A Smart Compression Scheme for GPU-Accelerated Volume Rendering of Time-Varying Data

Yi Cao, Guoqing Wu, Huawei Wang
High Performance Computing Center
Institute of Applied Physics and Computational Mathematics
Beijing, China
{cao_yi, wu_guoqing, wang_huawei}@iapcm.ac.cn

***Abstract*—The visualization of large-scale time-varying data can provide scientists a more in-depth understanding of the inherent physical phenomena behind the massive data. However, because of non-uniform data access speed and memory capacity bottlenecks, the interactive rendering ability for large-scale time-varying data is still a major challenge. Data compression can alleviate these two bottlenecks. But just simply applying the data compression strategy to the visualization pipeline, the interaction problem can not be effectively solved because a lot of redundant data still existed in volume data. In this paper, a smart compression scheme based on the information theory is present to accelerate large-scale time-varying volume rendering. A formula of entropy was proposed, which can be used to automatically calculate the data importance to help scientists analyze and extract feature from the massive data. Then lossy data compression and data transfer is directly operated on these feature data, the remaining non-critical data was discarded in the process, and GPU ray-casting volume render is used for fast rendering. The experiment results shown that our smart compression scheme can reduce the amount of data as much as possible while maintaining the characteristics of the data, and therefore greatly improved the time-varying volume rendering speed even when dealing with the large scale time-varying data.**

Keywords-volume rendering; visualization; time-varying data; information visualization

I. INTRODUCTION

Three-dimensional data visualization has been widely used in the field of science and engineering. However, to enhance the high scientific confidence in numerical simulation, the physical modeling must be scientific and the method must be precise. That will bring thousands of times increasing of the simulation computing, make a substantial increase in the output size of simulation data, and eventually resulting in the huge bottleneck in the visualization side.

For example, three-dimensional laser filament simulation program, 512 processor cores are used to simulate the physical model with 512×512×1024 grid cells. The output data size of each time-step can reach 5GB(Gigabytes), if the output time-step number is 200, then the total data capacity will be 1TB (terabytes). To better understand the simulated data, and find the physical information behind data, scientists need to interactively visual analysis the large-scale time-varying data, which need for the full visualization pipeline speed no less than 5 frames per second. The whole pipeline speed is relative to the concept of time-varying visualization, unlike the single-time data visualization, which includes data uploading, data processing and data rendering stages. The GPU-based hardware-accelerated volume rendering algorithms can significantly reduce the rendering time, and most of the data processing can advance to the pre-process stage and not affect the pipeline speed, so the data load process in the pipeline highlights out, with the main factors affecting the interactive speed.

Data uploading involve the data transfer between multi-level storage structure from disk to the CPU and GPU, and there is a huge difference within storage capacity and access bandwidth when accessing between different hardware. For example: The memory capacity of the current mainstream consumer PCs whose hard disk capacity is 1TB, main memory size is 4GB and video memory is generally 1GB. From the disk to the GPU, the performance of data storage capacity is in descending order of relationship. On the other hand, GPU on-chip copy can reach 100GB/s. To the data transfer from CPU to GPU via the PCI Express 2.0 16x motherboard bus, the one-way transfer bandwidth in theory can reach 8GB/s, but the actual bandwidth we can achieve is usually more than an order of magnitude slower at 8GB/s. The disk access speed is the slowest, for example: the theoretical throughput in RAID 0 mode is 220MB/s, while the actual throughput only can achieve around 100MB/se. Analysis shows that the bandwidth between the mass storage and CPU is an order of magnitude less than that between CPU and GPU. Thus the closer to the end of the disk, the data bottleneck will be more significant. In order to enhance the whole visualization pipeline speed, the data transfer bottlenecks that exits in the multi-level storage structure need to be resolved.

