

Graxels: Information Rich Primitives for the Visualization of Time-Dependent Spatial Data

S. Stoppel^{†1}, E. Hodneland^{‡2}, H. Hauser^{§1} and S. Bruckner^{¶1}

¹ University of Bergen, Norway
² Christian Michelsen Research, MedViz

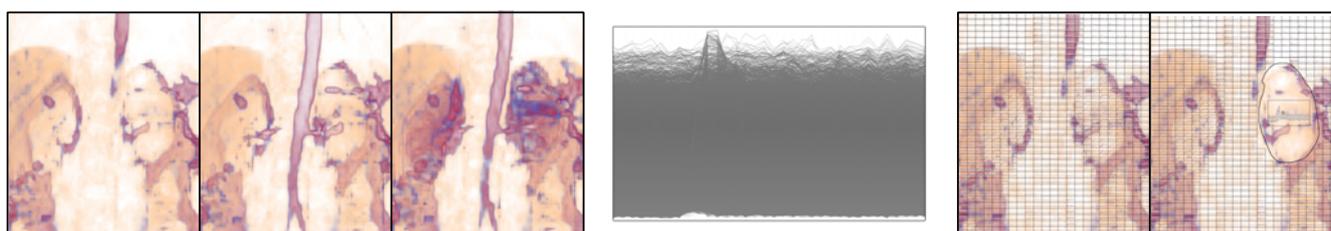


Figure 1: Conventional volume rendering shows time-dependent data as animation, which can convey qualitative temporal development well but does not provide an overview of the whole time span at once or the quantitative development of the data (left). Showing all time-intensity curves for a volume results in a cluttered appearance (middle). Our method (right) provides information about temporal developments in the form of small multiples in their spatial context. We allow multiple interactions in the spatial and value domain for further data exploration.

Abstract

Time-dependent volumetric data has important applications in areas as diverse as medicine, climatology, and engineering. However, the simultaneous quantitative assessment of spatial and temporal features is very challenging. Common visualization techniques show either the whole volume in one time step (for example using direct volume rendering) or let the user select a region of interest (ROI) for which a collection of time-intensity curves is shown. In this paper, we propose a novel approach that dynamically embeds quantitative detail views in a spatial layout. Inspired by the concept of small multiples, we introduce a new primitive graxel (graph pixel). Graxels are view dependent primitives of time-intensity graphs, generated on-the-fly by aggregating per-ray information over time and image regions. Our method enables the detailed feature-aligned visual analysis of time-dependent volume data and allows interactive refinement and filtering. Temporal behaviors like frequency relations, aperiodic or periodic oscillations and their spatial context are easily perceived with our method. We demonstrate the power of our approach using examples from medicine and the natural sciences.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation

1. Introduction

One of the most common and straight-forward strategies for visualizing time-dependent volume data is animation. While simple

and conceptually easy to understand, this approach is not well-suited for the detailed analysis of spatio-temporal patterns such as spatially-localized trends in the value domain. For example, in dynamic contrast enhanced MRI (DCE-MRI), the shape of a time-intensity curve in a particular region in space can provide important information about the malignancy of a tumor [KSEB*14]. For this reason, visualization methods typically combine an animated single time-step visualization with additional views that depict data values over time in the form of curve plots. While these additional

[†] e-mail: sergejsto@googlemail.com

[‡] e-mail: Erlend.Hodneland@uib.no

[§] e-mail: Helwig.Hauser@UiB.no

[¶] e-mail: Stefan.Bruckner@UiB.no

views facilitate the detailed analysis of time-dependent data, they do not provide a spatially-localized overview of temporal patterns.

For this reason, we propose a novel approach which provides a comprehensive overview of temporal relations directly embedded in a spatial context. Inspired by the concept of small multiples, we embed view-dependent interactive time-intensity graphs in a 3D visualization. Regions of interest can be easily generated by merging and splitting of these embedded graphs, and selections in time and space are possible. In contrast to previous approaches, our technique allows for the identification of frequency patterns, frequency relations and quantitative relations within time-dependent volumetric data. Furthermore, our method supports streaming data and is therefore well-suited for real-time applications such as the in-situ visualization of 4D ultrasound data.

The main contribution of this paper is a new information rich visualization primitive for the visual analysis of time-varying volume data. We demonstrate its applicability to the identification of temporally and spatially salient features. Furthermore, we show how our approach can be used for fast selection in the spatial and/or temporal domain. The strength of our approach is that it provides a spatially localized overview of time-intensity patterns in a single view. As such, our technique provides a useful addition to multi-view visual analysis systems for time-dependent data by establishing a link between the spatial and value domains which are usually treated separately.

2. Related Work

The visualization of temporal behavior has been extensively studied and the book by Aigner et al. [AMST11] presents a comprehensive overview of the topic. However, the majority of the existing techniques focus on data without a spatial reference or the visualization is decoupled from the spatial domain. We divide our related work section into three main categories: visualization in multiple views, aggregation of time for a visualization in a single view, and integrated depiction.

Multiple Views:

Multiple views are a common approach to visualize time-dependent data. By presenting different dimensions or abstractions in multiple windows and linking them via selections, a fast understanding of relationships can be provided. Roberts [Rob07] provided an extensive state of the art report on coordinated multiple views in exploratory visualization. Akiba et al. [AFM06] introduced an approach for the simultaneous classification of the entire time series and explored options for transfer function specification based on a time histogram. Chang et al. [CGK*07] developed WireVis, a set of coordinated visualizations based on identifying specific keywords within financial transactions. Fang et al. [FMHC07] presented three different similarity measures for time-intensity curves and encoded them in three different visualizations. Akiba et al. [AM07] introduced a three component interface, which abstracts the complexity of exploration. Wang et al. [WYM08] discussed an importance-driven approach to time-varying volume data visualization for enhancing the most essential aspects of time-varying data. Woodring et al. [WS09a] presented an approach which transforms data points into multiscale time series using the wavelet transformation. Angelelli et al. [ANOHH11] shown a technique for the exploration

and semi-automatic segmentation of 4D ultrasound data by defining regions of interest and exploring their temporal development.

One drawback of such methods is that attributes like time-intensity curves are shown without a spatial context or with only a limited indication of spatial relationships. When displaying the data in separate windows one has to keep track of the links between the windows. With a growing number of windows this task becomes increasingly difficult. While our method provides multiple views as well, we mainly focus on displaying temporal information directly in its spatial context, thus improving the linking of the value domain and the spatial domain.

Data Aggregation to a Single View:

Data aggregation techniques aim to provide an overview of the data in a single intuitive view. Aggregation can be applied to the temporal as well as to spatial domains and can vary from stacking of the data, for instance in the form of a space-time cube, to finding elaborate transfer functions in the spatio-temporal domain.

