Annual Report of
ADVENTURE Project
ADV-99-1 (1999)

PARALLEL VISUALIZATION OF FINITE ELEMENT SOLUTIONS WITH TEN MILLION DOFS USING PC CLUSTER IN A WINDOWS ENVIRONMENT

***Shinobu YOSHIMURA ,**Shinich SHOUI, and **Hiroshi AKIBA**

*School of Engineering, University of Tokyo
e-mail: yoshi@q.t.u-tokyo.ac.jp

**Allied Engineering Corporation
e-mail: shoui@alde.co.jp, akiba@alde.co.jp

1 INTRODUCTION

When visualization is processed in engineering design, physical phenomena are usually globally understood at first, and then the detailed behaviors are analyzed. In order to understand physical phenomena, the visualized image must be freely and interactively displayed from any perspective, even if the image is not precise. On the other hand, the details must be displayed and freely-searching functions are also required.

Another approach is to only visualize the important information which is extracted from the results of the analysis in advance. However, it is not easy to determine parameters to extract the information from large and complicated data. Also, data extraction makes a detailed search impossible. Therefore, if the significant factor in a large scale analysis is retaining realism, all the results obtained on the analysis must be preserved and a user can smoothly switch the visualization functions at any time.

This kind of visualization has been mainly performed by the cooperative use of supercomputers and workstations with specially equipped hardware for graphics. Within this frame, various kinds of architectures have been developed according to how much processing is performed on a supercomputer.

Recently, significant three-dimensional-graphics environments have been developed, and it is becoming possible to construct a system which offers real-time visual images with interactive operations. Even on personal computers, visualization processing is becoming widely available due to the improvement of its basic ability and the high cost-performance. However, present PCs are not powerful enough to visualize images in large scale analyses.

In order to achieve real-time display with interactiveness, the drawing rate should be kept at around 10Hz. This is the bottleneck in the visualization of large scale analyses on a PC. At this point, PCs can not oppose workstations with special graphics equipment that is processing 3D images. Recently, however, PC-class machines are available, and distributed-processing environment, which is made up with several kinds of computers connected by LAN, is becoming popular. Therfore, technology which uses these computer resources effectively is needed.

This study describes a system which can visualize results of three-dimensional-finite-element analyses with ten-million-scale degrees of freedom. It generates visualization information on a PC cluster working in parallel to display the results on a PC with Windows NT.

2 VISUALIZATION PROCESSES

In parallel visualization processing, the first issue is to decide which processes should be performed in parallel. In addition, the amount of the calculations and communication must be balanced carefully. Also, seveal parallel processing procedures must be implemented according to each visualization method.

Scientific visualization can be classified into two types: surface and volume visualization. In the former the surfaces of the objects or physical quantities on any cross section are visualized, in the latter, the direct volume rendering or the physical quantities inside an object, are visualized. The data format can be defined by regarding the visualization process as the data-conversion process abstractly. Figure 1 shows a flow of this abstraction. The visualization process is generally composed of two steps, mapping and rendering. The details of each process are described later.

