Parallel visualization of gigabyte datasets in GeoFEM

Issei Fujishiro (1) (2), Li Chen (2), Yuriko Takeshima (3) ,
Hiroko Nakamura (1), and Yasuko Suzuki (1)

(1) Department of Infromation Sciences, Ochanomizu University, Tokyo, Japan (e-mail: fuji@is.ocha.ac.jp; phone: +81-3-5978-5700, fax: +81-3-5978-5705). (2) Research Organization for Information Science & Technology, Tokyo, Japan (e-mail: chen@tokyo.rist.or.jp; phone: +81-3-3436-5271, fax: +81-3-3436-5274). (3) Institue of Fluid Science, Tohoku University, Sendai, Japan (e-mail: yuriko@ifs.tohoku.ac.jp)

Abstract

An initial overview of parallel visualization in the GeoFEM software system is provided. Our visualization subsystem offers many kinds of parallel visualization methods for the users to visualize their huge finite element analysis datasets for scalar, vector and/or tensor fields at a reasonable cost. A polygonal simplification scheme is developed to make more efficient, the transmission and rendition of output graphic primitives. A salient feature of the subsystem lies in its capability in automatic setting of visualization parameter values based on the analysis of scalar/flow field topology and volumetric coherence, to improve the quality of visualization results with a minimized number of batch re-executions. Representative experimental results illustrate the effectiveness of our subsystem.

Introduction

The numerical simulation and analysis in engineering and science disciplines usually consist of three main stages: generating computational grids; solving physical equations; and visualizing the resulting data. With the rapid development of computer hardware and software, the computational grid is becoming more and more complicated, and the amount of dataset produced by routine computations is becoming larger and larger. Therefore, it is very important to develop fast and effective visualization techniques and systems for very large datasets.

GeoFEM is known as a large-scale finite element analysis platform for solid earth simulation (Okuda, 2000[1]). The goal of its visualization subsystem is to provide the GeoFEM users with an interactive visual exploration environment for various types of volumetric datasets arising from its analysis subsystems. Visualizing large-scale datasets is a very challenging topic because it is very difficult to obtain sufficient information from such datasets at a reasonable cost. Our visualization subsystem mainly focuses on the following two aspects: one is to extend the parallel performance of the existing visualization algorithms to improve its speed, and the other is to augment the algorithms by exploiting the use of auxiliary feature-based data context analysis.

In order to choose and develop an effective set of visualization techniques, an advance analysis of functional requirements was performed with an aid of Hesselink et al's taxonomy of visualization techniques (Hesselink, 1994[2]). Their taxonomy attempts to classify existing visualization techniques based on the following three orthogonal aspects - the order of target datasets (scalar, vector, tensor); the level of information to be visualized (elementary, local, global); and the spatial dimensionality of visualization primitives (point, line, surface, volume). The original taxonomy is modified slightly herein to enhance its applicability by adding "bitmap", "solid" and "glyph" to the primitives. We selected major techniques currently available for GeoFEM visualization applications, as listed in Table 1.

Table 1. GeoFEM taxonomy of visualization techniques.

Techniques	Order of data	Information level	Vis. primitive
Isosurface fitting	Scalar	Elementary	Surface
Solid fitting	Scalar	Elementary	Solid
Volume rendering	Scalar	Elementary	Point
Arrow plots	Vector	Elementary	Glyph
Particle advection	Vector	Elementary	Point
Stream objects	Vector	Elementary	Surface
Probe	Vector	Local	Glyph
LIC	Vector	Local	Bitmap
Topological map	Vector	Global	Point&line
Hyper-streamlines	Tensor	Local	Surface

In this paper, we first describe some kinds of parallel visualization modules for scalar, vector, and tensor fields which we have developed in GeoFEM. A polygonal simplification scheme for making more efficient, the transmission and rendition of output graphic primitives is also presented. Particular focus will be placed on a salient feature of the subsystem, i.e., its capability in automatic setting of visualization parameter values based on the analysis of scalar/flow field topology and volumetric coherence, to improve the quality of visualization results with a minimized number of batch re-executions. Representative experimental results with GeoFEM simulation datasets illustrate the effectiveness of our subsystem.

