Interactive Visualization by Camera Cluster

Tomoki Yamada[1], Akira Kageyama[1], and Youhei Masada[1]

[1]Graduate School of System Informatics, Kobe University, Kobe, Japan

1 Introduction

Scientific visualization for large scale simulation data is usually performed in the post-process mode. A huge amount of numerical data produced by hours-long simulation job on a supercomputer is first stored in the storage system of the computer, then the data is transferred to a local storage system of the reseacher. The data transfer through the internet can take hours, or even more than a day, when the total size of the numerical data amounts to more than TB. It is not rare that the data transfer process takes more time than the simulation itself in the high performance computations.

The data transfer is not the worst part as a time consumer in the research cycle of the simulation study. The transferred data should be processed through various ways such as statistical analysis, mathematical conversion, compression/decompression, and, of course, visualizations to generate images and movies.

If the final goal of the above overall post process is visualized images or movies, it is a natural idea to apply the visualizations on the supercomputer when the simulation is running. This is called realtime or in-situ visualization [1][2].

A problem of the in-situ visualization method is that the target data as well as visualization parameters including the camera position that are used to generate intended images/movies have to be fixed before the submission of the simulation job. When the resulting movies are not satisfactory, one has to re-submit the simulation job again, with modified visualization parameters. This would be the reason why the in-situ visualization method has not become a mainstream in the large scale simulation study.

In this paper, we propose a new way of in-situ visualization that can overcome the previous flaw. The key point is to make a lot of movies at once, by placing thousands of cameras around and inside the simulation region. See Fig. 1. Each camera has thousands of visualization parameters and respective target data. The output of this in-situ visualization method by a cluster of cameras are many (millions) of movie files.

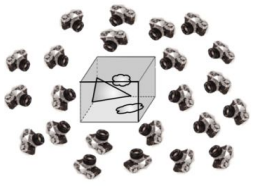

Fig. 1. Thousands of cameras around simulation region.

In this camera cluster method, (almost) any specific visualization image, for example an isosurface of specific scalar field at a specific level can be found in the set of movie files.

When the number of camera position is sufficiently large, one can view a visualized objects from an effectively any angle. This means that one can extract a sequence of rotating objects from the movies database *in real time*, by, for example, the mouse drag. A sequence of images taken from different view points are extracted from different movie files one after another in accordance with the mouse drag. The same technique can be applied to change the isosurface level in real time. All one has to do is to find the proper movie file in which the isosurface is visualized with the specified level. In principle, one can do almost all interactive operations that are allowed in the interactive visualization software by this camera cluster method.

2 Technical Challenges

2.1 Configuration of viewpoints

A key point of the camera cluster method proposed in this paper is to place thousands of cameras (or view points) that take (independently) in-situ visualizations of the simulation. To demonstrate the feasibility of the concept, we have developed a test code in which 130 cameras are placed around a simulation region. For the in-situ rendering, we use a visualization program developed by Furumura and Chen [6].

In the original concept of the camera cluster method, the cameras are supposed to be placed almost uniformly in the 3-D space inside and outside of the simulation box. In this test, however, we have placed 130 cameras in a two-dimensional distribution to minimize the camera number; they are placed on a spherical surface with a fixed radius from the center of the simulation box. All the cameras look at the the center.

It is not straightforward to place more than 100 cameras on a spherical surface as uniform as possible. Making use of the spherical coordinate system, or on the grid points of the so-called latitude-longitude mesh is not a good idea, because it leads to a unnecessary concentration of cameras around the poles. To avoid the pole concentration, we make use of Yin-Yang grid in which two congruent component grids are combined with partial overlap to cover a spherical surface [3].

2.2 Interactive movie player

A simulation job with in-situ visualization by 130 cameras generates 130 sets of image sequences. The image sequences are converted to movie files by making use of FFmpeg library. After the simulation, the movie files are retrieved to a local PC at hand.

Though these movie files contain a great deal of information, they are useless without a proper player. What common movie players do is basically simple; it reads one source movie file and show a sequence of the images contained in the source file on the PC's screen.

On the contrary, the special movie player we need here is more complicated one. It reads multiple (130 at present test and millions in future) source movie files. And it generates a sequence of images from them *in real time*, responding to the user's command through the PC's GUI event such as the mouse drag or key type. We have developed such a movie player with OpenCV [5].

The player we have developed here has mainly three functions; (1) to stop and play, (2) to interactively change the source movie (that is, to change the view point), and (3) to interactively zoom-in/out the scene.

3 Proof of Concept

To demonstrate the feasibility of the Camera Cluster method, we have chosen a seismic wave propagation simulation for the first application, since our rendering engine adopted in this test code is specifically designed to this kind of simulations.