To speed up data transfer between mass storage and GPU, data compression is most commonly used. The usual practice is compressing the entire data field, but scientists often do not necessarily need a global data processing during their research process. Scientists prefer to locate in the data field of interest sub-region, observe data in these regions with time-step evolution, and then get a deep understanding of these simulation results. Scientists often use this research model to feedback and guide them for the parameters modification and verification of their physical model. These data areas targeted by the scientists generally contain a lot of physics knowledge, so compared to other sub-regional data,

978-0-7695-4602-5/11 $26.00 © 2011 IEEE
DOI 10.1109/ICVRV.2011.56

they are more important. The traditional positioning methods have significant limitations, such as: approach which makes use of slices to find the interesting sub-regional often involve a large number of manual operation. When faced with the large-scale dataset, this can even lead to failure to locate, or ignore some important sub-region. Therefore, if the program can automatically find these important areas of dataset during the processing stage, and separate these importance data from the non-concerned and non-noise data. It will help us to filter the data properly and reduce the amount of data as much as possible, while maintaining the characteristics of the data. Hence, the bottlenecks pressure can be reduced, and improve the rendering speed of whole visualization pipeline. Therefore we present a smart compression scheme, which can automatically locate sub-region which physicists are interested in, and guide the data compression only to perform on these regions containing feature. Comparing to the traditional compression strategy, the algorithm we proposed can achieve higher data compression ratio and can provide a more interactive volume rendering performance for large-scale data.

II. Related Work

The volume rendering of GPU-based hardware-accelerated algorithms can significantly reduce the data rendering time. But when the algorithm is simply applied to the large-scale data visualization, the memory capacity bottlenecks and data transmission bottleneck, rendering speed may be substantially reduced or even fail to rendering. In this situation, many traditional compression algorithms can be use to reduce the pressure of the bottleneck. For example, lossless compression algorithm RLE and LZO [1], and lossy wavelet compression algorithms, etc. Data compression can also reduce the GPU memory capacity bottlenecks, but due to the hardware differences between the CPU and GPU, many commonly used compression algorithms on the GPU is still unable to perform, not all compression algorithms can be applied to GPU.

GUTHE et al. [2] proposed hierarchical wavelet for volume data compression to support GPU hardware-accelerated volume rendering, where CPU performs the data compression and decompression stage. FOUT and Ma [3] introduced a compressed volume render using vector quantization, which can perform the data decompression on the GPU side. The DXT texture compression[4], known also as S3TC (S3 Texture Compression), is a lossy algorithm based on block truncation coding, which is designed specifically for modern graphics hardware and can provide the real-time data decompression capability during rendering. The GPU-accelerated DXT compression and decompression is used to fast volume rendering large dataset by [5, 6]. The two-stage compression scheme combining CPU and GPU is also presented in [5], where the LZO compression is performed with CPU. The two-stage compression scheme can effectively reduce the bottleneck of data streaming from disk to memory for the twice compression. The above methods are all belong to uniform compression mode, but the distribution of data value is often associated with the physical properties, Therefore, if use the non-uniform partition mode instead of the uniform mode, we can not only better maintain the characteristics of data, but also can achieve higher data compression ratio.

To non-uniform compress the volume date based on the value distribution over the volume, JORG et al. [7] presented a variable-length coding approach for effective volume compression. When the value range of some bricks is smaller than the value range of the entire volume, the whole brick is reduced by storing a base value and the difference from the base for each voxel. The variable-length decoding can perform on GPU during rendering. PATRIC et al. [8] proposed a solution to reduce data by user defined transfer function. When a transfer function is applied, large subsets of the volume will give little or no contribution to the rendering. Then they can make use of the knowledge encoded in the transfer function to select a level-of-detail that reduces the data for data reconstruction and rendering when using wavelet-based volume data compression. [9] Using texture packing and compression of sparse volume data to speedup time-varying volume rendering. HIROSHI and MA [10] discard data blocks with values outside the data interval of interest, where the uninteresting data values are assigned transparent opacities similar to the editing of an opacity transfer function.

CHAO et al. [11] introduces an importance-driven approach based on the formulation of conditional entropy from information theory. The different temporal trends exhibited by importance are used to reveal the important aspects of time-varying data. CHAO [12] further utilize the importance values to lower the data streaming from mass storage to memory. They adopt non-uniform quantization coding based on the data range and the importance value which can be simple decoding on GPU side. Both [12] and [7] also adopt the two-stage compression approach hybrid CPU and GPU in their processing pipeline, the large-scale time-vary can then be more effectively handled by volume rendering. Similar to [5, 6, 7, 12], we take the two-stage compression scheme in process pipeline. The difference in our work is we utilize the importance analysis method to discard the un-importance data in the volume assisted the domain knowledge. This process is viewed as a lossy compression process in our work. Unlike the former works, we jump these discard areas in volume space during data compression and data uploading, Instead of coding them with a little bits or packing them. Hence we can further reduce the data transfer streams while keeping the physical phenomenon behind data feature.