Parallel visualization techniques for large-scale datasets in GeoFEM

Parallel visualization frame in GeoFEM

Much work has been done on parallel visualization architecture (Haimes, 1994[3], Doi, 1997[4]). The framework of our parallel visualization subsystem is schematically overviewed in Figure 1. We adopted a parallel architecture which intends to perform visualization concurrently with numerical calculation on the same high-performance parallel computer, followed by transmission to clients, graphic primitives rather than resulting images. On each client, the users are allowed to modify interactively the parameters related to viewing, illumination, and shading. The resulting images are displayed by the *GPPView* viewing software, which is also developed by the GeoFEM group. On the computational server, the users are only required to specify in the batch-control files, the list of visualization methods for use plus necessary parameter values. Since both the visualization and calculation processes are performed on the same computer at the same time, we need not retain the computational results on the disk, which can avoid the limitations of storage capacity for large-scale datasets. Meanwhile, we can make full use of huge memory space installed in the computational server to achieve sophisticated visualization. For the purpose of improving the throughput of the entire processing under the limitations of communication bandwidth between the server and a client, and the client's memory and storage space, we developed a specific geometric decimation module to reduce to some degree the size of the output graphic primitives.

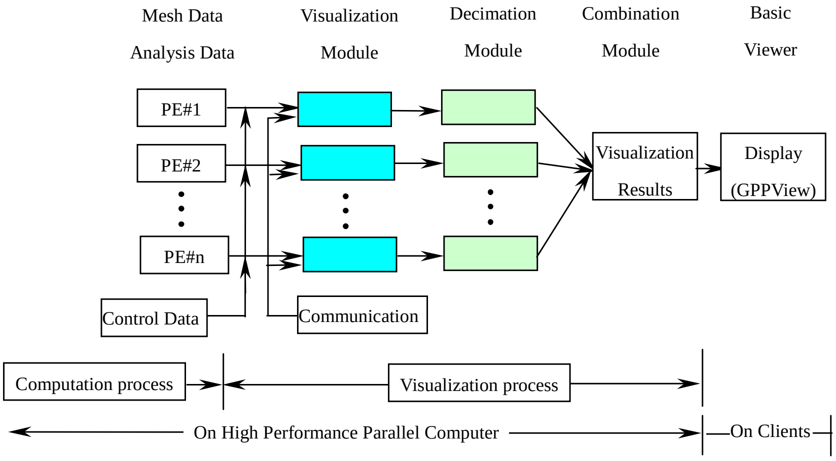


Figure 1: The parallel framework of visualization subsystem in GeoFEM

Based on the visualization framework, we have implemented many parallel visualization algorithms for the users to visualize their data for scalar, vector and tensor fields.

Parallel visualization techniques for scalar datasets

Compared with other commercial software systems, our cross-sectioning module not only has a highly parallel performance, but also possesses the following advantages: (1) Cross-sections are not limited to planes. The module can generate a surface which is convenient for the users who want to observe the distribution of some physical attribute on a surface such as a global surface. (2) We totally provide ten convenient ways for the users to define a plane cross-section or a surface cross-section, and also can display multi-parallel cross-sections or multi-scalars on different cross-sections in a flexible manner.

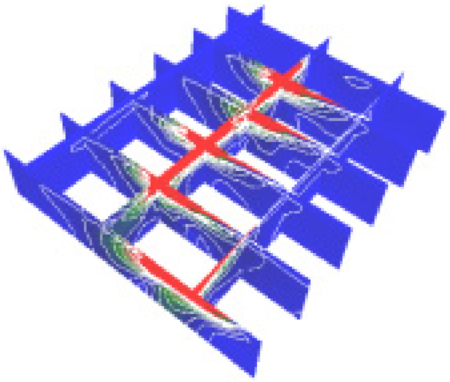


Figure 2: Cross-sectioning for large dataset with 456,365 nodes. Equivalent scalar values of stress are mapped to colors with isolines. (Data courtesy of Mikio Iizuka in GeoFEM)

