Interactive Transfer Function Design on Large Multiresolution Volumes

Steven Martin*
The Ohio State University

Han-Wei Shen†
The Ohio State University

Abstract

Interactive transfer function design techniques seek to leverage user knowledge to facilitate the discovery of data salience. In this process, interactive volume rendering is typically a necessity. Interactive volume rendering of large-scale data on workstations is often accomplished through the use of level of detail techniques, prioritizing information deemed to be salient over information deemed to be unimportant. If salience is not known a priori, and interactive transfer function design techniques that depend on volume rendering are to be applied to large-scale data using level of detail selection, then there is a cyclic dependency. Techniques must be applied that can support simultaneous development of salience both for the transfer function design technique and the level of detail selection technique. Building on recent work in LOD selection, we propose an interactive transfer function design technique that enables incremental salience discovery to support simultaneous construction of transfer functions and LOD selections on large-scale data.

Keywords: Transfer function design, Scalable visualization, Level of detail selection

Index Terms: Computer Graphics [I.3.6]: Graphics data structures and data types—Transfer function design, Scalable visualization, Level of detail selection

1 Introduction

Direct volume rendering (DVR) is widely used in the visualization of volume data. Key to the creation of high quality visualizations using DVR is the construction of effective transfer functions [5]. Effective transfer functions emphasize salient information while deemphasizing unimportant information. Interactive, semi-automatic, transfer function design seeks to leverage users' domain-specific knowledge [9] to progressively develop per-interval volume salience.

Interactive transfer function design techniques rely on iterative refinement, by users considering visual feedback, to guide a transfer function generation algorithm. In the case of DVR using optical models that consider opacity [7], modifications to the transfer function require re-rendering of the volume, placing DVR into the interactive portion of the workflow. Level of detail techniques are commonly applied to enable interactive DVR of large-scale data, seeking to take advantage of the typically nonuniform salience of volumes.

The use of interactive transfer function design with salience-dependent level of detail selection creates a cyclic dependency. Interactive transfer function design techniques seek to enable discovery of interval volume salience, but also depend on interactive volume rendering. At the same time, interactive volume rendering on large-scale data using workstations depends on level of detail selection, which, to be effective, depends on knowledge of the salience of different parts of the volume. Figure 1a depicts this cycle.

*e-mail:martinst@cse.ohio-state.edu
†e-mail:hwshen@cse.ohio-state.edu

IEEE Symposium on Large Data Analysis and Visualization 2012
October 14 - 15, Seattle, Washington, USA
978-1-4673-4733-4/12/$31.00 ©2012 IEEE

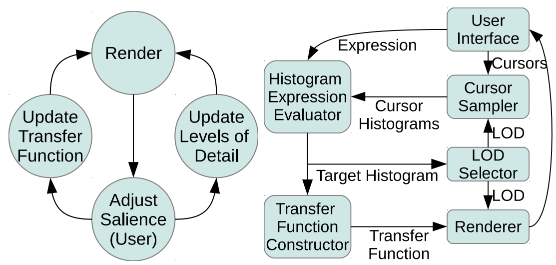
The core contribution of this work is a technique that reduces the impact of this cyclic dependency by enabling interactive, incremental construction of *target histograms* that can simultaneously be used to support transfer function construction and level of detail selection. Target histograms are used to drive the construction of both the transfer functions and the level of detail selection. This is accomplished by using *histogram expressions* to combine multiple local histograms into a target histogram. The target histogram is then used to generate a transfer function. Using Histogram Spectra [6], the target histogram is then also used to compute the optimal levels of detail for the multiresolution input volume. This enables interactive transfer function design in the context of DVR on data considerably larger than the available system memory.

2 Related Work

Due to its importance in direct volume rendering, transfer function design has been a widely explored topic. Pfister et al. [8] provide a comparison of a selection of trial and error, data-driven, and image-driven techniques. Kindlmann [3] extends this discussion to include feature detection based techniques. Fundamentally, it is unlikely that one type of technique will be appropriate for all applications. This paper concentrates on an interactive, data-driven approach intended to leverage the domain-specific knowledge of users.

