Scalable Visualization of Time-varying Multi-parameter Distributions Using Spatially Organized Histograms

Tyson Neuroth, *Member, IEEE,* Franz Sauer, *Member, IEEE,* Weixing Wang, Stephane Ethier, Choong-Seock Chang, and Kwan-Liu Ma, *Fellow, IEEE*

Abstract—Visualizing distributions from data samples as well as spatial and temporal trends of multiple variables is fundamental to analyzing the output of today's scientific simulations. However, traditional visualization techniques are often subject to a trade-off between visual clutter and loss of detail, especially in a large-scale setting. In this work, we extend the use of spatially organized histograms into a sophisticated visualization system that can more effectively study trends between multiple variables throughout a spatial domain. Furthermore, we exploit the use of isosurfaces to visualize time-varying trends found within histogram distributions. This technique is adapted into both an on-the-fly scheme as well as an in situ scheme to maintain real-time interactivity at a variety of data scales.

Index Terms—histograms, particle data, large-scale data, in situ processing, time-varying data, isosurfaces, scientific visualization

1 Introduction

The ability to visualize and understand the data distributions of multiple variables simultaneously has numerous scientific applications. One class of applications in particular, the study of fluids, can rely on analysis of field and/or particle data produced by simulations. The ability to intuitively explore such data types can lead to a better understanding of phenomena such as fusion [1], [2], particle acceleration [3], or combustion [4].

However, visualization techniques are often subject to a trade off between clutter and loss of information. Visualizing all data values of every available entity within the domain can result in a great deal of clutter, whereas averaging data values can result in a loss of information and can eliminate potentially important minor trends.

The use of histograms or distribution functions can be a powerful data reduction tool that can eliminate clutter while maintaining subtle trends found in the data. Spatially organizing these histograms throughout a simulation domain allows one to distinguish spatial variations as well. This results in an information dense visualization of data distributions, with a favorable structure that lends well to user interaction and efficient computation. Many challenges of using such a technique lie in the ability to present these information-rich histograms in a usable and easy to understand manner.

Our previous work [5] focused on using spatially organized histograms to represent the overall motion within sets of discrete objects. These "velocity histograms" were able to provide a low clutter and intuitive representation of velocity distributions, highlighting both major and minor trends. However, the work was limited to studying motion (velocity), and was only applied on a per time step basis.

This new work extends and generalizes some of the concepts introduced in [5]. The system now supports the visualization of 1D or 2D spatially organized histograms, and the user can construct them from any available or derivable variables. This allows users to explore a wide variety of relationships between the properties of entities, as well as their motion throughout the domain.

Additionally, this work focuses on developing new visual representations that can encode time-varying trends present in the histogram data. This is a challenging task since temporal properties of the histograms need to be represented in an intuitive manner. This becomes even more difficult when histograms are 2D rather than 1D. Since a temporal sequence of 2D histograms can be stacked into a 3D volume, we utilize isosurfaces and isocontours to describe time-varying trends in the data. Each isosurface represents a boundary between bins that contain a higher or lower frequency compared to the isovalue.

Another challenge lies in the massive amounts of data generated by large scale simulations. As a variety of parameters can be chosen when generating histograms to highlight different aspects of the data, this technique is best suited as an interactive visualization tool. However, special care must be taken in order to maintain fluid interactivity in large-scale datasets. As a result, we maintain both on-the-fly and in situ methods of histogram generation in order to efficiently handle a variety of dataset types and sizes.

In this paper, we present our new histogram-based visualization techniques and make the following contributions:

- *Tyson Neuroth, Franz Sauer, and Kwan-Liu Ma are with the Department of Computer Science, University of California at Davis, Davis, CA, 95616. E-mail: taneuroth@ucdavis.edu, fasauer@ucdavis.edu, ma@cs.ucdavis.edu*
- *Weixing Wang, Stephane Ethier and Choong-Seock Chang are with the Princeton Plasma Physics Laboratory, 100 Stellarator Road, Princeton, NJ, 08540. E-mail: wwang@pppl.gov, ethier@pppl.gov, cschang@pppl.gov*

- We extend our past work on spatially organized velocity histograms to visually represent the relationships between variables other than velocity.
- We develop a visual representation based on isosurfaces that can be used to explore time-varying properties in 2D histograms.
- We maintain two techniques for histogram generation: an on-the-fly scheme for smaller datasets, and an in situ scheme for large-scale datasets.
- We integrate these techniques into a coherent and usable visualization software.

We demonstrate the effectiveness of our approach using case studies from a variety of real-world datasets.

2 Related Work

There are many works that focus on reducing clutter in flow visualization that are applicable to this problem since many flow datasets can take a Lagrangian (particle-based) representation. Kirby and Laidlaw [6] used concepts from painting to design a multi-layer representation of colors and textured patterns to represent different parameters of a flow field in an easy to read manner. Obermaier and Joy [7] used a unique visualization of metric tensors to show deformations on 3D surfaces in a flow. Properties like the velocity gradient can be represented on these surfaces using elliptical glyphs. In addition, topology-based methods, which were first introduced by Helman and Hesselink [8], can be used to extract specific flow patterns of interest and present them in a low-clutter way. However, many of these methods are still limited in that they can hide useful details since major trends will tend to be favored over minor trends.

The use of histograms has made a very strong presence in the computer vision community. For example, Ihaddadene and Djeraba [9] utilize a block direction histogram to represent the overall motion of crowds in different "blocks" or fields of view. Romanoni et al. [10] utilize spatio-temporal histograms for background subtraction in cases where the camera is in motion. Another example is the use of "Histograms of Oriented Gradients" as feature descriptors. Dalal et al. [11] utilize these for the accurate detection of humans that are in motion relative to the background or camera. Lastly, histograms were utilized by Jung et al. [12] in the clustering of visually tracked objects by identifying similar neighborhoods in the histograms.

To explore spatial trends, we rely on a panning-window style partitioning to define the spatially organized histogram layout. This can be considered a type of "Magic Window" as defined by Tominski [13]. Other methods have been used to show spatial variation and uncertainty in a summarized manner. For example, glyphs have been designed for visualizing uncertain flow fields by showing ranges of possible velocities [14]. Furthermore, Hlawatsch et. al [15] have designed a glyph which encodes various statistical information.

Histograms also have prevalent use in domain specific areas, such as geophysics [16], superfluids [17], and fusion [18] to name a few. This is because they excel at summarizing distributions, and can be used to visualize statistically accurate correlations between multiple variables. While the spatially organized histograms that we use enable the exploration of multiple data parameters at once in a low clutter manner, there are numerous other techniques that have explored the topic of multi-dimensional data analysis. These include methods such as scatter plotting techniques, multivariate topology, and more, and are well summarized in the survey by Kehrer and Hauser [19].

Visualizations showing temporal changes in 1D histograms are used in areas such as electrical engineering, audio analysis, radar, and seismology [20], [21], [22], [23]. In these applications, the 1D histograms represent wave-frequency distributions and the temporal stacks are referred to as spectrograms, cumulative spectral decay plots, or more generally as waterfall charts. The 2D histogram stacks that we use in this work could be considered a higher dimensional extension of such plots, which require more advanced rendering methods and interactive parameter controls to be used effectively for visualization.

Using isosurfaces to visualize frequency distributions has been explored before, for example to study electrical activity involved in ventricular fibrillation [24]. Also, Kao et al. used a 3D histogram cube representation to visualize non-temporal features in 2D distributions from EOS satellite data [25]. Their 3D histogram cubes were termed "pixel-wise summaries" and used the third dimension to represent properties of the 2D histograms, such as the mean, median, standard deviation, interquartile range, kurtosis and skewness.

Isosurfacing and other volume based visualizations have been used to study time-varying data, for example Woodring et al. introduced direct volume rendering techniques for visualizing time-varying data, termed "Chronovolumes" [26]. Shen et al. proposed a fast algorithm for time-varying volume visualization based on the time-space partition tree [27]. Other useful techniques for temporal analysis of flow data rely on"time and streak surfaces" [28], [29]. Alternatively, Widanagamaachchi et al. [30] developed techniques to temporally correlate features and investigate their tracking graphs in turbulent combustion simulations. Applying these methods to features in the temporally evolving histograms in our work could provide an alternative way of visualizing their evolution over time.

Lastly, storing in situ generated distributions as a way of downscaling the data storage of results from large-scale simulations has also been investigated before. Thompson et al. [31] leveraged such in situ generated histograms for feature detection through statistical and topological methods. In situ generated histograms are also useful for analyzing results from gryokinetic simulations, such as GTC [2]. However, advanced interactive software designed for exploring this type of data has been lacking.

3 Background

Before we describe our methods, we introduce some preliminary background knowledge about the types of histograms we can make use of as well as the data types that our methods can support.

3.1 Histograms

A histogram can be used to represent the distribution of discrete data objects. The desired variables of each object

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2642103, IEEE Transactions on Visualization and Computer Graphics

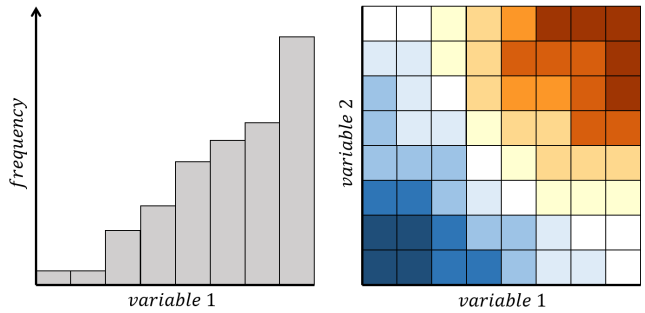


Fig. 1. An image depicting the visual representation used for 1D and 2D histograms in this work. Left) A 1D histogram discretized into bins. The frequency of entities sampled into each bin is represented by the height of the bar. Right) A 2D histogram with each bin represented by a colored cell. Darker colors represent bins with higher frequency magnitude, and when visualizing weighed histograms, hue can differentiate between positive (red-orange) and negative (blue) bin values.