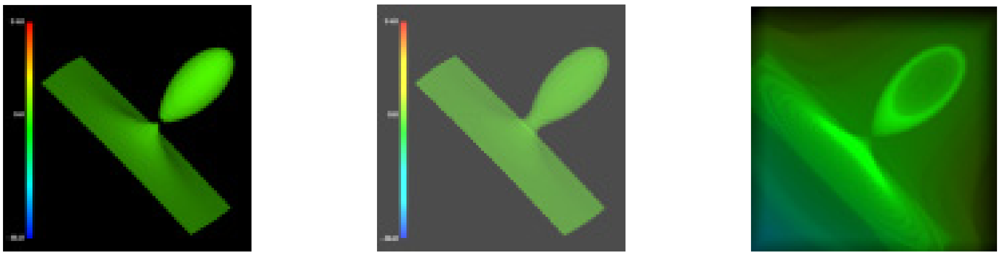
We have tested the module with a large unstructured 3D viscoelastic finite element analysis dataset for kinematic earthquake cycle in the Southwest Japan. It contains 456,365 grid points. We used the parallel computer SR2201, which is installed in the

Computer Center of University of Tokyo, and has 1024PEs, 300Gflops peak speed and 224MB memory for each PE. In the case of using 32 PEs, the module required about 33 seconds to generate the image shown in Figure 2. The analysis module took about 379 seconds. The increase due to visualization is about 8.7%.

Our visualization subsystem also provides the *Isosurface Fitting* (*IF*) method for visualizing 3D scalar fields. We take advantage of a topologically-consistent version of *Marching Cube*s (MC) (Lorensen, 1987[5]), which is well-known as a highly-parallelizable isosurface construction algorithm, and existing accelerated rendering facilities for getting resulting isosurfaces. However, the IF approach visualizes only a part of the target volumetric object at a time (see Figure 3(a), for example). The other is the *direct volume rendering* (DVR) approach, which projects the entire dataset semi-transparently onto a 2D image without the aid of any intermediate geometric representations. Although DVR can produce intuitive images (see Figure 3(c)), it is intrinsically viewing-dependent, and thus requiring expensive re-computations according to changes in projection parameter values.

To find a compromise for the accuracy versus efficiency trade-offs between these two approaches, we adopted yet another concept of *solid fitting* (Fujishiro, 1996[6]), which cannot be found in other commercial visualization software. Solid fitting can be viewed as a generalized IF approach, since it employs *interval volume*, which allows the users to represent as a solid, a 3D subvolume for which the associated scalar values lie within a user-specified closed interval. Such a less-constrained geometric feature extraction allows more intuitive and informative visualization of volumetric *Region Of Interest* (ROI) than the traditional IF approach, and more computationally efficient than the DVR approach (see Figure 3(b)). Interval volume is also suitable for morphological measurement of ROIs. The quantitative properties of interval volume, such as the surface area, total volume, and field integral, are useful for the understanding of the target datasets. Figure 3 compares the three approaches for visualizing a 3D version of the *Folium of Descartes*.

Note that all the three volume visualization algorithms are parallelized in the subsystem.



(a) Isosurface fitting (b) Solid fitting (c) Direct volume rendering

Figure 3: Three volume visualization approaches for 3D version of Folium of Descartes.

Parallel visualization techniques for vector and tensor datasets

Since *streamline* is the most popular way for visualizing vector datasets, we have included a parallel streamline generation algorithm in our subsystem. In order to reveal much more 3D orientation and depth information, we also implemented the *illuminated streamline* method (Zoeckler, 1996[7]), which makes each streamline have a radius to form a *streamtube* (see Figure 4, for example).