Most similar to our transfer function design technique are selection-based techniques, where the user selects regions of interest and the technique generates transfer functions based on the selections. Wu et al. [11] describe an image-space technique to define which regions are important and which regions are not. They then apply a genetic algorithm to generate transfer functions that expose details in the important regions. This technique bears some similarity to our technique in that it enables salient and non-salient regions to be identified by the user, though ours operates in the data space and avoids a highly iterative technique like genetic programming in the interest of interactivity. Ropinski et al. [9] propose a technique in which users use mouse strokes to identify which regions belong to which material, making use of the ray histograms of those regions. This technique is similar to ours in that it allows for salience to be interactively specified by the user, though it does not provide a similar scheme for providing logical combinations of different regions and it relies on having some amount of pre-segmentation of the data. Similarly to our technique, Huang et al. [2] apply an approach using slice planes as a tool to help provide a context in which users can interactively select regions to guide the construction of transfer functions.

Level of detail selection has also been a long-explored problem, with many solutions attempting to maximize salient data visible (or minimize error) for the application of interest. Guthe et al. [1] and Wang et al. [10] both propose techniques that optimize LOD selections, in the context of in-core data, using screen space error metrics. Both of these techniques consider the final image, including visibility, thus they both also take into account the salience implied by the transfer function. Ljung et al. [4] and Martin et al. [6] both propose methods that perform LOD selection by utilizing precomputed metadata to minimize error with respect to a target distribution. The former work concentrates more on potential compression aspects of the problem, while the latter concentrates more on the optimization aspects and extensions to multivariate data. Our work focuses on a less-explored aspect: interactive transfer function design in the context of a workflow using level-of-detail selection on large-scale out-of-core data.



(a) Work flow (b) Data flow

Figure 1: Level of detail selection and transfer function design both depend on interval salience.

3 Technique

The fundamental goal of our technique is to facilitate the identification of interval salience on large-scale data. With interval salience, transfer functions can be constructed and levels of detail can be automatically selected. Figure 1a illustrates the basic workflow.

In our system, the user interactively changes four controls to construct a transfer function: a set of cursors, a set of slice planes, an expression for combining the distributions from each cursor, and the camera. The level of detail selection and transfer function are both updated on the fly using the first three controls. The view is re-rendered for changes to any of the controls.

The data flow for the system is shown in figure 1b. The user interface provides cursors to the cursor sampler. The cursor sampler then, using the current LOD selection, evaluates the histogram of the region within each cursor. These *cursor histograms* are subsequently combined using *histogram expressions* to compute a *target histogram*. The target histogram is then used to generate both an LOD selection and a transfer function. The LOD selection and transfer function are then used by the renderer to generate an image that is then passed to the user interface. The entire process can be interactive, even on a workstation with considerably less memory than the size of the dataset, as exhibited in section 4.

The result of this data flow is that level of detail selection and transfer function design are both directly driven by the target histogram, which is incrementally constructed by the user using the cursors and expressions. By incrementally constructing a transfer function, the user is also incrementally constructing a level of detail selection that produces good quality for a given working set size constraint. This increases usability of the transfer function design algorithm on large-scale data. Additionally, incremental construction can help users maintain a mental mapping between color and value during the interaction process.

3.1 Cursor Histograms

A *cursor histogram* is the histogram of the set of sample values for all points within a cursor. We chose the cursors to be circular discs within slice planes due to their simplicity, but other shapes or a sketching interface could be used. Cursors are moved and resized by clicking on the slice planes. Slice plane rotation, translation, and visibility can be changed with other UI elements.

The method used for generating cursor histograms is important, because a poor sampling pattern with a large number of bins and large gradients may yield aliased histograms, producing ineffective transfer functions. Two potential approaches for sampling these cursor regions are direct histogram computation assuming a trilinear interpolation function, and adaptive point sampling. We found an adaptive point sampling algorithm to be effective. Sampling on a uniform grid is used within each block, with the resolution of the sampling grid being proportional to the resolution of the block. The resolution of each block is adaptively chosen by the level of detail selection algorithm to minimize error subject to a global size constraint.