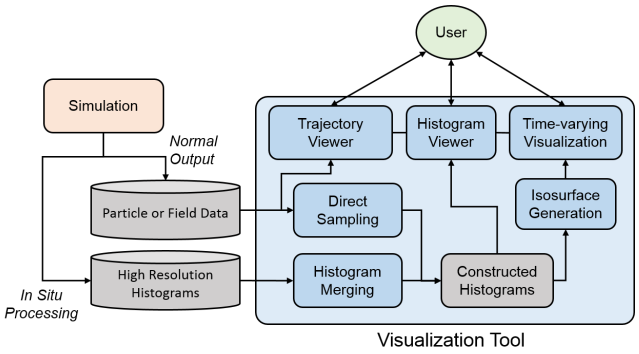


Fig. 2. An overview of our workflow. Histograms can be constructed on-the-fly (directly from particle data) or from pregenerated high resolution histograms that were computed in situ. A user interfaces with the visualization software and can explore the data using three primary linked views: a histogram viewer which shows the spatially organized histograms, a trajectory viewer which shows the trajectories of particles corresponding to selected bins, and a time-varying visualization which uses isosurfaces to show temporal patterns of a selected histogram.

are sampled and binned into histogram cells, as in Figure 1. A 1D histogram represents the distribution of values for a single variable, and bar height can be used to encode the bin frequencies (left). A 2D histogram represents the distribution of values in two variables (right). In this case, we can use color to represent bin frequency and, when appropriate, hue to differentiate bins with a net positive or negative value (see next section).

3.2 Weighted Histograms

In some cases, it is necessary or useful to weight the contributions of each sampled object. We compute our weighted histograms in raw form as follows, where $H(i,j)$ gives the value of the bin at row and column (i,j), $bin(i,j)$ represents the set of objects mapped to it, and w_k is the weight for the sampled object, o_k.

$$H(i,j) = \sum_{o_k \in bin(i,j)} w_k$$

In various particle-in-cell fusion simulations such as XGC [1] and GTC [2], the simulation particles each represent a variable number of real-particles. In this case, scientists need to use this value as a weight in order to see the proper distributions corresponding to the modeled physical system. Furthermore, these weights represent perturbations from a Maxwellian background and thus can be negative or positive. Still, visualizing the distributions of unweighted simulation-particles can be useful for making sense of how the simulation is behaving.

Alternatively, one could use any variable in the data as a weight when generating the histograms. This allows users to view additional information in the limited 1D or 2D spaces, since the frequency counts are now being adjusted by a separate variable. However, it must be considered that the "frequency" of a certain bin can be caused by a large number of objects with a small weight, or a small number of objects with a large weight (see Section 4.3.4). As an analogy, suppose you want to see distributions of campaign contributions in a local election. You could compute an unweighted histogram showing how many people contributed to a campaign. Alternatively, you could weight each contribution by its amount in order to see how money/influence is distributed. Furthermore, weighting contributions to one party negative and the other positive allows one to see how deviations from a neutral position are distributed.

3.3 Applicable Data Types

Any set of data points embedded in a mathematical space can be sampled in order to summarize their distributions. In this paper, we focus on particle data from fusion and accelerator simulations. However, we could also operate on data associated with discrete grid points. Furthermore, as we can also partition and sample entities in non-physical spaces, we can visualize and study many different combinations of variables in a single visualization. Thus the techniques in this paper can be applied to many multi-parameter datasets, and are not limited to just scientific simulation data. Without loss of generality, we will use the term particles throughout the rest of this paper to describe the discrete objects being sampled.

4 Methods

The main application of this technique is for interactive visualization of multiple data distributions, both spatially and temporally throughout a simulation domain. Desired variables of particle data are sampled in order to form a set of spatially organized histograms to be visualized by the user. The means of sampling and generating the histograms, constructing their visual representations, and interactively exploring their trends are all important factors and are described in more detail in the following sections.

4.1 Overview

One important aspect of such a technique is the ability to interactively explore the data through a variety of adjustable parameters and configurations, (e.g., chosen variables, size/distribution of sampling regions, histogram resolution, etc.). As a result, the histograms displayed to a user

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2642103, IEEE Transactions on Visualization and Computer Graphics

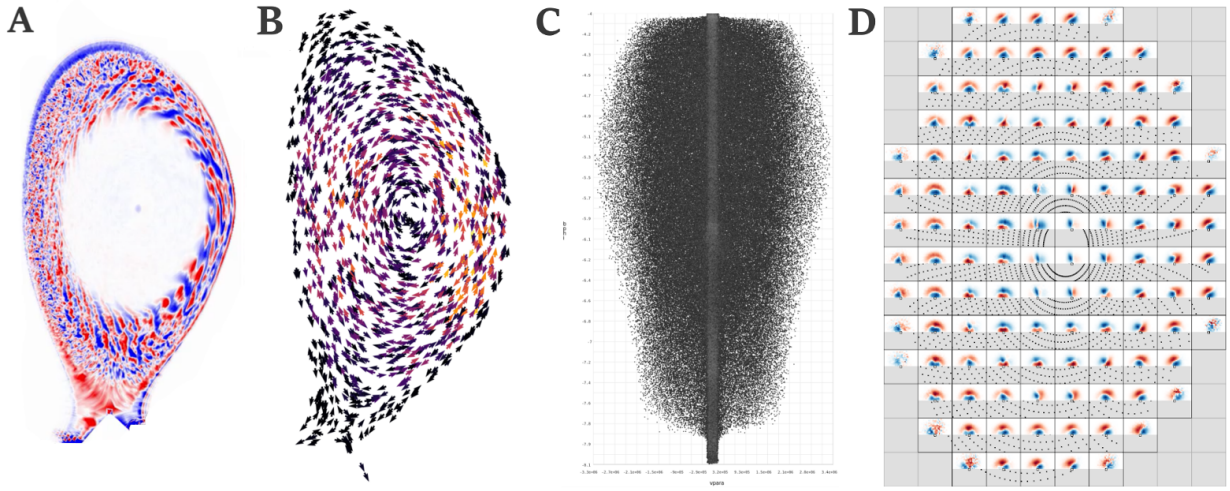


Fig. 3. a) Color can be used to highlight spatial variation throughout the domain, but does not show trends between variables. b) A vector plot showing particle motion (color mapped to magnitude) results in clutter, even when plotting only 0.5% of the full data. c) A scatter plot can be used to show trends between two variables in the particle data, but does not show how such trends differ throughout the domain. d) Spatially organized histograms can display both trends between different variables and spatial variation throughout the domain in a low clutter manner.

need to be constructed at runtime; however, this is one of the most computationally expensive steps. We therefore provide two schemes for histogram generation, each designed to handle different dataset sizes as shown in Figure 2. For smaller datasets, we utilize GPU acceleration to efficiently sample and bin the raw simulation data directly. The presented visualization updates in real time according to any parameter adjustments. For large-scale datasets, we provide an alternate method that can emulate the same interactivity. Instead of sampling the raw data values directly, a set of high resolution histograms are constructed in situ and saved to disk. These can then be sampled in real time according to the user controlled parameters.

The visualization tool itself provides multiple linked views which can be used to explore different aspects of the data. A trajectory viewer is used to directly view raw simulation data and provides spatial context as well as detailed information on demand. The histogram viewer presents a set of spatially organized histograms which can present major and minor trends in the data in a low clutter manner. This view focuses on exploring spatial differences in 1D or 2D histograms. Lastly, a time-varying visualization is provided for exploring temporal variation. Histograms of interest are stacked into a 3D volume, and isosurfaces are used to visually represent their temporal patterns and trends.

4.2 Spatially Organized Histograms

As previously described, histograms are a powerful method of visualizing overall trends in the variables of a group of objects. We extend this technique to also study how distributions differ throughout a 2D mathematical space of interest. This is done by partitioning the space into a desired number of subsections. All of the particles within a particular subsection are then sampled to form a histogram for that region. A comparison between these spatially organized histograms can show how the distributions differ according to spatial variation in the underlying data. Moreover, by adjusting the resolution of the spatial partition, one can highlight small-scale details, or general trends that permeate larger portions of the domain. This is discussed further in Section 4.3.

The main advantage to using spatially organized histograms is illustrated in Figure 3. One traditional visualization method for exposing spatial variation in data is to plot variables directly, using color to represent their values (part A). While this makes spatial variation clear, it is difficult to compare multiple variables at once without introducing confusion, over-plotting and clutter. Furthermore, using vector plots to describe the motion of a set of particles can result in a great deal of clutter (part B). To compare trends between multiple variables, scatter plots may be used where each axis represents a particular variable (part C). However, such a view does not show how the variables differ according to spatial variation. A spatially organized set of histograms can be used to represent both spatial variation as well as trends between variables simultaneously in a low clutter manner (part D).

4.2.1 *On-the-fly Sampling Using GPU Acceleration*

When the data sizes are manageable, histogram generation can be done on-the-fly using GPU acceleration. This is important since user controlled parameters strongly affect how the histograms need to be generated and presented in the visualization. By sampling the raw data values in real time, users can interactively explore multiple aspects of the data.

