Visualizing Time-Varying Features with TAC-based Distance Fields

Teng-Yok Lee* Han-Wei Shen†

The Ohio State University

ABSTRACT

To analyze time-varying data sets, tracking features over time is often necessary to better understand the dynamic nature of the underlying physical process. Tracking 3D time-varying features, however, is non-trivial when the boundaries of the features cannot be easily defined. In this paper, we propose a new framework to visualize time-varying features and their motion without explicit feature segmentation and tracking. In our framework, a time-varying feature is described by a time series or Time Activity Curve (TAC). To compute the distance, or similarity, between a voxel's time series and the feature, we use the Dynamic Time Warping (DTW) distance metric. The purpose of DTW is to compare the shape similarity between two time series with an optimal warping of time so that the phase shift of the feature in time can be accounted for. After applying DTW to compare each voxel's time series with the feature, a time-invariant distance field can be computed. The amount of time warping required for each voxel to match the feature provides an estimate of the time when the feature is most likely to occur. Based on the TAC-based distance field, several visualization methods can be derived to highlight the position and motion of the feature. We present several case studies to demonstrate and compare the effectiveness of our framework.

Index Terms: I.3.3 [Computing Methodologies]: COMPUTER GRAPHICS—Picture/Image Generation; I.3.7 [Computing Methodologies]: COMPUTER GRAPHICS—Three-Dimensional Graphics and Realism

1 INTRODUCTION

While animations are commonly used for visualizing time-varying data, there exist several major challenges that hinder effective analysis of complex time-dependent features. For instance, due to the 2D nature of the display, comparing the sizes and shapes, and tracking the motion paths of small 3D time-evolving features can be difficult. It is also non-trivial to identify complex temporal patterns in a data set from animations since the user needs to infer quantitative information and memorizes the trends at multiple data points in a long period of time. Finally, the transfer functions used in direct volume rendering (DVR) are often not designed to track data values of high dynamic ranges spanned across a large number of time steps. As a result, important temporal features can be easily missed.

To overcome these problems, feature-based visualization techniques have been studied in the past [3] [5] [10] [11] [20] [23] [24] [26]. With those techniques, not only a large data set can be reduced down to a smaller number of salient features, but also the complex dynamics of the time-varying data can be succinctly described by the features' characteristics in space and time. Most of the feature-based visualization techniques require extracting the features in the spatial domain first, and then tracking the evolution of the features by matching their positions over time. Feature extraction, nevertheless, can be nontrivial when the boundaries and the properties of the features are not well defined. The extraction and tracking of features can also be very time consuming.

In this paper, we propose a novel visualization framework to analyze time-varying data sets. A major goal of this framework is to allow the user to perform detailed analysis of a data set's temporal behavior, and to identify regions that exhibit different spatiotemporal properties. Specifically, we are interested in time-varying features that can be described as time series or called *Time Activity Curves* (TAC). TACs are common as feature representations in medical data, such as electrical signals of the heart via electrocardiography, brain signals via positron emission tomography (PET) imaging or functional Magnetic Resonance Imaging (fMRI), and electromagnetic radiation from dynamic SPECT. Previously, researchers have utilized TACs for analysis and visualization in various medical applications [6] [7] [30]. In scientific simulations, features of interest are also commonly represented as TACs. In an earthquake simulation, for example, the seismic wave measured at a fixed point over time forms a time series, whose shape can be used to determine the geometrical properties (e.g. the distance to the earthquake source) or the geological properties (e.g. the material) at the point.

To identify regions that exhibit a particular temporal trend represented by a user-specified feature TAC, we propose a novel method in our framework to convert the original time-varying volume into a distance field, called *TAC-based Distance Field*. The distance field stores the dissimilarity of each voxel's time series to the feature TAC, from which various visualization techniques can be applied to reveal the spatial distribution of the temporal feature. To measure the dissimilarity/distance between TACs, several issues need to be addressed. First, because the feature travels through space at a finite speed, different points in space will encounter the feature at different times. This phenomenon can be observed as a shift of time in the data points' respective TACs. The width of the feature on the TAC can also be stretched/compressed because the feature may travel at different speeds in different regions. Another point of consideration is that as the feature travels in space, its property, for example the magnitude of the earthquake wave, may also gradually change over time. This will cause the shape of the feature TAC to deform along its motion path. As an example, the three TACs shown in Figure 5 (a) illustrate how the feature TAC can change over time. It can be seen that the temporal patterns have different lengths, and the peaks appear in different positions inside the TACs.

Previously, L_1 distance metric, L_2 distance metric and cross-correlation were used to compare TACs [6] [7] [30], but these distance metrics cannot adequately address the issues mentioned above. To tackle the problems, we employ a metric called *Dynamic Time Warping* (DTW) to calculate the shift- and deform-invariant dissimilarity. DTW is a dynamic programming algorithm that aligns two time series with the smallest distortion. With DTW, TACs of similar shapes with different temporal shifts and time spans can be classified into the same group. Meanwhile, because DTW also provides a mapping between the time steps of two TACs, it is possible to estimate when the feature of interest emerges at different points in the spatial domain. As a result, the spatiotemporal revolution of the feature can be effectively depicted.

This paper is organized as follows. We review related works on time-varying data visualization and feature-tracking in Section 2. Detail about the TAC-based distance field is described in Section 3,

*e-mail: leeten@cse.ohio-state.edu

†e-mail: hswhen@cse.ohio-state.edu

including the TAC-based feature description, computation of DTW, and construction of the TAC-based distance field. The visualization algorithms are described in Section 4. Because the time complexity of DTW is quadratic to the input size, we describe how to accelerate the computation using GPUs in Section 5. Section 6 presents case studies using two time-varying data sets. Section 7 concludes this paper and presents possible future work.

2 Related Works

Visualizing time-varying data has been a focus of visualization research for the past decade. While there are several issues related to this research topic such as design of transfer function [1] [9], and efficient data structures for interactive rendering [8] [22] [34], here we mainly focus on techniques that are targeted at effective displaying and tracking of time-varying phenomenon.

2.1 Time-varying Data Visualization

To visualize time-varying features in illustrative styles, Post *et al.* proposed a technique to render time-varying data as iconic symbols to represent the essential attributes of salient features [19]. In [25], Svakhine *et al.* proposed various techniques to visualize time-varying 3D data by displaying the contour volumes with different visual enhancements. To render the motion of a time-varying feature, Joshi and Rheingans [12] drawed the motion with conventional illustrative techniques such as speedlines to create visualization in different styles; this technique, however, requires a priori feature tracking stage to create the traces.

