Interactive Storyboard for Overall Time-Varying Data Visualization

Aidong Lu[*]
University of North Carolina at Charlotte

Han-Wei Shen[†]
The Ohio State University

Abstract

Large amounts of time-varying datasets create great challenges for users to understand and explore them. This paper proposes an efficient visualization method for observing overall data contents and changes throughout an entire time-varying dataset. We develop an interactive storyboard approach by composing sample volume renderings and descriptive geometric primitives that are generated through data analysis processes. Our storyboard system integrates automatic visualization generation methods and interactive adjustment procedures to provide new tools for visualizing and exploring time-varying datasets. We also provide a flexible framework to quantify data differences and automatically select representative datasets through exploring scientific data distribution features. Since this approach reduces the visualized data amount into a more understandable size and format for users, it can be used to effectively visualize, represent, and explore a large time-varying dataset. Initial user study results show that our approach shortens the exploration time and reduces the number of datasets that users visualized individually. This visualization method is especially useful for situations that require close observance or are not capable of interactive rendering, such as documentation and demonstration.

Index Terms: I.3.6 [Methodology and Techniques]: Interaction techniques— [I.3.7]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

1 Introduction

The increasing amount of scientific data creates new challenges for developing effective visualization techniques, especially for time-varying datasets. Previous work on time-varying data visualization primarily focused on the topics of accelerated rendering, feature extraction, change detection, and feature tracking, etc. In this paper, we propose a new method to visualize and explore overall time-varying data contents and relations from the entire time range.

It is often difficult to visualize and analyze large scale time-varying data because of the enormous data volume. To analyze a time-varying dataset, the most common approach is to perform interactive rendering at each time step, or to generate snapshots / composed animations in a batch process. For a time-varying dataset that has a large number of time steps, both approaches can be quite ineffective for users to grasp the overall temporal trend and detailed data properties due to the limitation of human perception systems [32], as pointed out by Joshi and Rheingans that visually inspecting each snapshot of a time-varying dataset is not practical for a large number of time steps [16]. Especially when multiple objects are interacting and changing over time, it is very difficult for users to analyze complex data relations in mind from numerous separate information pieces. Therefore, we need to integrate data analysis results into representation processes for more effective visualization of time-varying datasets.

In this paper, we present a new method for visualizing overall temporal evolution and salient data features of time-varying datasets. To address the issue mentioned above, there is a need to develop new time-varying data visualization techniques that can summarize complex data dynamics in a concise but effective manner, while still allowing users to closely observe data in greater details. To achieve this goal, we design an *Interactive Storyboard*, which displays sample images and line drawings in a clear storyboard layout to depict data relevancies and differences. Our design enhances the function of a storyboard by appropriately arranging snapshots and primitives to assist users to understand essential data contents and changes. This approach improves time-varying visualization by reducing the amount of data needed to visualize complex data characteristics. It allows close exploration and observation, which are especially useful for documentation and demonstration.

To facilitate effective data viewing through our interactive storyboard, we propose an approach to reducing the number of time steps that users need to visualize individually to understand essential data features by selecting representative datasets. We have designed a flexible framework for quantifying data differences using multiple dissimilarity matrices. This dissimilarity information is further analyzed through an extremum position detection algorithm to choose representative datasets. This framework is capable of showing various data features and it can be easily adjusted according to the application requirements by modifying a potential data feature list. Similar to the previous work on feature extraction and feature tracking, we treat the problem of representative data selection as a feature extraction process along the time axis, where the volume data are viewed as features-of-interest. By combining the information of data relations and the selection process of representative datasets, we can preserve salient features in the underlying time-varying dataset while reducing the amount of time steps required to generate overall storyboard visualization. Our initial user study shows that this approach shortens the exploration time and reduces the number of visualized datasets that are required to understand a time-varying dataset.

The remainder of the paper is organized as follows: We first summarize related visualization and graphics work on time-varying data, motion, and key data selection techniques. In section 3, we describe our framework for quantifying data differences using multiple dissimilarity matrices and an optimized weight generation process. In section 4, we automatically choose representative datasets for scientific datasets by incorporating two data distribution features. Section 5 describes our interactive storyboard design, automatic generation, and integrated interaction approaches for visualizing overall contents of time-varying datasets. Finally, we will discuss our results and future work in section 6.

2 Related Work

Time-varying data visualization [15] is a challenging topic because of the large data size and volume. Feature tracking has been one important research direction, since it can provide the frame-to-frame correspondence between objects-of-interest to reveal the temporal trend of a time-varying dataset. The tracking information can be further studied to detect significant data changes. Currently, most feature tracking approaches are based on pre-defined feature models or user-specified regions-of-interest. The matching of data features is generally achieved by the following two mechanisms. First,

[*] e-mail: aidong.lu@uncc.edu
[†] e-mail: hwshen@cis.ohio-state.edu

based on selected regions-of-interest for feature tracking, data features are matched based on their corresponding positions [25] or topological features are tracked using high dimensional geometries [14]. Critical points of geometry models have also been studied in many applications [12, 26, 8, 10]. Second, feature attributes, such as position and size, are derived from data models and used to measure data changes. For example, Samtaney et al. [24] introduced several evolutionary events and tracked 3D data according to their feature attributes. Banks and Singer [3] used a predictor-corrector method to reconstruct and track vortex tubes from turbulent time-dependent flows. Reinders et al. [22] matched several attributes of features and tracked feature paths based on the motion continuity. Verma and Pang [29] proposed comparative visualization tools for analyzing vector datasets based on streamlines. We design a general method for comparing data dissimilarities, which does not require a dense sampling frequency to capture the object evolution and is not limited by specific feature models, such as geometry or interval volumes, and their attribute designs. Our method can also be used to visualize data distributions according to selected representative datasets.