Our implementation for computing the spatially organized histograms for a single time step leverages the graphics pipeline using GLSL shaders. Each histogram bin corresponds to a single texel of a single channel texture. The variables and spatial positions of the particles that will be sampled are transferred to the GPU in the form of a vertex buffer. Interactive parameters such as the position and structure of the sampling grid, the histogram resolution, and the normalization factors, are transferred to the GPU as well. The vertex shader then maps each object to a sampling

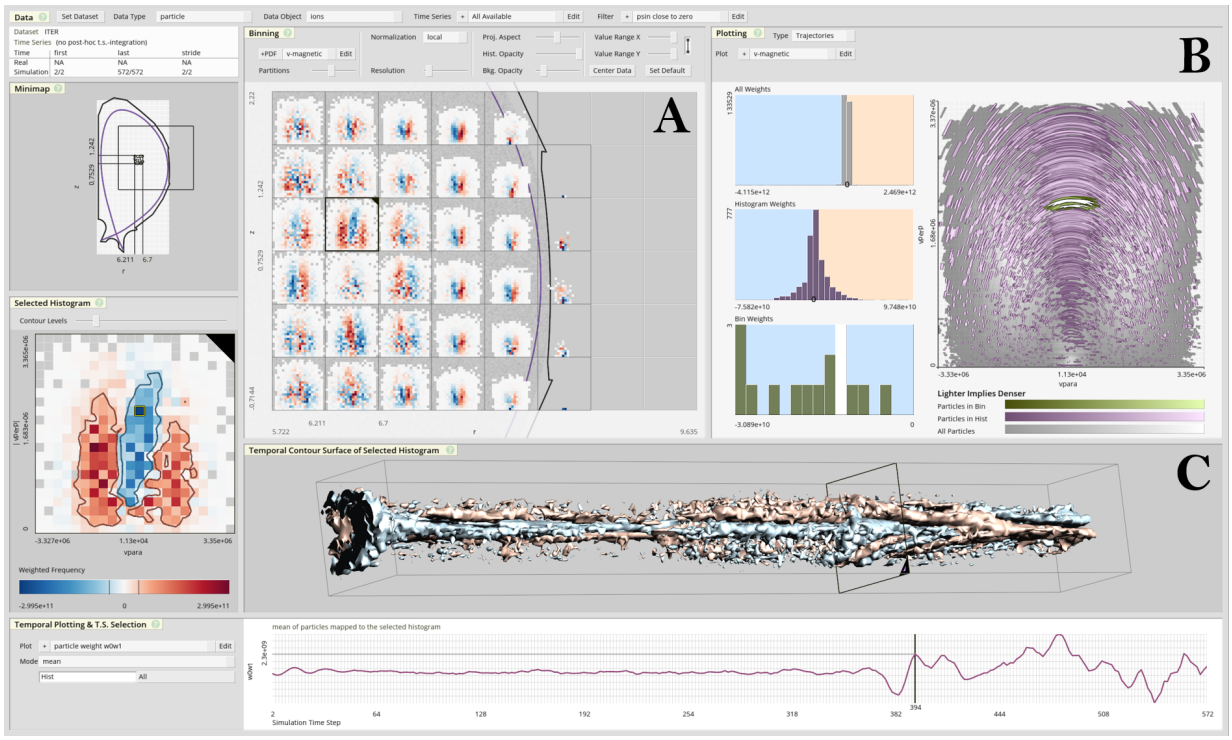


Fig. 4. A screenshot of the user interface. The interface consists of three main views: the histogram view (A), the trajectory view (B), the time-varying view (C). Additionally, a minimap depicting the zoom and location of the histogram view is shown on the top-left and a detailed view of the selected histogram is shown on the bottom-left.

region based on its spatial position and to a histogram bin based on its values. The result is a texel index which must then be converted into the correct position in normalized device coordinates before being input to the fragment shader. With additive blending enabled, the fragment shader then increases the value of the texel corresponding to the mapped histogram bin. The end result is a texture storing the computed spatially organized histograms.

4.2.2 *In Situ Generation and Sampling*

The interactive capabilities available through an on-the-fly sampling method are a major facet of our visualization tool. However, the computational overhead of histogram generation in larger datasets limits the responsiveness of the system, impeding real time exploration. As a result, we implement an alternative in situ processing based scheme which can be used to handle large scale datasets while minimizing loss of functionality in the visualization tool. In this scheme, raw data values are directly sampled and transformed into a histogram-based representation during the runtime of the simulation, allowing us to accumulate the statistical information of particles at much larger scales.

Each of these pregenerated histograms are saved so that they can be sampled in real time by the visualization tool during post hoc analysis. Each histogram is sampled onto a mesh, dependent on the type of simulation, where each grid point represents the center of a 3D volume from which particles are sampled. Since the size of each sampling region can vary, especially in unstructured grids, the volume of the sampling region is also saved. In turn, this can be used to normalize and more accurately sample histograms in the visualization tool. To suit their needs, users can control a balance between the temporal resolution in which to write the histograms to disk, the spatial resolution of the sampling mesh, and the resolution of the histograms themselves.

Our interactive system then samples this more manageable data representation for interactive exploration. Because the grid sizes can be dense and unstructured, it is often effective to visualize an overview at a reduced level of detail. For this reason, we partition the domain into disjoint sampling regions, as we do in the on-the-fly scheme, and sample grid points and their associated histograms by region. The histogram values of grid points that fall into the same sampling region are then merged using an appropriate normalization factor, N, and the net sampling volume, $\sum V_k$, where k is the grid point index. When integrating these techniques into GTS [2] (a fusion simulation), the normalization factor is equal to the inverse of the total phase space volume where particles are sampled from. The resulting normalized histogram values are computed as follows:

$$H(i,j) = \frac{N}{\sum V_k} \sum h_k(i,j)$$

where $h_k(i,j)$ is the 2D histogram of a sampled grid point with index k and $H(i,j)$ is the resulting merged histogram.

4.3 Visualization System

The visualization system ties together three main interactive views: a histogram view, a trajectory view, and a time-varying view. Each representation presents a different perspective into the data and allows for real time exploration. Furthermore, each view is linked so that selections and interactions in one view simultaneously affect the other. This

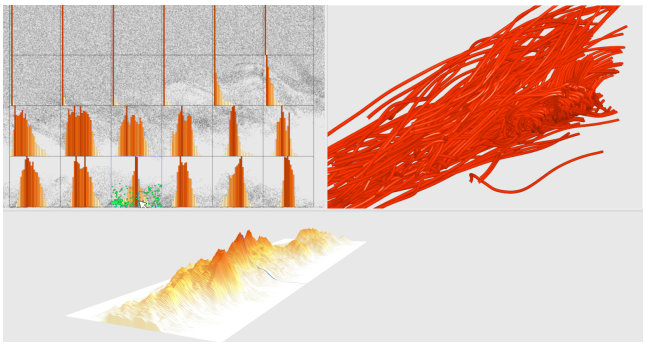


Fig. 5. An alternate example of the three main visualization views. A set of spatially organized 1D histograms is being generated from the particle data in the histogram view (top-left). Trajectories for particles corresponding to a moused-over bin are shown in 3D in the trajectory view (top-right). A spectrogram-like plot shows the evolution of the distribution in the selected histogram over time in the time-varying view (bottom).

multi-faceted approach allows users to investigate multiple aspects of the data at the same time. Figure 4 shows an overview of the user interface of the visualization tool.

4.3.1 The Histogram View

The histogram view presents the 1D or 2D histograms that were computed using either the on-the-fly or in situ sampling and binning schemes. By comparing the relative frequencies of different histogram bins, users can explore trends between multiple variables and spatial regions throughout the simulation domain simultaneously. Figure 5 shows the use of 1D histograms in the histogram view.

By default, this view uses a panning window approach. The screen space is partitioned into a regular grid of sampling regions. Entities within each sampling region are used to construct histograms in real time based on a configuration of parameters. The user interface can be used to control these parameters, such as the resolution of the spatial partition, the resolution of the histograms, and the variables that are used for the spatial organization and histogram generation. When the user drags the mouse over the view space, or zooms in, the sampled particles move relative to the screen, while the grid remains fixed. We also employ a layout that remains fixed with respect to the particles as the user pans and zooms. Such a layout has the benefit that zooming and panning can be done without affecting the sampling partition. However, the panning window approach gives more control over the locations of the sampling regions with respect to the particles.

One potential disadvantage of partitioning the space into rectangular sampling regions is that such regions may not conform well to the geometries underlying the data. This can be alleviated by projecting the data into different spatial layouts. For example, the tokamak device geometry is based on the magnetic flux surfaces that act to confine the plasma. The two spatial variables of interest in this space are the magnetic radius and the sweeping angle about the center of the poloidal plane. By projecting the data from this space into cartesian coordinates, such that poloidal angle is mapped to the x-axis, and the magnetic radius is mapped to the y-axis, the user can easily select regions that conform to

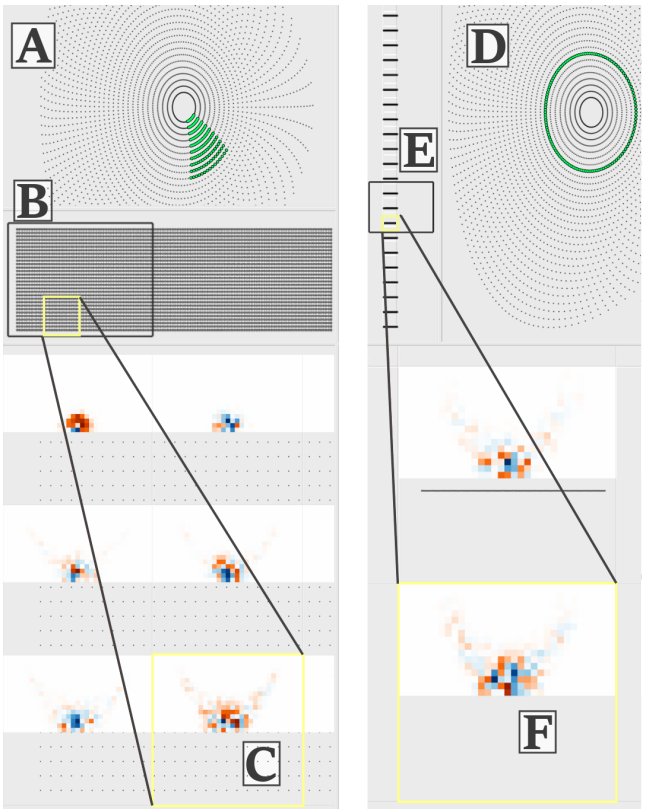


Fig. 6. Projecting curved geometry into a Cartesian viewport for easier visualization and interpretation. A,B,C) Projecting sampling points into a cartesian x-y layout based on radius and sweeping angle. A) The selected grid points in another spatial context, B) The mini-map view, C) The histograms overlaid over the projection. D,E,F) The same type of projection with one dimension scaled to an extreme in order to favor sampling the points based on one of the spatial variables over the other. D) The selected grid points, E) The mini-map, F) The associated histograms.

tokamak geometries of interest as in Figure 6. Furthermore, we can scale the projection in each dimension separately, allowing us to sample from larger ranges of one of the variables relative to the other.