Several works have been proposed to visualize time-varying data sets without animation. Similar to the Chronophotography technique, Woodring and Shen described an algorithm called Chronovolume to visualize data of multiple time steps in a single image [31]. Another way to visualize time-varying 3D data is to consider the whole data set as a 4D volume, and then visualization can be achieved by rendering a 3D hyperplane in the 4D domain from different perspectives, as proposed by Woodring *et al.* [33]. Compared to the previous works above, which are focused on visualizing isosurfaces or interval volumes, our framework provides a more general representation of the temporal features by using TACs with more rendering options.

In addition to the applications in medical imaging [6] [7] [30], TAC and similar ideas have been studied in scientific visualization in recent years [2] [27] [32]. Woodring and Shen employed TACs in multi-scale to represent the trends of temporal activities in various levels of detail. After all TACs in a volumetric data set are decomposed into multiple scales via Discrete Wavelet Transform and the TACs in each scale are grouped together, TACs in all groups are displayed as small multiples to help the users to quickly identify the trends in various scales [32]. In the TAC-related papers, each data point is associated with a function of time as the temporal trend. In a *function field*, instead, each data point is associated with functions of other parameters for different applications, such as the hyperspectrum images in remote sensing. Anderson *et al.* proposed visualization and query techniques to explore function fields [2]. Similar to TACs, another concept called *importance curves* was proposed by Wang *et al.* For each subvolume, the mutual information among consecutive time steps is calculated as a function of time, defining the importance curve of the subvolume. By displaying all importance curves or the clustered mean curves, users can categorize the temporal activities in the volumetric data sets [27].

Visualizing time-varying data by displaying salient features has been widely studied especially for large scale data sets since features require much less storage than the raw data. Feature-based visualization requires several components: feature definition, feature extraction and segmentation, feature tracking and visual representation of the features. In the next subsection several feature tracking algorithms are reviewed.

2.2 Feature Tracking

To track time-varying isosurfaces, Wang and Silver proposed several algorithms for regular grid [23] and unstructured grid [24] volumes. Their assumption is that there exists spatial overlap among corresponding isosurfaces in consecutive time steps. This assumption means that a series of overlapped isosurfaces is equivalent to a connected isosurface in 4D space. Ji *et al.* proposed an algorithm to take advantage of this property, where the tracking of 3D isosurfaces is reduced to generation and slicing of isosurfaces in 4D space [11].

To satisfy the assumption that there exists spatial overlap among corresponding features, the data needs to be sampled at a higher temporal resolution, which requires much larger storage space. To relax this constraint, different algorithms have been proposed. Reinders *et al.* proposed a tracking framework by predicting the variation of feature attributes, such as centroid, shape, size, orientation, etc. to estimate the temporal behavior of the features [20]. Bauer and Peikert transformed the input data into scale space, and then the non-overlaping features in the original scale can be tracked by searching in different scales [3].

One difference between feature tracking in scientific data and tracking video data is that the features in scientific data can be merged from a set of features or split into disjoint features over time. By taking advantage of the fact that the corresponding sets of disjoint features often have similar shapes, Ji and Shen utilized the Earth Mover Distance [21] and decision trees to match sets of features without spatial overlap [10].

All of the related works above assumed that it is possible to perform feature segmentation before tracking takes place. However, except isosurfaces, segmentation of features may not be trivial. To track features in 2D unsteady flow, Theisel and Seidel [26] proposed a concept that transforms the original dynamic flow data into a steady 3D vector field called Feature Flow Field, where the tracking in the original domain is equivalent to creating streamlines in the new vector field. Joshi and Rheingans [5] proposed a texture-based tracking method by treating the time-varying volumes as 3D textures and tracking the features by mapping overlapped subvolumes with similar textural patterns.

3 TAC-based Distance Fields

Detail about the TAC-based distance field is described in this section, including the representation of features as TACs, the algorithm of DTW, and the construction of the TAC-based distance field.

3.1 TAC-based Feature Description

To allow a detailed analysis of time-varying data, it is necessary to separate the data points that demonstrate different temporal trends. Here we create a framework to allow visualization of time-varying features modeled as TACs. TAC-like features are commonly encountered in medical applications since the time-varying signals captured by medical imaging devices already can be thought of as a set of time series. For data from scientific simulations, sometimes additional transformation may be needed to convert the raw data into TACs. A vector field, for example, needs to be converted into a scalar field such as velocity magnitudes, from which the TACs can be created.

To specify a feature, a TAC can be manually edited by the users or automatically extracted from the input data. As an example, in our experiment we generated the feature TAC by first clustering the time series from all voxels. We implement the Lloyd algorithm or called K-mean [14] for clustering. The distance metric during the clustering is L_2 norm. After the TACs have been grouped into distinct sets, one of the mean TACs from the K clusters is chosen as the feature of interest. In our case studies, K was set to 20 in order to capture the time shift of the salient patterns even though the number of distinct pattern shapes is often fewer. The user can switch the

TACs from one cluster to another and visualize the spatial distributions of different temporal features. This allows the user to perform detailed analysis of time-varying data in a more systematic way.

3.2 DTW Distance Metric

Once the feature has been specified as a TAC, in order to detect the regions that exhibit a similar temporal trend, a distance metric is needed to measure the dissimilarity to the feature TAC at each data point. To design such a distance metric for two TACs, we first use DTW to align them with the smallest distortion and then use L_2 distance between the warped TACs as the final distance. We call this distance metric as the *DTW distance metric*. DTW is an algorithm to nonlinearly align two data sequences, which has also been used in some other disciplines [13]. As mentioned previously, the reason for us to employ the DTW distance metric is to account for the possible time shift and shape deformation that can occur to the TACs as the feature travels through space and time.

To compute the DTW distance between two data sequences, three constraints need to be considered, which results in a dynamic programming algorithm. First, the first and last data points in one sequence are always mapped to the first and last data points in the other sequence, respectively. Second, the warping should preserve the original order of data in the sequence. Third, two adjacent data points in the same sequence cannot be mapped to non-adjacent locations.

