As printed in Volume Graphics 2001, Mueller and Kaufman eds.

A Study of Transfer Function Generation for Time-Varying Volume Data

T.J. Jankun-Kelly and Kwan-Liu Ma

Visualization and Graphics Research Group, Center for Image Processing and Integrated Computing, Department of Computer Science, University of California, Davis, CA 95616, {kelly,ma}@cs.ucdavis.edu

Abstract. The proper usage and creation of transfer functions for time-varying data sets is an often ignored problem in volume visualization. Although methods and guidelines exist for time-invariant data, little formal study for the time-varying case has been performed. This paper examines this problem, and reports the study that we have conducted to determine how the dynamic behavior of time-varying data may be captured by a single or small set of transfer functions. The criteria which dictate when more than one transfer function is needed were also investigated. Four data sets with different temporal characteristics were used for our study. Results obtained using two different classes of methods are discussed, along with lessons learned. These methods, including a new multi-resolution opacity map approach, can be used for semi-automatic generation of transfer functions to explore large-scale time-varying data sets.

1 Introduction

Transfer function generation for time-invariant volumetric data has been widely studied [2, 3, 5–7, 12]. Several methods exist to create both color and opacity maps for a variety of data types. In contrast, the complementary work for time-varying data has not been particularly addressed. One practice for dealing with such data is to apply a single transfer function—created using one of the volumes in the time-series—to all volumes in the series. It is clear that this practice is not always applicable to general time-varying data sets. This paper describes an investigation into the generation and use of transfer functions for time-varying data.

Our goal is to determine whether a single transfer function can capture the most relevant information in a time-varying data set. If such a transfer function does not exist, can we instead define a minimal set of transfer functions? Furthermore, can we classify the different types of temporal behavior which volume data exhibits and create transfer functions for them accordingly? To address these issues, we have designed a set of analyses on the time-varying volumes and the transfer functions derived from them.

Recent advances in graphics hardware have allowed for interactive viewing of time-invariant volumes [8, 13]. Efforts for interactive viewing of time-varying volume data are under way [10]. Consequently, in seems there would be little need to suggest transfer functions to the users of these systems since they can identify features of interest quite easily by themselves. However, the continuing growth of data size will outstrip the capabilities of these systems: pixel fill-rate, bus transfer speed, and out-of-core access times limit real-time interaction with very large data sets. Non-rectilinear volume data sets also pose a problem for conventional interactive renderers. Furthermore, because of bandwidth limitations on most conventional networks, data sets consisting of hundreds to thousands of time-steps cannot be transferred to scientific workstations at rates sufficient for interactive exploration. Thus, methods that pre- or post-process the data to create transfer functions for time-varying data, like those discussed here, are still needed. These methods are independent of grid topology and the method of rendering. Our work helps elucidate the effects that the temporal dimension has on the generation of transfer functions, which would assist visualization users decide what transfer functions best explore their data. They can even assist interactive renderers by “refining” a user’s transfer function during idle processor time.

2 Time-Varying Volume Visualization

In time-varying volume visualization, the phenomena under study evolves in some manner over time and space. As in time-invariant volume visualization, the features of interest are isosurfaces, boundary surfaces between materials, or semi-transparent clouds. These features exhibit several types of behavior when examined in a time-series. We define three such behaviors: *regular*, *periodic*, and *random/hot spot*. Regular behavior is characterized by a feature that moves steadily through the volume: the structure of the feature (i.e. the data values corresponding to that feature) neither vary dramatically nor follow a periodic path. Features exhibiting periodic behavior possess one or both of the properties excluded by regular behaviors—periodic motion or structure variation. In both cases, the features of interest persist for a significant percentage of the time interval. Transient features of interest (i.e., those that exist for short periods) or features that fluctuate randomly fall into the third category of behavior. Our study tries to suggest techniques for all three behaviors.

Previous time-varying volume visualization research has focused on one of two topics: efficient rendering or efficient storage. Both exploit spatial and temporal coherence in order to either reduce display time or storage size. Several hierarchical data structures have been suggested for rendering purposes [4, 9, 11, 15, 17]. Images are generated by traversing these hierarchies to a specified spatial and temporal error tolerance value. Multi-resolution methods can also be used in compression. Westermann [16] utilizes wavelets to represent time data at various levels-of-detail. Non-hierarchical techniques [1, 14] use differencing and run-length encoding to accelerate rendering and decrease the data size. None of these works focuses on the need for creating transfer functions for time-varying data, assuming instead that the functions are provided. This work outlines several methods to derive such mappings.

3 Transfer Function Generation for Time-Invariant Data

Considerable research has looked into the generation of transfer functions for volume visualization. To generate color an opacity transfer functions, Fujishiro et al. [5] use topological information from a hyper-Reed graph. Kindlmann and Durkin [7] use information from the first- and second-order directional derivatives to generate a volume of derivative histograms for generating opacity maps emphasizing boundary surfaces. Bajaj et al. [2] discuss using other volume function information to find isovalues of interest; these functions include the volume enclosed by an isosurface, the isosurface area, and the isosurface gradient. Isovalues chosen by this method can be emphasized in a corresponding opacity map. For color maps, Bergman et al. [3] lay out procedural rules for informative color choices. Both He et al. [6] and Marks et al. [12] describe systems for color and opacity parameter generation. In [6], genetic algorithms breed trial transfer functions. The user can either select functions from generated images or allow the system to be fully automated. In the later case, they evaluate the images with statistical qualities such as their entropy and variance. The Design Galleries system [12] addresses parameter manipulation in general by rendering a multidimensional space of those parameters. The user then navigates this space to discover a parameter setting.