The usages of snapshots have been explored for various purposes. First, multiple snapshots can be organized to compare and analyze complex information. Marks et al. [20] automatically generated and organized graphics or animations in the “Design Gallery” interface to help finding desirable input parameters. Ma [19] used image graphs to streamline the process of visual data exploration through dynamic graph features. Approaches that explore neural networks and information visualization techniques have also been explored to assist time-varying data visualizations [1]. Second, images can also be used to represent both static and moving objects, such as “moving images” [9]. Woodring et al. [35] simulated the chronophotography technique to depict time-varying data features using a high dimensional direct rendering method. Joshi and Rheingans [16] simulated techniques commonly used in comic books to convey changes over time. Similar to their objectives, we propose a different approach to improve the visualization effectiveness by decreasing the number of time steps for users to visualize for understanding overall data contents and relations. There are also relevant video summaries or visualization techniques [5, 6, 33], which generally focused on handling images over time.

“Key-poses” or “key-frames” have been mainly used in the domains of computer animation and video for motion retrieval, synthesis, activity recognition, etc. For example, a large number of key-poses were selected for motion synthesis [17] and video sequences [7]. Loy et al. [18] used a clustering algorithm to select key frames that are centers of frame clusters. Assa et al. [2] presented human motions in still images by selecting key poses based on the analysis of a skeletal animation sequence. We are mostly inspired by this paper to develop a general framework for visualizing and analyzing time-varying volumetric data, although a volume dataset typically does not have any specific feature models as human motions.

3 Data Relationship Measurement

To efficiently visualize a time-varying dataset with a large number of time steps, we design a new visualization approach that integrates data analysis results, which are achieved by measuring the degree of data similarity/difference and selecting important datasets that contain essential data features. This section discusses the key component in the comparison and selection processes, which is to compare all the time steps and measure their similarities or differences. As illustrated in Figure 1, a large amount of time steps are reduced to a much smaller number through the process of dissimilarity measurement and data distribution analysis. The quantitative results will be used to analyze representative datasets in section 4 and visualize an entire time-varying dataset in section 5.

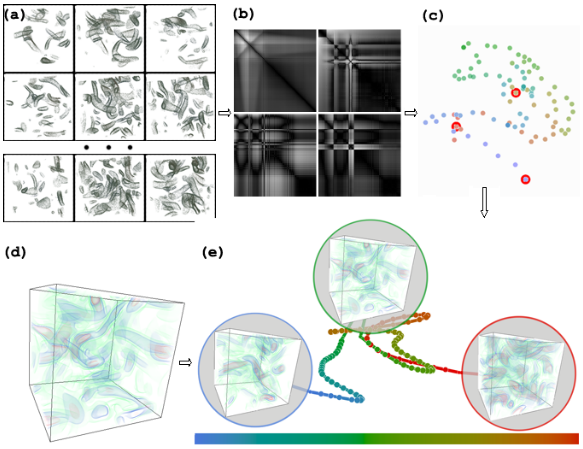

Figure 1: Our system architecture: We integrate the information of data analysis (b, c) and a single 3D data visualization method (d) for users to explore and visualize overall time-varying data contents (e). For a time-varying dataset (a), we calculate data dissimilarities according to selected data features (b) and select representative datasets by analyzing the distribution of time steps (c). The integration of data analysis results reduces the visualized data amount and keeps the essential information for more efficient time-varying data visualization.

Our approach allows users to compare 3D datasets from different time steps using a combination of various relevant data features. For each selected data feature, we calculate a dissimilarity matrix by comparing every data pair according to the feature definition. Then, we compose a final matrix as the quantified dissimilarity result through optimizing the calculation weights. We have explored a set of potential data features to measure data dissimilarities from different aspects, including geometry, texture, and statistical information. This framework is robust and easy for users to incorporate additional data comparison criteria. The final dissimilarity matrix will be affected by the selected data features to represent data relations that users are interested in.

3.1 Dissimilarity Matrix Computation

We first select relevant data features and regions-of-interest through visualizing single time steps using a direct 3D volume rendering approach. The data features can be selected from our sample list, as shown in Table 1, which includes multiple geometry, statistics, and texture differences. We have concentrated on general data features in object space, since feature space approaches require prior knowledge of the data models and image space algorithms need pre-selected viewpoint for volumetric data.

Assuming that a time-varying dataset includes n time steps, one $n \times n$ dissimilarity matrix will be generated for each potential data feature. To make sure that the final dissimilarity matrix is independent of the scales of different data features, we first calculate the maximum and minimum values of a feature in theory and then normalize the dissimilarity matrix using these two values. For example, the maximum and minimum values of the volume difference count are the data size and 0, and those of the χ^2 statistics are the histogram length and 0. If the volumes at two time steps have very similar data values, the matrix will mostly be filled with zero. This normalization process avoids having bias toward any particular data features, but preserves the degree of dissimilarity within any given feature criterion.

A time window, $T(d_1, d_2)$, can be used to modulate the dissimilarity matrix $M(d_1, d_2)$ based on their time interval, where d_1 and

Table 1: Our potential dissimilarity matrix computation list. The framework allows easy modifications for additional data features.

Dissimilarity Items	Measurements
Geometry & Topology	
Volume difference	A scanning process is performed to calculate the volume of regions-of-interest
Area difference	Approximated as the number of voxels that belong to the regions-of-interest
Center position shift	The shift of the weighted object center position
Boundingbox size change	The change of the boundingbox size for regions-of-interest
Shape change	The shape difference of regions-of-interest after the bounding boxes are aligned
Region number change	The number changes of separate geometries [25, 14]
Texture	
K-L divergence and J. divergence	$d_J(H,K) = \sum_i (h_i log \frac{h_i}{m_i} + k_i log \frac{k_i}{m_i}), where\ m_i = \frac{h_i+k_i}{2}$ [23]
χ^2 statistics	$d_{\chi^2}(H,K) = \sum_i \frac{(h_i-m_i)^2}{m_i}, where\ m_i = \frac{h_i+k_i}{2}$ [28]
Match distance	$d_M(H,K) = \sum_i \mid \hat{h}_i - \hat{k}_i \mid$, where $\hat{h}_i$ and $\hat{k}_i$ are the cumulative histogram of $\{h_i\}$ and $\{k_i\}$
EMD	The minimal cost need to transform H to K to the total flow [23]
Statistic difference	
Scalar value, gradient, curvature	Differences of the average and standard deviation
Gradient and curvature directions	Differences of the angular separation
Transfer functions	From extended distance matrices of the texture approaches