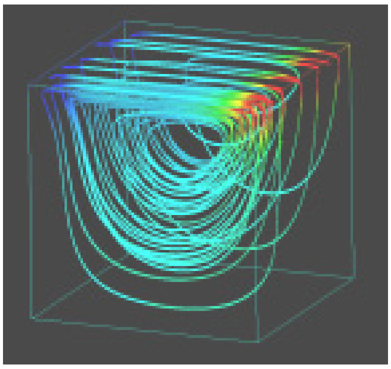


Figure 4: Illuminated streamlines for a lid-driven convection in a cubic cavity with a Reynolds number 1000. (Data courtesy of Hiroaki Matsui in GeoFEM)

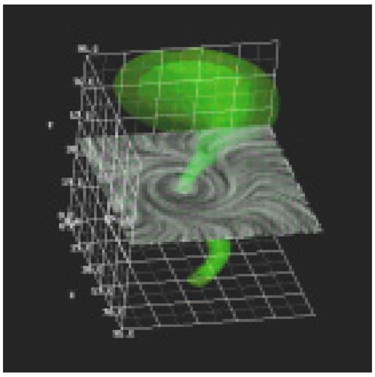


Figure 5: Near-velocity volume & LIC texture-mapped cross-section for flow volume (generation code courtesy of R. Crawfis, OSU)

We also implemented the typical texture-based visualization method—*LIC* method (Carbral, 1993[8]) for vector data fields. LIC (Line Integral Convolution) is a procedure that smears a given image along paths that are dictated by a vector field. It is local, one-dimensional and independent of any predefined geometry or texture, and is capable of showing the vector directions even in the area where they change quickly. It can avoid the sampling problem inherent in the streamline method. Figure 5 shows an LIC texture-mapped cross-section for a tornado flow dataset.

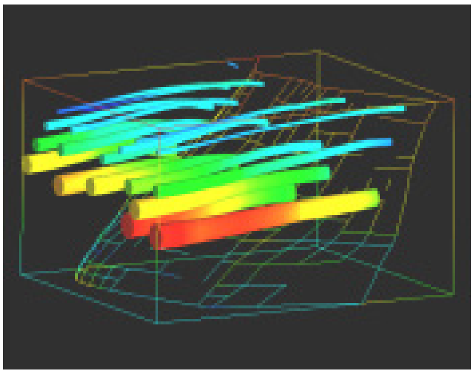


Figure 6: Hyperstreamlines for a fault analysis data around Japanese Islands. (Data courtesy of Mikio Iizuka in GeoFEM)

For tensor datasets, we have implemented a parallel *hyperstreamline* algorithm (Delmarcelle, 1993[9]), which can visualize 3D second-order tensor fields along continuous paths, and can display nine components (three eigenvectors) of a tensor field simultaneously. According to the direction of major eigenvector, the module first generates a trajectory from a seed point, then attaches ellipses at each point on the trajectory, and forming streamtubes. The direction and magnitude of the long axis and short axis of each ellipse are decided by the direction and magnitude of the other two eigenvectors at this point respectively. The colors on the tube surface can display the magnitude of major display eigenvector at each point on the trajectory. An example of stress tensor data is shown in Figure 6.