All of the techniques above, with the exception of the Contour Spectrum [2], focus on static volumetric data. The Contour Spectrum enables the user to display the changes of the underlying contour functions over time. Isovalues of interest can be chosen from different time and contour function values. The result is a set of opacity maps. The question is how these maps can be distilled into a single or minimal set of informative transfer functions for a time-varying data set.

4 Transfer Functions for Time-Varying Data

Transfer functions for time-varying data need to capture the three behaviors outlined in section 2. Ideally, these behaviors should be captured by a single transfer function. More than one opacity map can be misleading or physically meaningless—they suggest a sudden change in what is visible that can disorient the observer. For example, a data set displaying the motion of a single boundary surface through a medium requires only one transfer function to highlight the motion. However, multiple transfer functions may be required if several different types of features exist within the data, especially if they persist for different lengths of time. One transfer function per feature should be a sufficient upper bound. Even then, this number can be reduced by combining mappings which do not overlap in value. Visualizing the time series then becomes a task of choosing which features will be rendered and examined for a particular viewing.

The purpose of this research is to determine how to generate a transfer function(s) for a time-varying data set given only the volumes for each time step and a corresponding transfer function for each. These transfer functions were generated by some time-invariant generation technique for use in our experiments. No *a priori* transfer function generation technique is assumed, though for the purposes of testing an implementation of [7] was used. The findings of this work should be applicable to any current time-invariant transfer function generating method and for those yet discovered.

There are two classes of methods that can be used to generate transfer functions for time-varying data. The first class consists of algorithms which analyze each time-step separately, create a transfer function, and then try to combine these transfer functions into a summary function. The other class of techniques does not ignore the temporal di-

```
def findOpacity( start, end, threshold ):
    if start == end:
        return opacity[start], 0
    mean = the mean opacity in the range [start,end]
    std = the std. deviation in the range [start,end]
    cov = std / mean
    if cov ≤ threshold:
        return mean, cov
    else:
        mean1, cov1 =
            findOpacity( start, (start+end)/2, threshold )
        mean2, cov2 =
            findOpacity( (start+end)/2+1, end, threshold )
        if cov1 < cov2:
            return mean1, cov1
        else:
            return mean2, cov2
```

Fig. 1. The coherency-based transfer function algorithm

mension and operates upon the entire set of volumes to generate a transfer function. Our experiments fall into both categories. Our first set of experiments, the *summary-function based* tests, derive a single transfer function from the original transfer functions. This summary transfer function was then used to render the time series. Our second set of experiments, the *summary-volume based* tests, create a single volume summarizing the time series and generate a transfer function from this volume. This transfer function is then applied to the original volumes. The images rendered from both cases were compared with each other and the original images to determine which best captured the content of the volumes.

4.1 Summary-Function Experiments

Four methods were used to generate summary transfer functions for time-varying data from the original transfer functions: single representative, average, union, and coherency-based. The average and union methods are self-explanatory: in the first, the average of all the opacity maps is used as the transfer function; in the second, the union (or maximum opacity) of the opacity maps is used. The intuition for the first is that the average opacity map should represent the transfer function with the least deviation from any of the original transfer functions. The union operator was chosen to capture features that only exist over a small time interval—these features would be "smoothed-out" during averaging. These two methods can be seen as complementary.

The first method, single representative, models common behavior—a transfer function is created using a single time step in the series and applied to the others. This method thus serves as a test of current practice. For this test, the opacity map for the last time step was used for each data set; others could be chosen.

The final technique, the coherency-based method, is the most complex. It utilizes the coefficient of variation (COV) metric discussed in [4, 15]. The COV, the standard deviation divided by the mean of a sample, can be considered to be a normalized standard deviation. A large COV suggests that the opacity for the value varies rapidly over time (it is *incoherent*). The COV for a data value is:

$$c_v = \frac{\sigma_v}{\overline{o}_v}$$

where

$$\sigma_v = \sqrt{\frac{1}{n-1}\sum_t \left(o_{v,t} - \overline{o}_v\right)^2},$$

$$\overline{o}_v = \frac{1}{n}\sum_t o_{v,t},$$

$o_{v,t}$ is the opacity for data value v at time-step t, n is the total number of time steps (and thus opacity maps), $\overline{o}_v$ is the mean opacity for data value v, and σ_v is the standard deviation.

The coherency-based method uses the average of the opacity over a time interval if the COV for that interval falls under a specified threshold value. Opacity values that do not change significantly over time, and thus represent more coherent features, are favored over rapid fluctuations by this method. The opacity value for a given data value is determined by the algorithm described in Figure 1. In summary, the algorithm first calculates the COV for the entire time interval. If the COV is above the threshold, the interval is split and the child COVs' are calculated. The half with the lowest COV of its child intervals is used. This process is graphically depicted in Figure 2.

It is interesting to note that given an infinite threshold, the coherency-based method returns the average of the opacity maps. If the threshold is zero, one of the original opacity maps is returned. In our experiments, opacity maps with thresholds of varying percentages of the maximum COV were used.

4.2 Summary-Volume Experiments

Two techniques were used to create transfer functions from summary volumes: averaging and coherency. The first method forms a summary volume by averaging the values of a voxel over all time-steps. This again smoothes the details in the volume while capturing details that are persistent over time. The coherence-based technique mirrors that used for opacity maps where voxel values over time replace opacity values over time in the original analysis. Like the summary-function approach, coherency volumes were generated using different percentages of the maximum COV as the threshold.

The summary-volume methods must analyze the entire volume for each time-step. Thus, they are very time consuming to perform for very large data sets with many time steps. In contrast, all of the summary-function based experiments have low-cost: they operate directly upon the (much smaller) opacity maps. Excluding the necessary pre-processing step to generate the initial opacity maps from the function, the summary-function methods described here are have either a linear (for all but the coherency-based method) or quadratic running time in terms of the number of time steps.