3.2 Histogram Expressions

Histogram expressions combine multiple *cursor histograms* into a single *target histogram*. A target histogram defines the importance of different value ranges. Value ranges with high probability in the target histogram are deemed salient. Conversely, value ranges with low probability in the target histogram are deemed unimportant. In a histogram expression, three operators are defined: disjunction, conjunction, and negation.

Operator	Histogram Expression	Result histogram bin k
Conjunction	$A \wedge B$	$\min(A_k, B_k)$
Disjunction	$A \vee B$	$\max(A_k, B_k)$
Negation	$\neg A$	$\max_{\forall i}(A_i) - A_k$

The disjunction operator is useful for combining two cursor histograms such that both of their histograms appear in the target histogram. The conjunction operator is used to combine two cursor histograms to find the bins that share high values in both histograms, implying a common importance. The negation operator is useful for expressing that the frequent values within a cursor histogram are unimportant, but the infrequent ones are important.

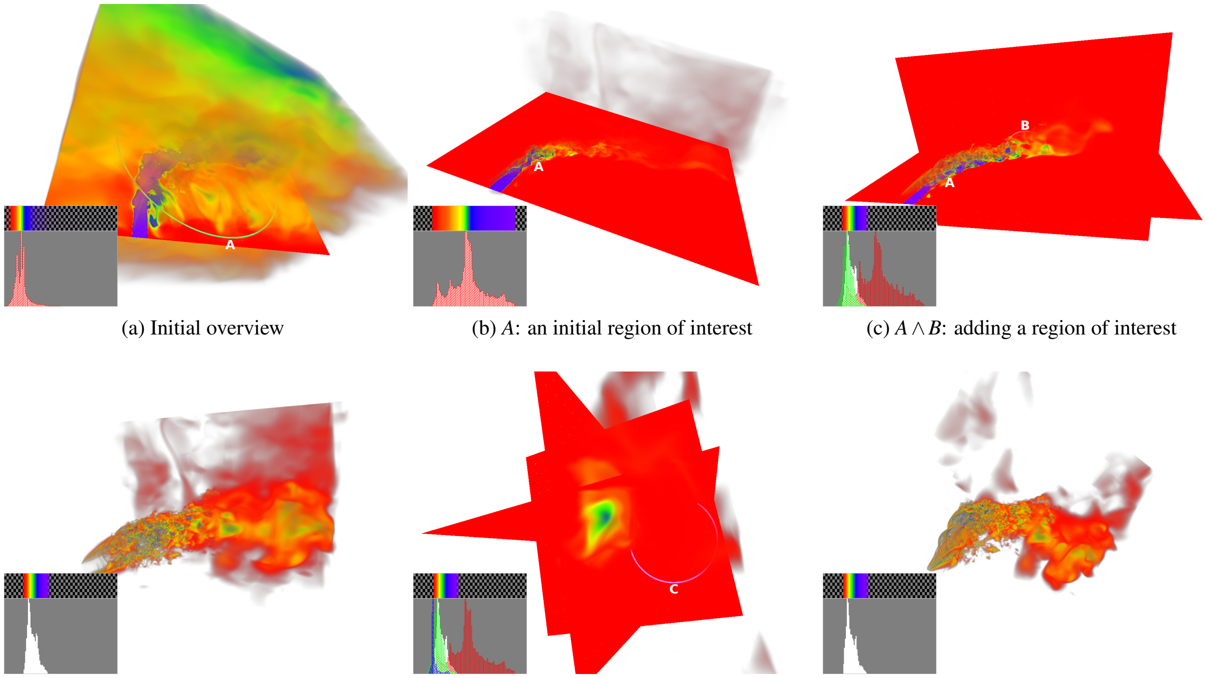
These operators can also be composed to form expressions, enabling the generation of target histograms using a combination of several cursors. For example, $D = (A \wedge B) \vee (B \wedge C)$ will combine three cursors into a single histogram, D. Bin values in D will be high only when they are high in B and A, or B and C. Intuitively, this kind of expression could be used to select two thin shells of values around a boundary region to explore its gradients and shape, filtering out background values and values immediately on the boundary.