d_2 are a data pair:

$$\hat{M}(d_1,d_2) = T(d_1,d_2) * M(d_1,d_2) \tag{1}$$

Two functions can be applied in different applications according to the requirement of enhancing or reducing the time dependency in the dissimilarity values [2, 30]. Generally, $e^{-\alpha|t_{d_1}-t_{d_2}|}$ is used to enhance the changes that are temporally closed and $1-e^{-\alpha|t_{d_1}-t_{d_2}|}$ is used to reduce it, where α is a constant. We use a small α in the second format to reduce the time dependency, since we want to choose representative datasets mainly from the information of data dissimilarities.

To accelerate the computation process, we collect and prepare information from all the data volumes during the preprocessing step, including detecting the number of separate objects and gathering basic data information (e.g., gradient and curvature). Figure 2 shows 11 dissimilarity matrices and the final matrix for analyzing a time-varying energy dataset.

3.2 Weight Optimization

After calculating individual dissimilarity matrices for a selected set of data features, we need to merge all of them into one final matrix, which will be used later to choose representative datasets. Assuming m dissimilarity matrices are generated, we use their weighted sum to compose a final matrix $D(d_1,d_2)$.

$$D(d_1,d_2) = \sum_{i=1}^{m} p_i * \hat{M}(d_1,d_2) \tag{2}$$

The weights $p_i (i = 1,...,m)$ play an important role in the final matrix, which will be used to select representative datasets. We propose an automatic process for generating the matrix weights by maximizing the data differences. We argue that the final matrix should catch the majority data differences and thereby compose a larger variety of values. Therefore, we use the standard deviation of the final matrix as our objective function in the optimization process. Since the different scales of data dissimilarities have already been considered in the matrices, the weights are only calculated according to their value distributions. The weights can be automatically solved by using the direction set method to minimize this objective function [21]:

$$f(p_i, i = 1,...,m) = \delta(D(d_1,d_2)) \tag{3}$$

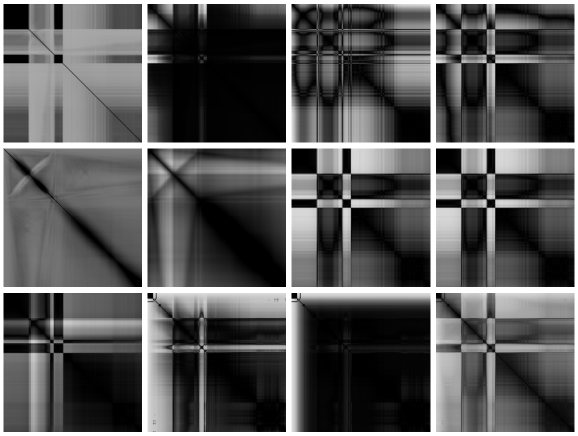

Figure 2: Dissimilarity matrices of an energy dataset for value-of-interest, volume value differences, value standard deviation, average value, gradient direction, gradient magnitude, volume of regions-of-interest, surface area, center position shift, KL divergence, χ^2 statistics, and the final matrix respectively. Brighter regions indicate larger dissimilarity values.

which does not require an explicit function format.

4 Representative Datasets Analysis

We automatically select representative datasets to reduce the required data amount for understanding time-varying data contents by analyzing the final dissimilarity matrix. Assa et al. [2] presented an approach to selecting key frames of animation sequences by measuring the similarities among a character's joint positions. Our main difference is that we want to interactively select representative datasets that include a significant portion of features for scientific data, whose data distribution requires more analysis than time sequence. The use of representative datesets reduces the amount of data to visualize and still keeps the essential data information, which can be used to improve the efficiency of time-varying data visualization.

4.1 Dimensionality Reduction

Because of the following three factors, we apply dimensionality reduction approaches to decrease the dimension of final dissimilarity matrix. First, since the dissimilarity matrix is composed of multiple measuring criteria, there may exist redundant information. Second, it is much faster when we perform the selection process in a lower dimensional space. Most importantly, we need to reduce the data information into a space where they can be visualized effectively.

Inspired by the human motion analysis work [2], we use the multi-dimensional scaling (MDS) [27, 4], which is a set of data analysis techniques that can display the pattern of proximities (i.e., similarities or distances) among multiple objects. Here, we can directly input the final dissimilarity matrix and outputs n point positions in a specified dimension, with each point corresponding to a time step. The Euclidean distances among output points are optimized to best express their dissimilarity values. Since the output point positions from our final dissimilarity matrix do not have real physical meanings, we test two types of non-classical MDS approaches and do not find significant differences between non-classical metric MDS and non-metric MDS methods. In this paper, we use the non-classical metric MDS for all the results.

To determine appropriate dimensions, we can use the MDS stress curve ($s_i,\ i = 1,2,\ldots$), which measures the difference between the dissimilarity values and output point distances. Starting from dimension 2, we calculate the difference of stress values between two adjacent dimensions ($|s_i - s_{i-1}|$) and automatically choose the one whose difference with previous dimension is smaller than a threshold, such as $\frac{|s_i - s_{i-1}|}{s_i} < 10\%$. For all the data used in this paper, the dimensions range between 2 to 12 were found to be appropriate for further analysis.

4.2 Representative Datasets Selection

Since we want to locate representative datasets mainly from the characteristics of data distributions, we do not take the order of time steps into consideration at this stage and it will be used later in the visualization process in section 5.