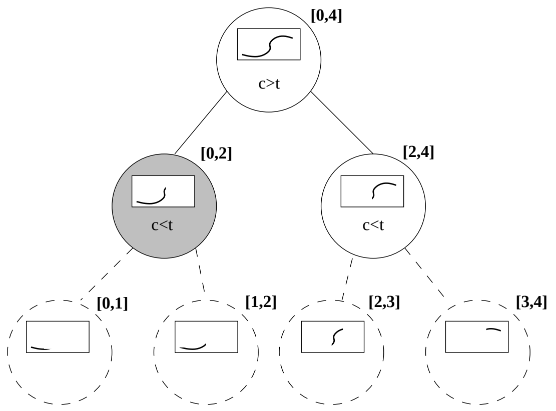


Fig. 2. An example of the coherency-based algorithm. Each circle represents a time-span interval of opacity values for a single data value. The algorithm starts at the top of the tree and works downward breadth-first until a node with the COV lower than the threshold is found. In this case, the interval $[0, 2]$ has a lower COV c than the interval $[2, 4]$ though both are below the threshold t. Thus, the average of the opacities in the first interval is used as the final opacity.

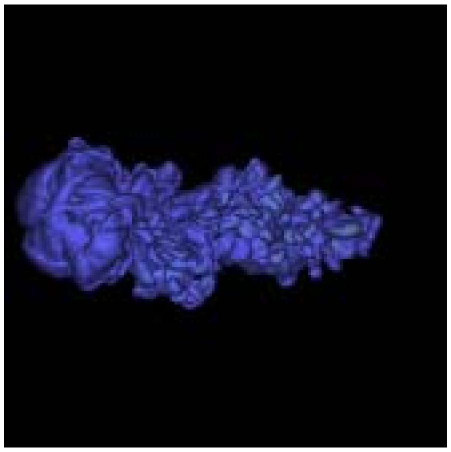
4.3 Test Data Sets

Four data sets were used to test the techniques. Figure 3 displays representative images from these data sets. The first data set models shock refraction and mixing. The data set shows the time evolution of an argon bubble after being disturbed by a shock wave. The bubble moves steadily from one side of the volume to the other while deforming. Only 90 time steps of this 640×256×256 voxels data set were used due to its large storage size. This data set is representative of a regularly behaved case: the argon-media boundary is essentially an isosurface and does not vary significantly in value over time.

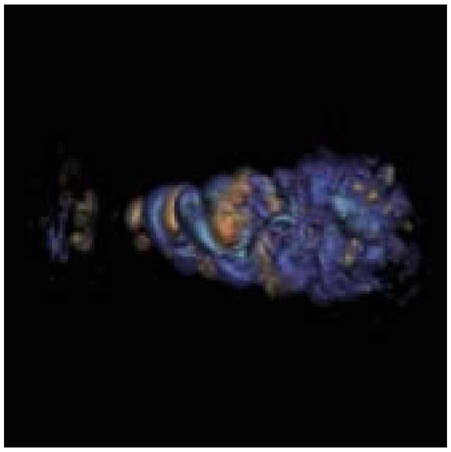
The next data set represents a turbulent jet flow simulation. It has 160 time steps (104×129×129 voxels each). The features of interest are the positive and negative vorticities of the jet. The vorticities are represented by the upper and lower halves of the data values, respectively. The vorticity surfaces rotate about the center of one axis of the volume. The surface boundaries are also fairly constant. Thus, the turbulent jet exhibits periodic motion with a regular boundary.

The third data set was obtained from a vortex flow simulation. It consists of 100 time steps (128×128×128 voxels each). "Tubes" within the vortex rotate while the boundaries of the tubes change over time. It thus exhibits periodic motion with non-regular boundaries.

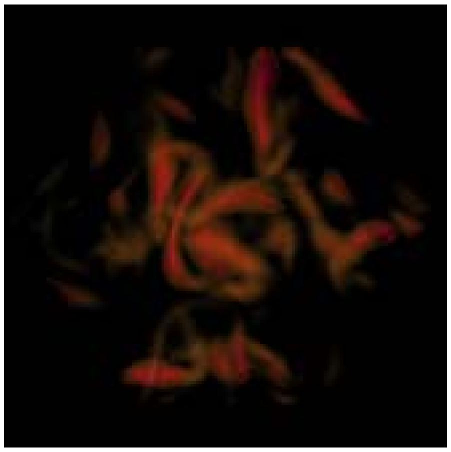
The final data set is a simulation of decaying quasi-geostrophic (QG) turbulence modeling atmospheric conditions. As the turbulence stabilizes, small "cells" form and begin to rotate in cyclone patterns and different locations. The data set exhibits a wide range of behavior as the simulation progresses. The 256×256×256 data set consisted of 935 time-steps, though only a subset was used for testing



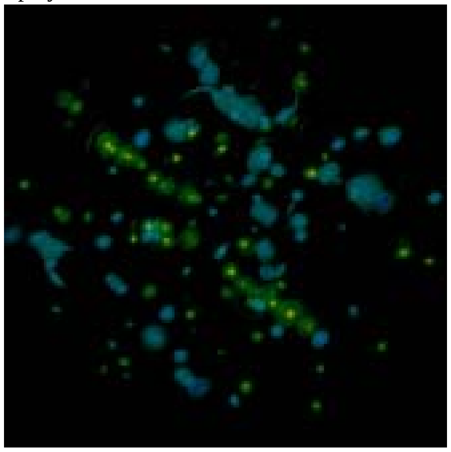
The argon bubble data set (640×256×256 voxels, 90 time-steps). The argon bubble is under motion from right-to-left.



