HDR VolVis: High Dynamic Range Volume Visualization

Xiaoru Yuan, *Student Member*, *IEEE*, Minh X. Nguyen,
Baoquan Chen, *Senior Member*, *IEEE*, and David H. Porter

Abstract—In this paper, we present an interactive high dynamic range volume visualization framework (HDR VolVis) for visualizing volumetric data with both high spatial and intensity resolutions. Volumes with high dynamic range values require high precision computing during the rendering process to preserve data precision. Furthermore, it is desirable to render high resolution volumes with low opacity values to reveal detailed internal structures, which also requires high precision compositing. High precision rendering will result in a high precision intermediate image (also known as high dynamic range image). Simply rounding up pixel values to regular display scales will result in loss of computed details. Our method performs high precision compositing followed by dynamic tone mapping to preserve details on regular display devices. Rendering high precision volume data requires corresponding resolution in the transfer function. To assist the users in designing a high resolution transfer function on a limited resolution display device, we propose a novel transfer function specification interface with nonlinear magnification of the density range and logarithmic scaling of the color/opacity range. By leveraging modern commodity graphics hardware, multiresolution rendering techniques and out-of-core acceleration, our system can effectively produce an interactive visualization of large volume data, such as $2,048^3$.

Index Terms—Volume visualization, high dynamic range, user interfaces, transfer function design, nonlinear magnification.

1 Introduction

Flow structures in fully developed 3D fluid turbulence span wide ranges of scale, both in space and in time. Volume rendering has been utilized to diagnose such fluid turbulence. We study large dynamical fluid simulations performed on supercomputer systems with both high spatial resolution and high precision values. For example, one typical data set we deal with is from a simulation of Mach 1 homogeneous compressible turbulence [27], [28] using the Piecewise-Parabolic Method (PPM) gas dynamics code [42], [44] on a grid of $2,048^3$ cells. The simulation was run on the TeraGrid cluster at the National Center for Supercomputing Applications (NCSA) using a number of dual-CPU machines that varied between 80 and 250 nodes over the course of a two-month run [43].

Such large turbulent flow data sets impose several issues when a high fidelity visualization using commodity graphics hardware is desired. First, turbulent flow structures are known to be both spatially and temporally intermittent, in the sense that fluctuations in flow quantities such as entropy and velocity are not uniformly distributed in space or time, but tend to be clustered. The level of this nonuniformity increases as the range of scales spanned by the turbulence increases. The large range of spatial scales and extreme nonuniformity of fluctuations lead to wide dynamic ranges in the values of flow quantities. The turbulent flow data is normally stored in 16-bit or 32-bit floating-point precision. Quantizing it to 8-bit scalar values to fit the texture-based volume rendering on currently available commodity graphics hardware will severely degrade the data quality and will inevitably result in loss of subtle, but important details. Second, to better reveal the internal flow structures, it is desirable to set the opacity of each slice small enough to minimize occlusion. Since the number of sampling slices must be comparable with the spatial resolution of the volume data to ensure sufficient volume sampling, large volume data requires very low opacity assignment for each slice; consequently, the opacity value will be rounded to zero in a low precision rendering system. Therefore, it is desirable to have high precision alpha compositing so that detailed structures residing in the data can be preserved.

For rendering such volume data with both high spatial and intensity resolutions, we present a novel high dynamic range volume visualization (HDR VolVis) framework [45]. Our method performs high precision volume compositing followed by dynamic tone mapping to preserve details on regular display devices. In the following, we summarize different stages of our *High Range (HDR) Volume Visualization* pipeline as illustrated in Fig. 1.

High Dynamic Range Input Data: The turbulent flow simulation data sets have both high spatial and intensity resolutions.

High Dynamic Range Transfer Function: The transfer function should have a comparable number of entries to the data intensity levels to handle high precision data sets. Displaying and editing such high resolution transfer functions on regular displays presents a challenge. We present a novel transfer function specification interface with nonlinear magnification of intensity range and logarithmic

• X. Yuan, M.X. Nguyen, and B. Chen are with the Department of Computer Science and Engineering and Digital Technology Center, University of Minnesota at Twin Cities, Minneapolis, MN 55455.
E-mail: {xyuan, mnguyen, baoquan}@cs.umn.edu.
• D.H. Porter is with the Laboratory for Computational Science and Engineering and Digital Technology Center, University of Minnesota at Twin Cities, Minneapolis, MN 55455. E-mail: dhp@lcse.umn.edu.

Manuscript received 12 Nov. 2005; revised 14 Jan. 2006; accepted 18 Jan. 2006; published online 10 May 2006.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-0161-1105.

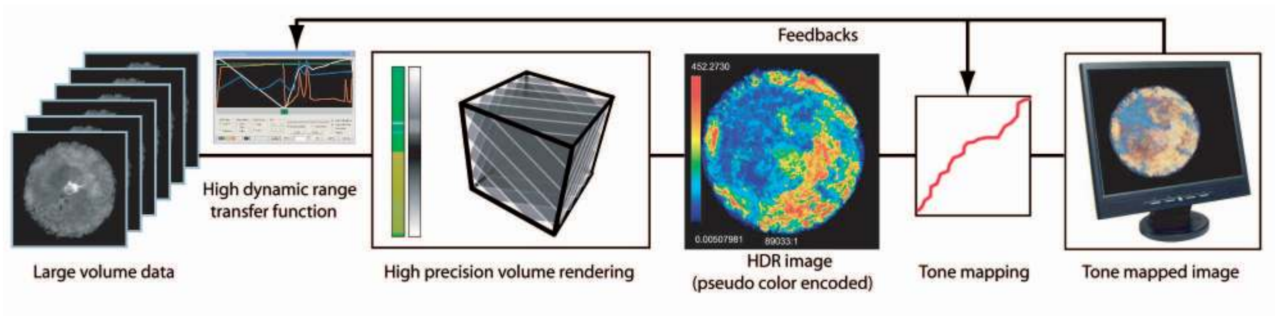
Feedbacks
High dynamic range transfer function
Large volume data
High precision volume rendering
452.2730
0.00507981
89033:1
HDR image (pseudo color encoded)
Tone mapping
Tone mapped image

Fig. 1. Pipeline of HDR VolVis. The input is a scalar volume with high precision and/or high resolution (e.g., $2,048^3$). The intermediate output is a high dynamic range image after high precision compositing. By applying a tone mapping operator, the final result can be displayed on a regular low dynamic range display device.

scaling of color/opacity range to facilitate design of high dynamic range transfer functions.

High Dynamic Range Volume Compositing: We perform compositing at high precision and leverage commodity graphics hardware for this purpose to retain volume details during rendering. The volume rendering output is in a high dynamic range image format.

High Dynamic Range Image Output: Simply rounding the pixel values of an intermediate image containing high precision values to regular display scales would result in loss of computed details. It is preferable to be able to retain as much of the computed high precision (i.e., high dynamic range) in the visualization as possible and provide dynamic color intensity mapping (i.e., tone mapping or tone reproduction) for regular displays or printing media. With input from a user, our framework can automatically adjust the tone reproduction for the final display to enhance selected features. We extend our tone mapping selections to other available operators and gain insights into better choices of tone mapping methods for visualization purposes. Our framework is also able to directly output the rendering results to high dynamic range video which can be played by HDR viewing software available in the public domain.

By leveraging modern commodity graphics hardware and out-of-core acceleration, we produce interactive visualization of volume data as large as $2,048^3$.

In this paper, we first briefly review related work in Section 2. We then discuss the three main technical issues involved in high dynamic range volume visualization: high precision compositing (Section 3), tone reproduction (Section 4) and transfer function design for high dynamic range volume rendering (Section 5). Although the transfer function design takes place first during the HDR volume rendering, it is discussed the last for clarity. Implementation details are described in Section 6 followed by results and discussions (Section 7). Finally we present our conclusions in Section 8.

2 Related Work

In the real-world, scenes with a wide range of colors and intensities are pervasive. Dynamic range here is defined as the ratio between the maximum and the minimum nonzero tonal values in an image or scene. Algorithms have been developed for capturing both photographs [3] and videos [15] with high dynamic range of over 10^5. The resulting image or video is stored in a floating-point format or in special coding schemes such as RGBE/XYZE, OpenEXR, LogLuv, and so on [22], [33], [21]. On the display or storage side, 8-bits-per-channel image representation is popular. The dynamic range of most available display devices is no more than two orders of magnitude. Paper or other printing media have much more limited dynamic range. Tone mapping operators [7], [8], [19], [25], [32], [38], [39] have been developed to bridge the gap between high dynamic images and low dynamic range display devices. In general, two types of tone reproduction operators have been proposed: global (spatially uniform) operators and local (spatially varying) operators [4]. Global operators apply the same function to every pixel throughout an image. One global operator may depend upon the contents of the image as a whole, so long as the same transformation is applied to every pixel. Conversely, local operators apply a different scale to different parts of an image. With the development of commodity graphics hardware, many tone mapping algorithms can be accelerated on graphics hardware [12]. Recently, a high dynamic range display device has been developed [35] based on a combination of an LCD (Liquid Crystal Display) panel and a DLP (Digital Light Processing) projector.