Feature analysis techniques

Adaptive extraction of isosurfaces/interval volumes based on Hyper Reeb graph

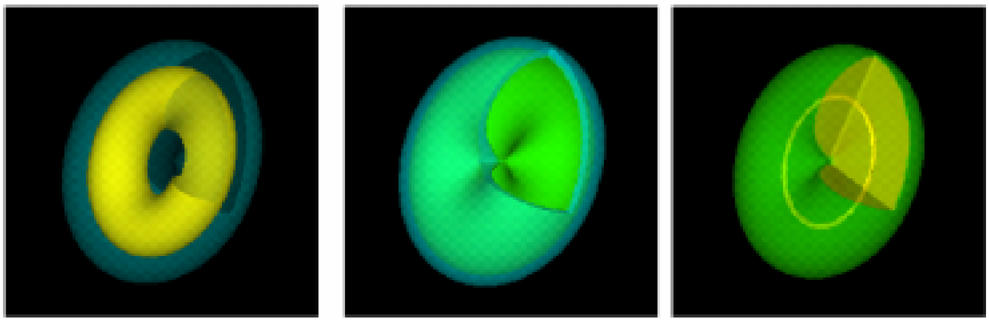
The concept of the *Reeb graph* was originally imported into CG fields by Shinagawa, et al. in order to reconstruct a topologically-correct surface from cross-sectional contours extracted from CT images (Shinagawa, 1991[10]). *Hyper Reeb graph* (*HRG*) is an extension of the Reeb graph concept to 3D volume fields (Fujishiro, 2000[11]). Theoretically, a volume can be decomposed into an infinite number of disjointed isosurfaces with different target values. The topological features of each isosurface can be captured by using the Reeb graph with a common direction for the height function. Therefore, by examining the sequence of isosurfaces in terms of the structure of Reeb graphs, we can find a particular field value, termed *critical field value* (*CFV*), for which the topological equivalence of consecutive isosurfaces is not maintained.

For example, by performing the HRG-based analysis of the topological feature embedded in a given scalar volume dataset beforehands, the subsystem can choose on the fly desirable settings of parameters related to indirect volume visualization techniques. We consider the following two options:

- Method 1: Simultaneous display of *m+1* semi-transparent isosurfaces, each of which is extracted with a field value $(f_i + f_{i+1})/2$ at the midpoint of the topologically-equivalent field interval $[f_i, f_{i+1}](i = 1,...,m)$. We can determine a plausible value for the opacity of each isosurface so as to reflect the mutual relationships among $l_{i,i+1} = f_{i+1} - f_i, (i = 1,...,m)$ in order to allow us to understand the relative thickness of topologically-equivalent field intervals.
- Method 2: Decomposition of a given volume V into a sequence of m+1 non-overlapping interval volumes $IV(f_i, f_{i+1})(i = 0,...,m)$;i.e. $V = \bigcup_{i=0}^{m} IV(f_i, f_{i+1})$. Topological equivalence gives the rigid basis for the volume decomposition. In addition, the boundaries of each interval volume convey informative shapes of isosurfaces with CFVs, at exactly the location where the topology of level surfaces changes.

Figure 7 visualizes the metatorus volume with the above two methods in a comprehensible manner. The selected isosurfaces in Figure 7(a) can also be utilized as an effective set of basic frames for the *flipbook* approach to volume rendering. On the other hand, the set of interval volumes in Figure 7(b) is expected to provide a good initial guess for more sophisticated volume segmentation.

Of course, HRG-based topological analysis can be used as a powerful tool to determine proper transfer functions to yield informative direct volume rendered images. For more details, see the reference [11].

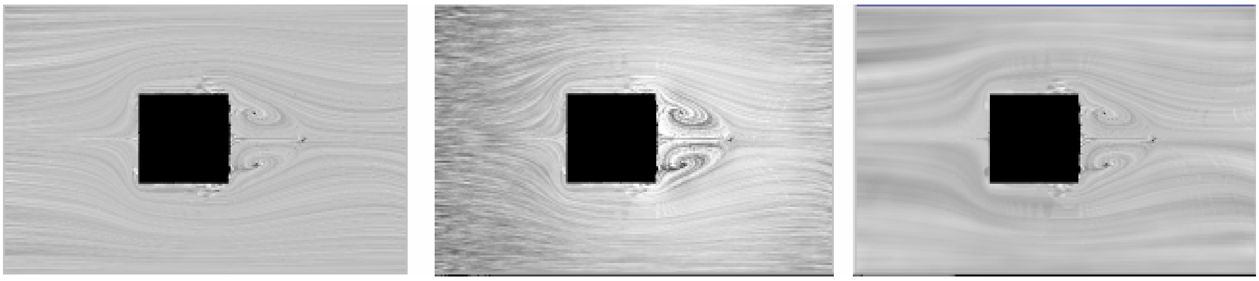


(a) (b)

Figure 7: Geometric object extraction from a metatorus volume based on the HRG. (a) Simultaneous display of two isosurfaces with 0.21 (ellipsoid) and 0.64 (torus); (b) Decomposition into two interval volumes $IV[0.15,0.271]$ and $IV[0.271,1.0]$.