The concept of the space-time cube, introduced by Hägerstrand [Häg70], provides a seamless integration of the spatial and the temporal domain. In this approach the three-dimensional space inside the cube is used to represent spatio-temporal data. Kraak [Kra03] used space-time cubes to place multiple layers along the time axis, each of them encoding the data at a specific time point, to visualize the development of the data over time. Jankun-Kelly et al. [JKM01] discuss how the dynamic behavior of time-varying data may be captured by a single or a small set of transfer functions.

Woodring et al. [WS03] presented an algorithm which aggregates volumes over time producing a single view volume to capture the essence of multiple time steps in a sequence. Woodring et al. [WWS03] also introduced an alternative approach to treat 4D data as one hypervolume instead of a collection of multiple time steps. Furthermore, Woodring et al. [WS06] discussed a method for comparative visualization by using logical operations on the data from multiple time steps. Lee et al. [LS09a] presented an algorithm which estimates the appearance of new trends and extracts important trend relationships. Another approach by Lee et al. [LS09b] computes the similarity between a voxel's time series and a feature. The similarity measure is then used for the visualization. Woodring et al. [WS09b] provided a method for semi-automatically generating transfer functions for temporal data. Lee et al. [LCPS10] introduced CycleStack, an ultrasound video visualization technique, which reduced the cognitive overhead by blending video and signal together in a stack-like layout. Wang et al. [WYM10] presented an application-driven approach to compressing and rendering large-scale time-varying scientific simulation data. Hsu et al. [HMCM10] discussed a set of visualization techniques for presenting the evolution of 3D flow.

Most of these techniques focus on 2D data. While our approach is suitable for 2D data as well, we focus mainly on time-varying 3D data. Furthermore, we do not aggregate the data into a single volume representation but rather focus on a comprehensive encoding, which captures the temporal behavior but is still easy to interpret.

Integrated Depiction:

Showing information in its spatial context has a critical impact on the understanding of local properties. The variation of integrated

depictions is immense. It varies from abstract values, such as street names on a map, to complex representations such as graphs or intensity curves for locally selected features.

Moere [Moe04] demonstrated how the principle of self-organization and behavior simulation can be used to represent dynamic data evolution by extending the concept of information flocking. Mlejnek et al. [MEV*05] presented ProfileFlags. In this approach a flag is placed on top of a T2 map of the anatomic scan of the patella. The tissue profile directly under the flagpole is then displayed on the flag. Lu et al. [LS08] discussed an interactive storyboard approach by composing sample volume renderings and descriptive geometric primitives that are generated through data analysis processes. Shearer et al. [SOMK08] introduced pixelplexing, a technique for the effective display of time-varying data on small screens. In a sense, their approach is inverse to ours, as they aim to increase the spatial resolution by animating static visualizations. Tikhonova et al. [TCM10] presented an exploratory technique that enables a coherent classification of time-varying volume data. Forlines et al. [FW10] described an interactive system for the visualization of multivariate spatio-temporal data. Elmqvist et al. [EDG*08] presented ZAME, a nested visualization for the exploration of large graphs. For a comprehensive overview on composite visualization we refer to Javed and Elmqvist [JE12].

Our approach differs from the discussed ones in the sense that we use a novel information rich primitive in a screen filling way and not only on selected locations. Hence, we provide an overview of spatial and temporal aspects of the data that can guide subsequent selection or filtering operations.

3. Graxels

Tufte [Tuf01] emphasized the strength of small multiples for data comparison as "shrunk, high-density graphics" which are "drawn almost entirely with data-ink". As Tufte writes: "Our eyes can make a remarkable number of distinctions within a small area. ... 100 points in one square centimeter". Small multiples provide a highly efficient, dense, comparative visualization. However, small multiples are mostly used for abstract data without a spatial relation. We propose information rich primitives, which we call graxels (graph pixels) as a dynamic extension of traditional small multiples for the visualization of time-dependent spatial data. We embed time-intensity curves directly in the spatially-oriented visualization of the data set, thus providing a direct relation between spatial and temporal patterns. We further allow several interactions with the graxels, such as grouping and change of size. The size of the graxels can vary in different image regions and is therefore *non-uniform*. Furthermore, we introduce interaction and selection methods in both the spatial and the temporal domain of the data set.

There are several ways to aggregate data for the graxels. A straight forward strategy is to aggregate the data in object space. Here two conceptually different approaches can be considered: aggregation over a fixed volume subdivision or clustering of data points according to a similarity measure. These two approaches are comparable to the Eulerian and the Lagrangian views of fluid flow in the sense that the Eulerian approach focuses on fixed locations in space through which the flow passes, while the Lagrangian

perspective follows particles as they move. Following the Eulerian perspective, we could regularly subdivide the data and perform aggregation into supervoxels. The time-intensity curves could then be displayed in each supervoxel. This approach has the disadvantage of being too rigid and may lead to arbitrary cuts in the volume.

Following the Lagrangian approach, the data points could be clustered according to some similarity measure. A graxel could be created for each cluster. This approach, while seemingly appealing, carries some disadvantages. For one, the results are highly dependent on whether an appropriate metric can be found, which in many cases is non-trivial. Moreover, the resulting clusters may have highly irregular shapes, which makes them badly suited as a canvas for graxels. Furthermore, while clustering methods for streaming data are advancing rapidly, the clustering process still imposes a high computational load on the process.

Our idea is inspired by the Eulerian perspective, but we avoid the rigid subdivision by performing the data aggregation in image space. This allows for an intuitive change of the aggregation through zooming, translation and rotation of the volume. The data aggregation is conceptually similar to conventional volume ray casting. Since our *graxels* cover an area larger than just a pixel, they can employ a more effective visual encoding than simple color mapping. We provide an aggregation technique which conceptually offers the same interaction as interactive volume rendering. Instead of casting rays for each pixel, we perform *tile casting* for *graxels*. This approach avoids the original rigidity of the Eulerian approach and allows for a flexible and dynamic adjustment of the aggregation through well-known interactions.

3.1. Overview

The goal of our approach is to capture the temporal behavior of the data without losing too much detail information or missing important temporal or spatial features. Most of the data sets discussed in this paper are 4D scalar data sets with time as the fourth dimension and we are interested in the temporal development of these scalar values. This development is captured by time-intensity curves (TICs) at each voxel position.

The common way to visualize 3D scenes is to apply a viewing transformation to the data which is then projected into image space. A similar approach can be applied to 4D data as well by defining a transfer function which captures the temporal aspect as well as the intensity values of the data. Such a technique was presented by Balabanian et al. [BVMG08], where a temporal style transfer function was defined and used in volume rendering in order to show the temporal development of the data.