Based on these constraints, given two data sequences denoted as $x[1\ldots m]$ and $y[1\ldots n]$, DTW can be modeled as a recursive equation:

For $i = 1\ldots m,\ j = 1\ldots n$,

$$D[i,j] = dist(x[i],y[j]) + \begin{cases} 0 & i = 1 \& j = 1 \\ D[i-1,j] & i > 1 \& j = 1 \\ D[i,j-1] & i = 1 \& j > 1 \\ D_2(i,j) & \text{otherwise} \end{cases} \quad (1)$$

where

$$D_2(i,j) = min\{D[i-1,j], D[i-1,j-1], D[i,j-1]\}. \quad (2)$$

Here $D[i,j]$ denotes the optimal distance after the warping between two subsequences $x[1\ldots i]$ and $y[1\ldots j]$ has been done, and $dist(\cdot,\cdot)$ denotes the distance function between two data items,which is the squared difference here. By storing the distances $D[i,j]$ for $i = 1\ldots m$ and $j = 1\ldots n$ in a 2D table D, the entries can be computed according to Equation 1 from $D[1,1]$ to $D[m,n]$ in scanline order. The warping can be then found by backtracking from $D[m,n]$ to $D[1,1]$. Figure 1 presents the result of DTW between two synthesized data sequences. The table D is plotted as a 2D image, and the backtracked warping is plotted as the red line. The DTW distance metric can be then defined as the entry $D[m,n]$, with a slight modification presented in the next subsection.

The main benefit of DTW is that it can consider both the shift and deformation of the time series, while other distance metrics either consider no transformation at all (e.g. L_1 and L_2 distance metrics) or only the shift of time steps (e.g. cross-correlation). Meanwhile, DTW allows the shift and deformation to be non-linear. The main drawback of DTW is its computation complexity, which is $O(mn)$ to align two sequences of lengths m and n. We accelerate the computation of DTW using GPUs, as described in Section 5.

3.3 Construction of TAC-based Distance Field

By computing the DTW distance from every data point to the feature TAC, the original time-varying volume is transformed into a distance field. This distance field can be visualized using conventional scalar data visualization techniques such as isosurfaces or DVR, described in the next section.

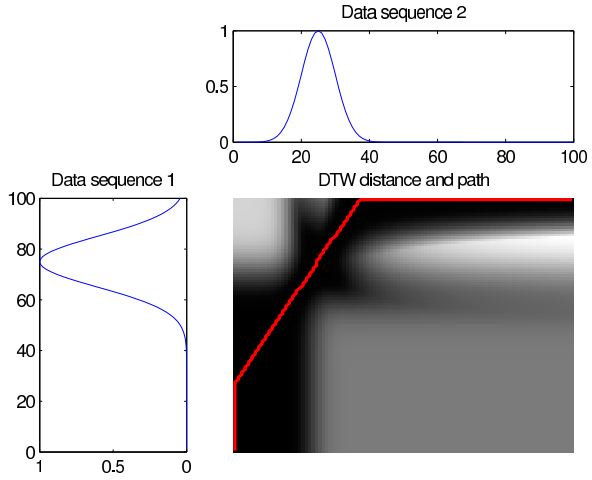


Figure 1: DTW between two synthesized data sequences. The two data sequences are normalized Gaussian functions with different means and variances. The warping is plotted as a red line in the lower right subfigure. The distance table is drawn as an image.

In addition to visualizing the distance field, which is to show the spatial distribution of data points that exhibit a similar temporal trend to the feature TAC, additional information such as when the main feature in the feature TAC moves through space can also be revealed. Using the seismic waves as an example, it is important to know when the peak of the earthquake wave passes through different areas. To highlight this information, along with the feature TAC the user can also indicate where the main feature of interest, the peak of the earthquake wave in this case, appears on the TAC by providing two bounding time steps, called *feature time range*. Since DTW will warp the time sequences when matching two TACs, the user-specified feature time range can be warped according to the local data point's TAC. The warped feature time range then indicates when the main feature occurs at the local point. We call the centroid of the warped time range a *feature time step*.

In order to emphasize the pattern within the feature time range, instead of $D[m,n]$, we modify the DTW distance metric as the accumulation of $dist(x[i],y[j])$ along the warping (i,j) where i is bounded by the feature time range. Besides, when comparing a feature TAC with two TACs that have similar patterns but different lengths of the warping, different accumulated sums will be computed as the results. To reduce the bias by the length, the sum is divided by both the length of the warping and that of the feature time range.

To summarize, in the TAC-based distance field, two attributes are computed for each data point: the DTW distance to the feature TAC and the feature time step when the main feature of interest occurs at that point. The DTW distance records the time-invariant shape dissimilarity between the point's TAC and the feature TAC. The feature time step records the estimated time when the feature passes through that point. By considering both attributes during the visualization process, spatial and temporal distribution of the feature can be revealed.

4 Visualization of TAC-based Distance Field

Once the TAC-based distance field has been constructed, the time-varying feature can be visualized using various visualization techniques. In this section, we use a synthetic data set to illustrate the concepts. This data set was created by moving a 3D Gaussian ker-

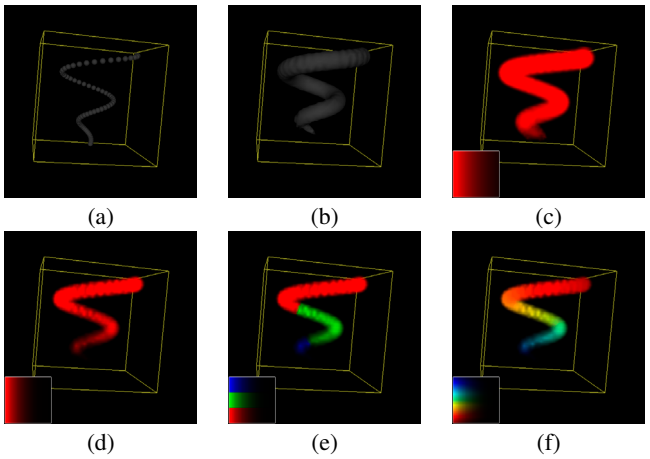


Figure 2: Visualization of the TAC-based distance field of a synthesized data set. This data set is a Gausian kernel moving over time. (a): the centers of the Gaussian kernels. (b): the isosurface. (c) and (d): DVR images where the opacities are decided according to Equation 3 with the drop-off parameter p of 8 and 16, respectively. (e) and (f): DVR images with discrete and continuous, respectively, color textures.