Adaptive LIC image generation for vector fields based on significance map

Texture-based methods provide a very promising way to visualizing vector fields. However, most of the existing methods treat every pixel equally, thus leading to fixed detail over the entire texture space without any designated highlights. In fact, it is quite common for a flow field to have extremely non-uniform distribution of details. It is very time-consuming to generate a finer image so as to ensure a sufficient precision everywhere for significant features.

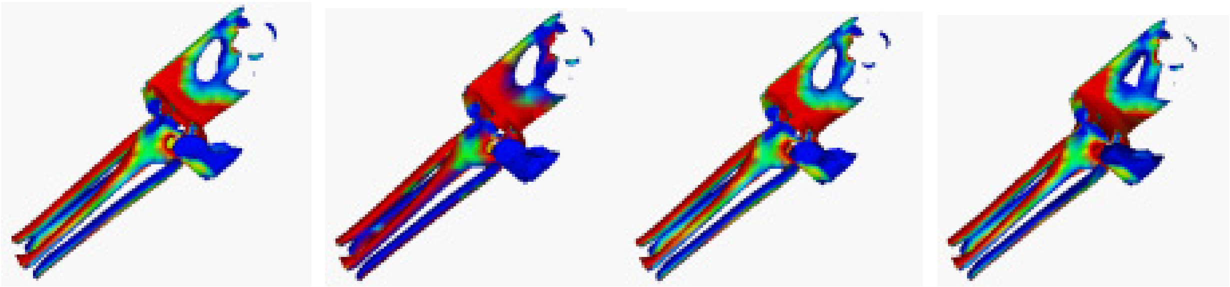


(a) (b) (c)

Figure 8: Adaptive LIC image generation based on significance map. (a) A cross-section texture image generated by the original LIC method; (b) A texture image generated by our significance-driven LIC method, which not only accelerates the image generation, but also highlights the vortices; (c) A texture image generated with different texture granularities. (Data courtesy of Anlu Ren, Zhejiang University, P.R. China)

We adopted a method to ameliorate the problem (Chen, 2000[12]). We introduce the “significance map”, which is derived from both intrinsic properties of a given vector field and user-guided highlights. We employ the flow topology analysis technique (Chong, 1990[13]) to determine the significance value at each point. In a case that a user is interested in certain areas, his/her specification may be used to adjust the topology-derived value. Based on the significance map, we propose techniques to accelerate LIC texture image generation, to highlight important structures in a vector field, and to generate an LIC texture image with different granularities. Figure 8 shows an example of our method. Figure 8(a) is generated by the original LIC method. Figure 8(b) is generated by adjusting texture opacities to highlight the vortices. And the convolution length at each pixel is shortened greatly in the regions far away vortices, by which it is 5 times faster than the original one. In Figure 8(c), a coarser granularity is selected in lower significant areas, which can involve less cells to generate the texture image. It is about 4 times faster than before.

Parallel decimation



(a) (b) (c) (d)

Figure 9. Simplifying interval volume of mechanical part. (a) Original (6,706 patches); (b) 50% patches reduced by accounting for geometry only (r = 0.0); (c) 50% patches reduced by optimized combination of color/geometry deterioration (r =0.63); (d) 50% patches reduced by accounting for color only (r = 1.0).

In order to make efficient the transmission and rendition of geometric primitives for the display on clients, a simplification scheme to decimate triangle patches (Schroeder, 1992[14]) was extended. The extended algorithm accounts for the color attributes as well as geometric features to select best edges to be collapsed (Nakamura, 2000[15]). Although analogous concepts can be found in the literature (Hoppe, 1999[16]), what distinguishes our algorithm from the others lies in its auxiliary mechanism to determine the combination ratio r of the color/geometry components in an error metric automatically by reflecting the coherence structure of a given two-scalar volumetric dataset. This is expected to be extremely useful for retaining meaningful details obtained by large-scale computations even within overall data reduction effects. Figure 9 uses a mechanical part volume to illustrate the feasibility of our optimized simplification scheme.