The benefits of rendering high dynamic range output, especially its applications to visualization, have not been seriously explored. For direct volume rendering, each voxel contributes to the final image by an alpha compositing operation. For large volume data with a highly uneven distribution of intensity, rendered images are likely to have very different brightnesses between highlights and dark regions. High dynamic range rendering techniques are necessary to preserve all rendering details. Only very recently, Ghosh et al. [11] investigated the use of high dynamic range display technology for volume rendering. Their emphasis is on mapping of the transfer function to a perceptually linear space over the range of intensities that a high dynamic range image display can produce. The goal is to reserve several just noticeable difference (JND) steps of intensities for spatial context apart from clearly depicting the main regions of interest. In this paper, we systematically investigate a new approach (HDR VolVis), focussing on visualizing large volumetric data sets with both high spatial resolution and high precision generated from large scale computer simulations. We also target popularly available 8-bit display devices.

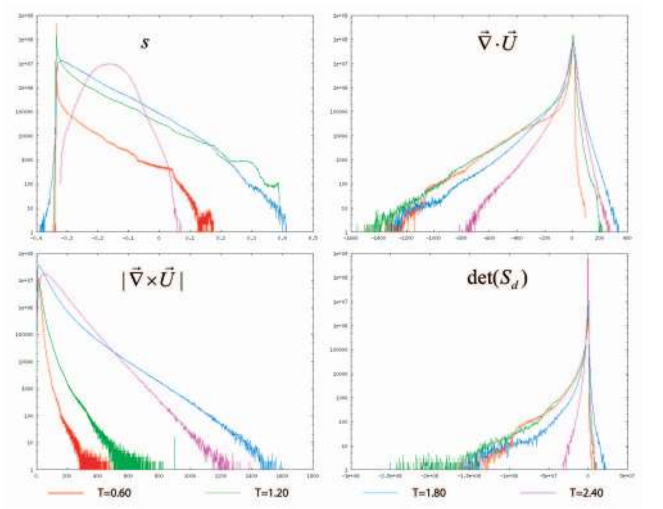

Fig. 2. Histogram of four flow quantities of a numerical turbulence simulation: entropy s; divergence of velocity $\vec{\nabla}\cdot\vec{U}$; vorticity magnitude $|\vec{\nabla}\times\vec{U}|$; determinant of the deviatoric part of the symmetric strain rate tensor $det(S_d)$. Four time steps are shown in each plot. Unit of time is defined as sound crossing time of energy containing scale.

3 High Precision Compositing

In this section, we discuss the first issue pertaining to high dynamic range volume visualization and display: high precision compositing.

First, we examine how several commonly used flow quantities stress the need for extended dynamic range in different ways. A numerical simulation of decaying, compressible, and homogeneous turbulence provides examples of flow structures (see Fig. 11), which are typical of fluid turbulence in general. The fluid modeled is dry air, which is approximated as an ideal gas. The initial state of this flow is constant density and pressure, with random and uncorrelated sinusoidal velocity perturbations in a periodic cubical volume. The amplitude of the initial power spectrum of velocity fluctuations is chosen so that the initial RMS Mach number is unity, leading to a significantly compressible flow, including shock waves. The numerical simulation used here was performed on a computational mesh of $1,000^3$. Here, we focus on four flow quantities, which have proven useful in diagnosing compressible turbulent flows. These four quantities are:

- the entropy, s,
- the divergence of velocity, $\vec{\nabla}\cdot\vec{U}$,
- the vorticity magnitude, $|\vec{\nabla}\times\vec{U}|$, and
- the determinant of the deviatoric part of the symmetric strain rate tensor, $det(S_d)$.

As shown in Fig. 2, the values of these four flow diagnostics have widely different probability distribution functions (PDF), which stress the need for high dynamic range in different ways.

The entropy is a thermodynamic quantity, very much like a temperature for compressible flows. The entropy of an element of gas only changes in response to dissipative processes. The initial state of this simulation is constant pressure and density. Hence, the initial entropy is constant. As the compressible flow develops, shock waves form and dissipate kinetic energy into heat, thereby increasing the entropy in isolated regions. Once the turbulence is fully developed, flow structures on the smallest scales also dissipate energy into heat. Since the initial entropy in this flow is constant and isolated patches of high entropy are generated by shocks, a wide range of variations from the initial entropy can develop from the turbulent mixing of the high entropy regions into the surrounding gas.

The divergence of velocity is a measure of the compressibility of the flow. Visualizations of $\vec{\nabla}\cdot\vec{U}$ show sound and shock waves. Since it depends linearly on spatial derivatives, and shock waves tend to concentrate velocity jumps on the smallest scales available, the PDF of $\vec{\nabla}\cdot\vec{U}$ increases linearly with the range of scales available, which corresponds to the mesh resolution in the case of a numerical simulation. In this model, where the linear spatial resolution of the mesh is 1,000, tails of the distribution extend from values of about $-1,600$ to $+300$ at time 0.3 (measured in flow times of the energy containing range). The PDF of the velocity divergence is seen to have extended exponential tails, especially for negative values corresponding to shock waves. The dynamic range of the divergence of velocity realistically spans a factor of 1,000 at time 0.3 in this flow. At later times the spread in the distribution of the divergence of velocity is not as large, since shock waves weaken with time.

Vorticity plays an important part in the development of turbulence. Vorticity magnitude measures shear in the velocity field. Vortex tubes are so predominant in typical 3D turbulent flows that their tube-like side effects are readily seen in nature, from which turbulence derives its name. Rendering large values of vorticity depicts both slip surfaces at early times in this flow and vortex tubes at late times. The dynamic range of vorticity here is not as high as the divergence of velocity. However, peak vorticity is organized in vortex tubes, which tend to be only a few zones thick. Furthermore, when the turbulence is fully developed, these vortex tubes do not occur in isolation, but are typically present in extremely complicated tangles or knots of turbulence, which can be hundreds of vortex tubes across. A volume visualization of the 3D structure on these vortex tube tangles would require rendering each tube with an optical depth of about 1/100. Hence, rendering with more than 8-bit deep color channels would be beneficial.

Finally, the determinant of the deviatoric part of the symmetric rate of strain tensor, $det(S_d)$, is a measure of how the flow changes the shape of fluid elements. Extreme values of $det(S_d)$ tend to scale as the cube of the range of spatial scales or the cube of the mesh resolution in numerical simulations. The resulting PDFs span a very wide range of values, both positive and negative, compared to typical fluctuations. At 0.3 flow times after the initial state, values of $det(S_d)$ range between extremes of -1.1×10^8 to $+1.4\times10^7$, while 96 percent of the values are in the range $[-10^4,+10^4]$. Hence, conservatively $det(S_d)$ has a dynamic range $>10^4$, and visualizations can definitely benefit from a high dynamic range treatment. In volume rendering, this means that the high dynamic range flow quantities must be mapped to high dynamic range alpha and color values during transfer function lookup, which further means high precision compositing.

Even without considering the dynamic range of the volume data, the nature of high resolution volume data sets requires high precision alpha compositing. In volume rendering, each sample is assigned an alpha value and alpha blending is performed during compositing on the rendered images in a back-to-front order. In this way, every sample contributes to the final rendering. On the other hand, the alpha blending exponentially decreases each sample's contribution. Assuming back-to-front compositing, the color contribution of the previous composited samples are modulated by a factor of $1 - \alpha(x_i)$, where $\alpha(x_i)$ is the corresponding opacity value of current sample x_i. As an example, let us look at a volume data of n slices. In the following discussion, we assume our sampling rate is the same as the data frequency of the volume data, e.g., one sample per voxel. Assuming that every voxel is assigned with the same opacity value α, the relative contribution of the furthest voxel to the rendered image will be decreased by a factor of $(1 - \alpha)^n$. To guarantee the final contribution to be no less than r, the α value must be set as:

$$\alpha = 1 - e^{log(r)/n}. \tag{1}$$

For data sets with a size of $2,048^3$, when r is 0.1, the α has to be less than 1.12^{-3}; when r is 0.01, α has to be smaller than 2.25^{-3}. In such cases, a rendering system with only 8 bits per channel will quantize the alpha value to zero and voxel regions with such low alpha values will not contribute to the rendered image. Furthermore, to achieve higher rendering quality, the number of slices (sampling rate) needs to be much higher than the size of volume (at least twice according to the Nyquist rate), hence the preferred α value would even be smaller. For high dynamic range volume rendering, samples of such low α values must still contribute to the final image. This requires high precision for the alpha values. As an example, for 16-bit floating-point representation, the highest precision is 2^{-24} [24]. This makes it possible to render volumes of $2,048^3$ resolution.