III. Information Theory

A. Information Assisted Smart Visualization

In order to find the interested data, the primary job is to quantitatively describe the characteristics of data, and the information theory has already provided an effective theory approach for that purpose. For scientific data sets, we can use the information entropy to quantify the inherent physical characteristics of the data. Entropy is a random variable or a measure of the information system, therefore, in order to evaluate that how much information is concerned by scientist

in the data set, we can calculate the entropy of the data set. The greater the entropy inherent in the data set, the more extensive information will be there, hence the more attention physicists should pay to it. From the concept of entropy, we can see that information entropy only relates to probability distribution, that is, the number of information contained in the data is only related to the numerical value distribution of the data. The evaluation of the amount of information contained in data set, that is, the process of irregular value distribution investigation. The more irregular data distribution, the greater the entropy, the data sets are more likely to contain important physical characteristics. On the contrary, if all the data values in dataset are identical, the entropy will then be zero. In this case, the data set does not contain characteristics, and the data set is equivalent to a blank data.

B. Entropy of Dataset

Entropy is a function of the probability distribution from the statistical point of view, which quantitatively describes the amount of information, irregular or complex of scientific dataset. Relative to the mean, extreme values, variance and other low-level statistics, it contains more information about the data set, so it can depicts the characteristics of the data more comprehensively.

Let x be a discrete random variable, range $\{x_1, x_2, \ldots, x_n\}$, corresponding probability distribution for the $\{p(x_1), p(x_2), \ldots, p(x_n)\}$, by $0 \le p(x_i) \le 1$, $\sum_{i=1}^{n} p(x_i) \log p(x_i)$. Values $I(x_i) = -\log p(x_i)$ is used to measure the amount of information provided x_i, called "self-information", Introduction to information from the average or mathematical expectation as entropy, denoted as:

$$H(X) = \sum_i p(x_i) \log p(x_i) \tag{1}$$

Where $H(X)$ is the entropy of random variable X, which value is a measure of the random variable measure of the uncertainty, and the information contained in the measure of random variable.

C. The Histogram Approach for Entropy Designment

Information measure is defined on the probability space. In this article, we use the histogram method to establish the probability distribution of scientific data. The main idea is to divide the data range of scientific dataset into several small boxes, use the statistical analysis of the data distribution to count the number of data down into each small box, and then we can approximate the probability distribution of the real probability distribution of the scientific data set.

Assuming a small the K th box as $B_k = [t_k, t_{k+1})$, $v_k = \#\{x \in [t_k, t_{k+1})\}$ is the sample size that fall into the K th box, a small box number is L, then the histogram density is defined as:

$$p(x) = \frac{v_k}{L(t_{k+1} - t_k)}, x \in B_k \tag{2}$$

In most practical applications, usually the uniform quantization, $t_{k+1} - t_k = h$, where h is quantization interval width, This estimate can be simplified histogram is expressed as:

$$p(x) = \frac{v_k}{Lh}, x \in B_k \tag{3}$$

Among them, the histogram density estimate only depends on the width h of quantization intervals, and when quantitative series $L \to \infty$, the resulting probability distribution approaching real smooth out the probability of infinite density.

D. Brick Importance of Dataset

To calculate the sub-regional importance, we define the importance of the i th sub-area D in the i th time step as follows:

$$A_i^t = \frac{H(D_i^t)}{\max_i(H(D_i^t))}, 1 \le i \le m \tag{4}$$

Where, m is the number of the region i th time steps, $0 \le A_i^t \le 1$. Obviously, when a sub-area contains a wealth of physical knowledge, the information entropy of that area will be larger. Meanwhile, the importance of that sub-region will also be larger than the others. We can sort all the partitioned sub-area in time-step accordance with the importance value or set the threshold value ε, which can be used to filter out the sub-region with the lower importance. When the total time overhead is fixed, the sub-regions with the more importance value has a higher priority in the allocation of computing resources and time slots, and then performs the high-resolution rendering. For the less important sub-region, we perform the low-resolution rendering, or even give up rendering.