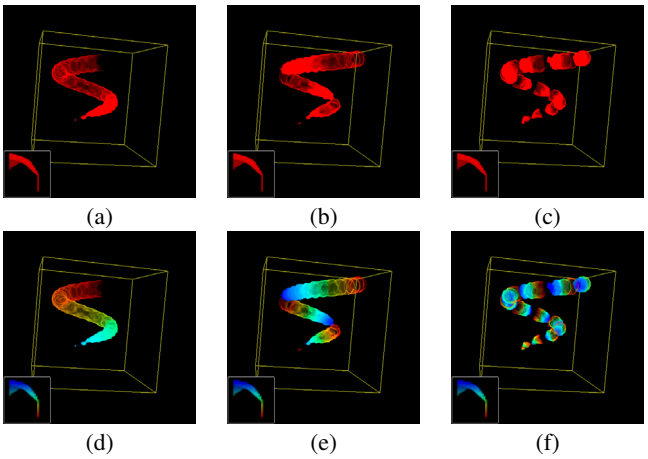


Figure 3: Creating motion trail via DVR. (a), (b) and (c): repeated patterns in different frequencies via discrete color map. (d), (e) and (f): repeated patterns in different frequencies via continuous color map.

nel in different locations over time. The path of the Gaussian kernel is shown in Figure 2 (a). The feature TAC is a 1D Gaussian kernel.

Isosurface To locate voxels that have similar TACs to the feature TAC, isosurfaces of small DTW distances can be extracted. Figure 2 (b) shows an isosurface of the test data set. The isosurface reveals the moving path of the Gaussian kernel, although no information about when the kernel moves through each data point is shown.

Volume Rendering via 1D and 2D Transfer Functions To reveal the temporal information of the feature, we can use DVR to display both the DTW distances and the feature time steps stored in the distance field. This is done using 2D transfer functions to map them to opacity and color.

When mapping the DTW distance to opacity, our goal is to hide the regions that have very dissimilar TACs to the feature TAC. We use the following equation to calculate the alpha channel for each voxel:

$$\alpha = (1-s)^p \tag{3}$$

where α is the opacity, s is the DTW distance normalized to [0,1] range, and p is a parameter to control the drop-off rate of the opacity as the voxel becomes more dissimilar to the feature TAC. Figure 2 (c) and (d) show images obtained using different drop-off values. Regions of different sizes are shown to indicate the levels of similarity to the feature.

To map the feature time step to color, there exist several possibilities. One way is to map the feature time step by a 1D transfer function. Figures 2 (e) and (f) present the DVR images via simple 1D transfer functions that map 'past' to 'red,' 'current' to 'green,' and 'future' to 'blue.' Note that in the figures, the 2D transfer functions are displayed in a small window at the lower left corner. Because the alpha value will drop to a small value near zero as the distance to the feature TAC gets larger, only about a quarter of the color bar is visible.

Another option is to use 2D textures to create different styles of images. The textures can be indexed by using the DTW distance as the row index and the feature time step as the column index. One desired effect is to show the feature's motion trail. To create this effect, the 2D texture can have texels with non-zero opacities arranged as a band covering the distances of interest. The opacities on those texels are controlled by the feature time step to emphasize a certain time range. Figure 3 (a) shows an image generated with this 2D texture. In addition, similar to conventional texture mapping, the texture coordinates can be scaled to create repeated patterns. In OpenGL, for instance, the warping parameters of the 2D texture can be set as `GL_REPEAT`, and the row indices of the texture coordinates can be multiplied by a user-specified scale s_t to control the frequencies of the patterns. Figures 3 (b) and (c) show images obtained by using the scale s_t as 4 and 16, respectively. Figures 3 (d), (e) and (f) show the images created by using a color texture with scale s_t as 1, 4 and 16, respectively. Also, effects of contour volumes can be created by using a texture whose opacity channel contains horizontal strips. The examples of contour volumes will be shown in Section 6.

Finally, even though our goal is not to rely on animations to visualize time-varying data, animated effect can be easily incorporated in our framework. By periodically shifting the column index when accessing the transfer functions, a dynamic trace of the feature can be produced.

5 GPU-based DTW Implementation

As mentioned in Section 3, the computation of DTW can be expensive. In this section, we describe how GPUs can be used to accelerate the DTW compuation using nVidia's CUDA library [16]. Other than accelerating the warping between the feature TAC and one input TAC, we also want to increase the throughput by computing the warping of more TACs to the feature TAC in parallel.

In principle, DTW between the feature TAC and all TACs can be accelerated by executing the TACs on multithreads in parallel. The dynamic programming table between each TAC and the feature TAC can be updated by a thread processor on GPUs. While this cannot be achieved by the shaders in GLSL or other shading language because these languages prohibit both the allocation of a large 2D array for dynamic programming and output to arbitrary memory location, both are supported in the thread programming in CUDA. The following properties make DTW suitable to be implemented on GPUs. First, the execution of DTW between each pair of TACs is independent of that of other pairs. Second, according to Equation 1, elements with the same index will be simultaneously accessed by all threads from all tables; thus this programming flow is suitable for the SIMD architecture of GPUs.

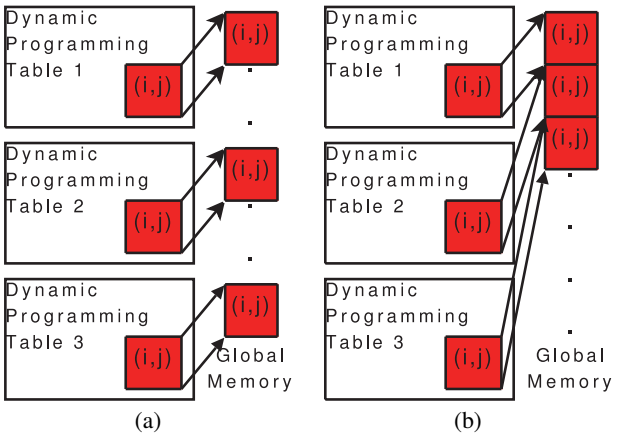


Figure 4: The memory layout of the dynamic programming tables on CUDA. (a): The layout without interleaving. (b): The layout with interleaving, where the memory requests to the (i, j)-th elements can be reduced to one request to a coalesced memory space.