In addition to the transfer function, to be discussed in Section 5, we define a global parameter α_g for adjusting the global opacity of the whole volume. This α_g value will multiply with the opacity returned by the transfer function to get each sample's final opacity for compositing. As an example, the α_g value is set to 0.5, 0.05, and 0.005 in Figs. 3a, 3b, and 3c, respectively. The opacity value in the base transfer function ranges from 0 to 1.0 (the same transfer function is used for all three images). All the figures are generated using high precision computations (16 bits) and, hence, are rendered using 16 bits per channel. The dynamic ranges are $3.32 \times 10^4 : 1,\ 1.19 \times 10^5 : 1,\ 2.76 \times 10^4 : 1$ for Figs. 3a, 3b, and 3c, respectively. The images are linearly mapped to 8 bits for display. Due to the high opacity value used, Fig. 3a shows an isosurface like rendering effect. The internal structures are occluded. Fig. 3b, in which the opacity values are reduced 10 times, illustrates more detailed structures. If alpha values are decreased further, a much darker image (Fig. 3c) is obtained compared to the other two images as the lowest alpha value is used. Not shown in the figure, 8-bits-per-channel rendering with the

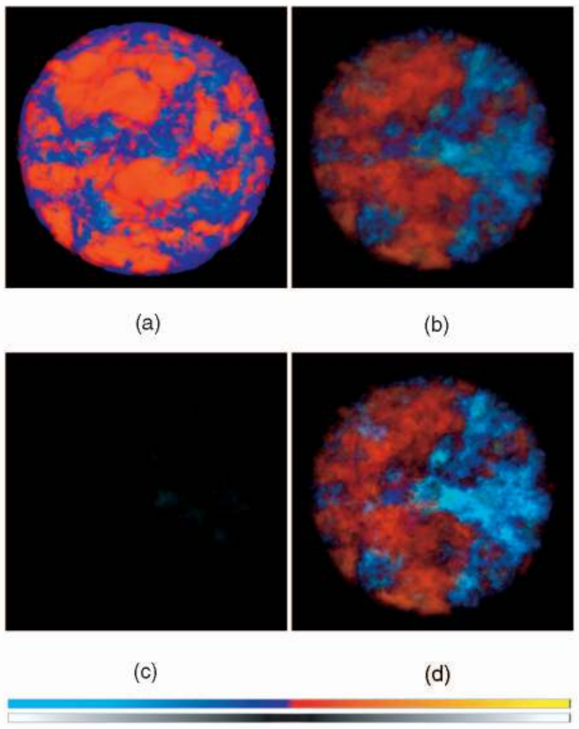

Fig. 3. High dynamic range volume rendering using different global alpha α_g: (a) α_g = 0.5, (b) α_g = 0.05, and (c) α_g = 0.005. All the images are generated using high precision computation (16 bits), but are linearly mapped to 8 bits for display. (d) Tone reproduction of (c). The corresponding transfer function is shown as the color bars (top: color, bottom: alpha).

same transfer function and α_g value as that used in Fig. 3c produces an almost black image. This is because when only 8-bits precision compositing is performed, the small α_g value will lead to zero alpha values for almost all the samples. Such quantization artifacts are also illustrated in Fig. 9. Fine details in Fig. 3c, however, can be appropriately displayed by using nonlinear tone mapping, to be discussed next, instead of linear mapping. The color bars show the transfer function applied during rendering (top: color specification; bottom: alpha values, dark regions indicates low opacity value). Similar transfer functions are also included in subsequent figures.

4 Tone Reproduction for High Dynamic Range Volume Rendering

The dynamic range of an output image from the high precision volume rendering is much higher than the dynamic range of a regular display device which is usually only several hundred levels. Fig. 4a shows an HDR image generated from our HDR volume visualization. To illustrate the HDR image on regular display devices or printing media, the HDR intensity in this image is pseudocolor encoded. One way to display such HDR images is to display subdynamic ranges of the original HDR image. This is analogous to the photography practice of using different exposures to capture a highly complex scene. Fig. 4b shows nine mapped images with increasing exposure levels. This demonstrates the rich information encoded in HDR visualization. However, this

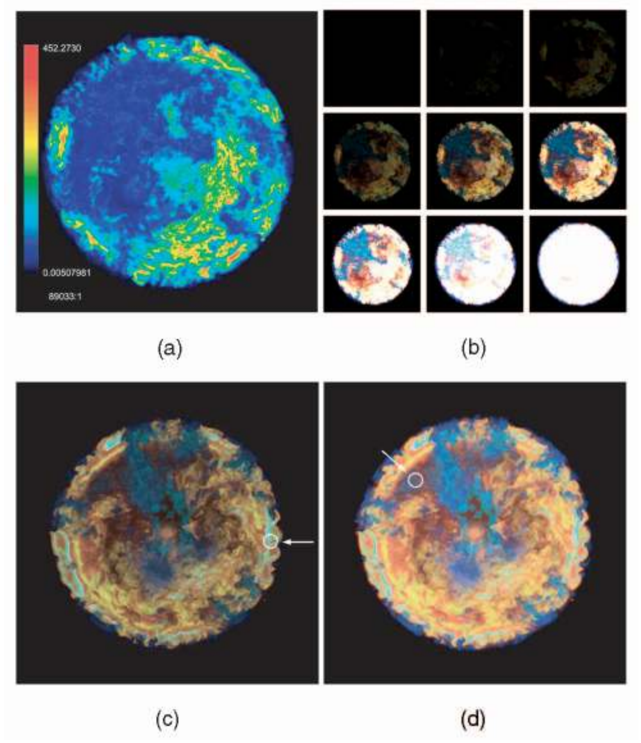

Fig. 4. High dynamic range image tone mapping. (a) The pseudocolor encoded intensity. (b) Images at different exposure levels. (c) and (d) Adaptive logarithmic tone mapping based on center selection. (c) Initial tone mapped image based on a user specified region of interest emphasizing highlight parts (white circle). (d) Tone mapping based on another user specified region of interest emphasizing parts with low illumination.

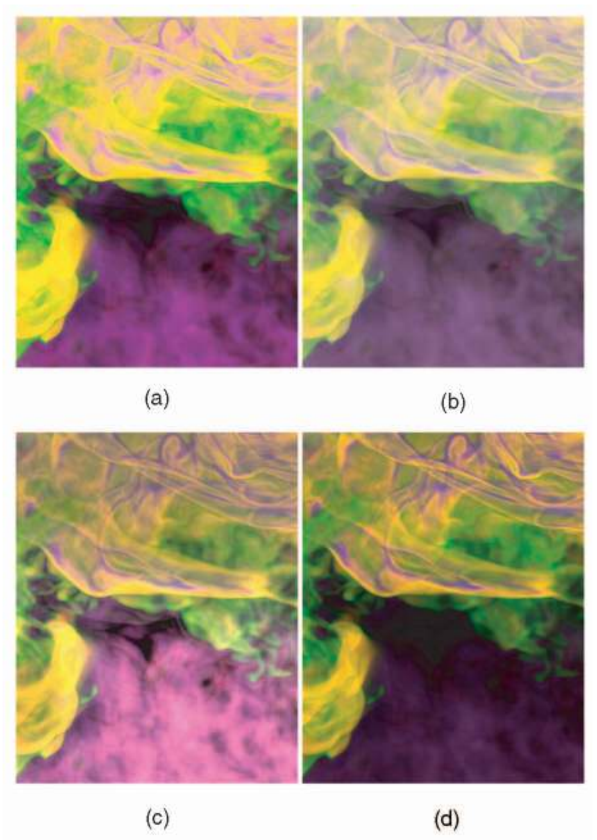

Fig. 5. Different tone mapping operators on HDR volume rendered results. (a) Drago et al.'s adaptive logarithmic mapping [6]. (b) Reihard et al.'s photographic tone reproduction method [32]. (c) Fattal et al.'s gradient domain high dynamic range compression [8]. (d) Durand and Dorsey's fast bilateral filtering [7].

method requires multiple images to display the full dynamic range of an HDR image. The rendered high dynamic range images could also leverage the recently developed HDR displays [35]. However, such devices are very expensive and are far from being commodity. In fact, even an HDR display device has fixed dynamic range when an HDR image has a higher dynamic range than that of the HDR display, tone mapping is still necessary for displaying. We seek to accommodate popularly available standard display devices.

We employ previously developed tone mapping techniques. Our system dynamically maps the volume rendered HDR images to regular images of 8 bits per color channel so that they can be displayed on regular 8-bit display devices. Fig. 4c and Fig. 4d show the regular 8-bits-per-channel images after tone mapping.

The goal of tone mapping or tone reproduction here is to compress the dynamic range and simultaneously maintain the contrast of the input high dynamic range image. The first tone mapping algorithm that we use belongs to the category of global operators. We choose this algorithm because it works at interactive speeds. Local tone mapping operators [7], [8], [32], which are relatively computationally expensive can achieve better compression results. The volume rendered images generated by our system can also be saved in HDR format images (e.g. Ward's RGBE format [40]) or in HDR movies, which can be viewed offline using HDR image viewers [13] or HDR movie players [6] available in public domain. For such offline viewing, other state-of-the-art local tone mapping methods can be applied. We choose the adaptive logarithmic mapping operator developed by Drago et al. [6]. This method is based on logarithmic compression of luminance values. A bias power function is introduced to adaptively vary logarithmic bases, resulting in good preservation of details and contrast. We integrate this tone mapping algorithm with our high dynamic range volume rendering. Fig. 4c and Fig. 4d shows how even a static HDR visualization allows users to interact with it and retrieve details from selected regions of interest. Fig. 4c shows the tone reproduction based on a user specified region of interest. Compared with Fig. 4d, which is based on a different user specified region, it can be observed that details and contrasts are retained the best in the regions of interest.

We also have conducted a preliminary experiment on several other tone mapping methods to display our HDR volume rendered images. As shown in Fig. 5, the tone mapping methods include: adaptive logarithmic mapping [6], photographic tone reproduction [32], gradient domain high dynamic range compression [8] and fast bilateral filtering [7]. All methods, except the first one, are local operators. In general, the global operators are computationally less expensive since all pixel values have the same operations resulting in efficient GPU implementation. Local tone mapping operators give results with higher local contrast since these methods tend to maintain local intensity contrast as much as possible. For our test image, gradient domain high dynamic range compression (Fig. 5c) and fast bilateral filtering (Fig. 5d) give better contrast in bright regions compared with Fig. 5a and Fig. 5b. For dark regions, Fig. 5d loses details. Although it is very difficult to compare different tone mapping methods having different goals and emphasis, we find that, based on the limited experiments we have performed, the gradient domain method gives higher contrast in both bright and dark regions (Fig. 5c). It is more suitable for visualization purposes because of the observed better detail preservation. Additional tone mapping operators and a full user study on their effectiveness will help us understand their specific applicability to HDR VolVis better.