From the reconstructed point cloud of MDS output (section 4.1), we have found two obvious distribution properties of scientific data which can be used to select representative datasets. As shown in Figure 3, when we connect points in the order of time steps, clear curve shapes can be seen from the original point cloud. Also, several clusters are formed among the point cloud, where close points indicate similar data contents at these time steps. We will need to combine these two distribution properties to locate representative datasets.

For each point in the MDS output, we calculate its suitability value of being a presentative dataset using the following three factors: representative size, change speed, and distances to the points that are already in the set. These factors are designed using geometry properties of the extremum locations in a high dimensional space, which indicate key time steps, according to the two data distribution properties.

First, the representative size $S(d)$ of each point d. The points are first clustered using the mean shift algorithm [11], which can be used without pre-knowledge of cluster number and shape. The cluster radius $r(c_i)$ is set as the maximum distance of the points belonging to a cluster c_i to the cluster center. We design a weight $g_i(d)$ for calculating $S(d)$ in a way that data closer to the center of larger clusters have bigger representative sizes, as shown below, where $\|c_i\|$ is the number of points in cluster c_i and $Dis_i(d)$ is the distance of point d to the center of cluster c_i.

$$S(d) = \Sigma_{clusters} \|c_i\| \cdot g_i(d) \tag{4}$$

$$where\ g_i(d) = \begin{cases} 0, Dis_i(d) > r(c_i) \\ 1 - (Dis_i(d)/r(c_i))^2, Dis_i(d) \leq r(c_i) \end{cases}$$

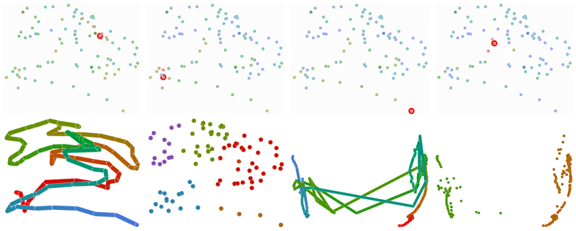

Figure 3: The top row illustrates the selection process of representative datasets. The bottom row demonstrates the two general properties of reconstructed data distributions: time sequence (left of each pair) and cluster tendency (right of each pair).

Second, data changes $C(d)$ of a point d within its local neighborhood, including changes in direction and distance. Assuming points d_1 and d_2 are two neighbors of point d, we use the direction change between $\overrightarrow{d_1 - d}$ and $\overrightarrow{d_2 - d}$ to approximate extremum locations in the MDS output space, with a constant p_c to control the effect of direction changes, and their lengths to measure the degree of local data changes. This is consistent with our observance that close points on a relative straight line represent smooth transitions and have small change values. The total data changes $C(d)$ of a point d is calculated by adding changes between every neighbor pair of point d.

$$C(d) = \sum_{d_1,d_2} \left(\frac{(|\overrightarrow{d_1 - d}| \cdot |\overrightarrow{d_2 - d}| + 1)}{2}\right)^{p_c} \|d_1 - d\| \|d_2 - d\| \tag{5}$$

Third, the distance of a point d to the points that are already selected as representative datasets. This can ensure the differences among selected representative datasets, which can be adjusted using a constant weight p_d.

$$Dif(d) = \sum_{d_i \in Set} (\|d_i - d\|)^{p_d} \tag{6}$$

Finally, the suitability of a point as a representative dataset is calculated by combining the above three factors:

$$V(d) = S(d) * C(d) * Dif(d) \tag{7}$$

The representative proportion of a set of selected datasets is measured as the sum of suitability values of selected datasets to the total value of all the points.

$$p(Set) = \frac{\Sigma_{d \in Set} V(d)}{\Sigma_{d \in Data} V(d)} \tag{8}$$

Given a desired number of representative datasets or a representative portion value from users, we can perform a greedy algorithm to select representative datasets. We continuously select a point with the largest suitability value $V(d)$, until the desired stop criteria is reached. When we set 100% as the desired representative portion, this process assigns each point a sequence number, which is used in the user interaction later for adjusting details shown from representative datasets. We can also select representative datasets without any parameter by calculating the maximum average representative proportion $p(Set)/ \parallel Set \parallel$. This can be achieved by traversing all possible combinations to find a best solution. Both procedures select representative datasets mainly from data distributions derived from the final data dissimilarity matrix. As shown in Figures 5-7, only the datasets that are special to the entire time range are selected.

We can significantly accelerate the selection procedure by precomputing the majority values, especially for multiple selection

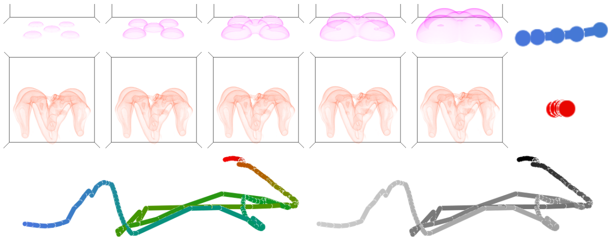

Figure 4: Visualization design. (Top) The right images show our timelines for the 5 left datasets respectively. Smaller data changes on the second row result closer MDS point positions. (Bottom) Similarly, point positions in a complete color/grey timeline represent information of data dissimilarity and time sequence, which will be further used to visualize overall time-varying data contents.

processes. Since $S(d)$ and $C(d)$ do not change once MDS is finished, they can be calculated before the selection. Although $Dif(d)$ varies, a $n \times n$ distance table between all the points can be pre-generated for fast lookup. By gathering all these values, the greedy selection process can run interactively.

5 Interactive Storyboard

We design a new visualization approach, interactive storyboard, to visualize and explore overall contents of time-varying datasets through composing suitable amount of information that can be efficiently understood by users. Our design principle is to visualize both data contents and relations through integrating data analysis results in this storyboard visualization system, including the final data dissimilarity matrix, point cloud from MDS output, and representative datasets from the previous two Sections. Since the selection of representative datasets preserves essential features of data contents and significantly reduces the number of datasets for users to visualize, it is more effective than asking users to visualize each time step individually and analyze all the datasets afterwards. We develop an automatic composition process for generating and rendering the interactive storyboard system. We also integrate several interaction approaches to allow users to control storyboard results and explore data evolution during different time periods.