In practice, however, special care should be taken when accessing the tables for dynamic programming due to the high memory access penalty on graphic hardware. Since the warping of feature time ranges needs the resulting table from dynamic programming, the tables for all TACs must be maintained when executing DTW, requiring an enormous memory space that can be only stored in the device memory (called *global memory* in CUDA terminology). Accessing the global memory, however, is slow since there is no cache. If these tables are unfolded and cascaded together to form a long vector, one instruction to access the elements of the same index will issue multiple memory requests to the global memory, causing high memory access penalty. In Figure 4 (a), for instance, accessing the (i, j)-th elements in all three tables will issue three memory requests to noncontiguous memory addresses. In our experiment, such GPU-based implementation was even slower than the CPU-based implementation on a quad-core system, in spite that GPUs can simultaneously execute DTW for more TACs. This result is consistent with the finding by Poli *et al.* that their CUDA-based DTW implementation was only 5% faster than a system with one Dual Core AMD Opteron processor [18].

To address the issue, we interleave the tables for all TACs to reduce the memory acess penalty. All tables are organized by cascading the elements with the same index into a contiguous memory address. Therefore, the instruction to access the elements of the same index will issue only one memory request. In Figure 4 (b), for instance, accessing the (i, j)-th elements in all three tables will only require one memory request to a coalesced memory space. This modification makes the GPU-based implementation at least ten times faster than the CPU-based implementation in our experiment. For example, for a data set of $150 \times 62 \times 62$ voxels with 200 time steps, the CPU-based implementation took 135 seconds on a workstation with two Dual Core AMD Opteron processors and 8GB system memory, while this optimized GPU-based implementation took only 10 seconds on an nVidia GeForce 8800GTX card.

6 Case Studies

To demonstrate the utility of our framework, we applied it to data generated from two different applications. One data set is TeraShake 2.1 [17], the benchmark for the IEEE Visualization Design Contest 2006. The other data set is the benchmark for the IEEE Visualization Design Contest 2008, generated by a simulation that models ionization front instability [28].

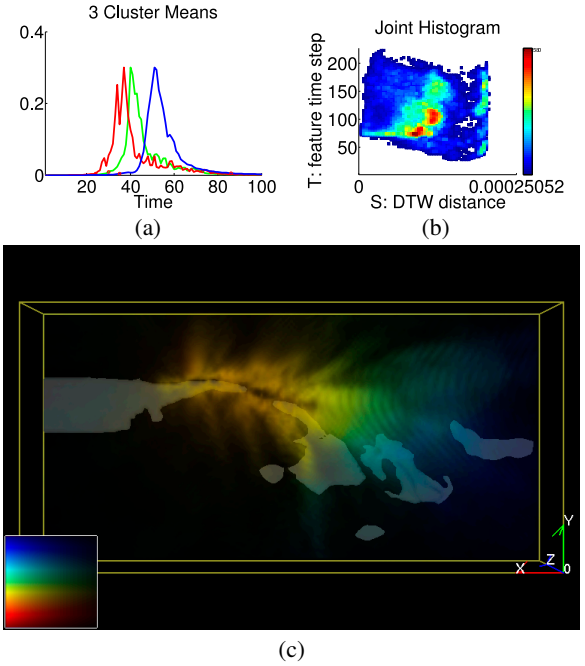


Figure 5: Visualization of the magnitude field of the data set TeraShake 2.1. (a): the mean TACs from three clusters. The selected feature TAC is plotted in blue. The values of the three TACs are normalized to match that of the selected feature TAC. (b): the joint histogram. (c): a DVR image. The parameter p in Equation 3 is 8. The basin is plotted as the grey patches.

6.1 Earthquake Wave Simulation

The data set TeraShake 2.1 records a simulated earthquake in 250 seconds on the Southern San Andreas Fault. During the first 60 seconds of simulation, a magnitude 7.7 earthquake began from South of Palm Springs toward Northwest on the San Andreas Fault, then the fault rupture stopped while the earthquake waves kept propagating. The original data set is a time-varying vector field, where each cell record the 3D velocity of the particle motion in this cell. The resolution of the data set is $750 \times 375 \times 100$ voxels with 227 time steps. In our test, the original data set was down-sampled, to reduce the overall storage, by approximately a factor of four in all three dimensions yielding a data set of $188 \times 94 \times 25$ voxels with 227 time steps.

The first property in which we were interested is the propagation of the seismic energy. We used the magnitude of seismic wave to create the TACs. Figure 5 (a) presents the mean TACs from the clusters, where the chosen feature TAC is plotted in blue. Only three mean TACs are plotted in order to reduce the visual clutter. Their values are normalized to match that of the feature TAC. It shows that all three mean TACs contain a peak in different locations, followed by a tailing signal of different lengths. This difference suggests that using DTW to calculate the dissimilarity will obtain a smaller distance than L_1 distance, L_2 distance, or the cross-correlation metrics.

Figure 5 (b) shows the joint histogram of the feature time step and DTW distance. The DTW distance and the feature time step are represented as s (horizontal) and t (vertical) coordinates, respectively. The occurrence of each bin is normalized to [0, 1] and then mapped to color according to the displayed color bar, which is from 'blue' to 'red.' To reduce the visual clutter, the bins whose occurrences are less than 1% of the maximal occurrence are not shown. In the joint histogram, after the 75-th time step, the horizontal distance between the t-axis and the first non-empty bin is increasing as

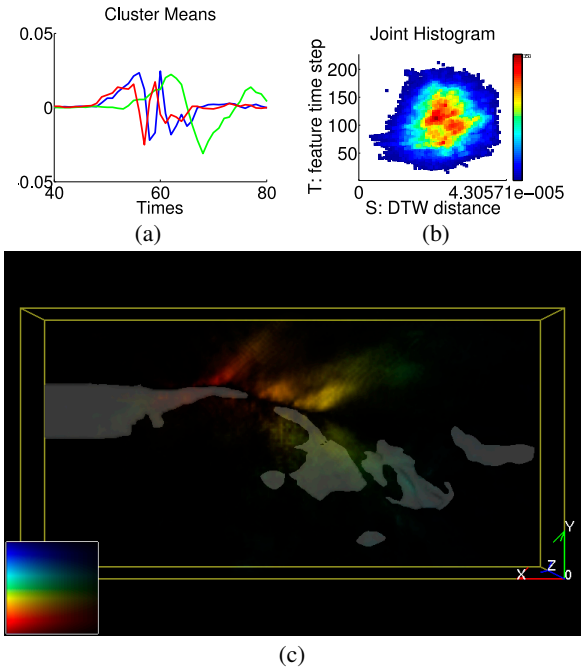


Figure 6: Visualization of the z components in the data set TeraShake 2.1. (a): the mean TACs from three clusters. The selected feature TAC is plotted in blue. (b): the joint histogram. (c): a DVR image. The parameter p in Equation 3 is 8. The basin is plotted as the grey patches.