4.3.2 The Trajectory View

The trajectory view is used to show the overall motion of sets of particles in either physical or phase space. Conveniently, the histogram view provides a powerful and unique way of selecting desired particle subsets. Due to the simple and regular structure of the spatially organized histograms, the user can apply mouse interactions over a sampling region and a bin in the associated histogram, and the system can unambiguously and efficiently extract the corresponding sampled particles. This process ultimately implements a 2-level range based selection, with the first level corresponding to a 2D spatial range (sampling region/histogram) and the lower level a 1D or 2D value range (bin of a 1D or 2D histogram). After the user makes such a selection, the particle trajectories are constructed and displayed in the trajectory view.

Since this process occurs in real-time, users can interactively explore the motion of particle subsets in either physical space, phase space, or both. An example of a selection can be seen in Figure 4B. In this case, the extent of all

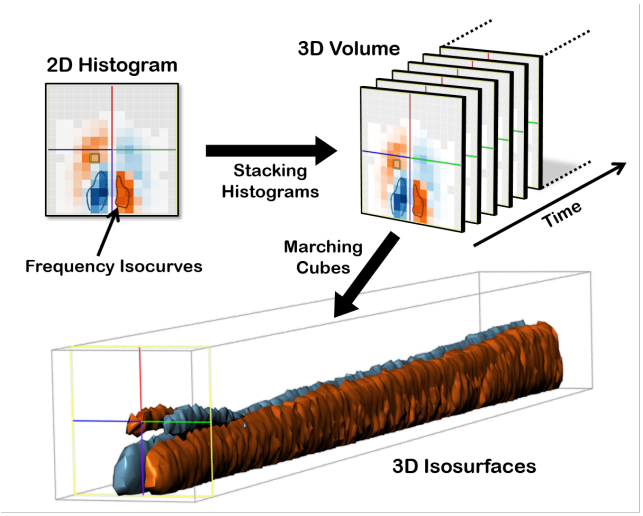


Fig. 7. The steps involved in generating the isosurfaces in the time-varying view. First a frequency isovalue is chosen and displayed as isocurves on the currently selected histogram. Next, histograms using the same sampling parameters are generated for all available time steps and stacked into a 3D volume. Lastly, an isosurface is generated with the selected isovalue using marching cubes.

trajectories is drawn in gray, the trajectories corresponding to the selected histogram are drawn in purple, and the trajectories corresponding to the selected bin are drawn in green. The lightness/darkness of the color represents the density of trajectories at that location. To the left of the trajectory view, bar charts show the distribution of weights in the dataset as well as the selected histogram and bin. This feature provides useful information in the case where the histograms are weighted, and also provides an interface for additional weight based levels of particle selection. See Section 4.3.4 for more details.

4.3.3 The Time-varying Visualization

While the histogram view is effective for comparing trends between variables as well as spatial differences throughout the domain, it's not effective for visualizing trends over time. As a result, we implement a time-varying view which can display temporal properties of a selected histogram. Occlusion and over-plotting make it difficult and confusing to view time-varying patterns for each bin in a 2D histogram. Instead we rely on isosurfacing techniques to extract surfaces from a volume that represent the histograms values at each time step. We choose isosurfaces because they are a simple and intuitive way of exploring a 3D volume. While a more flexible direct volume rendering approach could be another option, it would rely on careful selection of the transfer function, which could be a burden on the user, and could result in a more complex and difficult to interpret visualization.

Figure 7 describes the process through which we generate the time-varying view. First, a histogram of interest is selected. A user then chooses a frequency isovalue. This forms a set of contours which separates the 2D histogram into regions where the bins have a frequency higher than the isovalue and regions where the bins have a frequency lower than the isovalue. Next, additional 2D histograms over a sequence of time steps are generated using the same set of parameters (sampling region size, number of bins, etc.). These are then stacked into a 3D volume, where two dimensions represent each of the histogram variables and the third dimension represents time.

As the construction of the time-varying visualization requires computing histograms over many time steps, it often represents the primary performance bottleneck of the overall system. To accelerate this process we use the GPU. For good performance, we want to evenly balance the workload between GPU threads. We must also prevent multiple threads from attempting to update the same histogram bin (at a single memory location) simultaneously. These considerations affect our choice of how to parallelize the computation, and which GPU computing platform to use.

One option would be to assign each GPU thread a separate chunk of time steps. This has the benefit that concurrent writes to the same memory address are implicitly avoided as the computation for each time step is independent of each other. However, because the number of time steps is often small relative to the number of cores in modern GPUs, and each time step may correspond to a large chunk of data, this approach can lead to a load balancing problem. Another option is to assign threads work based on particle id. Because the number of particles is typically much larger than the number of time steps, as well as GPU cores, this approach can result in better load balancing. The drawback is that different particles handled by different threads may be mapped to the same bin, and therefore some locking mechanism is required to ensure that only one thread updates a bin's memory address at a time.

Modern GPU architectures, e.g. NVIDIA's compute architectures since Kepler [32], support efficient hardware based atomic operations on floating point data. Specifically, with CUDA, we can use the `atomic_add(float*, float)` function to increment a bin within the GPU kernel. Because we found parallelization over the particle ids using atomic operations to be more efficient than parallelization over the time steps, we choose to use this method.

We must also consider that all of the data that is needed to compute the temporal view may not fit in GPU memory at once. In this case, chunks of the data need to be processed one at a time, which means that large amounts of data need to be transferred to the GPU each time the histogram parameters have changed and the visualization needs to be recomputed. For our performance tests, we used a Titan X graphics card with 12 GB of memory, which could safely store over 500 time steps $\times$ 1,000,000 particles $\times$ 5 32-bit floating point variables persistently in memory. In addition to the particle data, GPU memory is allocated for the volume/histogram stack, but because the volume is typically very small relative to the size of the particle data, its memory footprint is insignificant.

When utilizing the in situ scheme, histograms have already been computed per grid point and thus typically represent a much smaller data size than the raw particles they were computed from. In addition, the grid points do not move over time, so the mapping of a grid point to a sampling region needs to be done only once while recomputing the time-varying view. As a result, this procedure can be done reasonably quickly using only CPU thread parallelism

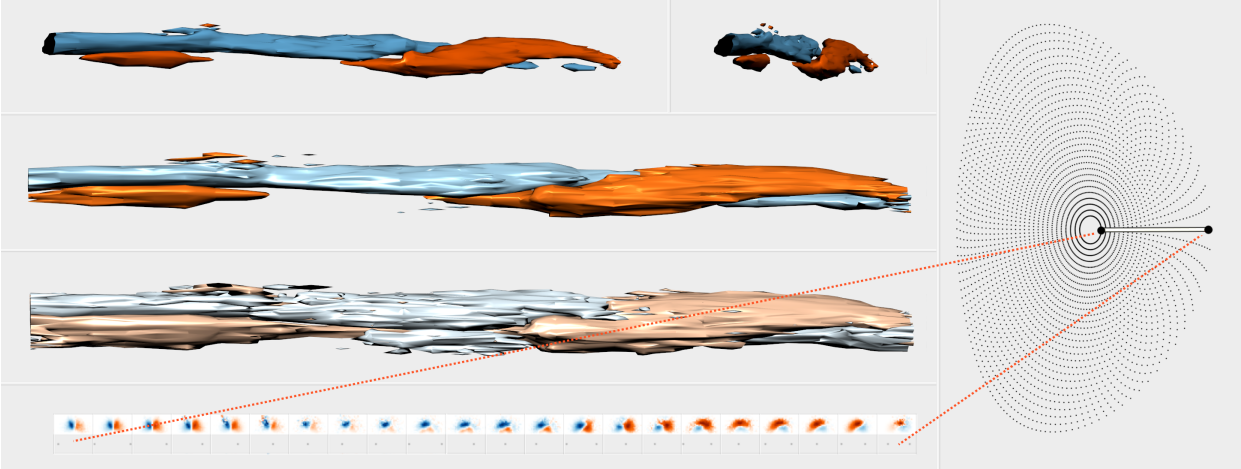


Fig. 8. An image demonstrating the ability to use the time-varying view to construct isosurfaces over a spatial path rather than through time. Histograms are sampled along this path (shown at the right). These are then stacked into a 3D volume to be visualized by isosurfaces. The views on the top left show isosurfaces from various isovalues and viewing angles and reveal distinct trends in the data.

along with CPU vector instructions.

Once the 3D volume has been computed, we perform marching cubes [33] to generate the isosurfaces. In general, the time it takes to construct the isosurface geometry is insignificant compared to the time it takes to generate the 3D volume. Performance tests for computing the 3D volume/2D histogram stack can be found in section 4.3.

Once the isosurfaces have been generated, users are free to pan, rotate, or zoom around the constructed mesh. An intersecting slicing plane indicates the currently selected time step and helps to orient the viewer. Users can also adjust any histogram or isovalue parameter and receive real time feedback on how the mesh changes form.

When visualizing a 1D histogram over time, we use a waterfall chart. In this case we can incorporate the values of each bin into the visualization. The 1D histograms are stacked and connected into complex polygons depth-wise over time. An example of this can be seen in Figure 5.

4.3.4 Other Features

As previously described, a particular bin in a weighted histogram could represent different numbers of discrete particles, depending on their weights; a small number of high weight particles and a large number of low weight particles could map to the same "weighted frequency". As a result, we also provide a visualization of the distribution of weights from a selected bin or histogram, as well as from the full particle data. This can be seen in the left portion of Figure 4B as 1D histograms. Selecting a subset of particle weights from the distribution highlights the bar and any corresponding trajectories in white.