While such an approach provides a useful overview, it lacks detail information of the temporal development and can suffer from occlusions. In our approach we do not combine the time steps and map them to local properties. Instead, we aim to use a more familiar and direct representation of the TIC as a function graph. Our approach is conceptually similar to traditional ray casting. However, instead of treating each ray separately, we subdivide the image space into small tiles. We aggregate the TIC behind the tiles in a parallel process called tile casting. Each tile is further subdivided into a predefined number of slabs, dividing the volume in

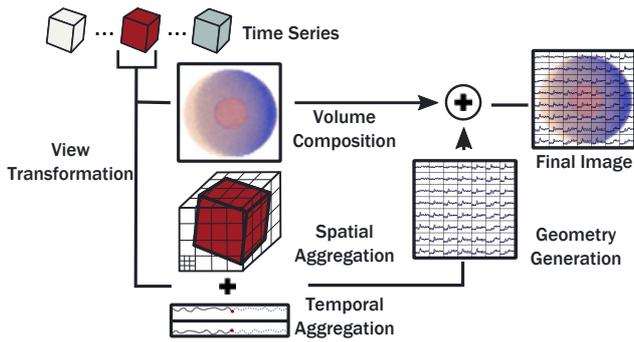


Figure 2: Each volume of the time series undergoes a view transformation. Afterwards the pipeline splits into the conventional volume composition and the tile casting. Each tile is subdivided into a user defined number of slabs. For each subdivision a representative set of TIC is computed and stored in the TICs cache. In the next step the curves for the graxels are generated. Finally, the volume rendering image and the graxels are optionally blended.

depth. For each of these slabs we compute the maximal, minimal and mean value of the subdivision. These values are used for the computation of the TIC of the corresponding areas in the volume. The tile casting is performed for each time step. An overview of our approach is shown in Figure 2. The tiles are grouped to graxels resulting in an aggregated set of minimal, maximal, and mean TICs. In the next step we generate the geometry of the TICs for each graxel. At this point the two parallel processes (volume composition and tile casting) join again and the TICs are displayed over a standard animated volume rendering. To investigate the data further, we allow interactions and selections in the spatial as well as in the temporal domain.

In the following we will discuss the main components of our approach. We subdivide the description of our method into three main parts: *tile casting*, *graxel rendering*, and *interaction*.

3.2. Tile Casting

As outlined in the previous section, our aim is to represent the temporal development in the value domain as familiar function graphs embedded in their respective spatial context. Our approach is to subdivide the image into a set of graxels. Each graxel contains a set of tiles. Each tile consists of fixed number of pixels arranged in a screen filling way, without overlapping each other. One can think of a graxel as a sensor which scans the volume as a projection of the covered image area. To increase the level of detail one can either use a finer sensor, i.e., a smaller number of pixels, or zoom into the data. If the camera position is inadequate, one can rotate the data to find the best viewing transformation, and the sensor will immediately adjust and record from the new viewing position.

We aim at being able to process streaming data as well as recorded data, and hence use an incremental approach which processes incoming volumes as they arrive. We summarize the tile casting as follows: For each time step the corresponding volume undergoes the volume rendering pipeline. In addition to conventional volume rendering, we aggregate the rays for each tile. In or-

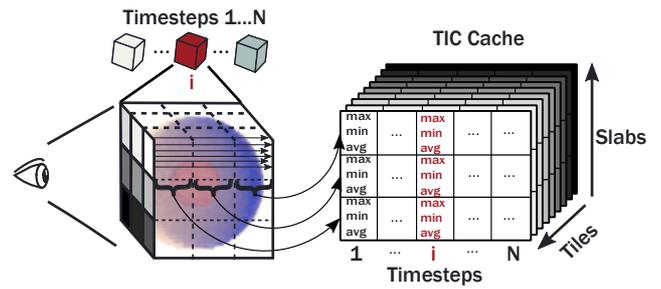


Figure 3: For each time step we shoot multiple rays for each tile. The rays are subdivided into slabs, the intensity values in those slabs are aggregated into an envelope of the maximal, minimal and the mean value. This process is done for each time step.

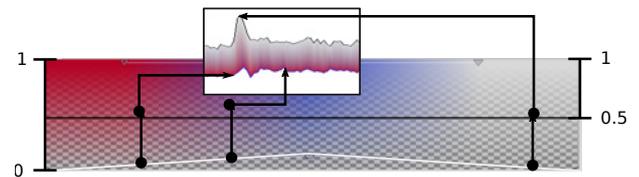


Figure 4: The color and opacity for the line strip are looked up in the transfer function. The opacity is mapped from the interval $[0,1]$ to an interval $[\alpha_{min}, 1]$. In our examples we used $\alpha_{min} = 0.7$. The mean curve is not displayed in this figure.

der to reduce memory consumption as well as to allow for interactive computation times, we evenly subdivide the volume behind the tile into a predefined number of slabs. For each slab we compute the maximal, minimal and mean value of the data covered by the slab and save those values in a TIC cache, see Figure 3. The user can change the number of slabs at any point, in this case the TIC cache is emptied and the computation is restarted.

After each tile and each slab are processed, we continue with the next volume. We repeat this procedure for each new time step until an already processed volume appears again and we completed a loop (in case of recorded data). The result is an envelope of the maximal and minimal value and the mean TIC for each slab of each tile, i.e. the curves represent aggregations in screen space and depth. While one could compute more statistical properties, the maximum, minimum, and mean are easy to interpret and have shown to be well-suited for characterizing the variation of the TICs within the aggregation regions.

In practice, for performance and memory reasons, our algorithm operates in two passes. First, we perform tile casting for a fixed tile size of $x \times y$ (where $x = y = 9$ throughout the paper). We then aggregate these tiles into graxels with $u \times v$ tiles, which are also stored in the TIC cache. The values of u and v can be freely chosen by the user and can be changed at any point. For the examples in the paper we use $u = 4$ and $v = 3$ to generate graxels of 36×27 pixels which results in a good trade-off between spatial resolution and readability for standard screen resolutions.

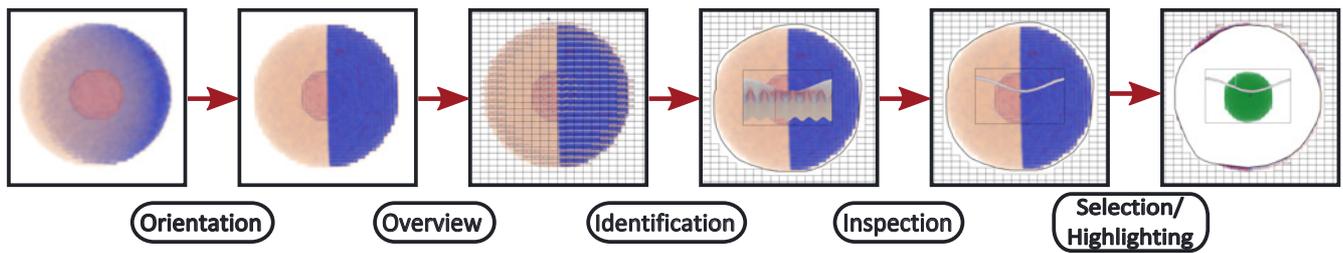


Figure 5: The first interaction step is orientation, where an interesting view is found. After the orientation the graxels fade in to provide an overview of the TICs. Using the overview the user can identify regions of interest and enclose them with the focus lasso to inspect the TICs. If desired the TICs can be selected, alternatively the user can highlight regions with interesting temporal development.