the t coordinate increases, which means the number of voxels that contain wave magnitudes closer to the maximal magnitude of the feature TAC is decreasing over time, indicating that the strength of the earthquake is reducing after the 75-th time step. This is consistent with the simulation setting that the fault rupture was terminated at the 60th second or equivalently the 55th time step. Figure 5 (c) presents the DVR image of the TAC-based distance field. Here the transition of color from yellow to green indicates the propagation paths of the earthquake energy. The wavefronts of the reflected waves are also depicted. Figure 5 (c) also plots the basin surfaces to reveal the correlation between the wave propagation and the basin structure. In the yellow region shown the image, a transparent strip follows, which is consistent with the structure of the basin.

The types of the seismic waves are of interest to the scientists. There are four types of seismic waves: *P*-, *S*-, *L*- and *R*- waves. The particles of the primary or compressional waves (*P*-waves) are moved in parallel to the propagation direction. The particles along the secondary or shear waves (*S*-waves) traverse perpendicular to both the propagation direction and the Earth. In the surface waves, the long waves (*L*-waves) move the particles in a direction perpendicular to the propagation direction and parallel to the Earth, while the motion of the Rayleigh waves (*R*-waves) is usually elliptical on a vertical plane. Details about the seismic waves can be found in several sources, such as the online demo created by Braile[4]. Since the particles along the *P* waves can only move horizontally, the vertical component of the velocity vectors (the z components in this data set) along the *P* waves should be small. Hence we used the z component of the particle motion to construct the TAC in order to distinguish the *P* waves from other types of waves. Figure 6 (a) presents the three mean TACs from selected clusters, where the chosen feature TAC is plotted in blue. It can be seen that the three mean TACs contain a similar pattern with different durations. In order to detect the region with such a pattern in the highest magnitude, we normalized the value of the selected feature TAC to $[-z_{max}, +z_{max}]$

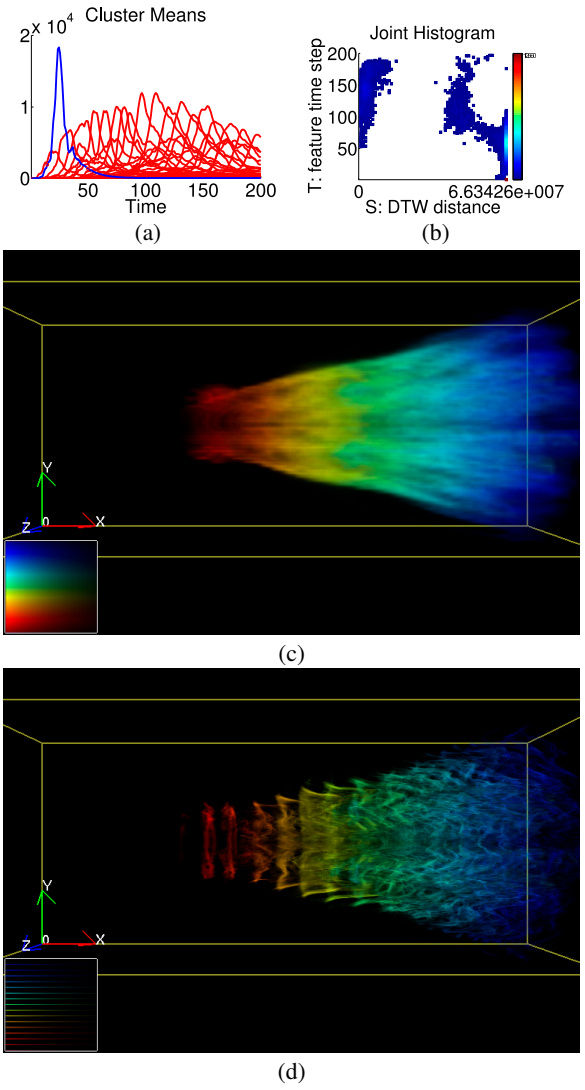
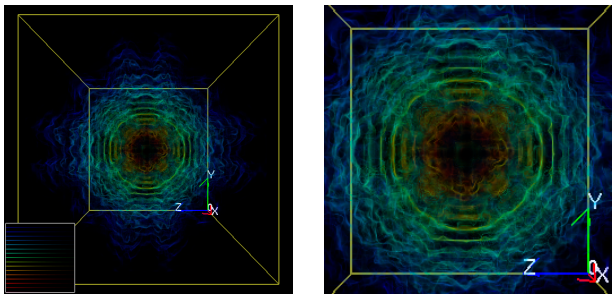


Figure 7: Visualization of the ionization front instability simulation. (a): the mean TACs of all clusters (red) and the feature TAC (blue). (b): the joint histogram of the respective TAC-based distance field. (c): a DVR image with continuous color map. (d): a DVR image with contour volumes. The parameter p in Equation 3 is 8 in both (c) and (d).

where z_{max} is the maximal z component of all vectors. The corresponding joint histogram is shown in Figure 6 (b). Figure 6 (c) shows the DVR image. Compared to Figure 5 (c), it can be seen that the color mainly changes from red to yellow, which indicates that the pattern of the feature TAC does not exist near the end of the simulation. Besides, the transparent strip that follows the basin also exists in Figure 6 (c).

6.2 Ionization Front Instability

Our second case study is to visualize the turbulence of ionization front instability. This data set was used for the 2008 IEEE Visualization Design Contest, whose aim is to reveal the influence of the I-front instabilities during the formation of the structure in early universe [28]. The first stars were initially very massive (more than 100 solar masses), causing high temperatures on the surface(greater than 100,000 K) and the propagation of the light energy as UV radiation. In a speed slower than light, these UV photons advanced



(a) (b)

Figure 8: Visualization of the ionization front instability simulation. Compared to Figure 7, the subfigures here were rendered along a viewing direction parallel to the x axis, thus providing better cue about the symmetricity of the turbulence structure. (a): the DVR image. The saturation parameter in Equation 3 is 8. (b): the enlarged image of subfigure (a).

