(c), several long segments in the two enlarged regions mean that they tend to be the outliers both in the data and in the local screen space. This experiment demonstrates that our splatting method can separate blended clusters and reveal more details in the data.

Lastly, we applied our approach to a large remotely-sensed dataset: named out5d [†] with 5 variables and 16,384 data items. After the polyline splatting, some clusters are revealed in Fig. 9(b). In order to disclose more details about the clusters, the system is set to splat the segments from the leftmost axis to the rightmost axis with $w_1 = 0, w_2 = 1.0, w_3 = 1.0,$ and $w_4 = 0$. Several resulting images are shown in Fig. 10. We can see that our method is good at separating intersected lines and showing local information. In Fig. 10(a), the thin segments in the purple cluster are at the cluster center. In Fig. 10(d), all the segments are of similar lengths and directions, which indicates that all the polylines in this 2-dimensional subspace formed by the last two axes tend to be parallel and overlapped.

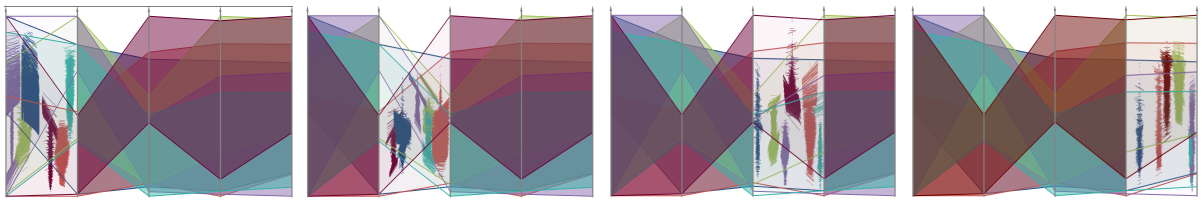
All our results were generated on a Macbook Pro with Intel Core 2 Duo 2.2GHz CPUs and 2GB Memory. In the polyline splatting animations for our three tested datasets, Fig. 2(c), Fig. 7(b), and Fig. 9(b) are selected time frames in 10s, 29s, and 1800s respectively. The preprocessing times for generating segment splatting results in Fig. 8 and Fig. 10 are 18s and 195s respectively.

7. Conclusions and Future Work

In this paper, we have presented a novel framework which consists of two splatters (*i.e.*, polyline splatter and segment splatter) to reduce visual clutter and reveal patterns in parallel coordinates. The polyline splatter splats the lines into the plot one by one and gradually shows opacity-enhanced images. Users can conveniently interact with the instant results and assist the clustering with their domain knowledge. Based on the clustered information, the proposed segment splatter splats segments from the leftmost axis to the rightmost axis at different speeds, colors, and lengths to reduce the original line crossings and reveal hidden patterns.

In the future, we plan to investigate the effectiveness of

[†] http://davis.wpi.edu/∼xmdv/datasets.html



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

Figure 10: *Segment splatting results on the out5d dataset.*

our system through formal user studies. We also plan to integrate more sophisticated interaction tools into our system such as hierarchical tree views, and apply our splatting ideas to other visual representations like graphs.

Acknowledgments

We thank Kai-Lun Peter Chung for his help on the first draft. This work was supported by Hong Kong RGC grant CERG 618706, Beijing NSF grant 4092021, and National "985 Project" Phase II at Peking University.

References

[ABK98] ANKERST M., BERCHTOLD S., KEIM D. A.: Similarity clustering of dimensions for an enhanced visualization of multidimensional data. In *Proc. of IEEE Symp. on Information Visualization* (1998), pp. 52–60.

[AdL04] ARTERO A. O., DE OLIVEIRA M. C. F., LEVKOWITZ H.: Uncovering clusters in crowded parallel coordinates visualizations. In *Proc. of IEEE Symp. on Information Visualization* (2004), pp. 81–88.

[BS04] BARLOW N., STUART L. J.: Animator: A tool for the animation of parallel coordinates. In *Proc. of Intl. Conf. on Information Visualization* (2004), pp. 725–730.

[DH73] DUDA R. O., HART P. E.: *Pattern Classification and Scene Analysis.* Wiley, 1973.