3.3. Graxel Rendering

Having generated a view-dependent representation of the spatio-temporal behavior of the underlying data over the area of each graxel, we now proceed to map this representation to visualization primitives. While there are many potential choices, we focus on a familiar and effective depiction of the TICs as line graphs where the horizontal axis represents time and the vertical axis represents the data value.

The graxel rendering can be subdivided into three stages: *curve filtering*, *curve positioning* and *curve rendering*. The *curve filtering* phase determines which curves will be shown. We provide several methods for selecting the curves in the image plane as well as in depth. After filtering of the curves, their position and size is determined in the *curve positioning* phase. All curves are displayed scaled to the size of their respective graxel. In the *curve rendering* phase the appearance of the TIC is determined. For each graxel we render the envelope of the maximal and minimal TIC as a line strip.

To establish a visual link between the graxels and the volume rendering, we color the line strip between the maximal and minimal value according to the corresponding colors of the maximal and minimal value on the graph for each time step. This means that for each time step the line strip consist of a color transition between two colors corresponding to the maximal and minimal value of the underlying graxel. This process is illustrated in Figure 4. The curves for the maximal, minimal and mean TIC are rendered over the line strip in constant colors. We use blue for the minimal values and red for the maximal values as default colors as red and blue carry the association of being on the opposite end of the color spectrum. The mean TIC is displayed in light gray. Other colors can be chosen as well. For opacity, we map the alpha values of the transfer function from $[0, 1]$ to $[\alpha_{min}, 1]$, where α_{min} is a positive real number between 0 and 1 with 0.7 as default value (see Figure 4). Changing α_{min} allows the user to control to which degree parts of the curve that are classified as transparent in the transfer function will be visible in the rendered graphs.

3.4. Interaction

The typical interaction with our technique can be subdivided into five steps, illustrated in Figure 5. The first step is *orientation*. During the orientation phase the graxels are transparent, which leads to exactly the same interactions as with conventional volume render-

ing. After a suitable view is found and no further rotation, zooming or translation is performed, the graxels start to fade in and the *overview* phase begins. During the overview phase the TICs of the whole volume can be inspected. In order to change between the different slabs the arrow buttons are used. Alternatively we allow the display of a user defined selection of slabs, in this case the user can select the slabs in a GUI and the browsing with arrow buttons is disabled. In this way regions of interest can be *identified*. After the identification of interesting regions, they can be *inspected* in more detail. As last step we allow *selections* in the temporal domain as well as *highlighting* in spatial domain. If a volume interaction occurs, the graxels will immediately fade out and fade in again as soon as the interaction stops.

One advantage of our approach is that it allows us to seamlessly integrate spatial and value domain selections. Our implementation features a wide variety of different interaction mechanisms, but here we focus on two primary tools. We allow selections in the spatial domain on the screen with the *focus lasso* and further interactions in the value domain with the *curve selector*. Our main aim is to enable the selection and extraction of temporal features, allow the highlighting of events with direct links to their spatial representation, and to provide mechanisms for grouping of and separation between different classes of temporal or spatial features. With the focus lasso we select and group the graxels by simply drawing an outline on the screen. All graxels that are enclosed by this lasso are then displayed in one single canvas (see Figure 6). To enable comparisons between different regions, multiple simultaneous focus lasso selections are possible.

Focus Lasso: The focus lasso is used after the *identification* phase and enables spatial grouping of several tiles by drawing a simple sketch. We create a binary mask from the convex hull of the selection where all selected pixels are set to one. From this mask, we create a global lasso region texture which keeps track of all the defined focus lasso regions and their intersections. This lasso region texture is used in the curve rendering phase. All graxels that are covered by a focus lasso region by at least 50% are displayed in a large common canvas inside the focus lasso. A focus lasso region is ignored if it is smaller than one initial graxel, as it would show less detail compared to the initial representation.

Focus lasso regions covering large areas contain more curves, which can lead to increased overdraw resulting in reduced readability. We use the following two strategies to provide a simple yet

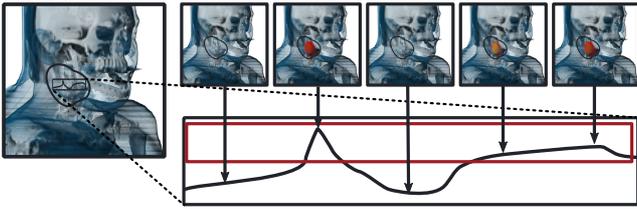


Figure 6: Selections in the spatial domain (upper left) cover specific spatial areas and all the time steps. Selections in the value domain (lower right) cover specific spatial areas and only specific time steps.

powerful *inspection* tool. We map the opacity of the time-intensity curves from the space $[\alpha_{min}, 1]$ to the space $[\hat{\alpha}_{min}, \hat{\alpha}_{max}]$, where both $\hat{\alpha}_{min}$ and $\hat{\alpha}_{max}$ can be chosen between zero and one. This mapping is done in order to allow similar time-intensity curves to form visual clusters. If the time-intensity curves have similar intensity values, they will mostly overlap, thus forming opaque visual clusters. We use a *point of interest* inside the focus lasso region. This point is located at the mouse position and can be fixed on the screen. Time-intensity curves closer to the point of interest are displayed more opaque while the ones further away become linearly more transparent until they reach zero opacity. The slope of the opacity transition can be freely modified by the user.

Curve selector: We aim at interactions that allow for the *selection* of curves in the spatial as well as in the value domain. The focus lasso already groups the graxels spatially. In order to perform selection on a finer scale, we provide a mechanism that only selects graphs above a certain opacity threshold. By hovering over a focus lasso region with the point of interest until the desired TIC is prominently shown and fixing the point of interest, we allow fine-scale filtering in the spatial domain. The selection in the value domain is performed by drawing a rectangle or a circle directly over the TICs. If a TIC crosses the drawn area, it is extracted and shown in a separate window.