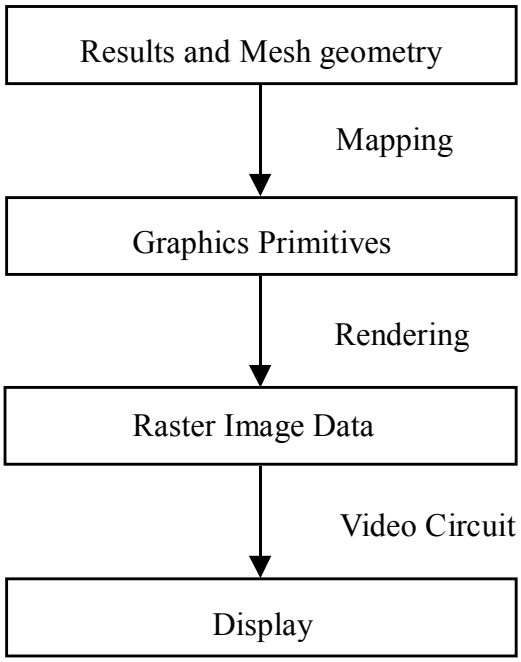


Figure 1 Data-conversion process in Visualization

An analysis calculation first creates the distribution of certain physical quantities which are requested to be visualized. The results of finite-element analysis are expressed as scalars, vectors or tensors on each element or node of the meshes. For example, in an elastostatic stress analysis, the strain and stress tensors are produced from the displacement vectors, and scalar fields like the equivalent stress are also obtained. The meshes consist of geometric elements like tetrahedrons or hexahedrons, and the integral points in each element possess the physical quantities obtained via the analysis. Since the visualization information is presented on the geometric elements such as points, lines and faces, an interpolation is necessary.

Usually, the physical quantities on the nodes are calculated by averaging the values on the elements which have been previously obtained from the values on the integral points. In order to make the images easy to understand, some kinds of geometric information must be taken into account, so as to make the physical quantities correspond to the visualization information. This procedure is called

"mapping".

There are two visualization methods in finite element analysis, i.e. data sampling and conversion to graphics primitives. These techniques are described as follows.

(1) Transformation to graphics primitives

In elastostatic stress analyses, scalar fields are displayed on the surfaces of the objects. By using the connectivity information of the meshes, the planner patch elements are extracted to represent the surface shapes of the analysis model, and the visualization information is displayed on them. A color map is determined that corresponds to the regularized physical quantities on the sets of the vertices. In other methods, free-form surfaces or patches generated on a pre-processor are sometimes used.

The physical quantities inside the objects are visualized by isosurfaces or cross sections for the scalar data. In order to display the cross section, the faces are extracted, where the cross section crosses each finite element. At the same time, the physical quantities are linearly interpolated to them. A surface within the same level is extracted by scanning the physical quantities on the nodes of each finite element so as to display the isosurfaces.

(2) Data sampling

In data sampling, the physical quantities on structured grids, which are obtained by the interpolation from the results of the analysis are sampled. This procedure is used in volume rendering and also is applicable to generating cross sections or isosurfaces. Displaying isosurfaces is generally achieved by the Marching Cube method with the grids for the sampling. In the finite element method, unstructured grids can decrease the number of the calculations even on complicated processing, as compared to structured grids. They are considered to be especially effective for high spatial resolutions.

(3) Rendering

In surface visualization, the physical quantities are made to correspond to the graphics primitives. They must be converted to a raster data to be displayed on the screen. The Z-buffer method is often used, in which graphics primitives are scanned with taking the front-and-back relationship from this viewpoint into account, and they are projected onto a plane to make a two-dimensional image. The ability of the interactive three-dimensional graphics is core primary to the Z-buffer method. Furthermore, optical effects are also added to the rendering procedure to make the object realistic. Flat or smooth shading are the typical methods used to produce optical effects.

3 PARALLEL VISUALIZATION SYSTEM

A basic approach to system architecture and an algorithm for an interactive parallel visualization system are described here. The meaning of being interactive is that an image can be smoothly updated in every operation by users, such as zooming, rotationing the displayed objects, or shifting in the viewspace.

3.1 Approach to parallel processing

Visualization for large scale analyses requires an excessive number of calculations and memory to be performed on a single processor system. For example, a problem with a-million-scale degrees of

freedom roughly needs 300 MB of memory. A single processor system therefore fails to deal with larger scale solutions, and parallel processing must be required. One of the problems in parallel visualization is that the data conversion process requires a large number of calculations and memory. Therefore one of the problems is how to distribute these processes effectively. This will be examined in the following section.