Noted that most of the current tone mapping methods work for photography or photorealistic rendering. Such methods focus on retaining perceptual fidelity. Until now, no tone mapping evaluation methods have been developed for visualization purposes. It is, therefore, necessary to develop algorithms to enable maximum detail conservation instead of perceptual fidelity, and corresponding evaluation methods. Research toward evaluating tone mapping operators [5] has been scarce. Ledda et al. [20] have used a high dynamic range display to evaluate existing tone mapping operators. It is possible to evaluate the effectiveness of tone mapping methods on volume rendered images with the assistance of a high dynamic range display. Compared to Ledda et al.'s method which is more focused on psychophysical validation, evaluation for visualization tasks should focus more on the preservation of information detail.

After the tone mapping operation, the original pixel intensities are altered to comply with the dynamic range of the target display device. When the tone mapped image is printed, it is desirable to know each pixel's original high dynamic range value. For global tone mapping methods, the operators are one-to-one mapping functions. Given the mapping parameters and operators, the pixel values in a tone mapped image can be mapped back to the original high dynamic range albeit, with limited precision. For local tone mapping operators, the same pixel values in the HDR format could be mapped to different levels in the low dynamic range output. Pixels with the same level in the low dynamic range may have different values in their corresponding HDR image. For such operators, the inverse process is not well-defined.

5 Transfer Function Design for High Dynamic Range Volume Rendering

A transfer function is an essential part in the volume visualization pipeline, which maps optical properties, such as color and opacity, to intensity values of the volume data sets being visualized. In practice, a transfer function is encoded in discrete form as a 1D array. Each entry represents the corresponding optical information of a quantized intensity value. In such a discrete scenario, the resolution (number of entries) of the transfer function table should match the data precision of the intensity values. An 8-bit volume only requires 2^8 (256) entries in the transfer function table; a 16-bit volume requires 2^{16} (65,536) entries. Such high resolution transfer functions for HDR VolVis imply two issues for transfer function design:

- Editing high resolution transfer function on display devices with much lesser resolution.
- Encoding high resolution transfer function in graphics hardware with a limited look-up table size.

In the following parts of this section, we discuss our approaches for handling the first issue. The encoding strategy for high resolution transfer functions will be discussed in the implementation section (Section 6).

As discussed above, rendering high precision input data introduces new issues in transfer function design. The resolution of currently available computer display devices is insufficient to display the full precision of each axis value (intensity and color/opacity). If a low resolution axis is used, different features with close intensity values cannot be differentiated. It is useful to be able to magnify one portion of the intensity region while still keeping other regions visible, following the general concept of *focus+context*. One possible way is to use a scrolling window. Another intuitive implementation is to have two separate windows in the transfer function specification interface: the first window displaying the whole transfer function and the second displaying a user specified zoom-in region in which the screen resolution matches that of the local transfer function. A drawback of such an implementation is that the user has to switch between two transfer function windows frequently. We develop a novel transfer function specification interface with nonlinear magnification for HDR VolVis. In our interface, the zoomed region and the whole transfer function context co-exist in the same editing window.

In the data visualization domain, the focus+context concept has been developed to allow the simultaneous presentation of global (context) and detail (focus) information in the same display. The fish-eye view technique [9], [10], which essentially is a nonlinear magnification transformation, allows the user to see an object in the region of interest in detail, and other lower resolution objects peripherally. The advantage of focus+context techniques is that the contextual relationship between the focus center and other context regions is preserved, while giving the user a useful level of control over the regions at hand.

We introduce a 1D version of the fish-eye visualization technique to the transfer function specification interface for our HDR VolVis system. As illustrated in Fig. 6, we develop a focus+context design interface which performs 1D nonlinear scaling on the intensity axis. In our current implementation, we employ a one-dimensional transfer function which assigns color and opacity to the volume based on the scalar voxel values. Our system allows the user to switch between RGB space and HSV (Hue, Saturation, Value) space [37] when specifying color. Color values are interpolated between user specified knots in HSV space, and then converted to RGB space for later volume rendering. In Fig. 6a, a transfer function is displayed in linear scale. The horizontal axis represents normalized intensity values. The green bar at the bottom indicates the intensity range to be magnified, which is specified by the user. The desired amount of magnification is also specified by the user. The goal is to nonlinearly scale

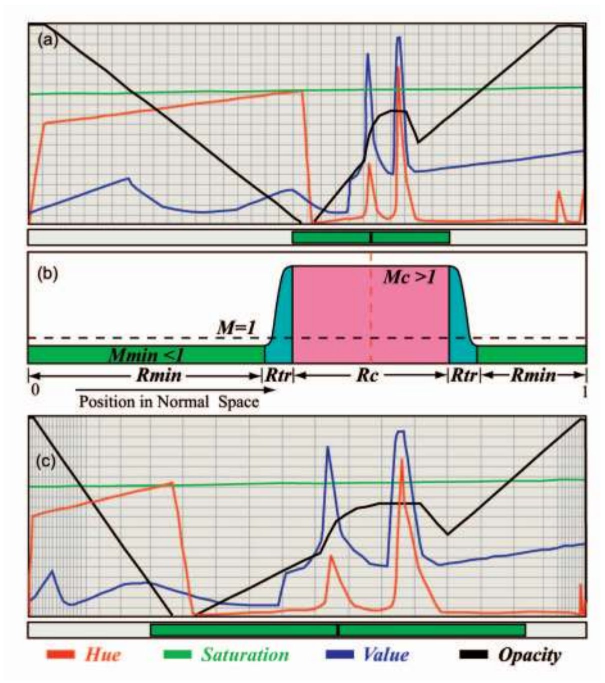

Fig. 6. Focus+context transfer function specification interface. (a) Transfer function displayed in normal scale. The green bar indicates the region of interest to be magnified. (b) Magnification curve. The region of interest has magnification value $M_c > 1$. (c) Transfer function displayed in nonlinearly scaled intensity space.

the intensity axis so that the full intensity range still fits in the same horizontal window range. The following formalizes the problem and explains our solution.

Fig. 6b illustrates a magnification curve. Through user input, the intensity range of interest R_c is magnified by a constant factor of $M_c > 1$. We assume that the rest of the intensity range R_{min} is minified with a factor of $M_{min} < 1$, which is to be computed. M_{tr}, the magnification factors of the transition region R_{tr} are linearly interpolated between M_c and M_{min}. R_{tr} is also predefined. The task is to compute the value of M_{min}. Since the entire intensity range should still fit in the horizontal window range, the following equation must be satisfied:

$$\int_{x\in R_{min}} M_{min} dx + \int_{x\in R_c} M_c dx + \int_{x\in R_{tr}} M_{tr} dx = 1. \qquad (2)$$

Since the intensity range is normalized into $[0, 1]$, M_c and M_{min} are constant (M_{min} is unknown), and as M_{tr} can be treated as $(M_{min} + M_c)/2$ throughout the transition regions, M_{min} can be computed. Nonlinear scaling of the intensity axis can then be performed, and the full intensity range is always presented in the view. Fig. 6c shows the transfer function of Fig. 6a in a nonlinearly scaled intensity space.

As illustrated in Fig. 6, we design a widget to indicate the intensity range of interest. This widget is described by a center and a width. During transfer function design, the user can click and drag the center of the widget to specify a region of interest for magnification. By clicking elsewhere, the user can change the width of the interested region to be magnified. It is also possible for the user to specify multiple magnification regions simultaneously. Because of this feature, transfer function design is not limited to having only one focus center. The whole design interface is still in a single window. In the zoom-out window method, one zoom-out window must be created for every magnification region.

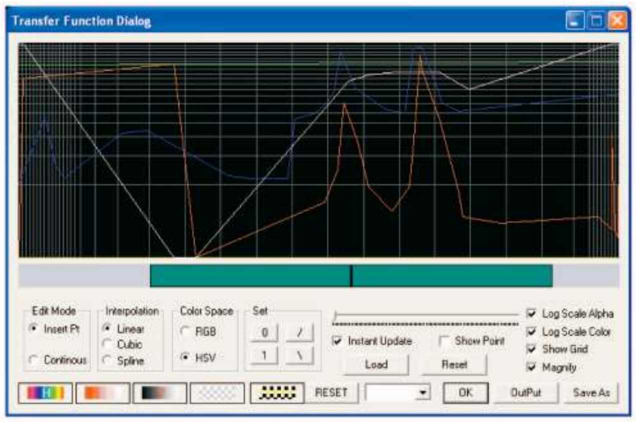

Fig. 7. A snapshot of the transfer function interface in our system. The curves are displayed with nonlinear magnification of the horizontal axis and logarithmic scaling of the vertical axis.

Another transfer function design issue pertains to the vertical axis for alpha and color values. As discussed in Section 3, the opacity values in the transfer function are preferred to be small to visualize detailed features in a large volume data. One goal of our volume rendering is to be able to reveal more internal structures simultaneously. A lower range of alpha values is desired because large alpha values will be quickly accumulated to full opacity. Therefore, our transfer function specification interface must accommodate the editing of alpha values near zero. The linearly scaled alpha axis provides limited resolution to small alpha values. To address this issue, we use logarithmically scaled alpha and color axes [30], through which ranges with small alpha and color values are magnified.