Conclusions

The parallel visualization subsystem in GeoFEM has been described. It can perform visualization concurrently with computation on a high-performance parallel computer, and output simplified geometric primitives to clients. It provides many kinds of parallel visualization techniques, covering from scalar data, vector data to tensor data. Feature analysis techniques are presented to improve the quality of visualization results. Future work includes developing more sophisticated visualization techniques, and improving the quality and efficiency for larger and complicated datasets.

Acknowledgments

The authors have benefited from continuous discussions with GeoFEM staff. This research was funded by Special Promoting Funds of Science & Technology.

References

[1] Okuda, H., Nakamura, H. and Yagawa, G., 2000, *GeoFEM: Parallel FE solid earth simulation platform ─ computational performance and solid earth applications*. CD-ROM Proceedings of the 4th International Conference on Supercomputing in Nuclear Applications, Tokyo, ES-G3.

[2] L.Hesselink, F.H.Post, and J.J. van Wijk, 1994, *Research Issues in Vector and Tensor Field Visualization.* IEEE CG&A, **14**(2):76-79.

[3] Haimes, R., *PV3: A distributed system for large-scale unsteady CFD visualization*, AIAA, paper 94-0321, 1994.

[4] Doi, S., et al., 1997, *RVSLIB: A library for concurrent network visualization of large-scale unsteady simulation.* SPEEDUP Journal, 11, 59-65.

[5] Lorensen, W.E. and Cline, H.E., 1987, *Marching Cubes: A high resolution 3D surface construction algorithm.* ACM Comp. Graph, **21**(4), 163-169.

[6] Fujishiro, I., Maeda, Y., Sato, H. and Takeshima, Y., 1996, *Volumetric data Exploration using interval volume*. IEEE TVCG, **2**(2), 144-155.

[7] Zoeckler, M., Stalling, D. and Hege, H.-C.,1996, *Interactive visualization of 3D vector fields using illuminated stream lines.* In Proceedings of IEEE Visualization, 107-113.

[8] Cabral, B., Leedom, C., 1993, *Image vector field using line integral convolution.* In Computer Graphics Proceedings, Annual Conference Series. ACM SIGGRAPH, 263-272.

[9] Delmarcelle, T. and Hesselink, L., 1993, *Visualizing second-order tensor fields with hyperstreamlines*. IEEE CG&A, **13**(4), 25-33.

[10] Shinagawa, Y. and Kunii, T.L., 1991, *Constructing a Reeb graph automatically from cross sections.* IEEE CG&A, **11**(6), 41-51.

[11] Fujishiro, I., Azuma, T., Takeshima, Y. and Takahashi, S., 2000, *Volume data mining using 3D field topology analysis*. IEEE CG&A, **20**(5), 46-51.

[12] Chen, L., Fujishiro, I. and Peng, Q.S., 2000, *Fast LIC image generation based on significance map*. In Proceedings of the Third International Symposium on High Performance Computing, 537-546.

[13] Chong, M.S., Perry, A.E., Cantwell, B.J., 1990, *A general classification of 3D flow fields*. Physics of Fluids Ann., **2**(5), 765-777.

[14] Schroeder,W.J., Zarge, J. A. and Lorensen, W.E.,1992, *Decimation of triangle meshes*. ACM

Comp. Graph., **26**(2), 65-70.

[15] Nakamura, H., Fujishiro, I., and Takeshima, Y., 2000, *Towards optimizing local feature metric for simplifying colored interval volumes*, In Proceedings of Work in Progress, IEEE Visualization 2000, Salt Lake City.

[16] Hoppe, H., 1999, *New quadric metric for simplifying meshes with appearance attributes.* In Proceedings of IEEE Visualization '99, 59-66.