First, the way we will discuss how to access the data is considered. The data is distributed in each parallel PE; this may reach several tens or hundred of mega-bytes in a large scale analysis. Since limited hardware can keep all data in the same place, the data must be preserved on the network in a distributed manner. This may lead to a client-server system for visualization processing [Pei-Wen Liu et al., 1996], where a large number of analysis results are managed on the parallel processing environment and load distribution is achieved during the processes between inputting the data and generating the visual images. In this project, the pre-processing system automatically generates the meshes to be decomposed based on the topology between the elements. The main solver uses these decomposed domains in its parallel calculations. This visualization system, therefore, can also use this decomposed data by distributing them on the cluster.

Next, we will describe how to generate visual images in parallel. There may be two methods used, depending on how many processes are performed in parallel: one generates only graphics primitives in parallel, and the other performs all procedures up to the rendering in parallel. For volume rendering, the volume data is directly rendered without extracting graphics primitives; a parallel algorithm for MPP has been developed along this way [Jaswinder Pal Singh et al., 1994].

Here, parallel surface visualization is examined. In a parallel rendering process on a PC cluster, a raster image, (i.e. the result of the rendering), needs to be transferred to the display PC. The load should be distributed every time drawing on the display. The efficiency in parallel processing must be sufficient, even taking the data-transferring cost into account, However, software-only rendering generally doesn't have any advantages, since processing by software is much slower than that by hardware and it is costly to transfer the raster images obtained by the rendering. From these reasons, this system uses the following approach: the procedures between the calculation of the physical quantities and the extraction of the graphics primitives are performed in parallel, and the data of the graphics primitives is transferred to the display PC, and the rendering is made on the single processor. Another problem for the visualization of large scale analyses is the trade-off between the number of the calculations and the resolution of the obtained image, i.e. the spatial resolution of the analysis can exceed that of the visual image. In other words, the refinement of the finite elements exceeds the resolution which can contain the smoothness of the displayed image, and the number of the graphics primitives corresponding to the pixels on the display are lager than that. When the wire-frames of the meshes are displayed, the elements are too small to be seen. The same problem also appears when displaying the cross sections or isosurfaces.

Furthermore, even if the graphics primitives can be extracted quickly enough in parallel processing, the amount of information must be saturated on the rendering process. For these reasons, the visualization should be achieved by a procedure based on the display resolution. This technique can retain the system's performance and avoid increasing the load in the rendering process by adjusting or keeping the number of the graphics primitives approximately constant. Therefore, an efficient algorithm and a practical system architecture is needed.

This technique also has the following advantages. Since the ability to display graphics is totally dependent on the hardware, one of the requirements can be achieved by making the software that can work on various kinds of machines with different graphics ability. This is not to prevent the use of

high-resolution visual images but to offer choices to users.

This technique is equivalent to developing an effective algorithm for a system with interactive operating functions at a practical level even on a large scale analysis, because the spatial resolution of the analysis model will exceed that of the visual image. The trade-off problem between the accuracy of the visualization images and the number of the calculations instantly leads to a procedure based on the resolution of the visualization image.

3.2 Architecture of the parallel visualization system

The parallel processing in this system is performed in perfect parallelism [Cherri M. Pancake, 1996]. This means that the problem is decomposed into independent domains which enable local processing. Since the data in this system is already decomposed as described above, this method and the successive processing algorithm are applicable to all surface-rendering processes.

The local disks on each PE have the data of decomposed finite element meshes and the results of the analysis. A slave, i.e. each PE, can only access the data and extracts the graphics primitives independently. Each process communicates by message passing. In volume rendering, the data access to a large shared space is abstracted via the message passing [Jaswinder Pal Singh et al., 1994].

This system is composed of a PC cluster which consists of several servers and a client, as shown in Fig. 2.