Some application scenarios require knowledge about the position of specific temporal features. In such a case we employ a highlighting mechanism which allows to highlight regions of the volume based on their values over time. Similar to the extraction of the graphs, a rectangle or a circle can be drawn directly over the graphs in a focus lasso region. All intensity values that are covered by this area at the corresponding timesteps will use a second transfer function, which can be defined freely, see Figure 6. This new transfer function can be applied to the current focus lasso, to all focus lassos, or to the whole data set. Furthermore, it is possible to render the non-selected values completely transparent for detailed inspection of the exact spatial position of the values of interest. An example is shown in Figures 10 and 12 and described in Section 5.2.

Through the combination of spatial and temporal selection in a single view, the curve selector provides a useful tool for selection and feature extraction. We allow grouping of features in the temporal and spatial domain with the focus lasso. Temporal events can be emphasized by selecting regions in focus with the focus lasso and then highlighting the volume with the curve selector.

Data set	Conv. Ray Casting	Graxels 10 slabs	Graxels 5 slabs
Synthetic data	0.049 s	0.114 s	0.079 s
Kidney Sim.	0.045 s	0.143 s	0.086 s
Perf. data	0.057 s	0.219 s	0.127 s
Supernova	0.127 s	0.354 s	0.148 s

Table 1: Performance of our method compared to conventional direct volume rendering as measured on an Intel CPU equipped with an NVidia GeForce GTX 780 GPU, with a viewport size of 987×967 pixels and graxels of size 4×3 (36×27 pixels) with 10 and 5 slabs and one slab rendered.

4. Implementation

Our method was implemented in C++ and OpenGL/GLSL. The tile casting approach is performed in two steps. First, the aggregation of data values into the TIC cache occurs in a GLSL shader using atomic operations of the `GL_ARB_shader_image_load_store` extension. Further aggregation over graxel areas is performed in a compute shader. The TIC cache itself is represented as a 2D texture array where each texture corresponds to one tile. Each row in this texture corresponds to one TIC, and the number of rows is the number of slabs multiplied by three (for the minimum, maximum, and mean TICs). We use multiple texture arrays if the number of tiles exceeds the maximum texture array size. Curve generation uses a geometry shader which performs lookups into the TIC cache.

Naturally, our method requires more computational time compared to conventional ray casting. The performance is mostly dependent on the number of total slabs and number of displayed slabs, as can be seen in Table 1. The computation time increases with the volume size as well but this is mostly due to the increased computation time for the volume ray casting. In this table we summarize the average computation time per frame for 30 consecutive frames while displaying the TICs for one slab. Displaying TICs for additional slabs increases the computation time for each frame by 0.087 s on average. The size of graxels does not have significant impact on the overall performance.

5. Application Examples

We show the advantages of our technique in four application examples. We demonstrate how our methods support the understanding of frequency patterns and we document the benefit of our technique's interactivity in the context of two medical examples. Further, we show how our approach captures relevant events in the temporal domain of a case from natural sciences.

5.1. Synthetic Data Set

A very natural application scenario of our method is the comparison of two volumes over time. In this example we study two synthetic data sets. The two data sets are very similar in terms of their geometry. However, they show different temporal behavior. In Figure 7 we show both data sets rendered at three different points in time, with one shown in the upper half and the other one below. Both data sets consist of two spheres, divided into a left and a right part. The two halves have time-varying intensity values described

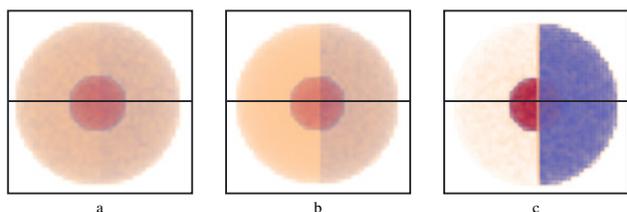


Figure 7: Volume rendering of consecutive time-steps of two synthetic data sets. The two data sets (lower and upper half) appear almost identical in the volume rendering.

by the function $i = a + b \cdot \sin(\theta + c \cdot \phi) + \text{rand}()$. The parameters a, b, θ and c define different temporal behaviors for the two halves. Inside each sphere is a spherical nucleus, with either a constant or a fluctuating value, described by the same formula. The rest of the data set consists of a constant value with random noise added to it. As shown in Figure 7, the two data sets look almost identical in a volume rendering. In fact a thorough tuning of the transfer function is needed to see any difference at all, but the animation still fails to convey *how* the two data sets differ. The same is true for comparing the left and right sides of the spheres. Volume rendering shows that the two halves differ, but it is not clear how, especially in terms of value change frequency.

The upper data set has a stable nucleus while the lower data set has a nucleus with varying intensities. Identifying and describing the temporal behavior of each region using direct volume rendering alone is demanding and difficult. Using our technique, however, one brings the sphere in the position where the halves are clearly distinct. As the interaction stops the graxels start to fade in providing an overview. Capturing the whole sphere with the focus lasso and investigating the three interesting regions reveals the characteristic temporal relations immediately. Figure 8 shows all time-intensity curves in one focus region (a) as well as the investigation by region, moving the point of interest, and showing the corresponding time-intensity curves (b–d). These examples demonstrate that integrating an explicit visualization of the temporal data behavior indeed helps to gain a fast overview of the temporal development in the data and different characteristic temporal developments become evident, in particular when compared to DVR alone.

5.2. Kidney Simulation Data

In the second example we present how our method was used by domain experts for the location of spatial features. In principle, highlighting parts of the data is possible through the tuning of the transfer function. However, if the exact value range for structures of interest is not known in advance, this process is difficult. Furthermore, in the case of time-dependent data, the relevant value range may change dynamically. This is the case in the following scenario.

We consider data from a numerical simulation of kidney compression during breathing. The goal of this setup was to be able to identify if a kidney is healthy or fibrotic. The latter is stiffer and thus compresses less than a healthy kidney. The simulation was done to study whether the degree of fibrosis and the location of the

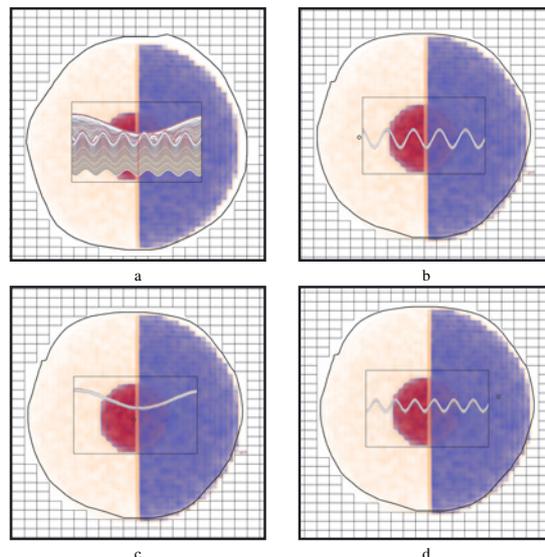


Figure 8: In alphabetical order: All time-intensity curves in one focus region, intensity in the left half-sphere, intensity in the nucleus and intensity in the right half-sphere.