The turbulent jet data set (104×129×129 voxels, 150 time-steps). Positive and negative vorticities in the flow are being displayed.



The vortex data set (128×128×128 voxels, 100 time-steps).



The decaying quasi-geostrophic (QG) turbulence simulation data set (256×256×256 voxels, 935 time-steps).

Fig. 3. Test data sets.

5 Results

To perform the various experiments, an opacity map was generated from each time step of the original volumes. These opacity maps were used by the summary-function experiments to derive their summary transfer function. Color maps were kept constant in order to limit the parameters of the study. For the summary-volume experiments, the original volumes were used. After each summary-volume was created, an opacity map was created using the same method use to generate the original transfer functions. Finally, for three time steps from each data set, test images were rendered using the original transfer function and the transfer functions created by the experiments. The mean difference of each test image from the original image was calculated for the three time steps. The differences were averaged to produce a measure of the dissimilarity between the test images and the originals. These values are recorded in Tables 1. Images from the tests can be found in Figures 7–9. These results are discussed below.

5.1 Single Transfer Function

Overall, the summary-function experiments images are more similar to the original images than the results from the summary-volume based techniques. The summary-volume methods are all based upon averages, and, for the smaller time-span data sets, this averaging does not preserve features of the data sets. The QG turbulence data set suggests that the volume coherency methods do perform better as more time steps are added. Since the COV is a statistical method, it is sensible that a larger sample size (i.e. more volumes) allows it to better represent the data.

Of the summary-function experiments, the single representative method produces images most like the original results, followed closely by the opacity coherency and averaging technique images. The union method are the most dissimilar. In data sets where the features of interest fluctuate (such as the vortex and QG turbulence data sets), the unioned opacity map blends all the boundaries from the time-steps, obscuring the result for any one time step (Figure 8c). The unioned opacity map is also sensitive to noise in the original transfer functions, as demonstrated by the fuzzy boundaries in Figure 7c.

Visually, all the summary-function experiments produce similar images for data sets with regular boundaries (the turbulent jet in Figure 7 and the argon bubble in Figure

Table 1. Mean image differences and variances of the test images from the originals for the summary-function experiments.

Data Set	Summary-Function Methods				Summary-Volume Methods	
	Single Rep.	Average	Union	Cohere 10%	Average	Cohere 10%
argon	2.6 ± 7.9	4.8 ± 12.9	16.3 ± 37.3	3.7 ± 12.5	11.8 ± 33.3	7.5 ± 19.8
jet	1.3 ± 3.8	1.5 ± 4.1	4.4 ± 11.0	1.6 ± 4.2	16.2 ± 23.7	7.8 ± 16.1
vortex	22.4 ± 27.9	30.9 ± 36.6	26.1 ± 37.0	26.0 ± 37.8	32.6 ± 40.8	23.7 ± 30.3
QG turbulence	3.3 ± 8.4	11.7 ± 27.9	4.6 ± 11.3	6.4 ± 17.2	80.0 ± 54.5	6.5 ± 16.6

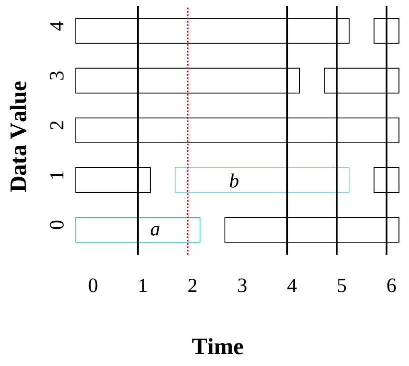


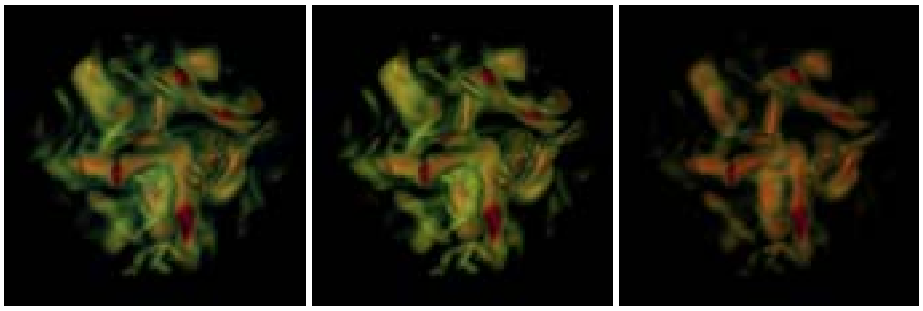
Fig. 4. An illustration of transfer function generation from the modified opacity coherency algorithm. Each block represents a span of time and a corresponding average opacity. The vertical lines are the midpoint of each interval. At each of these positions, a transfer function is generated from the intersected opacities.

6). Since the boundaries in these data sets are stable, the classification of each time-step volume is essentially the same, resulting in similar transfer functions. Data sets which exhibit non-regular boundaries produce more varied images when analyzed. For example, the images from the vortex data set tests (Figure 8) are very different from each other.

Though the motion of features does not affect the summary-function test results (since classification is by data value, not position), it significantly impacts the summary-volume based results. After the turbulence stabilizes in the QG turbulence data set, the range of motion of the features is small. This contributes to the low image difference for the volume coherence tests (Figure 9e). When summarized, volumes with periodic or regular motion can exhibit interesting effects. For example, since the range of rotation is constrained in the turbulent jet data set, this range is highlighted by the averaged volume (Figure 7e). In the argon bubble data set, where voxels change from one side of the argon boundary to the other as the simulation progresses, the averaged volume does not posses a coherent boundary; neither do images generated from its transfer function (Figure 6e).