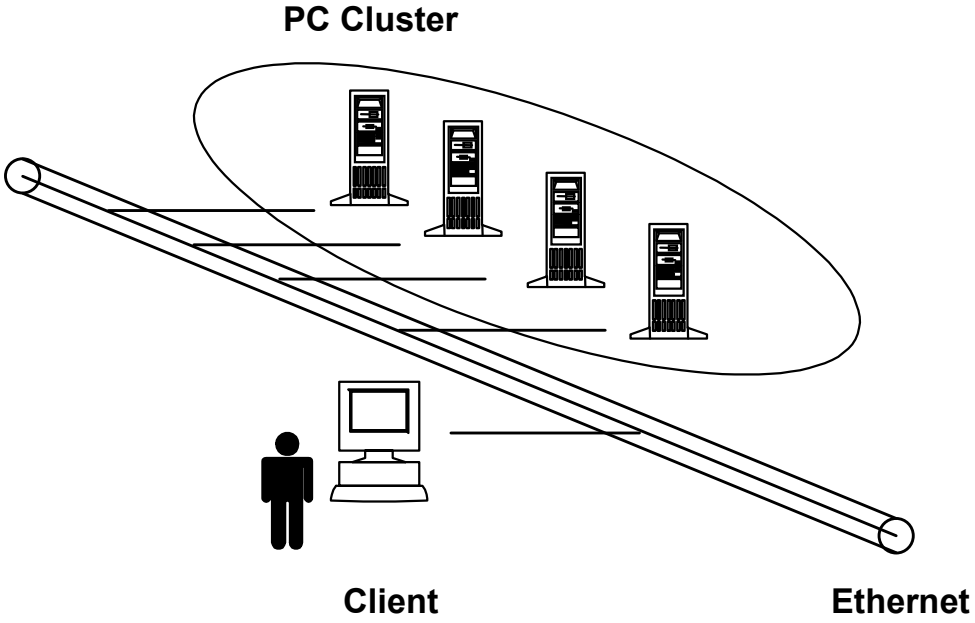


Figure 2 Client-Server System

Parallel processing is performed via the domain decomposition method [G. Yagawa et al., 1993] in this client-server system. It works with several slave processes and a master process which corresponds to each PE. A message passing model usually does not dynamically create a process. In addition, two programming models are known: in one model a multiplexed task is performed in each

single processor, and in the other different programs are performed as different tasks. Here, the latter one (i.e. SPMD) is used.

The following describes the details of the system's components, i.e. processes of the client, master and slaves. The relationship is shown in Fig. 3.

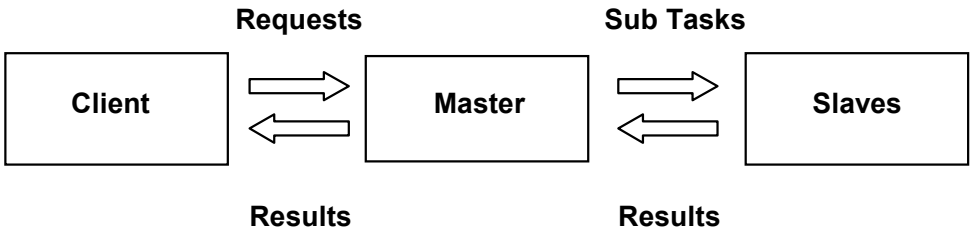


Figure 3 Master-Slave for Parallel Processing

The client displays the visualization information through the interactive operation on a GUI. Events happen via users like menu selection or button clicking on the window, and the request commands for the required processes are sent to the master. The client only receives the results from the master; this data is the extracted graphics primitives. Three dimensional graphics are displayed on the client and the user operates the displayed objects by zooming, rotating, or shifting in view space, etc.

The master receives the operation-request messages from the client and sends the request messages of decomposed task to the slaves. The master merges the processed results on the slaves in parallel and sends them to the client.

3.3 Visualization algorithms for large scale analyses

Some algorithms for visualization processing in large scale elastic stress analyses are described below.