In this simulation, the time development of the displacement $\mathbf{u}$ and it's time derivative (i.e., velocity) $\mathbf{v}$ at each grid point in a rectangular simulation box is calculated by solving the viscoelastic equation of motion with the finite difference method.

To visualize the s-wave (shear mode) and p-wave (compressional mode), we have defined two scalar fields $\phi_s \equiv |\nabla \times \mathbf{v}|$ and $\phi_p \equiv \nabla \cdot \mathbf{v}$.

The distributions of ϕ_p and ϕ_s calculated in the simulation code are stored to 3-dimensional arrays as volume data and they are visualized in real time.

Test simulations with in-situ camera cluster visualization are executed on a PC, a workstation, and a PC cluster system. 130 cameras are spherically distributed on the Yin-Yang grid points, 65 for Yin and 65 for Yang. Each camera generates a sequence of visualization images in the PPM format. The number of the sequential frames in each movie is 60 in this test. Fig. 2 shows an sample of interactive visualization by means of the camera cluster method. Propagation of p-wave (orange) and s-wave (green) are observed by the movie shown on a window of the PC. Typing a key on the keyboard send a signal to the movie player to change the source movie file (or to change the camera angle). One can observe the propagations from any angle in 3-D.

4 Conclusion

In this paper, we have proposed a new visualization method for large scale computer simulations. This is a kind of real-time, in-situ visualization with a twist. The key idea is to make many (millions of) movies files at once. The output data of this simulation/visualization strategy is not numbers, but movies.

We have developed an interactive movie player which reads multiple source movie files and generates an animation from those source files in real time. We can interactively "rotate" a visualized object by this movie player.

By a demonstration test applied to a seismic wave propagation simulation, we have confirmed that this method (camera cluster method) is not only feasible, but also practical. In future, we will apply this method to other simulation programs, with different rendering engines, and with much more cameras.

Acknowledgements

We thank Professor Furumura for providing us the simulation and volume rendering codes for seismic wave propagation.

References

[1] Hongfeng Yu, Chaoli Wang, Ray W. Grout, Jacqueline H. Chen, Kwan-Liu Ma, "A Study of In-Situ Visualization for Petascale Combustion Simulations", Technical Report CSE-2009-9, University of California at Davis, 2009

[2] Jerome Soumagne, John Biddiscombe, Jerry Clarke, "In-situ Visualization and Analysis of SPH Data using a ParaView Plugin and a Distributed Shared Memory Interface", 5th International SPHERIC Workshop, the University of Manchester UK, pp.186-193, 2010

[3] A. Kageyama, T. Sato, "'Yin-Yang grid': An overset grid in spherical geometry", GEOCHEMISTRY GEOPHYSICS GEOSYSTEMS, vol.5, pp.1-15, 2004

[4] http://www.ffmpeg.org

[5] http://www.sourceforge.net/projects/opencvlibrary

[6] T. Furumura, L. Chen, "Large Scale Parallel Simulation and Visualization of 3D Seismic Wavefield Using the Earth Simulator", Computer Modeling of Engeneering and Sciences, vol.6, pp.153-168, 2004

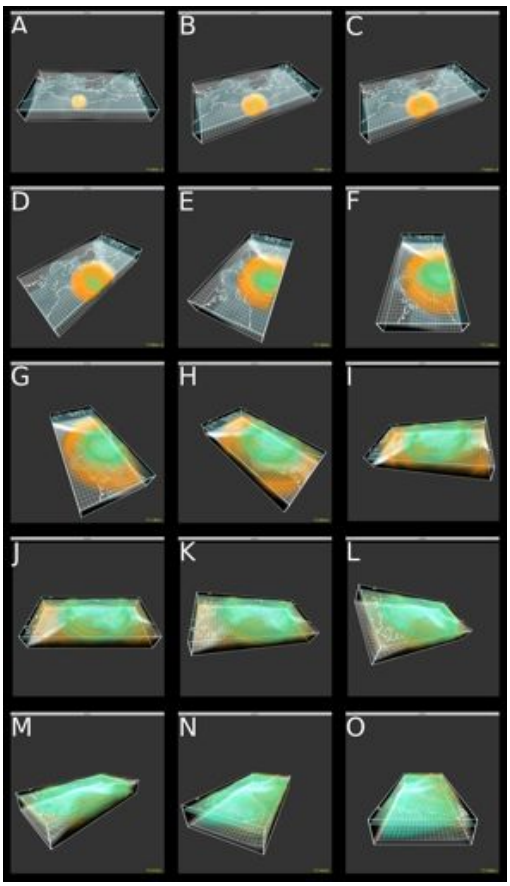

Fig. 2. Snapshot sequence from the video player.