5.2 Multiple Transfer Functions

Since using a single opacity map for all time-steps did not generate satisfactory images for the vortex data set, we experimented with a method to generate a set of transfer functions. We modified the opacity coherency algorithm in Figure 1 to return multiple intervals. During the traversal of time intervals in the algorithm, whenever an interval whose COV is less than the incoherency threshold is encountered, its mean opacity and the interval is returned. These intervals are gathered until a set of non-overlapping intervals span the entire time-series. For example, for the data value in Figure 2, the interval set would consist of the intervals $[0, 2]$ and $[2, 4]$, each with a corresponding



(a) Threshold 4.8 (b) Threshold 2.4 (c) Threshold 0.48

Fig. 5. A single time-step of the vortex data set rendered using three different transfer functions. The transfer functions were generated using consecutively lower thresholds in the modified opacity coherency algorithm.

mean opacity value. This process is repeated for each data value/element in the opacity map.

Given the interval set and mean opacity value for each data value, the actual transfer functions are generated as follows. The midpoint of each time interval over the entire range of data values is calculated; indices are rounded to the nearest integral index. A transfer function is created at each of these midpoint time indices. The opacities for this transfer functions are those whose time interval intersects the time step in question. Figure 4 illustrates this process. For example, the transfer function for time index 2 uses the mean opacity in interval a for data value 0 and the mean opacity of interval b for data value 1. These transfer functions are used to render the data set at their time index. For data sets between two generated transfer functions, the opacity value is linearly interpolated.

The method described is a multi-resolution transfer function generation scheme. As the incoherency threshold is decreased, the smaller and more numerous the time intervals created. These smaller intervals have opacities which are averaged over a smaller region of time. If the threshold is lowered enough, all the original transfer functions are re-created. The effect of changing the threshold is apparent in Figure 5. As the threshold changes, the emphasis on the boundary changes as well. The last image, with 10% of the maximum opacity incoherency as the threshold, almost reproduces the original image exactly. This threshold generated 63 opacity maps to cover the entire time-span. The other two images used one opacity map (the average) and 13 maps respectively.

This method can also be applied to data sets with stable boundaries. In Figure 10, a set of images highlighting the semi-transparent argon bubble boundary over three time steps is displayed. Since the boundary is regular, changing the threshold has less visible effect on the image than for the vortex data set.

6 Conclusions

Time-varying volume data displays behavior not present in static volume data. We have examined different methods to quickly generate transfer functions for these different types of volumes. From our study, it appears volumes which possess features with regular structure, regardless of their motion, can be adequately rendered with a single transfer function. Either the opacity coherency method we have introduced here or the single representative technique are recommended for these data sets. For data sets with dynamic boundaries, we have suggested a multi-resolution transfer function generation method. By changing the opacity incoherency threshold, the number of transfer functions created can be adjusted. The methods developed here can also be applied to the transfer functions generated by interactive renderers as an assisting tool. Using these renderers bypasses the expensive pre-processing needed to create the original maps. Instead, the transfer functions created by the interactive renderer are used as input to our methods. The final transfer function(s) can then help the user better understand their data.

6.1 Future Work

There are several directions for future work. The methods described here are only the first steps toward a robust transfer function generation system for time-varying volume data. For example, the static methods discussed in Section 3 could be extended to multi-time steps. These would provide better volume analysis techniques from those discussed here. Similarly, summary-volume methods looking at different statistical measures (min-max voxel values for example) may be interesting to investigate. It is also possible to consider the set of opacity maps over time as a surface with data value, time, and opacity as dimensions. This surface could then be simplified by various extant means to produce transfer functions cognizant of the time dimension. Transfer function creation algorithms which include the temporal-component in their analysis should be more effective than the single-time step methods we have examined.

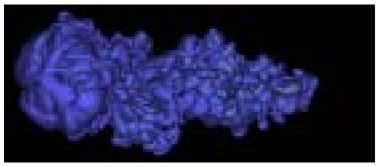
Complementary work deriving color maps for time-varying data should also be performed. These techniques could also be incorporated into a system to assist in transfer function generation during the actual creation of the data. While each time step is calculated by the solver, a set of suggested transfer functions can be calculated from the transfer functions generated previously. The non-coherency methods are ideal for this setting, as they can be incrementally calculated as the simulation progresses. Finally, an automated method to determine the characteristics of the data—whether features are regular, periodic, or random—would assist in both generating transfer functions and understanding the original data.

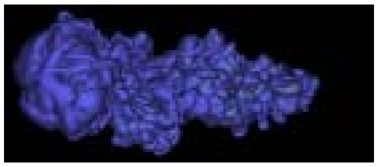
Acknowledgments

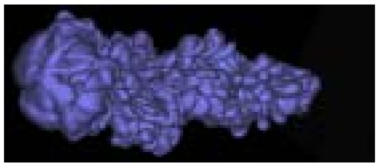
This work was supported by NASA Ames Research Center, the National Science Foundation, and Lawrence Livermore National Laboratory. The authors are grateful for those responsible for providing the data sets for our study. Finally, we thank the reviewers and the attendees of the Volume Graphics Workshop for their input.