We combine the logarithmic scaling in the vertical axis and nonlinear scaling in the horizontal axis for the transfer function design. Fig. 7 illustrates the same transfer function as in Fig. 6c, with the vertical axis of the interface scaled logarithmically. This makes it more convenient to edit small opacity and color values. Moreover, with one portion of the intensity scale magnified, the user can fine tune the transfer function in this region. We allow the user to switch between nonlinear and linear transfer function axes for convenience.

6 Implementation Details

Our high dynamic range volume visualization method is not limited to specific volume rendering implementations. Volume rendering with software implementation approaches can be easily adapted to the high dynamic range paradigm. Special purpose hardware for volume rendering, such as VolumePro [26], can also benefit from the high dynamic range volume by storing results in a high dynamic range format and integrating tone reproduction in the postrendering stage. We implement our HDR volume visualization system on a PC and leverage modern commodity graphics hardware to achieve high performance HDR visualization. All experiments have been performed

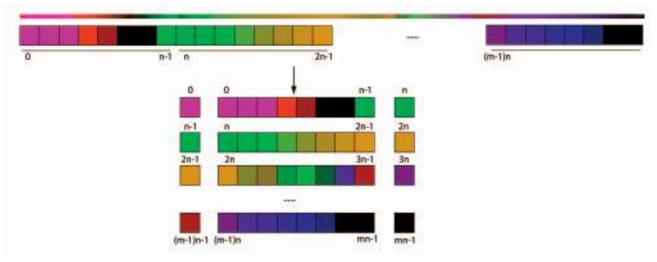

Fig. 8. Implementation of 1D transfer function with more than 4,096 entries which exceeds the limitation of texture size of current graphics hardware. A 1D array with size of mn is stored as a 2D texture with size of $m \times (n+2)$ in graphics hardware.

on a Dell Precision 530 workstation with a single Intel Xeon 2.20 GHz CPU, 1GB RAM, and an NVidia 256MB GeForce6800 Ultra graphics card. In the following sections, we discuss several implementation issues.

6.1 High Precision Rendering on Graphics Hardware

We first explain and analyze how to achieve maximum precision within the constraints of current graphics hardware. For conventional computations, we could use vendor extensions such as the NVidia extension GL_NV_float_buffer and support a floating-point texture of up to 32 bits per channel. However, for volume rendering, it is important to have alpha blending functionality. For this feature, currently only 16-bit floating-point buffers are supported natively at the hardware level. At 24 bits or higher, the performance reduces dramatically. Therefore, we use 16-bit floating-point buffers in our implementation. As discussed earlier, for 16-bit floating-point representation of alpha, the highest precision is 2^{-24}. This precision can be used to render a volume data of $2,048^3$ in size while reasonably preserving the details of the entire volume. For volume data sets with full floating-point precision, a quantization is necessary to fit the data into 16-bit precision. We expect higher precision alpha blending functionality to be supported better in the future commodity graphics hardware.

6.2 Transfer Function Encoding in 2D Texture

In hardware implementations of texture-based volume rendering, transfer functions are encoded as discretized lookup tables using 1D textures. Each entry in the 1D array stores the optical property (color and opacity); the texture index corresponds to the normalized intensity value. Graphics hardware uses the intensity value as the key to perform the lookup.

For current commodity graphics hardware, the texture size is limited by 4,096 in each dimension. However, for high precision rendering, 16 bits data requires 2^{16} (65,536) entries in the lookup tables. Higher precision data demands more levels in the lookup tables which can not be accommodated by the current available graphics hardware. Our solution is to encode a 1D lookup tables into multiple rows of a 2D texture. As illustrated in Fig. 8, for a lookup table with mn entries, we rearrange the entries into m rows. In such an implementation, 1D entries are broken between entry $kn-1$ and kn ($k = 0, 1, \ldots, m-1$). To facilitate linear interpolation between these pairs, we expand the horizontal texture dimension from n to $n+2$, and duplicate their corresponding neighbors in the 1D form. Such 2D encoding could contain up to $4,096 \times 4,094$ $(16,769,024)$ entries. Such an encoding scheme has also been discussed for implementing general parallel data structures on the GPU [1].

6.3 Interactive Rendering for Large Data Sets

We employ a texture mapping-based hardware implementation. Volume rendering is implemented using hardware accelerated 3D texture rendering [2], [14], [34], [36]. Parallel polygon slices perpendicular to the viewing direction are generated, texture-mapped with the 3D volume and composited together in back-to-front order. When rendering a volume of $2,048^3$ or larger in hardware, an immediate challenge is to cope with the limited graphics card memory. Naturally, we have to develop an out-of-core method so that partial volume data can be efficiently loaded and unloaded to and from the graphics card memory during the rendering.

We also seek time-critical visualization, i.e., we wish to provide instant feedback to user interaction. To achieve these goals, we have implemented a multiresolution volume data structure [17] and level-of-detail (LOD) volume rendering [41]. In our system, data sets are organized in a hierarchical brick structure. The multiresolution volume data structure is constructed by continuously down-sampling the higher level volume starting from the original volume (as the highest level). We subdivide large volumes of data into blocks small enough to fit in memory and large enough to allow efficient disk I/O. On PCs, disk read performance tends to increase with the size of the I/O request at least up to 256KB. On fiber-channel and striped disk systems I/O speeds continue to increase substantially up through request sizes of several MB. In our implementation, we typically choose block dimensions of 64 voxels on a side. For any level volume having size larger than 64^3, it is subdivided into blocks of 64^3.

We take a very straightforward out-of-core rendering approach. During rendering, two parallel threads, one for rendering, and another for I/O tasks (also known as loading data from hard disks), are executed simultaneously. Data bricks are read from the hard drive into the main memory upon the request of the rendering thread. When the user interacts with the volume data (such as user navigation or transfer function respecification), volume at a certain level (e.g., 128^3 or 256^3) is selected and rendered to keep up with the user interaction. The higher level volume is rendered only when the system enters the idle mode, i.e. user finishes the interaction and no change in rendering parameters is detected. The system automatically loads higher resolutions and renders higher quality images. Through this mechanism, we can quickly navigate the volume data and change the transfer function to achieve the desired visualization.

7 Results and Discussions

We have applied our HDR visualization technique to visualizing large fluid dynamics simulations (Table 1). One simulation, depicted in Fig. 3 and Fig. 4, shows temperature fluctuations in the deep convection zone of a red giant star [29]. Large amplitude positive and negative fluctuations are seen near the surface of the star, where they are driven by a nearly constant convective energy flux

TABLE 1
List of High Dynamic Range Volume Rendering Results

No	Simulation	Size	Parameter to be visualized
1	Convection zone of a red giant star	512^3	Temperature
2	Turbulent mixing of air and SF_6	$256 \times 512 \times 1024$	Fraction of SF_6
3	Homogeneous decaying compressible fluid turbulence	2048^3	Vorticity
4	Decaying, compressible and homogeneous turbulence	1000^3	s, $\vec{\nabla} \cdot \vec{U}$, $\lvert\vec{\nabla} \times \vec{U}\rvert$, and $det(S_d)$

going through very low density regions. Large negative temperature fluctuations are seen in the large, turbulent, and cool down flowing plume on one side of the star. The warm updraft on the other side of the star is relatively free of turbulence. The temperature field in such a warm updraft is characterized by extremely small amplitude, and small scale fluctuations around a slightly positive and smoothly varying background temperature. In order to see both large and small amplitude temperature fluctuations in a single visualization, a highly nonlinear mapping technique of temperature to color and intensity is of great importance. Our HDR volume rendering and dynamic tone mapping have made this possible.

The second simulation data shows turbulent mixing of fluids that typically produces a wide range of concentrations. In a turbulent layer between two pure fluids, some regions will be a substantial blend of both, other regions might be completely unmixed (100 percent one fluid or the other), while still others might have extremely small fractions of either fluid mixed into the other. The precise values of even minuscule amounts of mixing can be very important. Examples include the dilution of toxins in a blood stream, pollution into the Earth's atmosphere, flows with chemical combustion, and nuclear reaction chains in the flame zones of many stars. In Fig. 9, we show results from a two-fluid PPM simulation where turbulent mixing is driven by shear ($256 \times 512 \times 1,024$). Initially, a region of pure air is right next to, and is in relative motion to, a region of pure Sulfur Hexafluoride (SF_6). The interface between these two gasses starts as a plane, and the relative motion corresponds to the two blocks of gas sliding past each other. Hence, the initial contact discontinuity is also a slip surface. In addition to the large relative motion there are also small velocity perturbations, which grow with time due to the Kelvin-Helmholtz instability, thereby generating a turbulent mixing layer. The fractional volume of the heavier gas, SF_6, is followed as a dynamical variable in each computational cell. Fractional volumes range from 0 (pure air) to 1 (pure SF_6). In Fig. 9b, we show a rendering result of SF_6 fraction value using traditional 8-bit precision during rendering. The output image is also in 8-bits-per-channel. The image has been brightened seven times for easy inspection. It is clear that due to the quantization error, many flow regions have been rendered as blank areas. A lot of details are lost. Fig. 9a shows a tone mapped image from high dynamic range volume rendering. Many more details have been preserved. Fig. 9c and Fig. 9d show the corresponding high dynamic range image being rendered with two different exposures. As these visualizations show, very small amounts of the heavier gas (SF_6, concentrated in purple regions) can be mixed into the air via turbulent churning of the gas. Similarly, very small amounts of air can be mixed into the SF_6. Hence, if the user wishes to follow these very diluted mixtures, high dynamic range is needed for the fractional volume near both zero and unity. Note that all the rendered images in Fig. 9 use the same transfer function which is shown as the color bars. For low precision rendering, both volume data and transfer function are down-sampled to 8 bits precision or 256 levels.