(1) Displaying surface contours

Scalar fields are visualized in elastic stress analysis using geometric primitives of the analysis model. Here, the problem is the data format to display surface geometry. By using free-form surfaces which are used in the pre-system to represent surface geometry, the surfaces can be split into parts by any number. Then, it becomes relatively simple to make the patches by quadtrees for levels of detail. Since the physical quantities are specified on the nodes of the meshes, they must be projected onto the patches. Also, if the patches are not fine enough, quantities like the stress concentration cannot be accurately obtained because the density of the patches does not correspond to the surface used in the analysis calculation. The same problem arises also for patches used in automatic element generation. Therefore, in this system, the surface patches are extracted from the connectivity information between the mesh elements.

When displaying surface contours of the objects, the surface patches are extracted on each slave in parallel and the physical quantities are mapped onto them. They are merged by the master and sent to the client.

For an analysis with several hundred thousands of surface patches, it becomes difficult to display

the image smoothly. In order to resolve this problem, the surface patches should be simplified. The method is described later.

(2) Displaying cross section

When generating a cross section, the client sends the information to the master to preserve it. The master sends it to all the slaves and waits for orders. For each grid point found on the cross section, the slaves search the elements, which include the grid points. As shown in Fig. 4, the slaves keep singly linked list in the voxels, which is divided into three dimensional orthogonal lattice. These voxels are made by decomposing the box coordinates which include the meshes owned by the slave. The linked list holds the index of the elements inside the domain.

The slaves can quickly find the element, (including the grid point on the cross section) by calculating the coordinates from the element index kept in the voxel space. The physical quantities are obtained in this manner. Then, the slaves send them to the master. This is illustrated in Fig. 5.

After all the processes on the slaves are finished, the master sends them to the client for the display.

In this method, the precision of the visualized images depends on the grid number of the crosssection. Also, the number of the calculation depends on both the grid number and the time needed to search the elements. The processing time can be reduced by adjusting the number of the voxels according to the spatial mesh resolution used in the analysis.

(3) Displaying isosurfaces

The isosurfaces are generated by the data-sampling method. Triangular patches are made by Marching Cube method, as similarly to the cross-section generation, after the physical quantities are sampled with each orthogonal-grid point on the slaves. The master merges the patches extracted by the slaves, and send them to the client for display. Here, both the number of the calculation and the precision of the visualized images depend on the grid size in each slave, and not on the analysis scale.