IV. The Smart Compression Scheme

A. The Algorithm overview

Fig.1 shows our volume rendering flow chart based on the smart compression scheme. The entire process hybrid using CPU and GPU, which can take advantage of their respective advantages, hence will speed up the whole pipeline speed. The top part of Fig.1 is the pre-processing stage, it is executed before the real-time rendering stage, and its mission is to handle and filter the original time-varying data field. The efficient storage format can reduce the bottleneck pressures of data transmission during real-time rendering. The time-varying data field is divided into sub-blocks according to some rules at first, and an index file is used to record the field data file name of all sub-blocks. In the field of science and engineering data, the data is usually saved as single precision floating point type. However, the 8 bit color depth graphics can meet the demand of our visualization applications. Therefore, data quantization operation is also carried out in pre-processing stage, the data capacity can then be reduced four times. The entropy-based data analysis is performed toward all time-step data, the result is that each sub-block will be marked on the

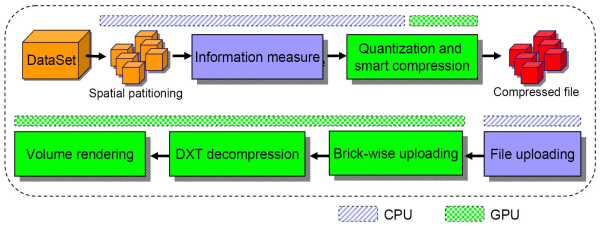


Figure 1. The flowchart of smart compression and rendering pipeline.

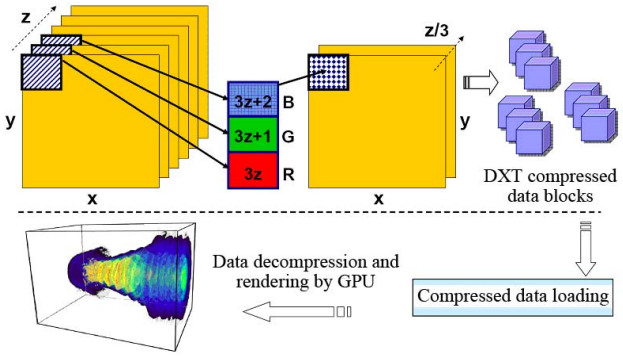


Figure 2. The GPU volume rendering with DXT compression data.

importance value. The importance value can be used to reflect the distribution of the data characteristics in the dataset, and help scientists to visual analysis data with domain knowledge. Data partition, entropy calculations and data quantization process are all done by the CPU. Each sub-block meeting the threshold of the importance value is finally stored to a disk file with compression format. The bottom part of Fig. 1 is the real-time rendering stage. The compressed block data is firstly loaded by the CPU, and then is uploading from main memory to the GPU texture memory. After all the data sub-blocks meet the threshold of importance value are uploaded on the GPU, the volume rendering started. The data decompression is performed by GPU during voxel data sampled.

B. Two-Stage Compression

The smart data compression which mentioned in this paper is based on automatic data analysis guidelines, which can keep the data characteristic during data compression. For the sub-blocks of data which not reflect the data characteristics, data compression processing will skip them. This data discard strategy based on importance value, we called it the first stage compression. The really DXT format data compression target on those feature data is called the second stage compression. For each sub-block data meet the threshold of importance value, the pipeline process will quantities and compress it, otherwise the sub-block will be discarded. In the first compression stage, the data analysis and the importance value calculation is based on the the information theory mentioned in chapter 3 using CPU. In the second compression stage, the data compression of DXT format is performed using GPU. The DXT texture compression, known also as S3TC (S3 Texture Compression), is a lossy algorithm based on block truncation coding, which is designed specifically for modern graphics hardware and can provide the real-time data decompression capability during rendering. The compressed data blocks with DXT format have the compression ratio of 6:1Equations

C. Spatial Partitioning of Volume Data

The subdivision intended mainly includes the following aspects: Firstly, the sub-block based processing strategy can effectively avoid the bottleneck caused by the large amount of data storage and data transfer. This is especially important when doing pre-processing for the large-scale raw data, because even single-time data may not be read into memory once. Secondly, the information theory based data analysis also require the data set is divided into sub-blocks, thus making it easy to calculate the distribution of the importance value, and assigns the value of the importance to each sub-block. Thirdly, most of the output data from the parallel numerical simulation has already been split, the split is based on domain knowledge of data modeling. For the time-varying data has been divided into sub-blocks in the numerical simulation phase, we will directly follow the sub-block partitioning strategy for visualization. Form a single block data, we need to split it in the volume space. A lot of space partitioning strategy can be used here, such as the uniform subdivision, octree based subdivision and non-uniform subdivision, and so on. In this paper, we use the uniform subdivision strategy.