The third simulation data, depicted in Fig. 10, shows magnitude of vorticity from a high-resolution ($2,048^3$) simulation of homogeneous decaying compressible fluid turbulence [27], [28] when the turbulence is fully developed. Fig. 10a shows a snapshot in time when the turbulence is in the process of developing. Fig. 10b is rendered from the same simulation after the turbulence is fully developed. Slip surfaces (Bright yellow/white structures in Fig. 10a) and intense vortex tubes (Elliptical structures in Fig. 10b) are clearly visible in the images. The goal of this study is to improve our understanding of inertial range turbulence, with applications in astrophysical flows, such as stellar convection, and the development and testing of subgrid-scale models of turbulence. Since no general theory of fluid turbulence is currently available, volume visualization is important for developing new insights into these flows. Of particular interest is the search for coherent three-dimensional structures, which control the dynamics of the flow. Three-dimensional fluid turbulence naturally produces slip surfaces and intense vortex tubes, which can be seen in the vorticity field here. In systems where a wide range of spatial scales is available, such as in this $2,048^3$ run, values of local vorticity span a wide dynamic range. Since the three-dimensional structure of both weak and strong vorticity is of interest, HDR visualization techniques that can handle such a wide dynamic range of values are very effective for examining these fields.

The last example is a numerical simulation of decaying, compressible, and homogeneous turbulence. The resolution of the volume is $1,000^3$. Four quantities (s, $\vec{\nabla} \cdot \vec{U}$, $|\vec{\nabla} \times \vec{U}|$, and $det(S_d)$) discussed in Section 3 are illustrated in Fig. 11. The top row images show the turbulence in its early stage, t = 0.30; the bottom row shows the flow in its later stage, t = 1.00. All eight images are rendered using the same viewing parameters. The same transfer functions are applied to each time-pair images. Details of this simulation have been discussed in Section 3. It is clear from the illustration that such time varying turbulent flows have a wide range of variations. The values of quantities vary through time. Such visualizations can definitely benefit from a high dynamic range treatment.

As is evident in these images, one clear advantage for HDR volume visualization is the rich information retained in one single visualization. A wide range of details can be dynamically viewed on a regular low range display. This can be

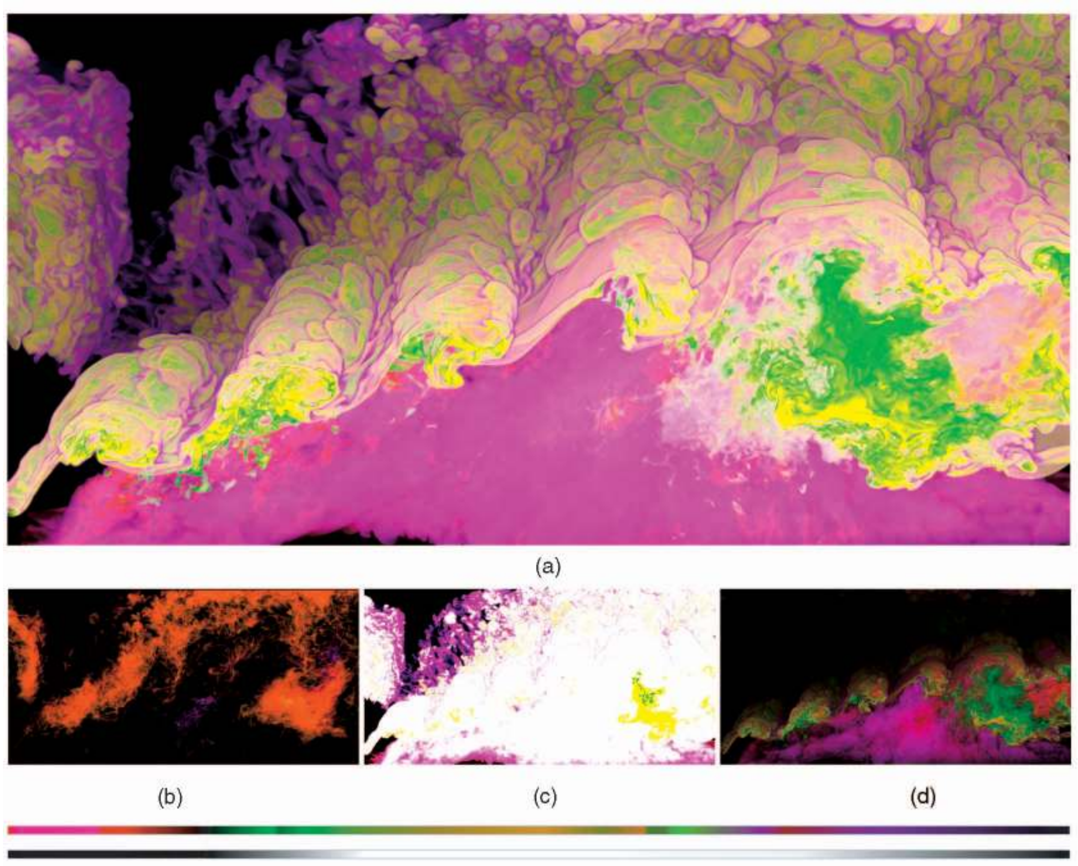

Fig. 9. High dynamic range volume rendering result for turbulent mixing of air and Sulfur Hexafluoride (SF_6). (a) Tone mapped image from high dynamic range volume rendering. (b) Rendering result using 8 bits precision in computation and output. The image is brightened seven times. (c) and (d) Images at different exposure levels from the same high dynamic range volume rendering. All rendered images use the same transfer function which is illustrated by the color bars (top: color, bottom: alpha).

effectively experienced by playing the HDR movies and tuning the tone mapping parameters. Here, even pregenerated animations offer interactive data exploration opportunities. (Videos clips of high resolution HDR volume rendering movies are available at: http://www.dtc.umn.edu/~xyuan/HDRVis/.)

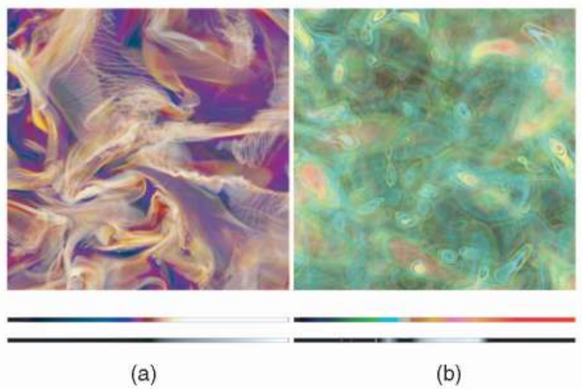

Fig. 10. Visualization results by the high dynamic range volume rendering system. Images depict magnitude of vorticity from a high-resolution simulation of homogenous decaying compressible fluid turbulence. (a) is at a time when the turbulence is in the process of developing. (b) is at a time when the turbulence is fully developed. The color bars below each rendered image are the corresponding transfer functions (top: color, bottom: alpha).

As aforementioned in Section 6, we render a volume in a down-sampled resolution during navigation for interactive performance. Starting from the highest resolution (level 1), we downsample the volume by a factor of two for each lower level. In our system, the resolution level corresponding to 128^3 is normally selected for interactive navigation. This way, a frame rate of over 30 fps can always be achieved during navigation. When the user stops navigation, volumes with higher resolutions are progressively loaded and rendered to achieve higher rendering quality; this continues until the highest quality is achieved, i.e., the full resolution volume is rendered. Whenever the user restarts the navigation, the system automatically lowers the rendering resolution to sustain an interactive rendering rate. For a high-resolution ($2,048^3$) simulation of homogeneous decaying compressible fluid turbulence data set (Fig. 10), over 15 seconds is required to achieve the highest rendering quality with the window size of $1,024^2$. For a smaller data set (red giant star, 512^3), less than 0.5 seconds are required for the full rendering. Note that the rendering time also depends on the number of slices applied per voxel distance. A sampling rate of two slices per voxel distance is used in our experiments.