For exploring time-varying datasets, our storyboard is designed by arranging data relations, data dissimilarity distributions, and snapshots of representative datasets to visualize overall data contents. Storyboard is a powerful descriptive tool that has been successfully used to describe events [13], actions [2], or visualize volume data [34]. We will show that various complex evolutions of time-varying datasets can be visualized through our flexible storyboard generation method.

5.1 Visualization Design

Our visualization layout is generated from two components: data relations and sample snapshots. The data relations are mainly represented by the MDS output and sample snapshots can be generated for representative datasets using any direct volume rendering approach (we use texture-based volume rendering for results in this paper). We use sample snapshots from key time steps to represent essential data contents at different levels, and reduce the details of others by showing their relations to the adjacent time steps.

We design the overall time-varying visualization by embedding sample snapshots generated from representative datasets into a layout that is organized from the point cloud of MDS output. Since close points represent similar datasets (small dissimilarity values), it is intuitive for users to understand that the contents of these datasets are similar. The effectiveness of this approach is similar to various MDS applications for demonstrating data relations in many social, science, and engineering fields. Our initial layout shape comes from the 2D/3D MDS reconstruction result, which is a series of 2D/3D point positions. Since the timeline may be difficult to understand directly when the points are connected in the order of time steps, we smooth the timeline between representative datasets using the weighted average position between each two adjacent points. This preserves their original distances, which represent data dissimilarity degrees, and displays them in a more readable format. As shown in Figure 4, both the data similarities (according to point locations) and time sequence (indicated by rainbow or grey colors) can be visualized through our timelines.

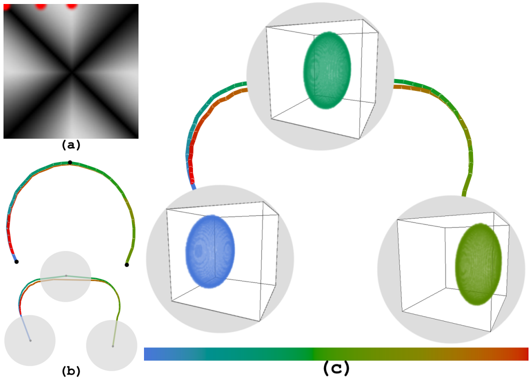

Figure 5: (a) Final dissimilarity matrix for a simple sphere time-varying data shows that it is difficult to select the representative datasets (in red dots) directly. (b) An example of the automatic layout generation process by adding circle templates and organizing point positions. (c) Our storyboard describing a sphere moves back and forth when the timeline changes from blue to red.

According to the selection process of representative datasets, we assign a rendering level for each time step to decide the size of rendering primitives. Representative datasets will be shown using their snapshots with different sizes and the rest will only be shown as points. Since a 3D volume may face any direction in a 3D space, we use a circular shape as the template for embedding sample snapshots, as shown in Figure 5. Each sample image will be zoomed to best fit the template around the circle center. We assign grey scale background colors to represent the importance of a time step and optional edge colors to strengthen its time sequence.

For smooth exploration and visualization of a time-varying dataset, the snapshots of all the time steps are pre-generated so that any selected time step can be displayed in real time during interaction. We also include volume boundaries in the snapshots to show the volume orientation. The snapshots from all the time steps are generated from the same view to avoid confusions in the case that objects are changing over time. The view direction can be selected automatically by maximizing entropy values or minimizing the occlusions of regions-of-interest [31].

5.2 Automatic Generation

We automatically adjust the storyboard layout and rendering settings through the following three steps: basic layout generation, automatic fitting, and primitive property assignment. Our basic storyboard layout is generated from processed timelines of MDS output, as shown in Figures 4 and 5.

We then automatically embed sample snapshots into the basic layout by using their previous assigned rendering levels and circle templates. For 3D layout, snapshots are embedded directly using the corresponding point positions as the centers of circle templates. For 2D layout, we re-arrange point locations to avoid snapshots overlapping in the storyboard. Our approach is to add extra space

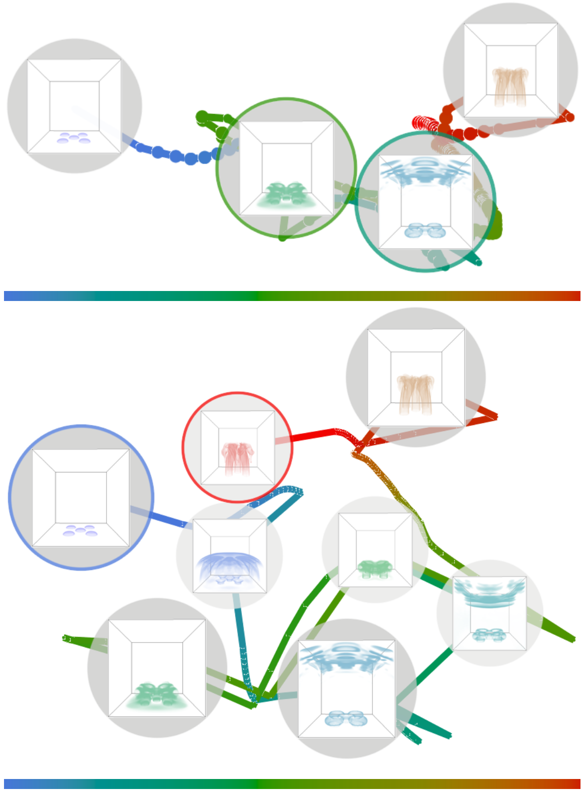

Figure 6: Storyboards for an energy dataset with different level-of-details. The storyboard on the top clearly shows the most important data information along the timeline: the main object starts from the bottom, expands to the top, shrinks to the bottom, and finalizes around the center. The bottom storyboard contains more details by using less smoothed timeline and more representative datasets.