3.3 Level of Detail Selection

Because the volume data is too large to fit in-core, level of detail selection is of critical importance. Our technique assumes the data is bricked into blocks of grid-centered cells, with each block having multiple levels of detail stored. Given a target histogram, we want to compute the level of detail that minimizes the error (maximizing the salient information available) for a given size constraint.

The technique described by Martin et al. [6] is used to compute an LOD selection. This technique stores metadata called Histogram Spectra. This metadata consists of a matrix, stored for each block, that contains the per-bin difference between the histogram of a block at a given level of detail and the ground truth histogram of a block computed at the maximum level of detail. For a given target histogram and a given LOD, this enables an estimate of the amount of salient information that has been lost as a result of downsampling. The metadata is generated during preprocessing, with the ground truth data only being needed for comparison during the preprocessing process. During the computation of level of detail selections in the interactive portion of the workflow, only this compact metadata needs to be accessed to estimate error, rather than needing to access the original volume. Using the data from this estimation algorithm, a greedy optimization algorithm is applied to compute the minimal error LOD selection subject to a user-defined working set size constraint. This enables fast, salience-aware, LOD selection for large out-of-core volumes even when interval volume salience changes within the interactive portion of the workflow.

The transient response of LOD selection algorithms is important in interactive workflows. Two aspects of this are data flow cycle convergence and working set stability. Data flow cycle convergence refers to the tendency for the LOD selection to converge to a single solution given a set of parameters, despite cycles in the data flow.



(a) Initial overview

(b) A: an initial region of interest

(c) $A \wedge B$: adding a region of interest

(d) $A \wedge B$: no slices, more opacity

(e) $A \wedge B \wedge \neg C$: reducing clutter

(f) $A \wedge B \wedge \neg C$: result

Figure 2: An example of the technique being applied to the Flame test volume, discussed in section 3.5

Working set stability refers to the magnitude of the change in the resulting working set for a given change in the target histogram.

In figure 1b, it can be seen that there are two cycles involving the level of detail selector, one completely automated, and one involving the user. In the automated cycle, the cursor sampler, histogram expression evaluator, and LOD selector are involved. When the LOD selection changes, it can affect the histograms sampled by the cursor sampler, which will change the target distribution, which may change the optimal LOD selection. In practice it was observed that the system converges within one or two iterations. This is reasonable because the potential change introduced in the target histogram by a change in LOD tends to be relatively small compared to the overall set of events contributing to the target histogram. In the cycle involving the user, all elements of the system are involved. However, the same property that allows the automated cycle to converge also allows the cycle involving the user to converge. In both cases, working set stability helps to contribute to fast convergence.

Working set stability affects both the cycle convergence and the number of transfers required from out-of-core to in-core for a given change in the target histogram. Consider the case that a small change is made to the target histogram. Because this is a small change, it will likely have a small influence on per-block salience, as computed with Histogram Spectra, used in the LOD selection algorithm. This reduces the chance that a large number of blocks will have different optimal LODs chosen. This stability was observed, and is reasonable, given that this interactive transfer function design technique involves incremental construction of transfer functions, where small changes tend to be made to the target distribution.

3.4 Transfer Function Construction

The transfer function is constructed by using a user-provided color ramp, opacity factor, and the target histogram. The general goal of the construction algorithm is to generate transfer functions that emphasize values with high frequencies in the target histogram, while deemphasizing values with low frequencies in the target histogram.

The opacity component of the transfer function, $T_a(u)$, is constructed such that it is linearly proportional to the target histogram, $H(u)$, for every value, u. A user-provided opacity factor is used as the coefficient of proportionality to adjust the compromise between clarity and occlusion. This approach is taken so that values with high frequency in the target histogram will be more strongly visible than those with low frequency.