An additional feature is the ability to stack histograms into a 3D volume based on a physical path through the domain rather than through time. In this case, histograms are generated along a discrete set of points along the path. Each point represents the center of a sampling region from which particles are sampled. In this case, the isosurfaces represent variations in the distribution between variables along this physical path. An example of this can be seen in Figure 8 where grid points are sampled along a line from the center of the poloidal plane to the edge of the high field (right) side. The histogram stack is ordered by radial distance and shows how the distributions change according to this spatial variation.

Normalization of the color scale is another factor that users can adjust interactively. Sometimes it is useful to visualize variations in bin frequency, or sample sizes, across multiple histograms at once. In such cases, it is important to ensure that the color mapping used is globally consistent across every region and time step. However, this can reduce the usable color range for histograms with relatively low maximum bin frequencies, and as a result can hide patterns and make variation in-perceivable for certain individual histograms. For this reason, the user interface includes a slider to apply a global normalization factor in order to amplify unpronounced features. We also provide a local normalization feature which allows the full range of color to be used in each histogram independent of its maximum frequency relative to the other histograms. Local normalization is favorable for showing how the variables are distributed within each region, individually. When the user chooses local normalization, the histogram view is affected as well as the temporal volume, in where the histograms will be normalized on a per-histogram, per-time step basis.

5 Results

We demonstrate the effectiveness of our system using a set of real world datasets in the fields of fusion and particle accelerator research. We use results from two different fusion simulations to test the on-the-fly and in situ sampling schemes provided by our system and demonstrate the unique patterns that the visualization can highlight. We then test the system using a particle accelerator dataset to show its applicability to other fields of science. Lastly, we provide performance results to justify the interactivity of our technique when using either sampling scheme.

5.1 Fusion Datasets

Nuclear fusion is a promising future energy source, but challenges remain before it can be made practical. Simulations of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2642103, IEEE Transactions on Visualization and Computer Graphics

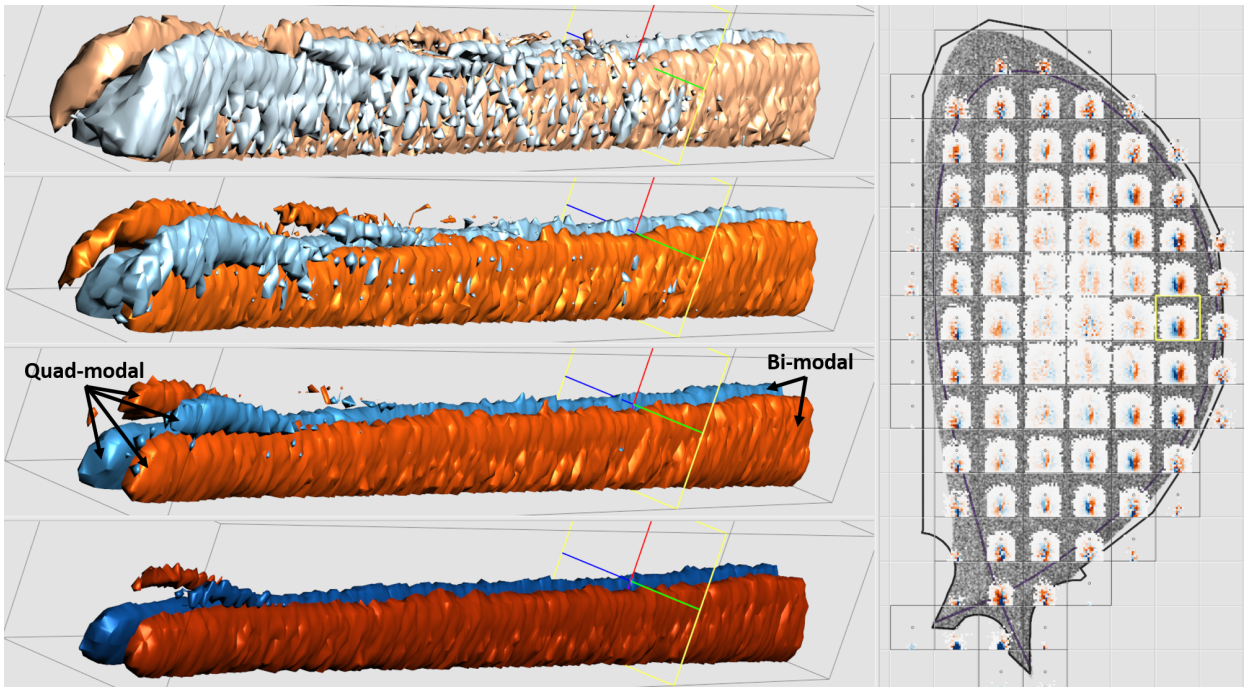


Fig. 9. Using the time-varying visualization to study temporal trends for a selected histogram in the XCG1 dataset. The horizontal histogram variable represents velocity parallel to the magnetic field, while the vertical histogram variable represents velocity perpendicular to the magnetic field. Left) The temporal view is shown using 4 different isovalues, low to high from top to bottom. Right) The histogram view. In the mapping of color to frequency, a global threshold was used, and the volume was normalized uniformly at each time step. The visualization highlights the differences in overall weighted particle frequency in different areas of the domain as well as over time. One interesting aspect shown in this visualization is how the distribution starts out quad-modal and evolves to become bi-modal.

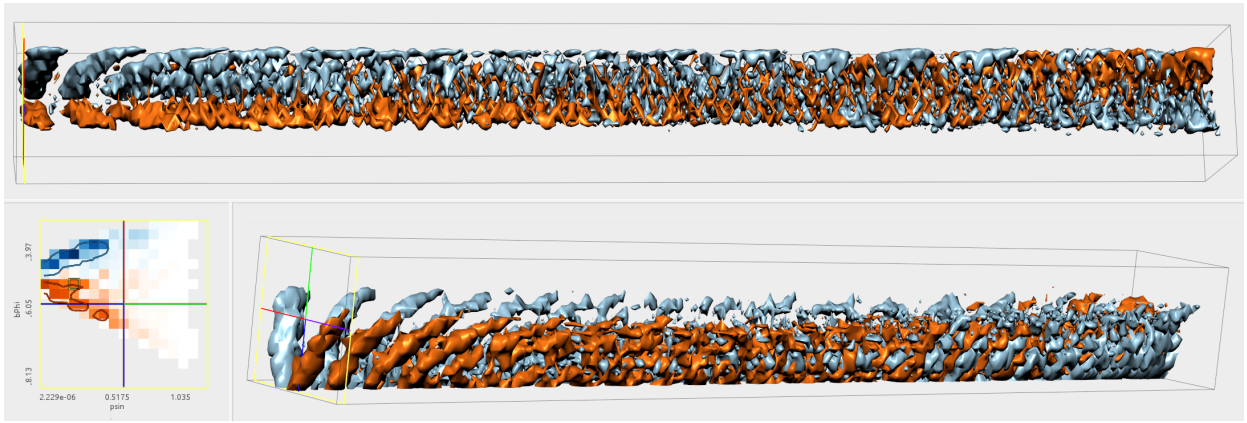


Fig. 10. Using the time-varying visualization to study temporal trends for a selected histogram in the XCG1 dataset. The horizontal histogram variable represents the magnetic radius, while the vertical histogram variable represents the magnetic field strength in a direction that runs around the torus geometry. The isosurfaces reveal complex wavelike patterns which tend to swap places with their positively and negatively weighted counterparts.

the physical devices designed to harness the power of this phenomena play a major role in acquiring the knowledge required to solve these challenges. The fusion datasets we explore come from large scale simulations developed by research teams at the Princeton Plasma Physics Laboratory.

The first dataset comes from a simulation called XGC1 [1], which is designed to study the physics of magnetically confined plasmas in the edge region of tokamak devices. The particular simulation run we study represents the International Thermonuclear Experimental Reactor (ITER). The dataset we use consists of about 286,000 particles over 282 time steps. We utilize the on-the-fly sampling and binning scheme with this data.

The second dataset comes from a simulation called GTS [2], which is designed to study microturbulence in fusion devices. In this case, we use the in situ sampling scheme, in which a set of histograms is computed while the simulation is running. These histograms are laid out on a mesh consisting of 3,840 grid points, each with a resolution of 33 by 17 bins. In addition, we use a dump of ~15,000,000 particles in order to compare histograms resulting from our on-the-fly and in situ schemes.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2642103, IEEE Transactions on Visualization and Computer Graphics

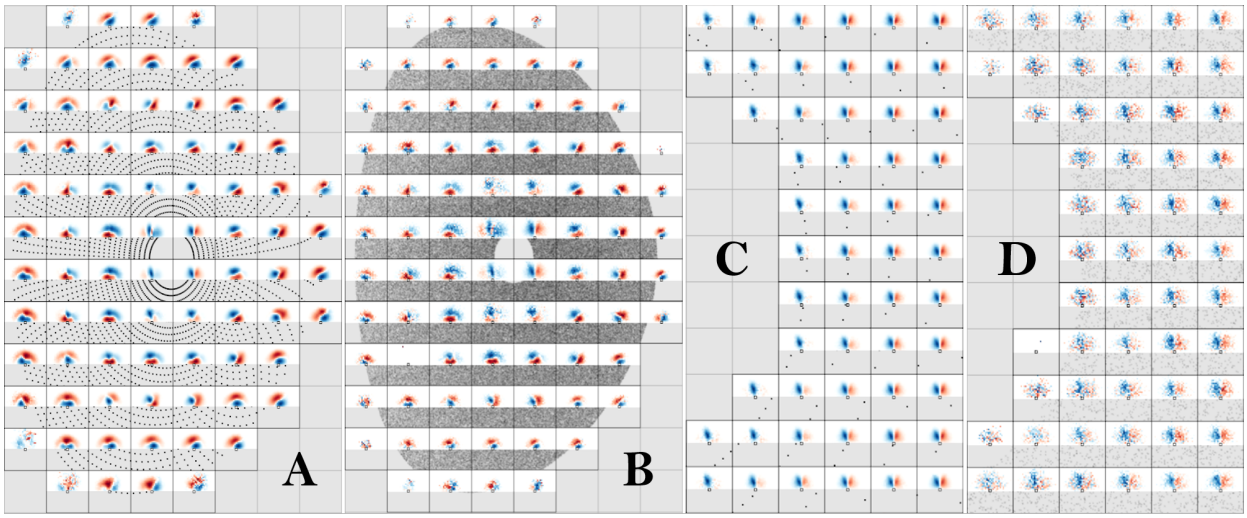


Fig. 11. A) Visualizing the in situ generated histograms by merging histograms from the simulation grid points within each sampling region. B) Comparing the same simulation using histograms generated on-the-fly from particles directly. C/D) Zooming in the domain from A and B respectively highlights the advantages of using the in situ method. When the sampling regions have too few particles to sample (D) the histograms degrade in quality and ability to faithfully capture the distributions. This is not an issue for the in situ case (C) since it was able to sample a much larger number of particles during the simulation.