for each snapshot and adjust storyboard according to accumulated size of all the points and snapshots. Assuming our circle templates have size r_l for rendering level l and there are totally n different levels. We first measure the distances between each snapshot pair and push them along the opposite direction if they are closer than the required circle template sizes. Then, starting from the first time step, we traverse all the point positions in the initial layout. A time step corresponding to a representative dataset with rendering level l will be expanded along the previous and following directions using a circle template with size r_l. During this accumulation process, we keep the proportion of adjacent point distances except representative datasets to preserve the overall data dissimilarity information. After we traverse all the time steps, we stretch the whole layout linearly to fit the assigned rendering space with the same scale on both x and y axes. A user can control the mapping direction from the accumulated layout to the rendering space. We leave this control to the user to keep the interface consistent during the interaction. In our examples, the maximum snapshot size is assigned to be 10 times of the average point distance, and a lower level snapshot size is 60% of the higher one.

The time steps that correspond to non-representative datasets are simply shown as points. We use point size to represent local data density, which is approximated by the distances to the closest time steps. The point colors are used to represent time sequence by using the blue to red portion of a rainbow, where blue indicates the first time step and red indicates the last time step. The widths and colors of line segments are interpolated between the attributes of connecting points.

5.3 User Interaction

We provide several interaction and exploration functions that allow users to select important time ranges and control storyboard results. The amount of user effort to achieve these interactions is largely reduced through integrating user interaction and our automatic timeline adjustment process. These interaction functions especially enhance the exploration and analysis capability of our interactive storyboard.

We first provide a function to adjust the details of storyboard contents with a scalar value *scale* between *min* and *max*. This allow users to expand storyboard to observe more details or shrink it for a higher level view. When the scale of details is increased, we enrich the storyboard contents by providing more detailed timeline layouts and adding snapshots of lower level representative time steps. The timeline is less smoothed for representing more accurate information of data distribution. Lower level representative time steps are selected by continuously locating the next representative dataset from the time steps that have not been included, according to the selection sequence calculated using suitability values $V(d)$ in section 4.2, until the new representative portion $p(Set)$ (calculated from equation 8) is larger or equal to the user-specified degree $\frac{scale}{max}$. The sizes of snapshots are used to indicate their "importance" in the entire time range. Figure 6 shows the storyboard at two representative levels, noticing that the representative datasets for a larger scale value (bottom) include all the selected datasets on the top. When the value of details is decreased, the shrinking process is achieved by reversing the increasing process: we use less detailed timeline and reduce the number of snapshots in the layouts. The *min* level only includes one snapshot and the *max* level uses snapshots for all the time steps. The usages of one scalar value and automatic update process make it very convenient for users to adjust the level-of-details.

We also allow users to select their interested time periods and modify the scale of details for each time period respectively. The important time periods are selected by indicating the start and end time steps, represented by two small black triangles on the top of colorbar in Figure 7. For every selected time period, users can control its level of details using the above detail adjustment tool. The rest of the time periods will be rendered at the default highest level. As shown in Figure 7, the second half of the time range is enriched with more timeline details and sample snapshots.

Another useful interaction function is to provide an overview of data distributions surrounding a particular time step selected by users. To achieve this function, we automatically modify the storyboard contents from the following two aspects. First, we add the specified time step as a representative dataset. Then, we select representative datasets by using difference values to the specified time step d_s in the final dissimilarity matrix as the weights of suitability values:

$$V'(d) = \|d - d_s\| \times V(d) \tag{9}$$

This modification favors the datasets that are more different from the specified time step. As shown in Figure 8, the storyboard provides an overview of data relations around the selected time step from the entire time range.

We also add a direct 3D volume rendering window in addition to the storyboard for enhancing the exploration function of our system, as shown in Figure 9. The storyboard portion is used as a guideline and summary of the data contents throughout the entire time range. The users can still visualize each individual dataset through the 3D volume rendering portion. Users can interactively select a time step, as indicated on the left bottom corner of the storyboard, to visualize in the 3D rendering window. This combination provides a more comprehensive tool for exploring time-varying datasets, especially those with a large number of time steps.

Our storyboard system also includes a key frame display window that can be used to enlarge snapshots from multiple selected time

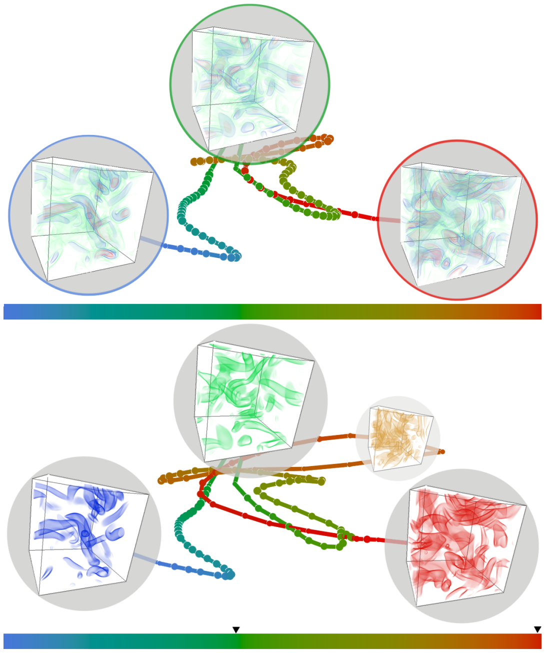

Figure 7: Storyboards for a vortex dataset. Representative datasets are connected by smooth timelines to visualize overall time-varying data contents and changes. A user can interactively select their interested time ranges and explore additional information by expanding corresponding portions of the storyboard.