D. Decompression and Rendering

The data decompression is executed by GPU during volume rendering stage. As the light continues to advance in the data field, GPU is also needed to continuously perform decompression operation to obtain samples around the grid cell data for the subsequent interpolation operation. During the three-dimensional volume rendering, although this will cause a lot of repetitive data decompression operation, but thanks to the high-speed decompression performance of GPU hardware, these repeated decompression will not have significant performance impact.

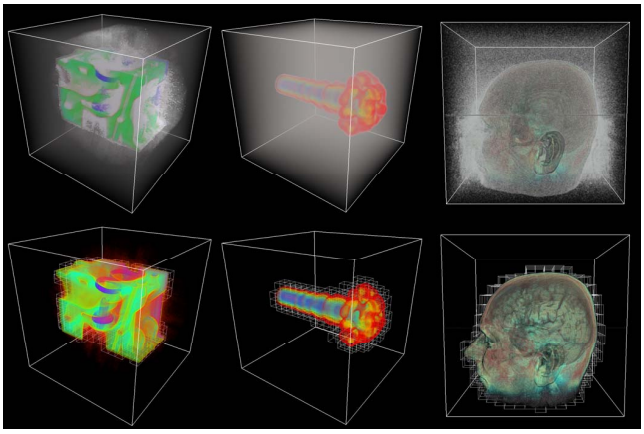
In contrast to the original volume data where each volume elements is represent as a physical variable value, each new element of the packing volume data contains three sub-elements, which correspond to the RGB channels in OpenGL external data format. The packing only change the data filling order in direction of z-axis. Hence every three elements in the original volume data will be filled in the RGB channels of a new volume element in turn. As a result, the grid resolution in z-axis of the new constructed volume data approximately equals to 1/3 of the original grid resolution. Therefore, the data decompression concerned about the correctly unpack of the DXT compressed data from the RGB format using GPU, shown in Fig.2, and then GPU will quickly submit these unpacked data to the graphics processing pipeline in order to meet the needs of volume rendering.

V. RESULTS AND DISCUSSION

To verify the entropy algorithm proposed in this paper, three volume data, including science, engineering and medical science dataset, has been used for experimental test. The top row of Fig.3 shows the distribution of information within the data field in which meaningful data and noise data and non-interest data is mixed together. In this case, the inappropriate transfer function will cause people to misunderstand the data field. We first split the volume data into small homogeneous sub-block, and then use the entropy formula proposed in section 3 to calculate the importance of each sub-block. The white bounding boxes draw in bottom row of Fig.3 represents the sub-blocks that meet the threshold of importance value. Fig.3 (a) shows that when the important value of sub-blocks is greater than or equal to the threshold 0.45, these sub-blocks can accurately describe the engine area. When the meaningful Engine data is just limited by the bounding box, the ratio of meaningful sub-blocks reached 15.9%.

To test the performance of the smart compression scheme for time-varying volume rendering proposed in this paper, a numerical datasets have be used. The dataset is generated in a simulation program of laser filament, the grid resolutions of this datasets is 512×512×1024, and the dataset contains 200 time steps and a single physical variable represented by the single-precision float type. The compressed volume rendering algorithm is tested on a PC equipped with an Intel Core2 Duo processor and NVIDIA Geforce GTX 260 graphics card, where the video memory is 896MB and the operating system is Red Hat Enterprise 5.0.

Base on the entropy formula proposed in section 3, we can calculate the importance value for each sub-block, and then get the volume rendering results. For each time-step data in our test time-varying dataset, the distribution of importance value is basically the same. So we can use the importance value of one time step data to unified descript the importance distribution of the other time-step data. The suitable time-step is usually selected by the scientists with domain knowledge, the general physical state of that time-step has evolved to a stable state.