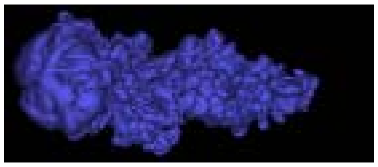
References

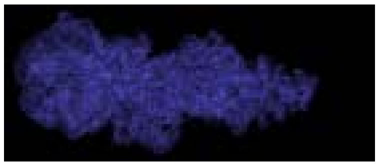
1. Dostas Anagnostou, Tim J. Atherton, and Andrew E. Waterfall. 4D volume rendering with the shear warp factorisation. In *Proceedings of the 2000 Symposium on Volume Visualization*, pages 129–137, 2000.
2. Chandrajit L. Bajaj, Valerio Pascucci, and Daniel R. Schikore. The contour spectrum. In *Proceedings of IEEE Visualization 1997*, pages 167–173. IEEE, 1997.
3. Lawrence D. Bergman, Bernice E. Rogowitz, and Lloyd A. Treinish. A rule-based tool for assiting colormap selection. In *Proceedings of IEEE Visualization 1995*, pages 118–125. IEEE, 1996.
4. David Ellsworth, Ling-Jen Chiang, and Han-Wei Shen. Accelerating time-varying hardware volume rendering using tsp trees and color-based error metrics. In *Proceeings of the 2000 Symposium on Volume Visualization*, pages 119–128, 2000.
5. Issei Fujishiro, Taeko Azuma, and Yuriko Takeshima. Automating transfer function design for comprehensible volume rendering based on 3d field topology analysis. In *Proceedings of IEEE Visualization 1999*, pages 467–470. IEEE, 1999.
6. Taosong He, Lichan Hon, Aire Kaufman, and Hanspeter Pfister. Generation of transfter functions with stocastic search techniques. In *Proceedings of IEEE Visualization 1996*, pages 227–234. IEEE, 1996.
7. Gordon Kindlmann and James W. Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In *Proceedings of the IEEE Symposium on Volume Visualization*, pages 79–86. IEEE, ACM SIGGRAPH, 1998.
8. Eric LaMar, Bernd Hamann, and Kenneth I. Joy. Multiresolution techniques for interactive texture-based volume visualization. In *Proceedings of IEEE Visualization 1999*, pages 355–362. IEEE, 1999.
9. David Laur and Pat Hanrahan. Hierarchical splatting: A progressive refinement algorithm for volume rendering. In *Proceedings of SIGGRAPH91*, pages 285–287, 1991.
10. Eric B. Lum, Kwan-Liu Ma, and John Clyne. Texture hardware assisted rendering of time-varying volume data. In *Proceedings of IEEE Visualization 2001*. IEEE, 2001.
11. Kwan-Liu Ma and Han-Wei Shen. Compression and accelerated rendering of time-varying volume data. In *Proceedings of the 2000 International Computer Symposium - Workshop on Computer Graphics and Virtual Reality*, pages 82–89, December 2000.
12. J. Marks, B. Andalman, P. A. Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, K. Ryall, J. Seims, and S. Shieber. Design galleries: A general approach to setting parameters for computer graphics and animation. In *Proceedings of SIGGRAPH97*, pages 389–400. ACM SIGGRAPH, August 1997.
13. Hanspeter Pfister, Jan Hardenbergh, Jim Knittel, Hugh Lauer, and Larry Seiler. The VolumePro real-time ray-casting system. In *Proceedings of SIGGRAPH99*, pages 251–260, 1999.
14. Han-Wai Shen and Chris Johnson. Differential volume rendering: A fast volume visualization technique for flow animation. In *Proceedings of IEEE Visualization 1994*, pages 180–187, 1994.
15. Han-Wei Shen, Ling-Jen Chaing, and Kwan-Liu Ma. A fast volume rendering algorithm for time-varying fields using a time-space partitioning (TSP) tree. In *Proceedings of IEEE Visualization 1999*, pages 371–377. IEEE, 1999.
16. Rüdiger Westermann. Compression domain rendering of time-resolved volume data. In *Proceedings of IEEE Visualization 1995*, pages 168–175, 1995.
17. Jane Wilhelms and Allen van Gelder. Multi-dimensional tree for controlled volume rendering and compression. In *Proceedings of the 1994 Symposium on Volume Visualization*, pages 27–34, 1994.


(a) Single Representative


(b) Opacity Average


(c) Opacity Union


(d) Opacity Coherence 10%


(e) Summary-Volume Average

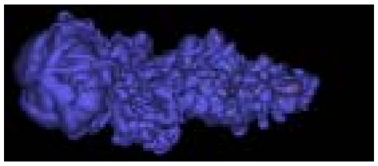
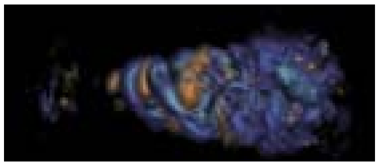
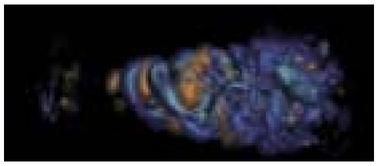
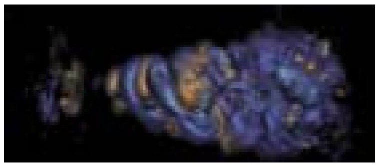
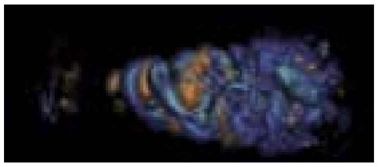

(f) Summary-Volume Coherence 10%

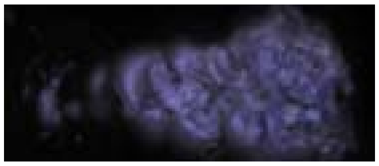
Fig. 6. Results for argon bubble data set.