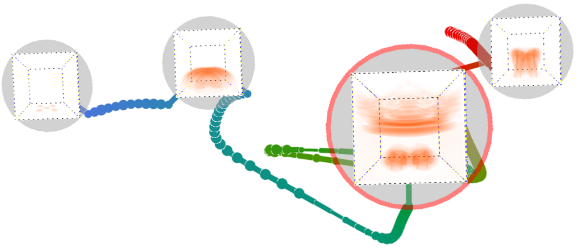

Figure 8: Concentration on a particular time step. When a user selects time step 58, which is highlighted with a red template boundary, the storyboard automatically update representative datasets for visualizing overall data relations around the selected time step.

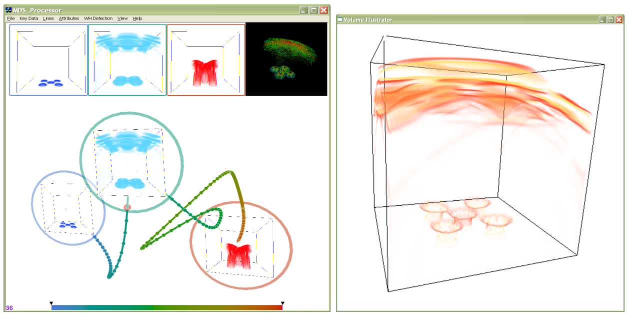

Figure 9: The storyboard system is composed of the bottom timeline portion and the top key frames portion. In this figure, the bottom is a 3D storyboard for an energy data with three key time steps. The red dot in the middle is used to control the time step shown in the right top corner and a separate single data rendering window where a user can perform common interaction tasks, such as rotating and selecting regions-of-interest.

steps for better comparison, as shown in Figure 9. For each time step, users can increase its rendering level by adding it to representative datasets or shrink it to a point. A simple interaction interface is also provided to adjust the potential feature list for measuring data dissimilarity matrices. The regions-of-interest are selected using standard 2D transfer function.

5.4 Results and Discussions

Figures 5-8 show several 2D storyboards and Figure 9 shows a 3D storyboard for visualizing overall time-varying data contents and relationships throughout the entire time range. The dimension of all the time-varying datasets used in this paper is $128 \times 128 \times 128$. The number of time steps for the sphere data is 128, energy data is 200, and vortex data is 100. The system performance of the storyboard is interactive for the above datasets. The preparation time can be long according to the selected feature combinations: the final dissimilarity matrix takes hours and all the snapshots are generated within a few minutes. This process can be shortened by optimizing our distance matrix calculation algorithm or with parallel methods. Since user time is viewed as much more precious than computer time, we believe that it is practical to utilize computing resources to shorten the required user interaction time and allow users to visualize a time-varying dataset with a large number of time steps interactively.

We find that 2D layout has less occlusion problem caused by displaying 3D objects; thereby more suitable for representation and demonstration purposes. Since a 3D layout can be integrated with more interactions, such as rotation, it is more interesting for exploring and interacting with the contents of a time-varying dataset. There is a tradeoff issue for adjusting the timeline shape: smooth lines are easier to understand and winding lines are better in representing the original data distribution. We perform a small amount of smoothing operations and a user can adjust the modification degree with a variable. We also make the circle template transparent for a clearer view on the underlying timelines.

5.5 User Study

We have conducted an initial user study to evaluate the effectiveness of the proposed interactive storyboard method. We use two systems, our interactive storyboard and a standard direct volume rendering system, for visualizing six different time-varying datasets (each with 20 time steps). Each subject is asked to explore the contents of these six datasets until he or she is fully confident in understanding the entire datasets. We randomize test data sequence and alternate the two provided systems for balancing other factors. During the experiments, we recorded the time steps that subjects chose to visualize using the 3D rendering window/system, so that we can summarize the total numbers of visualized time steps and the durations of experiments. Our initial data analysis results from 7 subjects (students and faculties in the field of visualization) show that the storyboard method can shorten the average performance time and decrease the number of visualized time steps. This is consistent with our expectation since the storyboard is designed for reducing avoidable comparison and visualization operations for users. We plan to perform a formal user study to test more subjects and the significance of these results.

6 Conclusions and Future Work

This paper presents an interactive storyboard method that can be used to visualize and explore overall contents of time-varying datasets. Through this new data/information visualization format, we integrate data analysis results into visualization processes so that users can understand overall data contents without visualizing each individual time step. The effectiveness of this method is derived from the suitable amount of information that are composed from data analysis results, including essential data contents, distributions, and relationships. The essential data information preserves a significant portion of data features from the entire time range, greatly reduces the amount of information our users need to digest, and provides new visualization capabilities to interact with time-varying datasets. We show that this approach can provide new visualization tools with convenient user interactions, such as exploring and representing time-series datasets for scientific studies.

We have developed a framework for analyzing data relations and selecting representative time steps for time-varying datasets. This is achieved by quantifying data differences into multiple dissimilarity matrices and choosing representative datasets through detecting extremum positions in the final dissimilarity matrix. This framework is flexible for measuring various data features and can be easily modified according to the application requirements by adjusting the potential data feature list. In this paper, we consider representative datasets selection as the feature extraction process from a group of temporally related datasets. Therefore, our applications using representative datasets can reveal the essential data features from a large number of time steps. We demonstrate the usages of representative datasets for better data digestion effect in our interactive storyboard method.

Our future work includes investigating the following approaches to extend the proposed interactive storyboard method. We will perform a formal user study and test the scalability of the storyboard system. We plan to accelerate the preparation process for selecting representative datasets by optimizing the computation components and developing parallel algorithms. We are interested in including snapshots from different viewpoints into the storyboard layout to provide more comprehensive information. We also plan to extend this approach to improve the direct time-varying visualization approaches by utilizing the information of representative datasets. Finally, we will develop representative dataset selection methods for 3D vector data by exploring additional vector dissimilarity measurements.