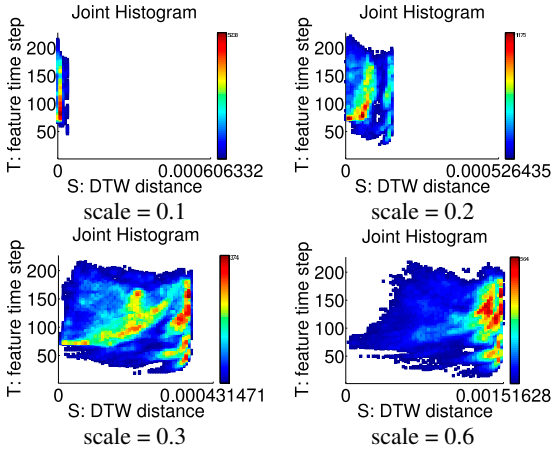


Figure 9: The joint histograms under the same TAC in different scale. The test data set is the magnitude field of the data set TeraShake 2.1. The feature TAC is the blue curve in Figure 5 (a).

behind an abrupt wall of radiation known as the *ionization front* (I-front), which formed a boundary between the hot ionized gas and the cold neutral gas beyond the front. Because the propagation speed of the radiation wave was reduced over time, the hot ionized gas pushed past the front, which drove a shock that led to instabilities, thus sweeping and deforming the planar I-front over a spherical blob of gas. A detailed description about this data set can be found on the website of this contest [28] or the article by Whalen and Norman [29].

The data set is a multifield time-varying volume. Each voxel records 10 fields and a 3D velocity vector. This data set contains 200 time steps of $600 \times 248 \times 248$ voxels. We downsampled the data set by a factor of four in all three dimensions to reduce the storage. In this case study, we only used the velocity vector field to reveal the turbulence structure of the sweeping I-front. Because the scale of the turbulence is related to the scale of the curl field, we converted the velocity field into the curl field, from which the TACs were created.

Figure 7 (a) shows the feature TAC in blue and other selected TACs. This selected TACs were extracted via K-mean clustering. It can be seen that most mean TACs show impulses at different time steps and with different durations. We normalized the value of the feature TAC in order to capture the regions with the highest curl magnitude. Figure 7 (b) shows the joint histogram of the corresponding TAC-based distance field. In the joint histogram, the numbers of the shown bins near the t axis increase after the 50-th time step. This indicates that the turbulence mainly occurred after the 50th time step, and its structure was expanding over time.

We are mainly interested in the spatiotemporal structure of the turbulence. Figure 7 (c) presents a DVR image. The colors of voxels from left, the origin, to right ($+x$) vary from red to blue, indicating the propagation path of the turbulence. This is consistent with the simulation setting that the I-front propagated along the x axis. While Figure 7 (c) provides an overview of the propagation, it cannot reveal the detail of the turbulence structure. Thus a texture with horizontal strips was applied to create contour volumes. The result is shown in Figure 7 (d). The contours show that the structure of the turbulence was distorting as the I-front propagated. The symmetry of the turbulent structure is also of interest. Figure 8 contains subfigures rendered from a viewing direction parallel to the $x-$ axis. The contours in Figure 8 (a) and the enlarged image Figure 8 (b) indicate that, even though there exists minor non-symmetry in Figure 8 (b), the major structure of the turbulence is symmetric about the x axis.

7 Conclusion

7.1 Summary

In this paper, we present a new framework for visualizing time-varying data sets. We are interested in those features that can be modeled as time series patterns. Since the shapes of the patterns could be shifted, stretched, or compressed over time, we use DTW as the time-invariant distance metric. By converting the original sequence of volumes into a distance field to a specified feature, we can visualize the spatiotemporal behavior of the feature via volume rendering.

The benefit of our framework is multifold. By using DTW, The distance between feature descriptors represented as TACs can be computed invariantly under time warp. DTW also provides a scheme to measure temporal inference to the feature. An overview of the time-varying feature can be displayed in a single image rather than in animation, and different visualization styles can be created.

7.2 Limitations

Despite the advantages, there exist several limitations in our current framework. One issue is in the specification of the feature TAC. It is noteworthy that DTW metric does not account for the change of scales in the TACs; therefore, in addition to the shape of the feature TAC, the specification of the scale is also crucial. Figure lists the joint histograms for the the magnitude field of the data set TeraShake 2.1 where the feature TAC is the blue curve in Figure 5 (a). Here, however, the scale of the feature TAC is normalized to different values. It can be seen that the distributions in the joint histograms are skewed toward the t axis as the scale decreases. This distortion can create different DVR images, requiring further user interaction to specify different textures as the 2D transfer functions to filter out the desired regions.

Another issue is the time complexity of DTW. Even though we have gained a huge acceleration using GPUs, for very large data sets, it is still too slow for interactive exploration. To reduce the number of TACs, in this paper we downsampled the spatial resolution of the data set. In the future, we will study other ways such as hierarchical data structures to reduce the number of TACs to be computed.

Meanwhile, the TAC-based distance field cannot be created until the entire volume of time sequences becomes available. Thus our method could be inadequate for online visualization. Also, our algorithm prefers data that exhibits spatial coherence. For features

with little spatial overlap in adjacent time steps, discontinuous patterns may appear in the visualization.

7.3 Future Works

In the future, in order to define more types of features, the use of multi-variants TACs rather than 1D scalars should be studied. In the applications where TACs with the same shape but in different scales are considered equivalent, modified DTW such as Derivative Dynamic Time Warping [13] will be needed. Moreover, currently only the DTW distance and the feature time step are considered. The feature time ranges can also be embedded in the rendering process to create more effective visualizations. Yet another direction is to transform the distance field into the scale space, where a feature can be detected across different scales. One example of such techniques is the SIFT feature detector [15] used in computer vision applications.

Acknowledgements

We would like to thank for the useful discussions with Jon Woodring and Thomas Kerwin, and the anonymous reviewers for their helpful comments. This work was supported in part by NSF ITR Grant ACI-0325934, NSF RI Grant CNS-0403342, NSF Career Award CCF-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 *EuroVis '07: Proceeding of the Joint Eurographics - IEEE VGTC Symposium on Visualization 2007*, pages 115–122, 2007.

[2] J. C. Anderson, L. Gosink, M. A. Duchaineau, and K. I. Joy. Feature identification and extraction in function fields. In *EuroVis '07: Proceeding of the Joint Eurographics - IEEE VGTC Symposium on Visualization 2007*, pages 195–201, May 2007.