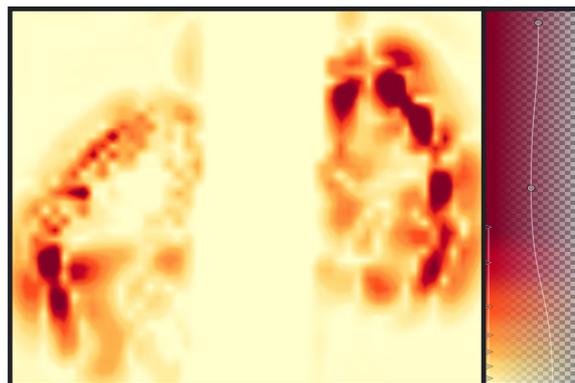


Figure 9: The upper half of the kidney on the left side was simulated to be fibrotic, while the kidney on the right was simulated to be healthy. The transfer function is shown on the right side with low values in yellow and high values in red.

fibrotic tissue can be diagnosed on the basis of local volume deformation information. The data set is a 64×64 2D set with 34 time steps. A visualization of the data set, based on a sequential multi-hue transfer-function, is shown in Figure 9, depicting the time-step with the most prominent differences between the healthy and the fibrotic tissue. A typical task for a domain expert is the identification of the fibrotic tissue inside the kidney. However, the compression rate is highly dependent on the inhaled air volume, thus the compression values of the fibrotic tissue vary for each patient. Using a transfer function with a gradual color change helps to see the difference between healthy and fibrotic tissue but this smears out the exact location of the fibrotic tissue as depicted in Figure 9.

In this example, the upper part of the left kidney was simulated to be fibrotic. While the visualization of the most prominent differ-

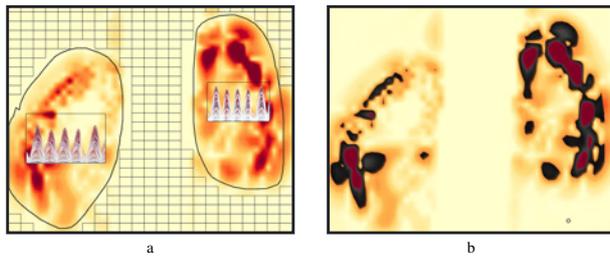


Figure 10: *a: Selecting the kidneys with the focus lasso allows to explore the TIC of each kidney and to select and highlight value range characteristic for fibrotic tissue. b: Coloring the characteristic values of fibrotic tissue in dark grey, we identify real fibrotic tissue as isolated patches of dark grey, this reveals that only the kidney cortex is affected by fibrosis.*

ences between the healthy and the fibrotic kidney does not require much time, these images give only little insight into the exact location of the fibrotic tissue. As only the absence of red color is noticeable in the upper part of the left kidney, one perceives the whole upper part of this kidney to be fibrotic and interprets the brighter colors as fibrotic tissue.

The relative volume change occurs mostly in the kidney cortex, as the rest is not compressed but moves along. Furthermore, the fibrotic cortex is not completely stiff, just stiffer than a healthy cortex, while still being more elastic than the inner kidney. Domain experts are interested in isolated regions with values in the middle range of the whole relative volume range. Selecting the kidneys with the focus lasso allows us to investigate the TICs of each kidney as shown in Figure 10 (a). Selecting the characteristic values of fibrotic tissue allows us to render the data with a different transfer function. We immediately see isolated structures in the upper part of the left kidney, as shown in Figure 10 (b). This way we can determine that the fibrotic tissue does not correspond to the tissue colored in yellow. Our domain experts used our tool to confirm their simulations. In the future they plan to use our approach as a promising means for detection of fibrotic tissue in real patient data, which naturally is more complex and noisy compared to the simulated example.

5.3. Perfusion Data

As mentioned earlier, the shape of the TICs in contrast-enhanced MRI can provide valuable information on the pathological nature of tissue. TICs can reveal relevant information on the possible malignancy of a tumor or on the functionality of an organ. Domain experts are also interested in the shape of the tissue with certain perfusion characteristics. The kidney cortex, for example, has a very high absorption rate and a very distinct shape. If the anatomical shape with particular perfusion characteristics substantially differs from the expected shape, this may also indicate a relevant organ malfunction. With our method, we can assess both criteria together by extracting and comparing TICs in multiple windows.

In this example we look at a 4D contrast-enhanced MR scan of the abdominal region. The data set consists of 48 time steps with a dimension of $61 \times 56 \times 48$. Sometimes practitioners are interested

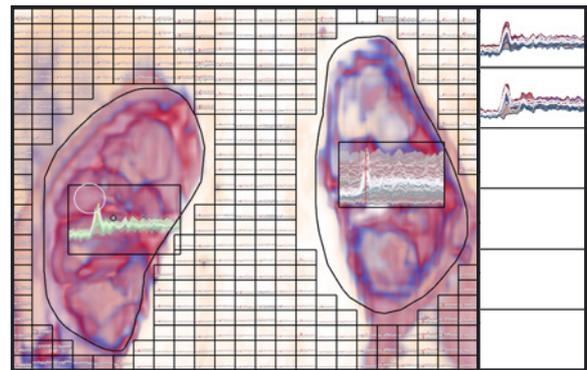


Figure 11: *The TICs from both kidneys are extracted and displayed in separate windows. We zoomed in on the left kidney in order to show the selection in more detail.*

in the comparison the perfusion rate of the two kidneys, to find out if one kidney may have a malfunctioning filtering process. The standard procedure for this is to select a point of interest in each kidney and to compare the two TICs. However for this one needs to know a priori where to set the point of interest. Furthermore, such a process is sensitive to outliers. Our approach helps to identify the region of interest and provides additional robustness compared to the standard procedures. In Figure 11 we show the extracted TICs from the left and right kidney. The kidneys are roughly outlined with the focus lasso. Then the point of interest is fixed and a circle is used to make a selection in the value domain. Only curves that cross the circle are selected. Now one can compare the perfusion rates of both kidneys in the windows on the right side. We can observe that this data set was obtained from a healthy volunteer where both kidneys show regular perfusion rates.

As mentioned before a practitioner might be interested in the shape of the kidney cortex with certain TIC characteristics. Seeing the time-intensity curves of the kidney cortex, we can highlight the voxels with the corresponding values. By selecting their outline with the focus lasso, we can hover over a characteristic time-intensity and then select the high values with the rectangle tool. Now we can pause the animation and fix the time step, which corresponds to the peak of the perfusion TIC. This selection can be seen in Figure 12 (a). After this selection we can hide the graxels and observe the shape of the kidney cortex more clearly, see Figure 12 (b).