Without high dynamic range volume rendering, to perform the above rendering on commodity graphics hardware, a mapping must be conducted which takes the original scalar values to integers in the range $[0-255]$. Linear mapping is the simplest approach. A log scaling is

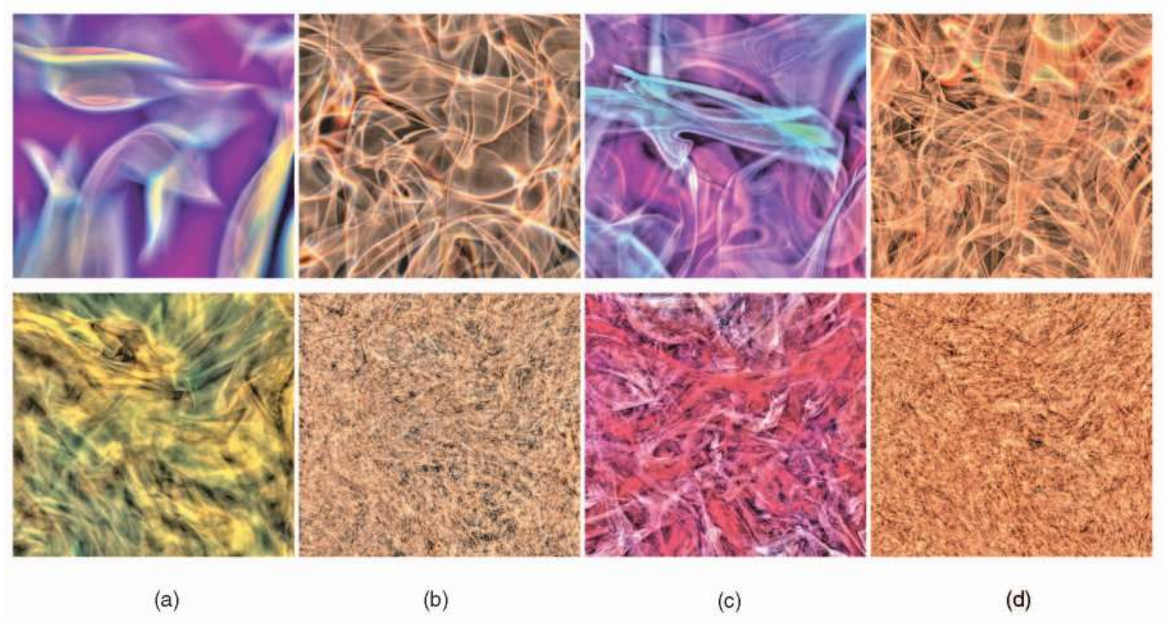

Fig. 11. High dynamic range volume rendering of four flow quantities of a numerical compressible turbulent simulation. (a) Entropy s, (b) divergence of velocity $\vec{\nabla} \cdot \vec{U}$, (c) vorticity magnitude $|\vec{\nabla} \times \vec{U}|$, and (d) determinant of the deviatoric part of the symmetric strain rate tensor $det(S_d)$. The top row is the turbulence in its early stage, t = 0.30; the bottom row is the flow in its later stage, t = 1.00.

effective for positive definite quantities with a large dynamic range, such as mass density in stellar convection. Hyperbolic tangent mappings work well for quantities distributed with wide tails, such as vorticity components. However, such mappings require prior knowledge of the volume data set and substantial effort in the preprocessing. In addition, the loss of information is inevitable during the quantization for such mappings. Even with the best nonlinear mapping to handle an extended dynamic range of values, HDR might still be necessary to retain precision. For example, in the compressible decaying turbulence flow computed on a $1,000^3$ mesh mentioned above, negative values of $det(S_d)$ span at least four decades (four orders of magnitude) of range (i.e., from the magnitude of the RMS value to magnitude of the most negative value). Positive values of $det(S_d)$ span at least three decades of range (from the RMS value to the maximum). A combined scale, covering both positive and negative values, from $1/10$ of the RMS value to each extreme value, would span a total of nine decades: five decades for negative values and four decades for positive values. If only 256 levels were available to describe $det(S_d)$, then there would only be about 28 levels per factor of 10, leading to a jump of 8.5 percent between consecutive levels. Our HDR volume rendering provides a direct and flexible way of exploiting full information for volume data sets.

8 Concluding Remarks

In this paper, we have presented a high dynamic range volume visualization (HDR VolVis) system for rendering volume data with both high spatial and intensity resolutions. Our method performs high precision volume rendering followed by dynamic tone mapping to preserve details on regular display devices. With interactive input from the user, our system automatically adjusts the tone reproduction for the final display to enhance selected features. In this work, we also present a novel transfer function specification interface with nonlinear scaling of intensity range and logarithmic scaling of color/opacity range to facilitate HDR volume visualization. By leveraging modern commodity graphics hardware, multiresolution rendering techniques and existing out-of-core acceleration, our system can produce interactive visualization of huge volume data. We have demonstrated the suitability of our system to visualize large simulation data that has a wide range of physical properties.

We plan to enhance our transfer function specification interface by incorporating some recently developed advanced techniques. For example, we intend to incorporate high dimensional transfer function design [16]. In such cases, techniques of 2D or higher dimensional magnification such as fisheye [31], hyperbolic space [18], [23] are useful. It is also possible to utilize the information of the volume data, analyze topology, and then apply semantic magnification. As we discussed in Section 4, since most existing tone mapping techniques are developed for real world photography, we are developing new tone mapping operators suitable for our visualization purposes and evaluation methods for such new operators.

To further improve interactivity, we are currently working on porting the HDR VolVis system to a Linux cluster which drives the PowerWall in the Laboratory of Computational Science and Engineering (LCSE) in University of Minnesota. The cluster consists of 14 nodes. Each node is equipped with an NVidia Quadro4400 graphics card, 8GB of memory, and 12 Seagate 400GB SATA disks. Nodes are interconnected with Infiniband 4X HCA. Each PC node will perform the high precision volume rendering and directly send the generated tone-mapped rendered images to its corresponding screen of the PowerWall.

Acknowledgments

Support for this work includes the University of Minnesota Computer Science Department Start-up funds, University of

Minnesota Digital Technology Center Seed Grants 2002-4, US National Science Foundation (NSF) ACI-0238486 (CAREER), NSF EIA-0324864 (ITR), and the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement number DAAD19-01-2-0014. Its content does not necessarily reflect the position or the policy of this agency, and no official endorsement should be inferred. The last author would like to acknowledge the following support: DoE grant Nos. DE-FG02-87ER25035 and DE-FG02-94ER25207, NSF PACI at NCSA grant, CDA-950297, and UNISYS Corp hardware gift. The authors would like to thank Paul Woodward for his support. They also thank Michael Knox for his technical support and Amit Shesh and Nathan Gossett for proof reading. Finally, the authors would like to thank the reviewers for their constructive reviews.

References

[1] I. Buck, T. Foley, D. Horn, J. Sugerman, K. Fatahalian, M. Houston, and P. Hanrahan, "Brook for GPUs: Stream Computing on Graphics Hardware," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 777-786, 2004.

[2] B. Cabral, N. Cam, and J. Foran, "Accelerated Volume Rendering and Tomographic Reconstruction Using Texture Mapping Hardware," *Proc. Symp. Volume Visualization,* pp. 91-98, 1994.

[3] P.E. Debevec and J. Malik, "Recovering High Dynamic Range Radiance Maps from Photographs," *Proc. SIGGRAPH '97,* pp. 369-378, 1997.

[4] K. Devlin, A. Chalmers, A. Wilkie, and W. Purgathofer, "STAR: Tone Reproduction and Physically Based Spectral Rendering," *Proc. Eurographics '02,* pp. 101-123, 2002.

[5] F. Drago, W.L. Martens, K. Myszkowski, and H.-P. Seidel, "Perceptual Evaluation of Tone Mapping Operators," *Proc. ACM SIGGRAPH Conf. Abstract and Applications,* 2003.

[6] F. Drago, K. Myszkowski, T. Annen, and N. Chiba, "Adaptive Logarithmic Mapping for Displaying High Contrast Scenes," *Computer Graphics Forum,* vol. 22, no. 3, pp. 419-419, 2003.

[7] F. Durand and J. Dorsey, "Fast Bilateral Filtering for the Display of High-Dynamic-Range Images," *Proc. SIGGRAPH '02,* pp. 257-266, 2002.

[8] R. Fattal, D. Lischinski, and M. Werman, "Gradient Domain High Dynamic Range Compression," *Proc. SIGGRAPH '02,* pp. 249-256, 2002.

[9] G.W. Furnas, "Generalized Fisheye Views," *Proc. SIGCHI Conf. Human Factors in Computing Systems,* pp. 16-23, 1986.

[10] G.W. Furnas, "The FISHEYE View: A New Look at Structured Files," *Readings in Information Visualization: Using Vision to Think,* pp. 312-330, 1999.

[11] A. Ghosh, M. Trentacoste, and W. Heidrich, "Volume Rendering for High Dynamic Range Displays," *Proc. EG/IEEE VGTC Workshop Volume Graphics '05,* pp. 91-98, 2005.

[12] N. Goodnight, R. Wang, C. Woolley, and G. Humphreys, "Interactive Time-Dependent Tone Mapping Using Programmable Graphics Hardware," *Proc. 14th Eurographics Symp. Rendering,* pp. 26-37, 2003.

[13] HDRshop, http://www.ict.usc.edu/graphics/HDRShop/, 2006.

[14] M. Ikits, J. Kniss, A. Lefohn, and C. Hansen, *GPU Gems: Programming Techniques, Tips, and Tricks for Real-Time Graphics,* chapter on volume rendering techniques, pp. 667-692, Addison Wesley, 2004.

[15] S.B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High Dynamic Range Video," *ACM Trans. Graphics,* vol. 22, no. 3, pp. 319-325, 2003.

[16] J. Kniss, G. Kindlmann, and C. Hansen, "Multidimensional Transfer Functions for Interactive Volume Rendering," *IEEE Trans. Visualization and Computer Graphics,* vol. 8, no. 3, pp. 270-285, July/Sept. 2002.