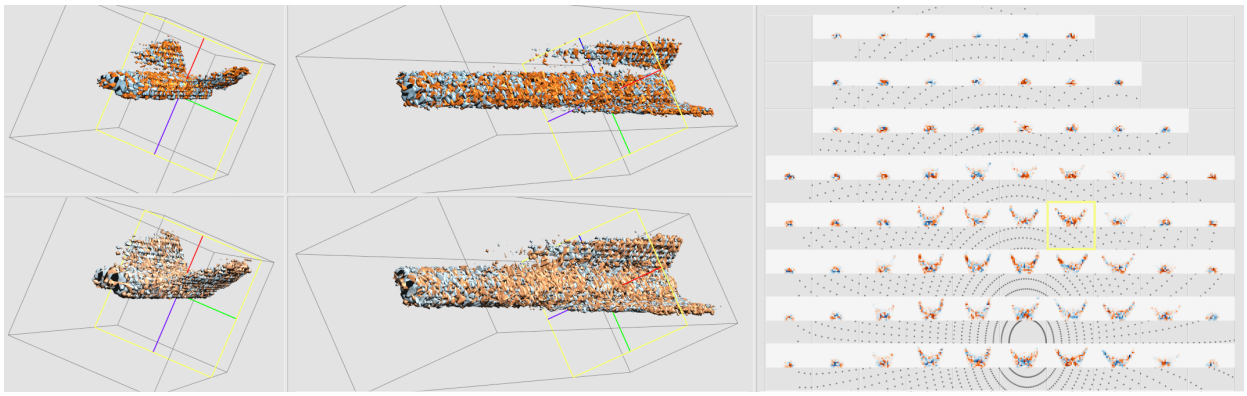


Fig. 12. An image depicting the in situ generated histograms from the GTS dataset. The horizontal histogram variable represents the velocity parallel to the magnetic field, while the vertical histogram variable represents the velocity perpendicular to the magnetic field. The right side shows spatial variations in the distributions throughout the simulation domain. The left side shows two time-varying isosurfaces of the selected histogram over time at different isovalues. This reveals a unimodal to trimodal evolution of the distribution.

5.1.1 On-the-fly generation with XGC1

The fusion simulation represents a complex torus-like shape. However, scientists are interested in the motion of the plasma towards or away from chamber walls (in the direction of the "minor radius" of the torus). As a result, many visualizations (including our own) focus on presenting information on a 2D "poloidal" slice where the less interesting motion in the third dimension is hidden. We project all particles throughout the torus onto the slice view so that the resulting histogram-based visualization, while 2D in nature, represents information from the entire 3D domain.

Figure 9 shows an example of the system employing our on-the-fly sampling and binning scheme to visualize the XGC1 dataset. In this case, the weight of the particle describes the perturbation from the background (Maxwellian) distribution. Positive/negative weights describe an increase/decrease of particle population from the background. The histograms are computed using these weights, and a divergent color map is used to differentiate negatively (blue) and positively (red) valued bins. From the figure, it can be seen that the distribution is initially quad-modal and eventually becomes bi-modal. Additionally, one can see how the contours expand over time; In this case, that effect is primarily due to an overall growth in particle weight as the simulation progressed.

When using our visualization tool, physicists have described that it can easily investigate the perturbed distribution function and the particle trajectories which are responsible for the perturbation. They mention that one very useful example is monitoring growth of particle weights. In their simulations, excessive growth of particle weight could induce statistical noises and degrade the accuracy of the simulations. Hence, monitoring the growth of particle weights and identifying the cause are important to regulate the statistical noises.

Another example can be seen in Figure 10. In this example, the horizontal histogram variable represents the

magnetic radius, while the vertical histogram variable represents the magnetic field strength in a direction that runs around the torus geometry. The time-varying isosurfaces reveal complex wavelike patterns forming in distinct positively and negatively weighted groups. Furthermore, these groups tend to swap positions over time occupying different regions of phases space.

5.1.2 In Situ Generation with GTS

Next we test the in situ sampling and binning scheme using data from the GTS simulation. Histograms are generated during the simulation with access to the full particle data. These histograms are then sampled based on user controlled parameters. The values in the histograms represent energy distributions in that the particles are weighted by energy relative to a background value (which allows them to be positive or negative).

The horizontal axis represents velocity parallel to the direction of the magnetic field, whereas the vertical axis represents velocity perpendicular to the magnetic field. Since the perpendicular velocity is represented as a magnitude, it is always positive and results in only the top half of our histograms bearing values. These types of velocity plots are commonly used when studying gyrokinetic simulations.

Figure 11 shows an example of the in situ sampled histograms using our visualization tool (A/C) vs. a comparison to the on-the-fly particle-based sampling scheme (B/D). Due to I/O limitations, the simulation is only able to dump a small subset of the full particle data, forcing the on-the-fly method to sample only 3% of all particles. The points in the background show the grid points over which the in situ generated histograms were computed. From the images it is clear that the in situ version was able to capture more detail since it represents statistical information from a much larger sample size. This is further exacerbated when zooming into the domain; as fewer particles are sampled per histogram, eventually we observe excessive degradation of statistical quality(D).

Figure 12 shows an alternate run of the GTS simulation and focuses on studying time-varying properties. The left side of the figure shows three different isosurfaces of the selected histogram (which is outlined in yellow) with time increasing towards the right. Each of these isosurfaces shows a unimodal distribution towards the start of the run with nearly all particles exhibiting a small parallel and perpendicular velocity. This evolves into an overall trimodal distribution consisting of: a small parallel and perpendicular velocity, a large positive parallel and large positive perpendicular velocity, and lastly a large negative parallel and large positive perpendicular velocity. This is what forms the distinct "V-like" shape in the images.

We can also look at spatial variations within a single time step as shown in the right side of the image. We can see that these "V-like" distributions are more prevalent near the center of the cutting plane, whereas distribution closer to the edges are smaller in shape, and consist primarily of particles with small parallel and perpendicular velocities.

5.2 Accelerator Dataset

Our next example uses data from a simulation called ACE3P [3], which is used to study the electromagnetic dynamics within particle accelerators. The specific device this simulation run studies is called a cryomodule, which uses a set of resonating cavities to accelerate the particles. A better understanding of certain processes such as dark current, in which charged particles become emitted from the cavity surfaces and enter the accelerating beam, can lead to an improved design of the device.

In this application, we were interested in examining the behavior of different clusters of particles rather than specific spatial regions. Therefore, we constructed a temporal histogram stack for each particle cluster separately, and used the sampling grid only for our initial cluster selection. The histograms here represent momentum in the x and y directions (perpendicular to the length of the device). The visualization was made from about 15,000 particles over about 3,000 time steps. This gives an overview of the movement of the particles perpendicular to the beam and easily highlights segments in time where apparent extreme or abnormal behavior occurs.

Figure 13 shows a comparison of 5 different clusters of particles (labeled A-E). In the time-varying view, time is increasing towards the right. Since each isosurface represents momentum perpendicular to the motion of the beam, expansion of the thickness of the isosurface tube can represent a point in time where instabilities are occurring. Such instabilities can cause particles to exit the beam and become deposited into a cavity downstream, potentially damaging the device. Comparatively, it appears that cluster E is less stable than clusters A through D.

5.3 Performance Results

The performance results for generating spatially organized 2D histograms for a single time step can be found in our previous work [5]. This section will instead focus on performance results for generating the 3D temporal volume of stacked histograms for both the on-the-fly and in situ methods. Generating the histogram volume is typically the largest bottleneck of the overall system. Isosurface construction times are insignificant in comparison, considering the typical sizes of the associated volumes. For example, a volume representing a 32×32 bin 2D histogram over time could include over 16,000 time steps (larger than the typical use case) before exceeding the size of a 256^3 volume (relatively small by today's standards). For testing, we used an Intel i7-5939K processor with 6 cores at 3.5 GHz and a Titan X graphics card with 12 GB of memory. The main code was compiled using the g++ compiler (version 4.8.4), while the CUDA GPU kernel was compiled using the nvcc compiler (version 7.0), both with `-O3` optimizations enabled.