5.4. Supernova

The three previous examples showed data which was to a certain degree fixed in space, with little movement of the structures themselves. In this example we look at a highly dynamic data set of a 4D simulation of a supernova, depicted in Figure 13. Each time step has the dimensions $432 \times 432 \times 432$ with four bytes per voxel. We used the first 20 time steps of the data set with intensity values representing the entropy field.

This example demonstrates our approach as an Eulerian view of flow very clearly. The TICs separate into several types, of which three are particularly distinct. These characteristic TICs are shown

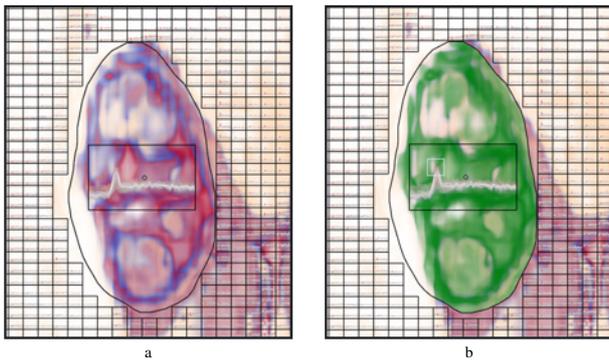


Figure 12: By selecting the characteristic values of the kidney (a) we easily highlight the kidney in the rendered image (a, b). In (b), the volume corresponding to the selected values is highlighted.

in more detail below the actual visualization, indicating that there are three regions with characteristically different temporal behavior. The region on the left shows high entropy in the first time steps, with a decrease towards the end. The region in the middle shows a high entropy field throughout the whole time period. The region on the right shows relatively low entropy in the beginning and high entropy in the last time steps. With volume rendering, we can confirm these developments, but multiple renderings are necessary, while our method captures this complex behavior in one view.

We do not claim that our method depicts the data more clearly than a volume rendering, which is essentially a Lagrangian representation of the flow, in this example, but it provides a fast overview of the whole time series in one image and manages to characterize the distinct areas with different behaviors. As such we see our technique as complementary to volume rendering, as it provides a quick overview of the data for identifying potential features of interest.

6. Discussion

In our experiments we found that using *graxels* in a screen filling manner provides a fast overview of the temporal development of the data. Distinct events manifest themselves as peaks in the TICs, which form visual clusters among the *graxels*. The focus lasso and curve selections allow fast and intuitive exploration of the data. Due to the choice to use an envelope of the TICs and the mean TIC rather than rendering all TICs at once, we reduced clutter but still provide some statistical information of the data. We do not see our method as a competitor to existing approaches, such as multiple coordinated views, as we do utilize these techniques as well. Our technique is meant to complement existing approaches and to provide a different view of the data. By overlaying our visualization over an animated volume rendering, for example, the temporal evolution of different image regions can be made more apparent. This could also be used selectively or on-demand, for example as a type of magic lens or only during certain operations.

Our current implementation has some limitations. The need for atomic operations during tile casting is a bottleneck and limits the number of slabs when interactive performance is desired. However, tile casting only has to be performed when the view is changed.

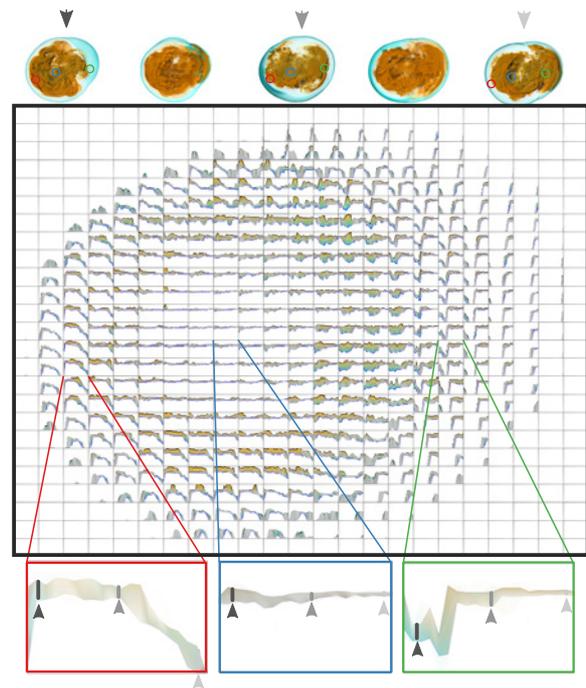


Figure 13: Our method reveals different parts of the data set, with decreasing, constant and increasing entropy fields. These temporal developments are integrated in one single view.

When using the presented selection methods, the performance is fully interactive. Also, when the transfer function is changed, only curve generation has to be performed. Hence, for most use cases the performance of our current implementation is sufficient and there are several options for further optimization.

While our visualization is quite dense, it nonetheless has shown to be effective in conveying even complex temporal patterns. As screen resolutions constantly increase, we believe that this type of visualization is promising to provide a concise overview of complex data. As pointed out by Tufte [Tuf01], humans are capable of a vast number of distinctions within a small area. Furthermore, one could display the *graxels* in a checkerboard fashion, thus cutting their amount in half. As already mentioned, hiding the *graxels* during interactions such as rotation and then gradually letting them fade in again can be used to improve orientation during view-point changes and also reduces the impact on computational performance. Since the *graxels* are placed in the screen space, they may not be aligned to boundaries in the spatial domain. This drawback can be overcome by either adjusting the tile size or by using the focus lasso which allows to define meaningful region borders.

The *graxels* are organized in slabs perpendicular to the viewing direction naturally many data sets contain structures which do not follow this orientation. These structures would have a better representation in deformed slabs, which align with the structures. We partially address these cases by allowing the user to define the number of slabs and the shown slabs freely.

We provide several interaction techniques with the data. Addi-

tional more complex techniques, such as selection of curves with a certain shape, can be easily integrated. It should be noted that our visualization is most effective when the size of the graxels is adjusted according to the screen resolution which is difficult to accurately reproduce in this paper as the maximum image size is limited by space restrictions. Hence, most of the images in this paper are best viewed by zooming in.

7. Conclusion

In this paper we introduced a flexible visualization technique for time varying volumetric data. Our method uses view-dependent small multiples of time-intensity curves, which we term graxels (graph pixels), to provide an overview of the spatio-temporal patterns in the data. We presented a GPU-based algorithm for generating these views and showed how they facilitate interactive exploration and analysis in space and time. Our method was designed for time-dependent volume data, but the approach is also applicable to other types of multivariate data.