Acknowledgements

This work was supported by DOE Grant DE-FG02-06ER25733, NSF Grant Nos. 0633150, 0325934, 0403342, NSF Career 0346883, and DOE SciDAC Grant DE-FC02-06ER25779.

References

[1] H. Akiba and K.-L. Ma. A tri-space visualization interface for analyzing time-varying multivariate volume data. In *Proceedings of The Joint Eurographics-IEEE VGTC Symposium on Visualization*, 2007.

[2] J. Assa, Y. Caspi, and D. Cohen-Or. Action synopsis: Pose selection and illustration. In *Proceedings of ACM SIGGRAPH*, pages 667–676, 2005.

[3] D. C. Banks and B. A. Singer. A predictor-corrector technique for visualizing unsteady flow. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):151–163, 1995.

[4] I. Borg and P. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer, 1997.

[5] J. Ćalić and N. W. Campbell. Compact visualisation of video summaries. *EURASIP J. Adv. Signal Process*, 2007(2):17–17, 2007.

[6] M. Chen, R. Botchen, R. Hashim, and I. Thornton. Visual signatures in video visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1093–1100, 2006. Member-Daniel Weiskopf and Member-Thomas Ertl.

[7] D. DeMenthon, V. Kobla, and D. Doermann. Video summarization by curve simplification. In *Proceedings of the sixth ACM international conference on Multimedia table of contents*, pages 211–218, 1998.

[8] H. Edelsbrunner, J. Harer, A. Mascarenhas, and V. Pascucci. Time-varying reeb graphs for continuous space-time data. In *Proceedings of 20th Ann. Sympos. Comput. Geom.*, pages 366–372, 2004.

[9] W. Freeman, E. Adelson, and D. Heeger. Motion without movement. *Computer Graphics*, 25(4):27–30, 1991.

[10] I. Fujishiro, R. Otsuka, Y. Takeshima, and S. Takahashi. T-map: A topological approach to visual exploration of time-varying volume data. In *Proceedings of ISHPC2005, Springer Lecture Notes in Computer Science*, volume 4759, 2007.

[11] B. Georgescu, I. Shimshoni, and P. Meer. Mean shift based clustering in high dimensions: A texture classification example. In *International Conference on Computer Vision*, pages 456–463, 2003.

[12] T. Gerstner and R. Pajarola. Topology preserving and controlled topology simplifying multiresolution isosurface extraction. In *Proceedings of Visualization*, pages 259–266, 2000.

[13] D. B. Goldman, B. Curless, S. M. Seitz, and D. Salesin. Schematic storyboarding for video visualization and editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 25(3):862–871, 2006.

[14] G. Ji, H.-W. Shen, and R. Wenger. Volume tracking using higher dimensional isosurfacing. In *Proceedings of IEEE Visualization*, 2003.

[15] C. Johnson and C. Hansen. *Visualization Handbook*. Academic Press, Inc., Orlando, FL, USA, 2004.

[16] A. Joshi and P. Rheingans. Illustration-inspired techniques for visualizing time-varying data. In *Proceedings of IEEE Visualization*, pages 679–686, 2005.

[17] J. Lee, J. Chai, P. Reitsma, J. K. Hodgins, and N. Porllard. Interactive control of avatars animated with human motion data. In *ACM Siggraph*, pages 491–500, 2002.

[18] G. Loy, J. Sullivan, and S. Carlsson. Pose-based clustering in action sequences. In *Workshop on Higher-Level Knowledge in 3D Modeling and Motion Analysis*, pages 66–72, 2003.

[19] K.-L. Ma. Image graphs - a novel approach to visual data exploration. In *Proceedings of IEEE Visualization*, pages 81–88, 1997.

[20] J. Marks, B. Andalman, P. A. Beardsley, and et al. Design galleries: a general approach to setting parameters for computer graphics and animation. In *Proceedings of Siggraph*, pages 389–400, 1997.

[21] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C : The Art of Scientific Computing*. Cambridge University Press, 1992.

[22] F. Reinders, F. H. Post, and H. J. Spoelder. Visualization of time-dependent data using feature tracking and event detection. *The Visual Computer*, 17(1):55–71, 2001.

[23] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

[24] R. Samtaney, D. Silver, N. Zabusky, and J. Cao. Visualizing features and tracking their evolution. *IEEE Trans. Comput.*, 27:20–27, 1994.

[25] D. Silver and X. Wang. Tracking and visualizing turbulent 3d features. *IEEE Transaction on Visualization and Computer Graphics*, 3(2):129–141, 1997.

[26] B.-S. Sohn and C. Bajaj. Time-varying contour topology. *IEEE Transactions on Visualization and Computer Graphics*, 12(1):14–125, 2006.

[27] W. Torgeson. Multidimensional scaling of similarity. *Psychometrika*, 30:379–393, 1965.

[28] J. M. Utts. *Seeing Through Statistics*. Duxbury Press, 2004.

[29] V. Verma and A. Pang. Comparative flow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 10(6):609–624, 2004.

[30] J. Vermaak, P. Perez, M. Gangnet, and A. Blake. Rapid summarization and browsing of video sequences. In *British Machine Vision Conference*, 2002.

[31] I. Viola, M. Feixas, M. Sbert, and M. E. Gröller. Importance-driven focus of attention. In *Proceedings of IEEE Visualization*, pages 933–940, 2006.

[32] B. A. Wandell. *Foundations of Vision*. Sinauer Associates, 1995.

[33] M. Waschbüsch, S. Würmlin, D. Cotting, F. Sadlo, and M. Gross. Scalable 3d video of dynamic scenes. *The Visual Computer*, (2):629–638, 2005.

[34] M. Wohlfart and H. Hauser. Story telling for presentation in volume visualization. In *Proceedings of The Joint Eurographics-IEEE VGTC Symposium on Visualization*, 2007.

[35] J. Woodring, C. Wang, and H.-W. Shen. High dimensional direct rendering of time-varying volumes. In *Proceedings of IEEE Visualization*, pages 417–424, 2003.