(a) Single Representative


(b) Opacity Average


(c) Opacity Union


(d) Opacity Coherence 10%


(e) Summary-Volume Average

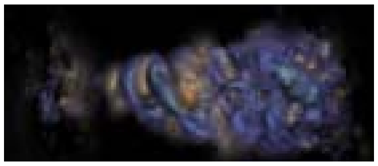
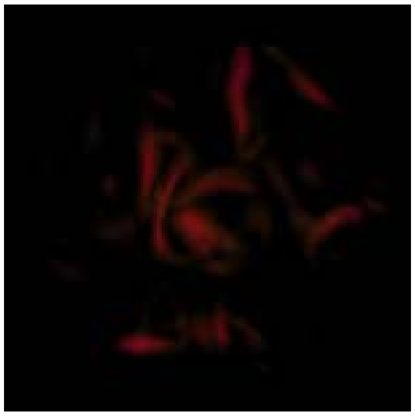
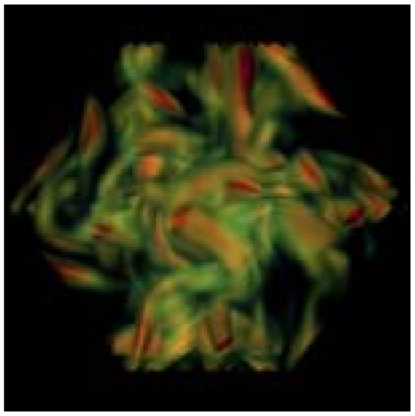

(f) Summary-Volume Coherence 10%

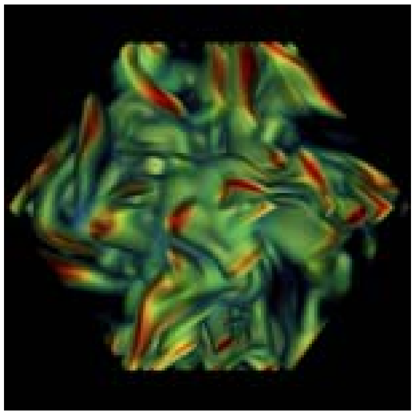
Fig. 7. Results for turbulent jet data set.