[3] D. Bauer and R. Peikert. Vortex tracking in scale-space. In *VisSym '02: Proceedings of the Joint Eurographics-IEEE TCVG Symposium on Visualization 2002*, pages 140–147, 2002.

[4] L. Braile. Seismic wave demonstrations and animations, December 2005. http://web.ics.purdue.edu/ braile/edumod/waves/WaveDemo.htm.

[5] J. Caban, A. Joshi, and P. Rheingans. Texture-based feature tracking for effective time-varying data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1472–1479, Nov.-Dec. 2007.

[6] Z. Fang, T. Möller, G. Hamarneh, and A. Celler. Visualization and exploration of time-varying medical image data sets. In *GI '07: Proceedings of Graphics Interface 2007*, pages 281–288, 2007.

[7] H. Guo, R. Renaut, K. Chen, and E. Reiman. Clustering huge data sets for parametric pet imaging. *BioSystems*, 71(1–2):81–92, 2003.

[8] S. Guthe and W. Straßer. Real-time decompression and visualization of animated volume data. In *VIS '01: Proceedings of the IEEE Visualization 2001*, 2001.

[9] T. J. Jankun-Kelly and K.-L. Ma. A study of transfer function generation for time-varying volume data. In *VG '01: Proceedings of the Joint IEEE TCVG and Eurographics Workshop on Volume Graphics 2001*, 2001.

[10] G. Ji and H.-W. Shen. Feature tracking using earth mover's distance and global optimization. In *Pacific Graphics 2006*, 2006.

[11] G. Ji, H.-W. Shen, and R. Wenger. Volume tracking using higher dimensional isosurfacing. In *VIS '03: Proceedings of the IEEE Visualization 2003*, pages 209–216, 2003.

[12] A. Joshi and P. Rheingans. Illustration-inspired techniques for visualizing time-varying data. In *VIS '05: Proceedings of the IEEE Visualization 2005*, pages 679–686, 2005.

[13] E. J. Keogh and M. J. Pazzani. Derivative dynamic time warping. In *SDM '01: Proceedings of the First SIAM International Conference on Data Mining 2001*, 2001.

[14] S. P. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, Mar. 1982.

[15] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov. 2004.

[16] NVIDIA Corporation. *NVIDIA CUDA Compute Unified Device Architecture Programming Guide 2.0*, 2008.

[17] K. B. Olsen, S. M. Day, J. B. Minster, Y. Cui, A. Chourasia, M. Faerman, R. Moore, P. Maechling, and T. Jordan. Strong shaking in los angeles expected from southern san andreas earthquake. *GEOPHYSICAL RESEARCH LETTERS*, 33:L07305, 2006.

[18] G. Poli, A. L. M. Levada, J. F. Mari, and J. H. Saito. Voice command recognition with dynamic time warping (dtw) using graphics processing units (gpu) with compute unified device architecture (cuda). In *SBAC-PAD '07: Proceedings of the 19th International Symposium on Computer Architecture and High Performance Computing 2007*, pages 19–25, Oct. 2007.

[19] F. H. Post, F. J. Post, T. V. Walsum, and D. Silver. Iconic techniques for feature visualization. In *VIS '95: Proceedings of the IEEE Visualization '95*, pages 288–295, 1995.

[20] F. Reinders, F. H. Post, and H. J. Spoelder. Attribute-based feature tracking. In E. Gröller, H. Löffelmann, and W. Ribarsky, editors, *VisSym '99: Proceedings of the Joint Eurographics-IEEE TCVG Symposium On Visualization '99*, pages 63–72, 1999.

[21] 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, Nov. 2000.

[22] H.-W. Shen, L.-J. Chiang, and K.-L. Ma. A fast volume rendering algorithm for time-varying fields using a time-space partitioning (tsp) tree. In *VIS '99: Proceedings of the IEEE Visualization '99*, pages 371–377, 1999.

[23] D. Silver and X. Wang. Volume tracking. In *VIS '96: Proceedings of the IEEE Visualization '96*, pages 157–164, 1996.

[24] D. Silver and X. Wang. Tracking scalar features in unstructured datasets. In *VIS '98: Proceedings of the IEEE Visualization '98*, pages 79–86, 1998.

[25] N. A. Svakhine, Y. Jang, D. S. Ebert, and K. P. Gaither. Illustration and photography inspired visualization of flows and volumes. In *VIS '05: Proceedings of the IEEE Visualization 2005*, pages 687–694, 2005.

[26] H. Theisel and H.-P. Seidel. Feature flow fields. In *VisSym '03: Proceedings of the Joint Eurographics-IEEE TCVG Symposium on Visualization 2003*, pages 141–148, 2003.

[27] C. Wang, H. Yu, and K.-L. Ma. Importance-driven time-varying data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1547–1554, Nov.-Dec. 2008.

[28] D. Whalen and M. L. Norman. Competition data set and description. 2008 IEEE Visualization Design Contest, 2008. http://vis.computer.org/VisWeek2008/vis/contests.html.

[29] D. Whalen and M. L. Norman. Ionization front instabilities in primordial h ii regions. *The Astrophysical Journal*, 673:664–675, Feb. 2008.

[30] K.-P. Wong, D. Feng, S. Meikle, and M. Fulham. Segmentation of dynamic pet images using cluster analysis. *IEEE Transactions on Nuclear Science*, 49(1):200–207, Feb. 2002.

[31] J. Woodring and H.-W. Shen. Chronovolumes: A direct rendering technique for visualizing time-varying data. In *VG '03: Proceedings of the Third International Workshop on Volume Graphics 2003*, pages 27–34, 2003.

[32] J. Woodring and H.-W. Shen. Multiscale time activity data exploration via temporal clustering visualization spreadsheet. *IEEE Transactions on Visualization and Computer Graphics*, 15(1):123–137, Jan.-Feb. 2009.

[33] J. Woodring, C. Wang, and H.-W. Shen. High dimensional direct rendering of time-varying volumetric data. In *VIS '03: Proceedings of the IEEE Visualization 2003*, pages 417–424, 2003.

[34] H. Younesy, T. Möller, and H. Carr. Visualization of time-varying volumetric data using differential time-histogram table. In *VG '05: Proceedings of the Fourth International Workshop on Volume Graphics 2005*, pages 21–224, June 2005.