The color component of the transfer function, $T_{rgb}(u)$, is constructed such that its contrast is linearly proportional to the target histogram, $H(u)$, for every value, u. This is accomplished by warping a user-provided color ramp. The contrast, $C(u)$, for a point, u, in the transfer function is defined as the color difference between a color in the transfer function at $u-h$ and at $u+h$, where h is a step size.

When using a linearly interpolated texture as a transfer function, using the width of one texel was found to be an effective step size. While many color difference metrics could be applied, we found that using the L2 norm of the difference between the colors in the CIE 1976 color space to be effective. We found this metric to be more effective in giving visually-intuitive results than using the L2 norm of the difference between the colors in the sRGB color space.

3.5 Interaction

An example interaction sequence of this technique on the Flame volume is shown in figure 2. Each cursor is assigned a letter which is subsequently used in the expressions used to compute the target histograms. Figure 2a shows the initial cursor the user is presented with, which provides a general overview of the dataset. In the next step (figure 2b), the user moves the cursor on the slice plane to a region that looks interesting, shrinking the cursor to focus on that region. That region is then grown by adding another cursor and applying the conjunction operator between it and the previous cursor. This exposes the common areas of interest between the two, allowing for greater opacity to be applied in the result as seen in figure 2c. Removing the slice planes, it can be seen in figure 2d that there is still some clutter in the background. This can be removed with another cursor and expression as shown in figure 2e, yielding the result in figure 2f. The entire process is interactive and allows for incremental exploration.

4 Results

The technique was implemented using C++ and CUDA, with CUDA being used for the volume rendering and cursor sampling. The test platform was a Linux workstation with 4 GiB of main memory, an NVIDIA GPU, and a 128 GB OCZ Vertex 2 SSD. The test implementation maintained the loaded volume data within GPU memory, while the standard Linux VM was allowed to manage the file cache. Between 100MiB and 800MiB of GPU memory was used for working set space.

Two multiresolution CFD datasets were used: the 14GiB Flame dataset shown in figure 2 and the 62GiB Nek dataset used to produce figures 3 and 4. Both of them were interactively manipulated on our test platform. Timings were analyzed to identify scalability as well as transient response.

Scalability was analyzed by performing an automated sequence of actions for different dataset sizes and different working set size constraints. The action sequence was similar to that used in figure 2, involving the movement of histogram cursors and the editing of histogram expressions. Figure 3 shows the results. The per-frame times depend primarily on the user-defined working set size, rather than the volume size. This is because the amount of data that can be possibly loaded for a frame, the number of samples that need to be taken for cursor histograms, and the number of samples that need to be taken for rendering all depend on the working set size rather than the volume size. The ratio of the working set size to the volume size affects the quality of the results, but does not strongly affect the frame times. Due to this relationship, it is possible for the user to apply the technique to very large data, even with fairly limited working set sizes.

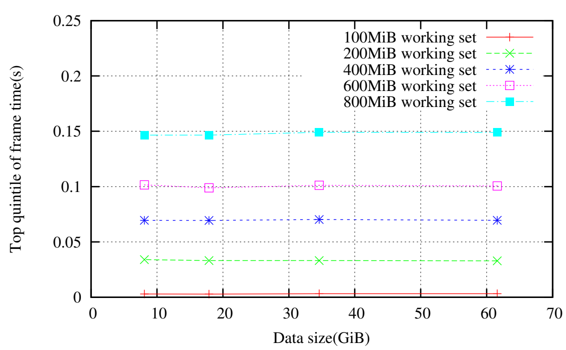


Figure 3: The performance as a function of volume size and working set size is largely a function of the working set size, rather than the volume size, facilitating scalability for large-scale data.

Figure 4 shows the typical amounts of time needed to process the various aspects needed for a given frame, as a function of the time since the program has started, using the same automated test procedure used to generate figure 3. The system file caches were flushed immediately prior to execution of the run, thus no volume data was resident at the start of execution and some modest initial loading is necessary. Importantly, it can be seen that the loading times required during interactions with the system are reasonable. This further reinforces the case that the technique is well conditioned – a small change in the input target histogram will tend to yield a small change in the resulting LOD selection. This facilitates interactive, incremental construction of interval value salience for transfer function design and level of detail selection.