8. Acknowledgements

This work has been supported by the MedViz lighthouse project IllustraSound.

References

- [AFM06] AKIBA H., FOUT N., MA K.-L.: Simultaneous classification of time-varying volume data based on the time histogram. In *Proc. of EuroVis* (2006), pp. 171–178. 2
- [AM07] AKIBA H., MA K.-L.: A tri-space visualization interface for analyzing time-varying multivariate volume data. In *Proc. of EuroVis* (2007), pp. 115–122. 2
- [AMST11] AIGNER W., MIKSCH S., SCHUMANN H., TOMINSKI C.: *Visualization of Time-Oriented Data*, sec. ed. Springer London, 2011. 2
- [ANOHH11] ANGELELLI P., NYLUND K., ODD HILJA H., HAUSER H.: Interactive visual analysis of contrast-enhanced ultrasound data based on small neighborhood statistics. *Computers and Graphics* 35, 2 (2011), 218–226. 2
- [BVMG08] BALABANIAN J.-P., VIOLA I., MÖLLER T., GRÖLLER M. E.: Temporal styles for time-varying volume data. In *Proc. of the International Symposium on 3D Data Processing, Visualization and Transmission* (2008), pp. 81–89. 3
- [CGK*07] CANG R., GHONIEM M., KOSARA R., RIBARSKY W., YANG J.: Wirevis: Visualization of categorical, time-varying data from financial transactions. In *Proc. of IEEE VAST* (2007), pp. 155–162. 2
- [EDG*08] ELMQVIST N., DO T. N., GOODELL H., HENRY N., FEKETE J. D.: Zame: Interactive large-scale graph visualization. In *IEEE PacificVis* (2008), pp. 215–222. 3
- [FMHC07] FANG Z., MÖLLER T., HAMARNEH G., CELLER A.: Visualization and exploration of time-varying medical image data sets. In *Proc. of Graphics Interface* (2007), pp. 281–288. 2
- [FW10] FORLINES C., WITTENBURG K.: Wakame: Sense making of multi-dimensional spatial-temporal data. In *Proc. of the International Conference on Advanced Visual Interfaces* (2010), pp. 33–40. 3
- [Häg70] HÄGERSTRAND T.: What about people in regional science? *Papers of the Regional Science Association* 24, 1 (1970), 7–21. 2
- [HMCM10] HSU W.-H., MEI J., CORREA C., MA K.-L.: Depicting time evolving flow with illustrative visualization techniques. In *Proc. of Arts and Technology* (2010), pp. 136–147. 2
- [JE12] JAVED W., ELMQVIST N.: Exploring the design space of composite visualization. In *IEEE PacificVis* (Feb 2012), pp. 1–8. 3
- [JKM01] JANKUN-KELLY T. J., MA K.-L.: A study of transfer functions generation for time-varying volume data. In *Proc. of Volume Graphics* (2001), pp. 51–68. 2
- [Kra03] KRAAK M. J.: The space-time cube revisited from a geovisualization perspective. In *Proc. of the International Cartographic Conference* (2003), pp. 1988–1995. 2
- [KSEB*14] KHALIFA F., SOLIMAN A., EL-BAZ A., ABOU EL-GHAR M., EL-DIASTY T., GIMEL 'FARB G.: Models and methods for analyzing DCE-MRI: a review. *Medical Physics* 41, 12 (2014). 1
- [LCPS10] LEE T.-Y., CHAUDHURI A., POIKLI F., SHEN H.-W.: Cyclestack: Inferring periodic behavior via temporal sequence visualization in ultrasound video. In *Proc. of IEEE PacificVis* (2010), pp. 89–96. 2
- [LS08] LU A., SHEN H.-W.: Interactive storyboard for overall time-varying data visualization. In *Proc. of IEEE PacificVis* (2008), pp. 143–150. 3
- [LS09a] LEE T.-Y., SHEN H.-W.: Visualization and exploration of temporal trend relationships in multivariate time-varying data. *IEEE TVCG* 15, 6 (2009), 1359–1366. 2
- [LS09b] LEE T.-Y., SHEN H.-W.: Visualizing time-varying features with tac-based distance fields. In *Proc. of PacificVis* (2009), pp. 1–8. 2
- [MEV*05] MLEJNEK M., ERNEST P., VILANOVA A., VAN DEN BOSCH H., GERRITSEN F., GRÖLLER M. E.: Profile flags: a novel metaphor for probing of T2 maps. In *Proc. of IEEE Visualization* (2005), pp. 599–606. 3
- [Moe04] MOERE A. V.: Time-varying data visualization using information flocking boids. In *Proc. of IEEE InfoVis* (2004), pp. 97–104. 3
- [Rob07] ROBERTS J. C.: State of the art: Coordinated and multiple views in exploratory visualization. In *Proc. of International Conference on Coordinated and Multiple Views in Exploratory Visualization* (2007), pp. 61–71. 2
- [SOMK08] SHEARER J., OGAWA M., MA K.-L., KOHLENBERG T.: Pixelplexing: Gaining display resolution through time. In *Proc. of PacificVis* (2008), pp. 159–266. 3
- [TCM10] TIKHONOVA A., CORREA C., MA K.-L.: An exploratory technique for coherent visualization of time-varying volume data. *CGF* 29, 3 (2010), 783–792. 3
- [Tuf01] TUFT E. R.: *The Visual Display of Quantitative Information*, second ed. Graphics Press Cheshire, 2001. 3, 9
- [WS03] WOODRING J., SHEN H.-W.: Chronovolumes: a direct rendering technique for visualizing time-varying data. In *Proc. of Volume Graphics* (2003), pp. 27–34. 2
- [WS06] WOODRING J., SHEN H.-W.: Multi-variate, time varying, and comparative visualization with contextual cues. *IEEE TVCG* 12, 5 (2006), 909–916. 2
- [WS09a] WOODRING J., SHEN H.-W.: Multiscale time activity data exploration via temporal clustering visualization spreadsheet. *IEEE TVCG* 15, 2 (2009), 123–137. 2
- [WS09b] WOODRING J., SHEN H.-W.: Semi-automatic time-series transfer functions via temporal clustering and sequencing. *CGF* 28, 3 (2009), 791–198. 2
- [WWS03] WOODRING J., WANG S., SHEN H.-W.: High dimensional direct rendering of time-varying volumetric data. In *Proc. of IEEE Visualization* (2003), pp. 417–424. 2
- [WYM08] WANG C., YU H., MA K.-L.: Importance-driven time-varying data visualization. *IEEE TVCG* 14, 6 (2008), 1547–1554. 2
- [WYM10] WANG C., YU H., MA K.-L.: Application-driven compression for visualizing large-scale time-varying data. *IEEE Computer Graphics and Applications* 30, 1 (2010), 59–69. 2