[17] E. LaMar, B. Hamann, and K.I. Joy, "Multiresolution Techniques for Interactive Texture-Based Volume Visualization," *Proc. IEEE Conf. Visualization '99,* pp. 355-361, 1999.

[18] J. Lamping, R. Rao, and P. Pirolli, "A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies," *Proc. SIGCHI Conf. Human Factors in Computing Systems,* pp. 401-408, 1995.

[19] G.W. Larson, H. Rushmeier, and C. Piatko, "A Visibility Matching Tone Reproduction Operator for High Dynamic Range Scenes," *IEEE Trans. Visualization and Computer Graphics,* vol. 3, no. 4, pp. 291-306, Oct./Dec. 1997.

[20] P. Ledda, A. Chalmers, T. Troscianko, and H. Seetzen, "Evaluation of Tone Mapping Operators Using a High Dynamic Range Display," *ACM Trans. Graphics,* vol. 24, no. 3, pp. 640-648, 2005.

[21] Y. Li, L. Sharan, and E.H. Adelson, "Compressing and Companding High Dynamic Range Images with Subband Architectures," *ACM Trans. Graphics,* vol. 24, no. 3, pp. 836-844, 2005.

[22] R. Mantiuk, G. Krawczyk, K. Myszkowski, and H.-P. Seidel, "Perception-Motivated High Dynamic Range Video Encoding," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 733-741, 2004.

[23] T. Munzner, "H3: Laying Out Large Directed Graphs in 3D Hyperbolic Space," *Proc. 1997 IEEE Symp. Information Visualization,* pp. 2-10, 1997.

[24] NVidia Corp., Nvidia Opengl Extension Specifications, http://developer.nvidia.com/object/nvidia_opengl_specs.html, 2005.

[25] S.N. Pattanaik, J.A. Ferwerda, M.D. Fairchild, and D.P. Greenberg, "A Multiscale Model of Adaptation and Spatial Vision for Realistic Image Display," *Proc. SIGGRAPH '98,* pp. 287-298, 1998.

[26] H. Pfister, J. Hardenbergh, J. Knittel, H. Lauer, and L. Seiler, "The VolumePro Real-Time Ray-Casting System," *Proc. SIGGRAPH '99,* pp. 251-260, 1999.

[27] D. Porter, A. Pouquet, I. Sytine, and P. Woodward, "Turbulence in Compressible Flows," *Physica A,* pp. 263-270, 1999.

[28] D. Porter, A. Pouquet, and P. Woodward, "Measures of Intermittency in Driven Supersonic Flows," *Physical Rev. E,* vol. 66, 2002.

[29] D. Porter and P. Woodward, "3-D Simulations of Turbulent Compressible Convection," *The Astrophysical Supplement Series,* 2000.

[30] S. Potts and T. Möller, "Transfer Functions on a Logarithmic Scale for Volume Rendering," *Proc. 2004 Conf. Graphics Interface,* pp. 57-63, 2004.

[31] U. Rauschenbach, "The Rectangular Fish Eye View as an Efficient Method for the Transmission and Display of Large Images," *Proc. IEEE Int'l Conf. Image Processing,* pp. 115-119, 1999.

[32] E. Reinhard, M. Stark, P. Shirley, and J. Ferwerda, "Photographic Tone Reproduction for Digital Images," *Proc. SIGGRAPH '02,* pp. 267-276, 2002.

[33] E. Reinhard, G. Ward, S. Pattanaik, and P. Debevec, *High Dynamic Range Imaging, First Edition: Acquisition, Display, and Image-Based Lighting.* Morgan Kaufmann, 2005.

[34] C. Rezk-Salama, K. Engel, M. Bauer, G. Greiner, and T. Ertl, "Interactive Volume Rendering on Standard PC Graphics Hardware Using Multi-Textures and Multi-Stage Rasterization," *Proc. SIGGRAPH/EUROGRAPHICS Workshop Graphics Hardware,* pp. 109-118, 2000.

[35] H. Seetzen, W. Heidrich, W. Stuerzlinger, G. Ward, L. Whitehead, M. Trentacoste, A. Ghosh, and A. Vorozcovs, "High Dynamic Range Display Systems," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 760-768, 2004.

[36] C.T. Silva, J.L.D. Comba, S.P. Callahan, and F.F. Bernardon, "A Survey of GPU-Based Volume Rendering of Unstructured Grids," *Brazilian J. Theoretic and Applied Computing (RITA),* vol. 12, no. 2, pp. 9-29, Oct. 2005.

[37] A.R. Smith, "Color Gamut Transform Pairs," *Proc. SIGGRAPH '78,* pp. 12-19, 1978.

[38] J. Tumblin and H. Rushmeier, "Tone Reproduction for Realistic Images," *IEEE Computer Graphics and Applications,* vol. 13, no. 6, pp. 42-48, 1993.

[39] J. Tumblin and G. Turk, "LCIS: A Boundary Hierarchy for Detail-Preserving Contrast Reduction," *Proc. SIGGRAPH '99,* pp. 83-90, 1999.

[40] G. Ward, "Real Pixels," *Graphics Gems II,* J. Arvo, ed., pp. 80-83. Academic Press, 1991.

[41] M. Weiler, R. Westermann, C. Hansen, K. Zimmermann, and T. Ertl, "Level-of-Detail Volume Rendering via 3D Textures," *Proc. 2000 IEEE Symp. Volume Visualization,* pp. 7-13, 2000.

[42] P.R. Woodward, "Numerical Methods for Astrophysicists," *Astrophysical Radiation Hydrodynamics,* vol. 54, pp. 245-326, 1986.

[43] P.R. Woodward, S.E. Anderson, D.H. Porter, and A. Iyer, "Distributed Computing in the SHMOD Framework on the NSF Teragrid," technical report, LCSE, UMN, Feb. 2004.

[44] P.R. Woodward and P. Colella, "The Numerical Simulation of Two-Dimensional Fluid Flow with Strong Shocks," *J. Computational Physics,* vol. 54, pp. 115-173, 1984.

[45] X. Yuan, M.X. Nguyen, B. Chen, and D.H. Porter, "High Dynamic Range Volume Visualization," *Proc. IEEE Conf. Visualization '05,* pp. 327-334, 2005.

Xiaoru Yuan received the BS degree in chemistry and the BA degree in law from Peking University, China, in 1997 and 1998, respectively. He received the MS degree in computer engineering from the University of Minnesota at Twin Cities in 2005. He is a PhD candidate in computer science at the University of Minnesota at Twin Cities. His primary research interests fall in the field of computer graphics and visualization with emphasis on illustrative visualization (nonphotorealistic rendering and its application in visualization), high dynamic range imaging and rendering, novel visualization user interface, and computational geometry. He is a student member of the IEEE. For more information, see http://www.cs.umn.edu/~xyuan.

Minh X. Nguyen received the BS degree in computer science from Ho Chi Minh City University (now, Ho Chi Minh City University of Natural Sciences), Vietnam in 1994. He is a PhD candidate in computer science from the University of Minnesota, Twin Cities. Before going back to graduate study, he was a lead programmer at a cartography company (DolSoft Co. Ltd., Ho Chi Minh City, Vietnam) specializing in GIS and a junior researcher at the Institute of Applied Mechanics, Ho Chi Minh City, Vietnam. His research interests are interactive 3D visualization with emphasis on hardware support, geometry modeling, scientific visualization, and computational geometry. For more information, see http://www.cs.umn.edu/~mnguyen.

Baoquan Chen received the MS degree in electronic engineering from Tsinghua University, Beijing (1994), and a second MS (1997) degree and a PhD (1999) degree in computer science from the State University of New York at Stony Brook. He is an assistant professor of computer science and engineering at the University of Minnesota at Twin Cities, where he is also a member of the Digital Technology Center and Digital Design Consortium. His research interests generally lie in computer graphics and visualization, focusing specially on 3D data acquisition, illustrative rendering, visualization, and interactive techniques. His research is supported by the US National Science Foundation (NSF), Army Research, Microsoft Research, and a private donation. He is the recipient of the Microsoft Innovation Excellence Program 2002, the NSF CAREER award 2003, and McKnight Land-Grant Professorship of the University of Minnesota 2004-2006. Dr. Chen has served, or is serving, on program and paper committees of several conferences in the field, most notably, IEEE Visualization (program cochair 2004, general cochair 2005/2006), and the Symposium on Point Based Graphics (2004/2005, papers cochair 2006). For more information, see http://www.cs.umn.edu/~baoquan. He is a senior member of the IEEE.

David H. Porter received the AB degree in physics and applied math from the University of California, Berkeley in 1979, and the PhD degree in physics, also from the University of California, Berkeley, in 1985. In 1985, he started working in computational fluid dynamics (CFD) as a staff physicist at the Lawrence Livermore National Laboratory. Since 1986, he has been a research associate in the Astronomy Department of the University of Minnesota (UM), where he has continued his studies in CFD. In 1994, he was promoted to senior research associate. In Minnesota, he has been a prime user of, as well as involved in the development of, numerical laboratories at the Minnesota Supercomputer Institute, the Army High Performance Computing Research Center, and the Laboratory for Computational Science and Engineering (LCSE) at UM. Along with Paul Woodward and Sarah Anderson at the LCSE and 10 colleagues at Livermore National Lab and IBM, he was awarded the 1999 IEEE Gordon Bell Award in the Performance Category for an 8-billion cell sPPM simulation of the instability and turbulent mixing of a shock-accelerated fluid interface.