Figure 14 shows the timing results for the on-the-fly method. The graph shows the time it takes to construct the 3D volume as a function of the number of time steps in the data. Each curve represents a different number of particles that are sampled. Since the number of particles in each sampling region can vary over time, we artificially ensured that each particle would be mapped to the selected sampling region, which represents the worst case computationally. In practice, the performance should be higher on average, as in the intended use case the selected sampling region will contain only a fraction of the full data.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2016.2642103, IEEE Transactions on Visualization and Computer Graphics

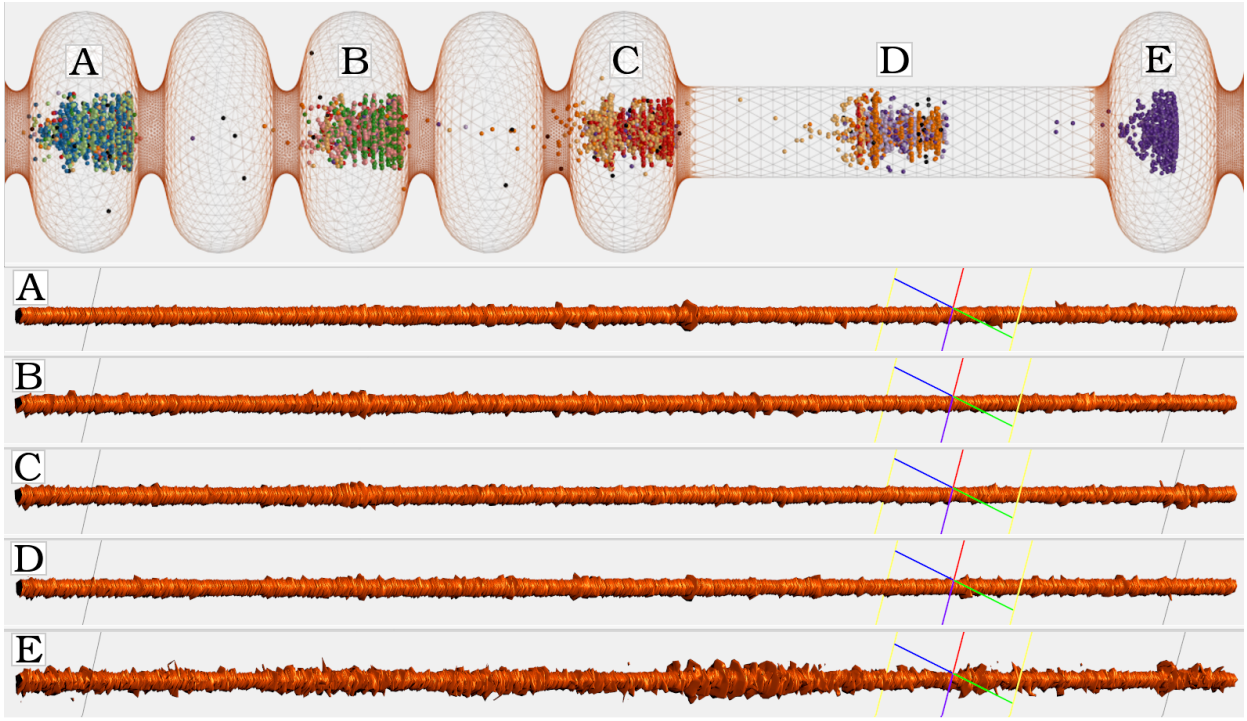


Fig. 13. The top of the image shows a visualization of clusters of particles in the cryomodule device with the mesh depicting the shape of the device/individual cavities. In this case, they were colored categorically according to which cavity they were emitted from. A temporal histogram isosurface was computed for each cluster, based on momentum in the x and y directions (perpendicular to the beam), and are shown side by side for comparison. As opposed to the other clusters, the leading cluster (E) is made up almost entirely from particles that were emitted from the same cavity. Additionally, the particles that make up this cluster experienced a higher than normal level of instability, especially during a time segment starting about halfway though the available time steps.

The graph itself shows a linear increase in the time it takes to compute the 3D volume, with distinct jumps that occur at the points where all of the particle data cannot fit into GPU memory. At those points, multiple chunks of data need to be transferred to the GPU and processed one at a time whenever the volume needs to be recomputed. The red curve shows the most extreme test case, where we were able to process 22 GB of particle data (1,000,000 particles $\times$ 1,100 time steps $\times$ (2 spatial variables + 2 histogram variables + 1 weighting variable) $\times$ 4 bytes) into a volume in less than 2 seconds. As the number of particles increases, the user may need to reduce the number of time steps and vice versa, however the visualization can remain interactive with reasonably large data sizes. For example, with 1,000,000 particles per time step, we can generate volumes representing 500 time steps and still get reasonably high frame rates. This delay could be alleviated further with the use of additional GPUs which could each process separate chunks of data simultaneously.

Figure 15 shows the performance tests of the in situ version. Each curve represents a different number of grid points that are within the selected sampling region. Because the grid points remain fixed over time (unlike particles), mapping them to sampling regions only needs to be done once, each time the sampling layout changes, and the rest of the computation involves only summations, which can be done very efficiently. These timing results are based on our parallel CPU implementation (using OpenMP) which also utilizes vector CPU operations to further increase performance.

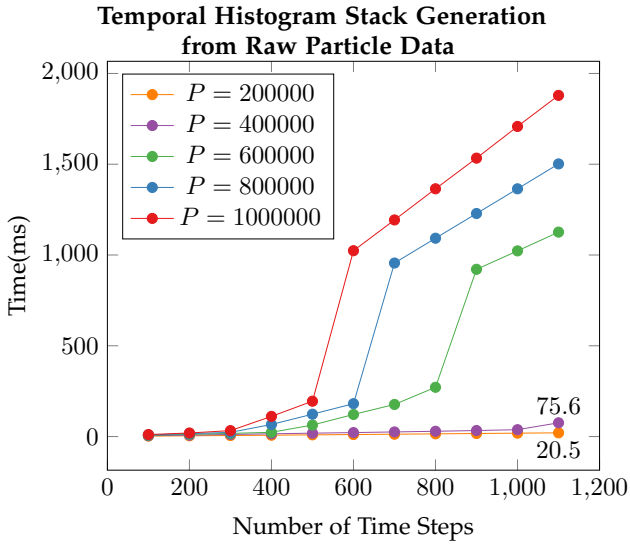


Fig. 14. Timing results for the time-varying visualization using the on-the-fly method with varying numbers of particles (P). Jumps occur when the GPU can no longer keep all of the time steps persistently in memory. Sampling 1,000,000 particles, we were able to achieve fluid interaction while computing up to 500 time steps per volume.

6 Discussion

The above results demonstrate the ability of our system to visualize trends between variables in conjunction with spatial trends, and to analyze time-varying properties. Maintaining generality allows the system to be applicable to a

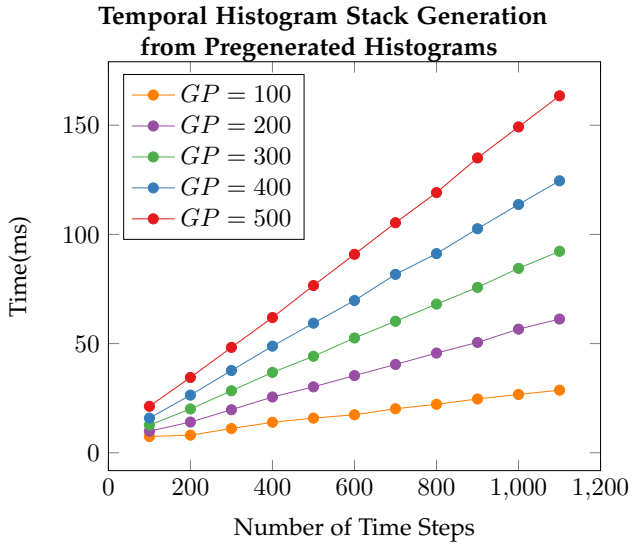


Fig. 15. Timing results for the time-varying visualization using the in situ method with varying numbers of pregenerated histograms sampled (GP). Since each thread is simply computing summations over the number of merged grid points, it scales linearly and is computationally efficient.

variety of data types, while employing two data processing schemes allows the system to handle various data scales.

6.1 Scalability

Using the on-the-fly scheme with GPU acceleration, the data scales that desktop computers can handle depend on the number of data entities (sample size $\times$ time steps). The system can handle around $\sim$ 1 billion data entities in under a few seconds when requesting a time-varying representation. Limits can be pushed further by utilizing multiple GPUs or even a distributed setting which can process chunks of particles simultaneously.

For larger data scales, the system can utilize our in situ scheme. Since histograms are generated during simulation time, they can represent information from massive numbers of particles. While potentially limiting the ability to explore trajectories (if the particle data is not available post hoc), the larger sample size can be a huge advantage.

6.2 Future Work

Future work will involve further integration of these methods into the simulations in order to visualize trends directly while they are running, further minimize expensive I/O costs, and support preselection of data subsets to be saved to disk. We can also improve the scheme for sampling pregenerated histograms. For example, as each grid point also represents a volumetric cell, which may overlap multiple sampling regions in our post-hoc system, we consider weighting each merged grid point according to the fraction of its volume which overlaps the sampling region.

7 Conclusion

Overall, this work presents an extension to our previous work [5] on using spatially organized velocity histograms to visualize motion. We have extended the capabilities of the technique by generalizing it to any type of variable and employed the use of isosurfaces to effectively visualize the time-varying trends of distributions within the histograms. By employing an on-the-fly scheme for small and medium-scale datasets, and an in situ scheme for large-scale datasets, we can handle a variety of simulation sizes. Furthermore, we demonstrate the usefulness of the system using real world datasets in the fields of fusion and particle accelerator science and present performance tests to demonstrate its interactive capabilities.

Acknowledgments

We would like to thank our collaborators at the Princeton Plasma Physics Laboratory who worked with us, specifically Seung-Hoe Ku, Robert Hager, and Randy Michael Churchill. We would also like to thank our collaborators at the SLAC National Accelerator Laboratory for providing the particle accelerator dataset. This research is sponsored in part by the U.S. Department of Energy through grants DE-SC0007443 and DE-SC0012610.

References

[1] M. Adams, S.-H. Ku, P. Worley, E. D'Azevedo, J. Cummings, and C.-S. Chang, "Scaling to 150k cores: Recent algorithm and performance engineering developments enabling XGC1 to run at scale," *Journal of Physics: Conference Series.*, vol. 180, no. 1, 2009.