(a) Engine (b) fuel (c) MRI-Head

Figure 3. The volume rendering results of three volume data. Top row: the meaningful data mix with the noise data and non-interest data, Bottom row: the sub-blocks that meet the threshold of importance value can accurately describe the meaningful area (white bounding box area).

TABLE I. RESULTS OF THE SMART COMPRESSION SCHEME

method	data size(MB)	Compression time(s)	Disk-CPU(s)	CPU-GPU(s)	time-step fps
Quant.	256	5.530	0.335	0.227	1.1
DXT	42.7	10.164	0.060	0.163	3.5
Smart	2.6	21.526	0.004	0.016	14

The detailed analysis of the various parts of our algorithm performance is shown in table 1. Three compression algorithms are used in the performance test, including the quantization from 32-bit to 8-bit, DXT compression, and the smart compression. Among them, the DXT compression process includes quantization process, and the smart compression process includes both quantization and DXT compression process. The time-varying volume rendering result of the large-scale laser filament simulation is shown in Fig.4. At time step 34000, the physical evolution has reached a steady state. At this point, the meaningful sub-block ratio only reached 6.7%, a small part of the data space is used for numerical simulation. Based on the smart compression scheme we present, the time-varying rendering performance can be greatly improved. When the threshold of importance value is taken as 0.1, the area of the sub-block meets the threshold and the meaningful physical area just coincidence.

From table 1, we find the test bandwidth between Disk-CPU is higher than the theoretical value, reaching 700 MB/s. This is because the disk caches the recent used dataset. The bandwidth between CPU-GPU is lower than the theoretical value, reaching 1GB/s. Thus, instead of the disk read, the GPU uploading became the largest bottleneck. Compared with the DXT algorithm, smart algorithm can improve the data transmission speed an order of magnitude. As a result, the large-scale laser filament data can smoothly achieve 14 fps during time-step volume rendering using our smart compression scheme, which meets the needs of interactive rendering. The original time-step rendering speed of the laser filament data without optimization can only reach 1.1 fps, which does not have the interactive performance.

VI. CONCLUSION

In this paper, we proposed a smart compression scheme based on information theory used to accelerate the large-scale time-varying volume rendering. By using the entropy formula proposed in section 3, the program can automatically calculate the importance of each sub-block in volume, hence help scientists analyze and extract feature from the massive data. The two-stage compression is utilized for non-critical data discarding and lossy data compression, and the subsequent data transfer is only operated on these feature data in the visualization pipeline, so the bandwidth bottleneck can be greatly eased. The experiment results shown that our smart compression scheme can reduce the amount of data as much as possible while maintaining the

characteristics of the data, and therefore greatly improved the time-varying volume rendering speed even when dealing with the large scale time-varying data.

ACKNOWLEDGMENT

This research was supported by the National Natural Science Foundation of China NO.61033009, and the Science and Technology Funds of CAEP under grant NO. 2010B0403057.

REFERENCES

[1] Oberhumer M. F. X. J, "LZO real-time data compression library", http://www.oberhumer.com/opensource/lzo, 2008.

[2] Guthe S., Wand M., Gonser J. and Straber W, "Interactive Rendering of Large Volume Data Sets", Proceedings of IEEE Visualization '02, IEEE Computer Society Press, pages 53–60. 2002.

[3] Fout N. and MA K.-L, "Transform coding for hardware-accelerated volume rendering", IEEE Transactions on visualization and Computer Graphics, volume 13, pages 1600-1607. 2007.

[4] Pat Brown. EXT texture compression s3tc. http://opengl.org/registry/specs/EXT/texture_compression_s3tc.txt.

[5] Nagayasu D., INO F. and Hagihara K, "A decompression pipeline for accelerating out-of-core volume rendering of time-varying data", Computers & Graphics, volume 32, pages 350–362. 2008.

[6] Yi Cao, Li Xiao, Huawei Wang, "Hardware-accelerated Volume Rendering Based on DXT Compressed Data Sets", Proceedings of Audio Language and Image Processing (ICALIP 2010), Shanghai University Press, pages 523 – 527. 2010.