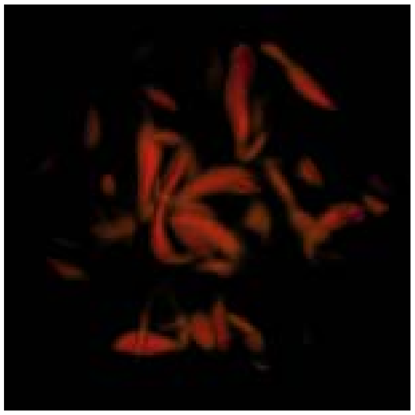
(a) Single Representative



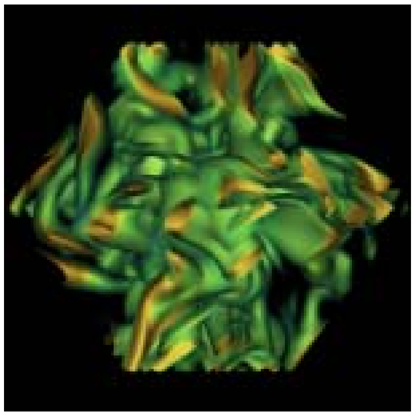
(b) Opacity Average



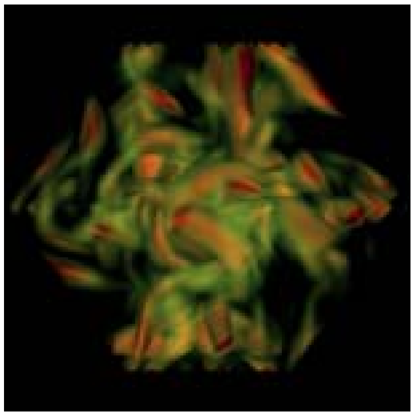
(c) Opacity Union



(d) Opacity Coherence 10%

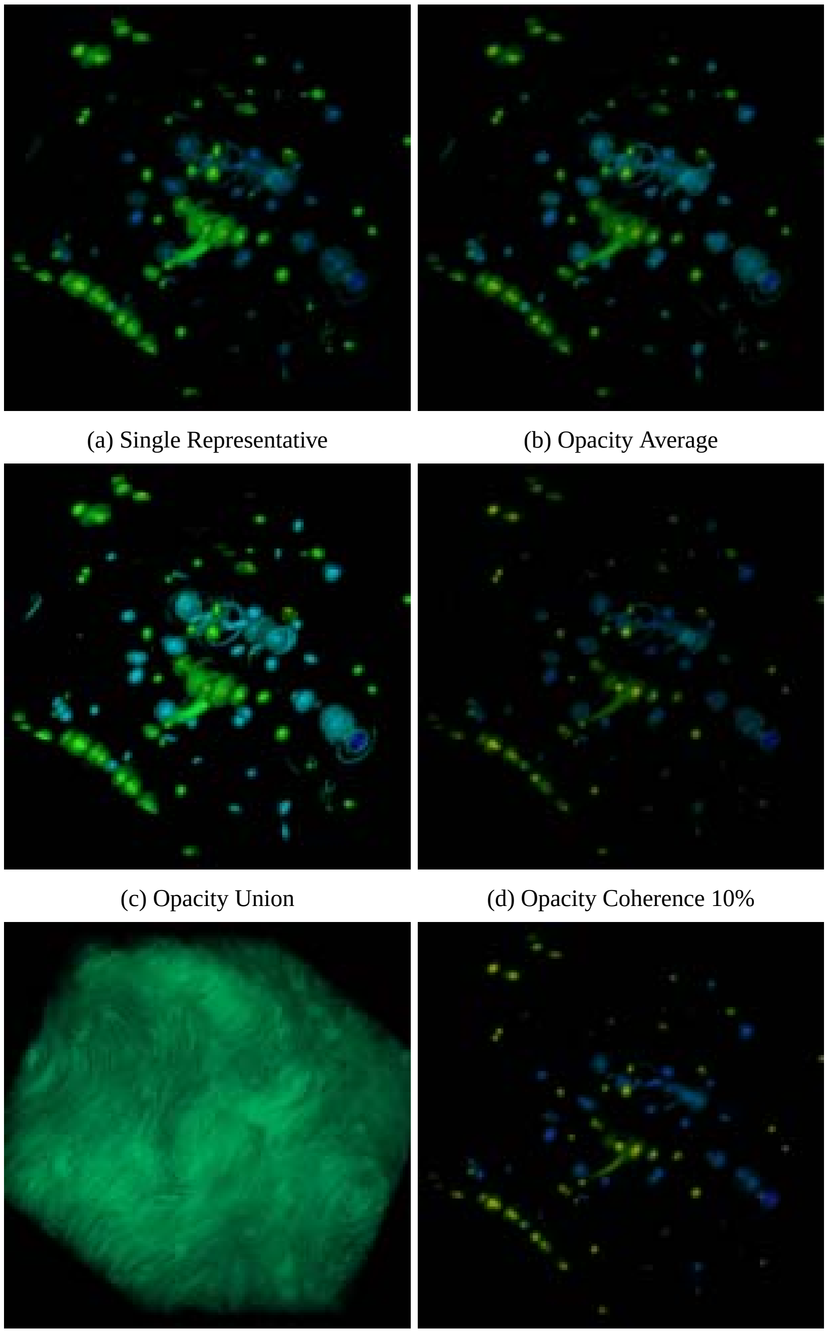


(e) Summary-Volume Average



(f) Summary-Volume Coherence 10%

Fig. 8. Results for vortex data set.



(a) Single Representative

(b) Opacity Average

(c) Opacity Union

(d) Opacity Coherence 10%

(e) Summary-Volume Average

(f) Summary-Volume Coherence 10%

Fig. 9. Results for weather data set.

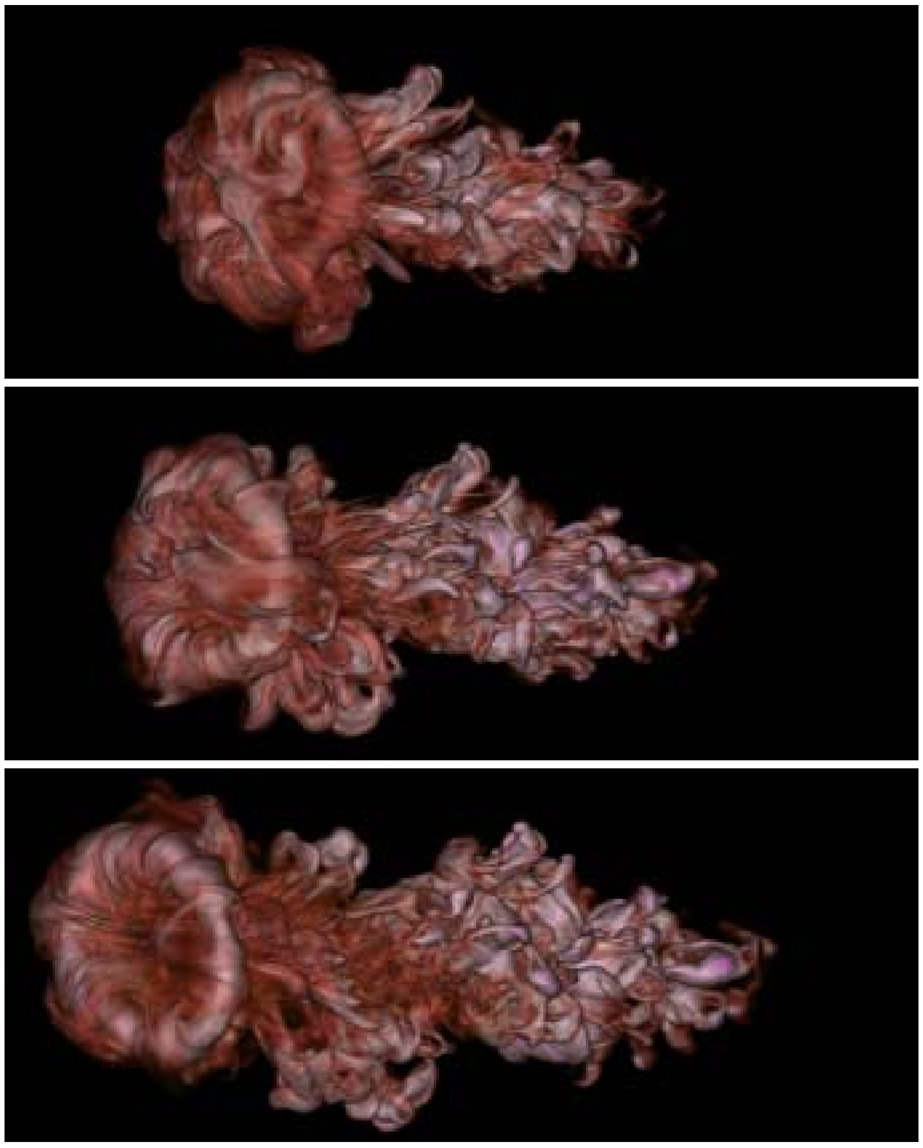


Fig. 10. Our method applied to the semi-transparent boundary transfer functions of three time-steps in the argon bubble data set.

[Color Plate: Jankun-Kelly and Ma, "A Study of Transfer Function Generation for Time-Varying Volume Data"]