5 Conclusion and Future Work

Histogram expressions combined with an interactive slice plane interface enable incremental, interactive construction of target histograms describing salience. These target histograms are then used to directly construct transfer functions and level of detail selections simultaneously, enabling interactive transfer function design on large-scale data.

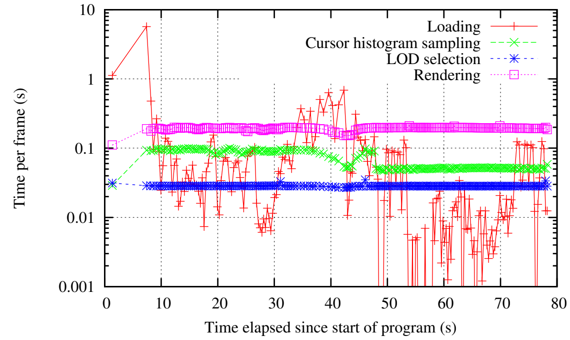


Figure 4: An example of the per-frame performance, as a function of running time, for a test run using the 62GiB Nek dataset with a 600MiB working set limit. In this case cursors are being moved around and expressions edited, yielding incremental updates to the target histogram.

The technique could easily be extended to support joint distributions between two variables, such as gradient magnitude and value, to enable more complex transfer function design techniques. Additionally, preservation of the mental mapping between color and value could be considered in transfer function construction for time-varying data. Finally, view-dependent transfer function construction and LOD selection could also be useful extensions.

References

[1] S. Guthe and W. Strasser. Advanced techniques for high quality multiresolution volume rendering. In *In Computers and Graphics (2004*, pages 51–58. Elsevier Science, 2004.

[2] R. Huang and K.-L. Ma. Rgvis: Region growing based techniques for volume visualization. In *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*, PG '03, pages 355–, Washington, DC, USA, 2003. IEEE Computer Society.

[3] G. Kindlmann. Transfer functions in direct volume rendering: Design, interface, interaction. In *SIGGRAPH 2002 Course Notes*, 2002.

[4] P. Ljung, C. Lundstrom, A. Ynnerman, and K. Museth. Transfer function based adaptive decompression for volume rendering of large medical data sets. In *Proceedings of the 2004 IEEE Symposium on Volume Visualization and Graphics*, VV '04, pages 25–32, Washington, DC, USA, 2004. IEEE Computer Society.

[5] C. Lundstrom, P. Ljung, and A. Ynnerman. Local histograms for design of transfer functions in direct volume rendering. *Visualization and Computer Graphics, IEEE Transactions on*, 12(6):1570 –1579, nov.-dec. 2006.

[6] S. Martin and H.-W. Shen. Histogram spectra for multivariate time-varying volume lod selection. In *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*, pages 39 –46, oct. 2011.

[7] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, June 1995.

[8] H. Pfister, B. Lorensen, C. Bajaj, G. Kindlmann, W. Schroeder, L. S. Avila, K. Martin, R. Machiraju, and J. Lee. The transfer function bake-off. *IEEE Comput. Graph. Appl.*, 21(3):16–22, May 2001.

[9] T. Ropinski, J.-S. Praßni, F. Steinicke, and K. H. Hinrichs. Stroke-based transfer function design. In *IEEE/EG International Symposium on Volume and Point-Based Graphics*, pages 41–48. IEEE, 2008.

[10] C. Wang, A. Garcia, and H.-W. Shen. Interactive level-of-detail selection using image-based quality metric for large volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(1):122–134, Jan. 2007.

[11] Y. Wu and H. Qu. Interactive transfer function design based on editing direct volume rendered images. *Visualization and Computer Graphics, IEEE Transactions on*, 13(5):1027 –1040, sept.-oct. 2007.