[7] J. Mensmann, T. Ropinski and K. Hinrichs, "A GPU-Supported Lossless Compression Schemefor Rendering Time-Varying Volume Data", in R. Westermann and G. Kindlmann, editors, IEEE/EG International Symposium on Volume Graphics '10, 2010.

[8] Patric Ljung, Claes Lundstr¨om, Anders Ynnerman and Ken Museth, "Transfer Funtion based adaptive decompression for Volume Rendering of Large Medical data Sets", IEEE Symposium on Volume Visualization and Graphics '04 (Austin, Texas, USA, October 11-12, 2004), pages 25-32. October 2004.

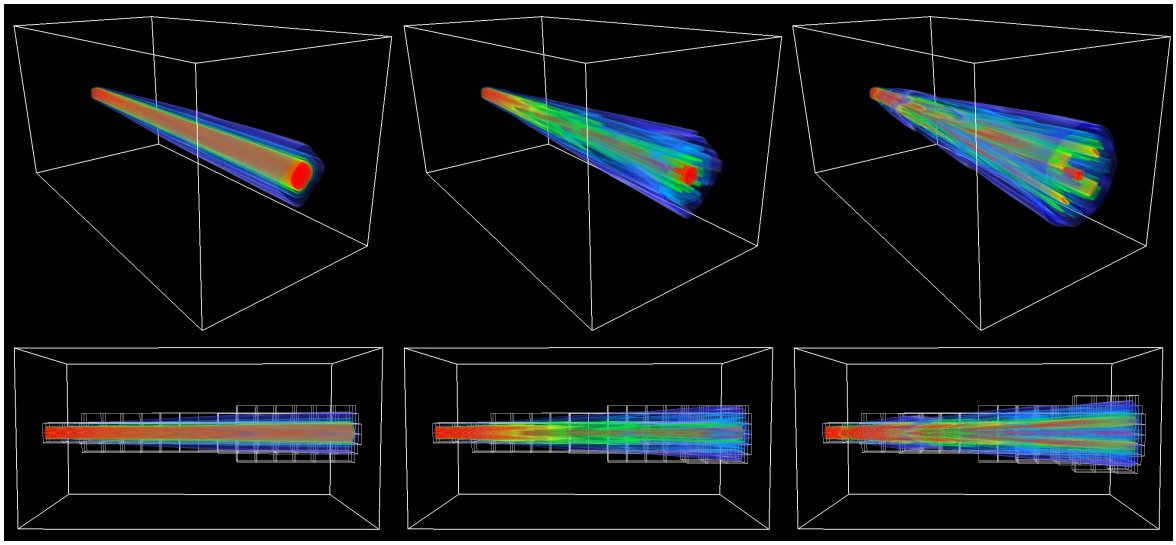
[9] A. P. D. Binotto, J. L. D. Comba and C. M. D. Freitas, "Real-time volume rendering of Time-Varying Data using a Fragment-Shader Compression Approach", IEEE Symposium on Parallel and Large-Data Visualization and Graphics '03 (Seattle, Washington, USA, October 20-21, 2003), pages 69-75. October 2003.

[10] Hiroshi Akiba, Kwan-Liu Ma. and John Clyne, "End-to-End Data Reduction and Hardware Accelerated Rendering Techniques for Visualizing Time-Varying Non-uniform Grid Volume Data", Volume Graphics, 2005.

[11] Chaoli Wang, Hongfeng Yu and Kwan-Liu Ma, "Importance-Driven Time-Varying Data Visualization", IEEE Transactions on Visualization and Computer Graphics, volume 14, pages 1547-1554. 2008.

[12] Chaoli Wang, Hongfeng Yu and Kwan-Liu Ma, "Application-Driven Compression for Visualizing Large-Scale time-varying Volume Data", IEEE Computer Graphics and Applications, volume 10, pages 59-69. 2010.

[13] IvanViola, Armin Kanitsar and Meister Eduard Groller, "Importance-Driven Volume Rendeimg", Proceedings of IEEE Visualization '04, 2004.



(a) time-step 10000 (b) time-step 18000 (c) time-step 34000

Figure 4. The volume rendering results based on the smart compression scheme. Top row: time-varying volume rendering of laser filament dataset. Bottom row: the sub-blocks that meet the threshold of importance value can accurately describe the meaningful area (white bounding box area)