[2] W. X. Wang, Z. Lin, W. M. Tang, W. W. Lee, S. Ethier, J. L. V. Lewandowski, G. Rewoldt, T. S. Hahm, and J. Manickam, "Gyro-kinetic simulation of global turbulent transport properties in tokamak experiments," *Phys. Plasmas*, vol. 13, no. 19, p. 092505, 2006.

[3] O. Kononenko, L. Ge, K. Ko, Z. Li, C. K. Ng, and L. Xiao, "Progress on the multiphysics capabilities of the parallel electromagnetic ACE3P simulation suite," in *31st International Review of Progress in Applied Computational Electromagnetics (ACES)*, Mar 2015, pp. 1–2.

[4] C. S. Yoo, E. S. Richardson, R. Sankaran, and J. H. Chen, "A DNS study on the stabilization mechanism of a turbulent lifted ethylene jet flame in highly-heated coflow," *Proceedings of the Combustion Institute*, vol. 33, no. 1, pp. 1619–1627, 2011.

[5] T. Neuroth, F. Sauer, W. Wang, S. Ethier, and K.-L. Ma, "Scalable visualization of discrete velocity decompositions using spatially organized histograms," in *IEEE 5th Symposium on Large Data Analysis and Visualization (LDAV)*, Oct 2015, pp. 65–72.

[6] R. M. Kirby, H. Marmanis, and D. H. Laidlaw, "Visualizing multivalued data from 2D incompressible flows using concepts from painting," in *Proceedings of IEEE Visualization Conference*, Oct 1999, pp. 333–340.

[7] H. Obermaier and K. Joy, "Derived metric tensors for flow surface visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2149–2158, Dec 2012.

[8] J. L. Helman and L. Hesselink, "Representation and display of vector field topology in fluid flow data sets," *Computer*, vol. 22, no. 8, pp. 27–36, Aug 1989.

[9] N. Ihaddadene and C. Djeraba, "Real-time crowd motion analysis," in *ICPR 19th International Conference on Pattern Recognition*, Dec 2008, pp. 1–4.

[10] A. Romanoni, M. Matteucci, and D. G. Sorrenti, "Background subtraction by combining temporal and spatio-temporal histograms in the presence of camera movement," *Machine Vision and Applications*, vol. 25, no. 6, pp. 1573–1584, Dec 2014.

[11] N. Dalal, B. Triggs, and C. Schmidi, "Human detection using oriented histograms of flow and appearance," *Computer Vision - ECCV*, vol. 3952, pp. 428–441, Dec 2006.

[12] C. Jung, L. Hennemann, and S. Raupp Musse, "Event detection using trajectory clustering and 4-D histograms," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1565–1575, Nov 2008.

[13] C. Tominski, S. Gladisch, U. Kister, R. Dachselt, and H. Schumann, "A survey on interactive lenses in visualization," in *EuroVis - STARs*, R. Borgo, R. Maciejewski, and I. Viola, Eds. The Eurographics Association, 2014.

[14] C. Wittenbrink, A. Pang, and S. Lodha, "Glyphs for visualizing uncertainty in vector fields," *IEEE Transactions on Visualization and Computer Graphics*, vol. 2, no. 3, pp. 266–279, Sep 1996.

[15] M. Hlawatsch, P. Leube, W. Nowak, and D. Weiskopf, "Flow radar glyphs - static visualization of unsteady flow with uncertainty," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 1949–1958, Dec 2011.

[16] E. Marsch, X.-Z. Ao, and C.-Y. Tu, "On the temperature anisotropy of the core part of the proton velocity distribution function in the solar wind," *Journal of Geophysical Research*, vol. 109, p. A04102, 2004.

[17] M. S. Paoletti, M. E. Fisher, K. R. Sreenivasan, and D. P. Lathrop, "Velocity statistics distinguish quantum turbulence from classical turbulence," *Physical Review Letters*, vol. 101, p. 154501, Oct. 2008.

[18] M. Salewski, B. Geiger, A. S. Jacobsen, M. García-Muñoz, W. Heidbrink, S. B. Korsholm, F. Leipold, J. Madsen, D. Moseev, S. K. Nielsen *et al.*, "Measurement of a 2D fast-ion velocity distribution function by tomographic inversion of fast-ion D-alpha spectra," *Nuclear fusion*, vol. 54, no. 2, p. 023005, 2014.

[19] J. Kehrer and H. Hauser, "Visualization and visual analysis of multifaceted scientific data: A survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 3, pp. 495–513, Mar 2013.

[20] D. A. Kozak, T. H. Stievater, M. W. Pruessner, and W. S. Rabinovich, "Waveguide modal analysis using an FFT spectrogram technique: Application to mode filters," in *Advanced Photonics 2016 (IPR, NOMA, Sensors, Networks, SPPCom, SOF)*. Optical Society of America, 2016, p. JTu4A.3.

[21] Y. Ojima, E. Nakamura, K. Itoyama, and K. Yoshii, "A hierarchical bayesian model of chords, pitches, and spectrograms for multipitch analysis," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2016, pp. 309–315.

[22] A. D. Luca, M. Contu, S. Hristov, L. Daniel, M. Gashinova, and M. Cherniakov, "FSR velocity estimation using spectrogram," in *17th International Radar Symposium (IRS)*, May 2016, pp. 1–5.

[23] R. Rekapalli and R. Tiwari, "Singular spectral analysis based filtering of seismic signal using new weighted eigen spectrogram," *Journal of Applied Geophysics*, vol. 132, pp. 33–37, 2016.

[24] B.-R. Choi, W. Nho, T. Liu, and G. Salama, "Life span of ventricular fibrillation frequencies," *Circulation research*, vol. 91, no. 4, pp. 339–345, 2002.

[25] D. Kao, J. L. Dungan, and A. Pang, "Visualizing 2D probability distributions from EOS satellite image-derived data sets: a case study," in *Proceedings of IEEE Visualization Conference*, Oct 2001, pp. 457–589.

[26] J. Woodring and H.-W. Shen, "Chronovolumes: A direct rendering technique for visualizing time-varying data," in *Proceedings of the Eurographics/IEEE TVCG Workshop on Volume Graphics*, 2003, pp. 27–34.

[27] 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 *Proceedings of IEEE Visualization Conference*, Oct 1999, pp. 371–545.

[28] H. Krishnan, C. Garth, and K. Joy, "Time and streak surfaces for flow visualization in large time-varying data sets," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 1267–1274, Nov 2009.

[29] J. P. M. Hultquist, "Constructing stream surfaces in steady 3D vector fields," in *Proceedings of IEEE Visualization Conference*, Oct 1992, pp. 171–178.

[30] W. Widanagamaachchi, J. Chen, P. Klacansky, V. Pascucci, H. Kolla, A. Bhagatwala, and P. T. Bremer, "Tracking features in embedded surfaces: Understanding extinction in turbulent combustion," in *IEEE 5th Symposium on Large Data Analysis and Visualization (LDAV)*, Oct 2015, pp. 9–16.

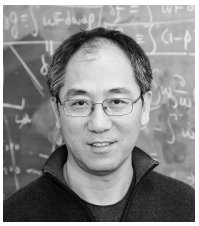
[31] D. Thompson, J. A. Levine, J. C. Bennett, P. T. Bremer, A. Gyulassy, V. Pascucci, and P. P. Pbay, "Analysis of large-scale scalar data using hixels," in *IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, Oct 2011, pp. 23–30.

[32] NVIDIA, "Kepler GK110 whitepaper." [Online]. Available: http://www.nvidia.com/content/PDF/kepler/NVIDIA-Kepler-GK110-Architecture-Whitepaper.pdf

[33] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *SIGGRAPH Comput. Graph.*, vol. 21, no. 4, pp. 163–169, Aug 1987.

Tyson Neuroth is a second year graduate student at the University of California, Davis, studying computer science and visualization under Kwan-Liu Ma. His research interests include scientific visualization, high performance computing, and human computer interaction. He received a BS in computer science from the University of California, Davis.

Franz Sauer is a Ph.D. candidate at the University of California, Davis, studying computer science and scientific visualization under Kwan-Liu Ma. His research interests include data visualization, large-scale scientific simulations, computer graphics, and physics. Sauer received a BS in physics from the California Institute of Technology.



Dr. Weixing Wang is a principal research physicist at the Princeton Plasma Physics Laboratory. He received a Ph.D. in plasma physics from the Graduate University for Advanced Studies at the National Institute for Fusion Science in Japan. His professional interests cover theory and computation of plasma micro-instabilities, turbulent and collisonal plasma transport in magnetic fusion experiments, physics of plasma confinement, gyrokinetic simulation, and advanced simulation algorithms.

Dr. Stephane Ethier is a Principal Computational Physicist at the Princeton Plasma Physics Laboratory. He is Deputy Head of the Computational Plasma Physics Group (CPPG) and leads the High Performance Computing effort. He obtained a Ph.D. in 1996 from the Institut National de la Recherche Scientific (INRS) in Varennes, Canada. Dr. Ethiers work covers all aspects of high performance computing in support of PPPLs large-scale scientific codes.

C.S. Chang has led multiple major projects in the past decade in both fusion physics and computational science. He is the present head of the SciDAC-3 Center for Edge Plasma Simulation (EPSI), actively leading an extreme scale computing research on multiscale self-organization physics in magnetic confinement plasma. He is a Fellow of the American Physical Society, and member of numerous national and international advisory and executive committees. Prior to joining Princeton Plasma Physics Laboratory, C.S. Chang was a Professor of Physics at Korea Advanced Institute of Science and Technology, and jointly, a research faculty member with rolling tenure at Courant Institute of Mathematical Sciences, NYU.

Kwan-Liu Ma is a professor of computer science and the chair of the Graduate Group in Computer Science (GGCS) at the University of California, Davis, where he leads the VIDi research group and directs the UC Davis Center for Visualization. His research interests include visualization, high-performance computing, and user interface design. Ma received a Ph.D. in computer science from the University of Utah. He is an IEEE Fellow.