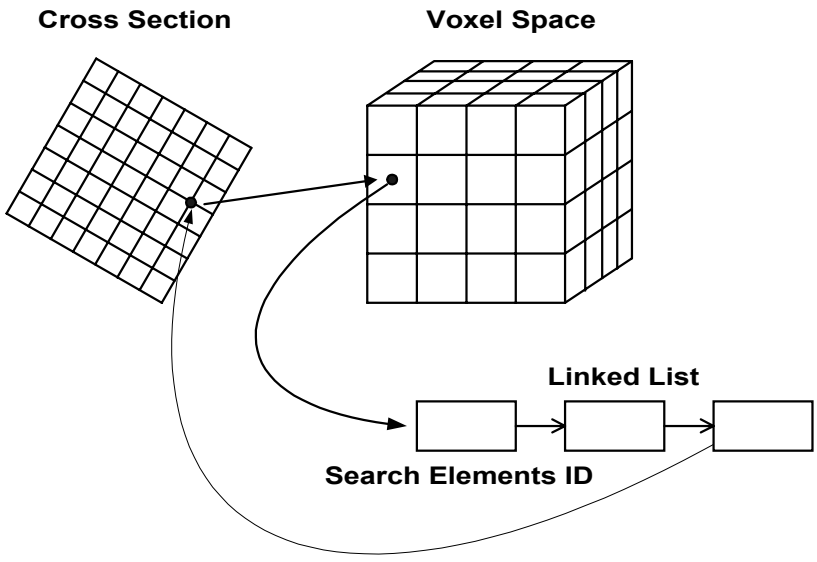


Figure 4 Data Sampling to a Cross Section

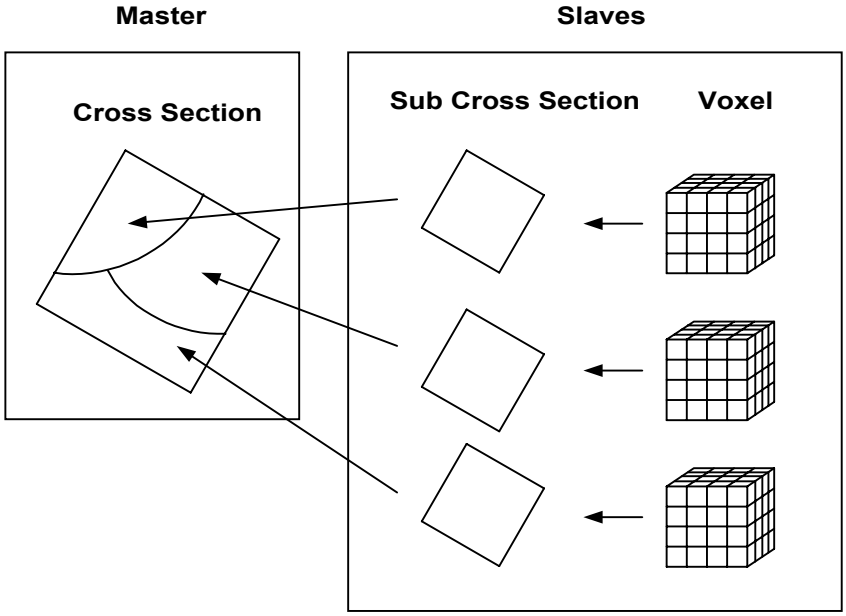


Figure 5 Cross Section Generation in Parallel

(4) Simplification of surface patches

There are several methods that can simplify the surface patches. In this study, triangulars are introduced as the surface patches which are extracted from the meshes, and the vertex pairs of each triangular are contracted to one side [Michel Garland and Paul Heckbert, 1997]. A parameter which is determined by the length of the edge decides which pair should be contracted, i.e. the length of the edge on every extracted elements are calculated first.

Next, an adequate length is chosen from their distribution. If an edge is shorter than this length, the edge is contacted to the other vertex and this procedure is recursively performed. The advantage of this procedure is that the coordinates of the triangular vertices after the simplification are the same as the nodes in the original analysis. This method is efficient. Each slave can perform this step in parallel, since there is no need to calculate physical quantities again using interpolation. The problem here is that the simplification is done only on short edges or dense parts of the elements. Therefore, the obtained stress concentration could not be accurate enough, because fine meshes are made around the region which may cause the stress concentration.

So, the vertex clustering should be done to reduce this effect. At first, the triangular vertices are stored in octtrees, as shown in Fig. 6. The domains are recursively divided by the number of the vertices included in each domain, and a contraction, (described above for vertex pairs of the edge), is performed for the sets of the vertices in each divided domain.

Furthermore, the parameter for the contraction is not only the length of edge, but the gradient of the physical quantities is also considerd here. It is possible to simplify the patches and also to contain the visualization information in parts with a rapid gradient of the scalar field by regularizing the scalar

quantities at each vertex and adding the difference as the parameters. For the isosurfaces, the contraction for the vertex pairs is accurate for the simplification.

(5) Levels of detail

By using the above-described simplification method for surface patches, it is possible to set levels of simplification and to generate several patches with different fineness. They can be easily obtained by recording the history of the contraction on each vertex. As shown in Fig. 7, the history is composed of the index of the vertices, which is made by the contraction of the surface patches on each level of the simplification. This is performed in parallel in a manner similar to the simplification; each slave independently creates a history and sends it to the master. In elastostatic stress analyses, sending it from the master to the client once, is adequate. On the other hand, it should be sent to each step in an unsteady analysis.