[ED06] ELLIS G., DIX A.: Enabling automatic clutter reduction in parallel coordinate plots. *IEEE Trans. on Vis. and Comp. Graph. 12*, 5 (2006), 717–724.

[ED07] ELLIS G., DIX A.: A taxonomy of clutter reduction for information visualization. *IEEE Trans. on Vis. and Comp. Graph. 13*, 6 (2007), 1216–1223.

[EDF08] ELMQVIST N., DRAGICEVIC P., FEKETE J.-D.: Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation. *IEEE Trans. on Vis. and Comp. Graph. 14*, 6 (2008), 1141–1148.

[FWR99] FUA Y.-H., WARD M. O., RUNDENSTEINER E. A.: Hierarchical parallel coordinates for exploration of large datasets. In *Proc. of IEEE Visualization* (1999), pp. 43–50.

[HLD02] HAUSER H., LEDERMANN F., DOLEISCH H.: Angular brushing of extended parallel coordinates. In *Proc. of IEEE Symp. on Information Visualization* (2002), pp. 127–130.

[HR07] HEER J., ROBERTSON G.: Animated transitions in statistical data graphics. *IEEE Trans. on Vis. and Comp. Graph. 13*, 6 (2007), 1240–1247.

[ID90] INSELBERG A., DIMSDALE B.: Parallel coordinates: a tool for visualizing multi-dimensional geometry. In *Proc. of IEEE Visualization* (1990), pp. 361–378.

[JC08] JOHANSSON J., COOPER M.: A screen space quality method for data abstraction. *Computer Graphics Forum 27*, 3 (2008), 1039–1046.

[JLJC05] JOHANSSON J., LJUNG P., JERN M., COOPER M.: Revealing structure within clustered parallel coordinates displays. In *Proc. of IEEE Symp. on Information Visualization* (2005), pp. 125–132.

[MSHC99] MUELLER K., SHAREEF N., HUANG J., CRAWFIS R.: High-quality splatting on rectilinear grids with efficient culling of occluded voxels. *IEEE Trans. on Vis. and Comp. Graph. 5*, 2 (1999), 116–134.

[NH06] NOVOTNY M., HAUSER H.: Outlier-preserving focus+context visualization in parallel coordinates. *IEEE Trans. on Vis. and Comp. Graph. 12*, 5 (2006), 893–900.

[Nov04] NOVOTNY M.: Visually effective information visualization of large data. In *Proc. of Central European Seminar on Computer Graphics* (2004), pp. 41–48.

[PWR04] PENG W., WARD M. O., RUNDENSTEINER E. A.: Clutter reduction in multi-dimensional data visualization using dimension reordering. In *Proc. of IEEE Symp. on Information Visualization* (2004), pp. 89–96.

[RFF*08] ROBERTSON G., FERNANDEZ R., FISHER D., LEE B., STASKO J.: Effectiveness of animation in trend visualization. *IEEE Trans. on Vis. and Comp. Graph. 14*, 6 (2008), 1325–1332.

[SOMK08] SHEARER J., OGAWA M., MA K.-L., KOHLENBERG T.: Pixelplexing: Gaining display resolution through time. In *Proc. of IEEE Pacific Vis. Symp.* (2008), pp. 159–266.

[STQ08] SANTAMARIA R., THERON R., QUINTALES L.: A visual analytics approach for understanding biclustering results from microarray data. *BMC Bioinformatics 9*, 247 (2008).

[WB96] WONG P. C., BERGERON R. D.: Multiresolution multidimensional wavelet brushing. In *Proc. of IEEE Visualization* (1996), pp. 141–148.

[WB06] WARE C., BOBROW R.: Motion coding for pattern detection. In *Proc. of symp. on Applied perception in graph. and vis.* (2006), pp. 107–110.

[YPWR03] YANG J., PENG W., WARD M. O., RUNDENSTEINER E. A.: Interactive hierarchical dimension ordering, spacing and filtering for exploration of high dimensional datasets. In *Proc. of IEEE Symp. on Information Visualization* (2003), pp. 105–112.

[ZYQ*08] ZHOU H., YUAN X., QU H., CUI W., CHEN B.: Visual clustering in parallel coordinates. *Computer Graphics Forum 27*, 3 (2008), 1047–1054.