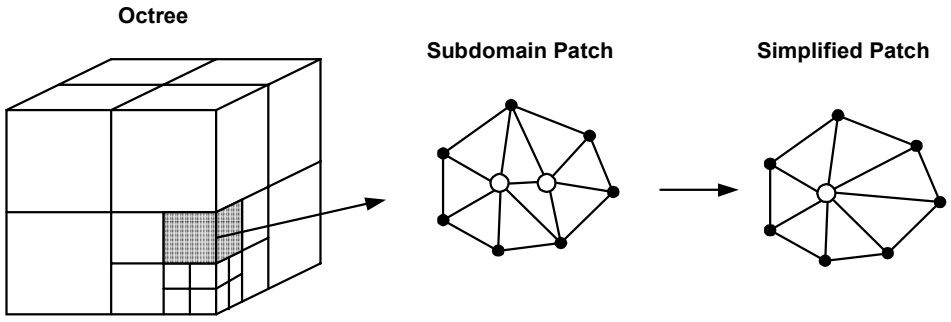


Figure 6 Simplification of Surface Pathes

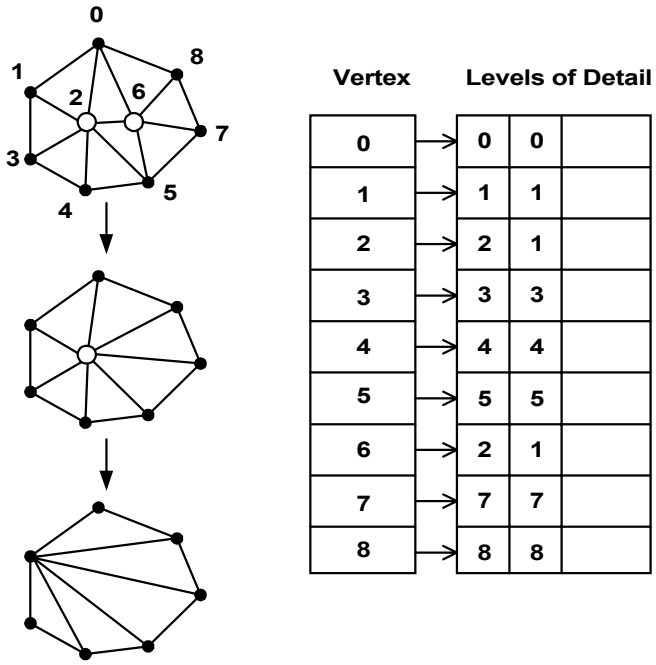


Figure 7 Leveles of Detail

4 IMPLEMENTAION

Object oriented design and implementation are employed in this development. Here, the programming language is C++. MFC (Microsoft Foundation Class Library) and OpenGL are used for the GUI and for the three-dimensional-graphics library, respectively. OpenGL is widely available on both WSs and PCs, and now this is regarded as one of the standard graphics libraries. The TCP/IP sockets are also used for the communication.

5 ESTIMATING PERFORMANCE

The system is examined using the data of an elastic stress analysis with a-million-scale degrees of freedom, and this analysis is performed on massively parallel processors, (Hitachi SR2201), by using a code for parallel-finite-element analyses based on the domain decomposition method. Five VT-Alphas (600MHz) for the parallel processing and a PentiumII (450MHz) machine are connected to communicate through a Fast Ethernet (100Mbps). Examples of visialization from static stress analysis with a million DOSs scale are shown in Figure 8 and 9.

6 CONCLUSION

This study outlines a parallel visualization system, which can process ten-million degrees of freedom. This system can process visual images without any interactiveness by parallel processing of the graphics primitives with the aid of the data sampling method. Fundamental performances of the developed system were demonstrated through the visualization of elasto static analysis results of one million DOF ploblem.

REFERENCES

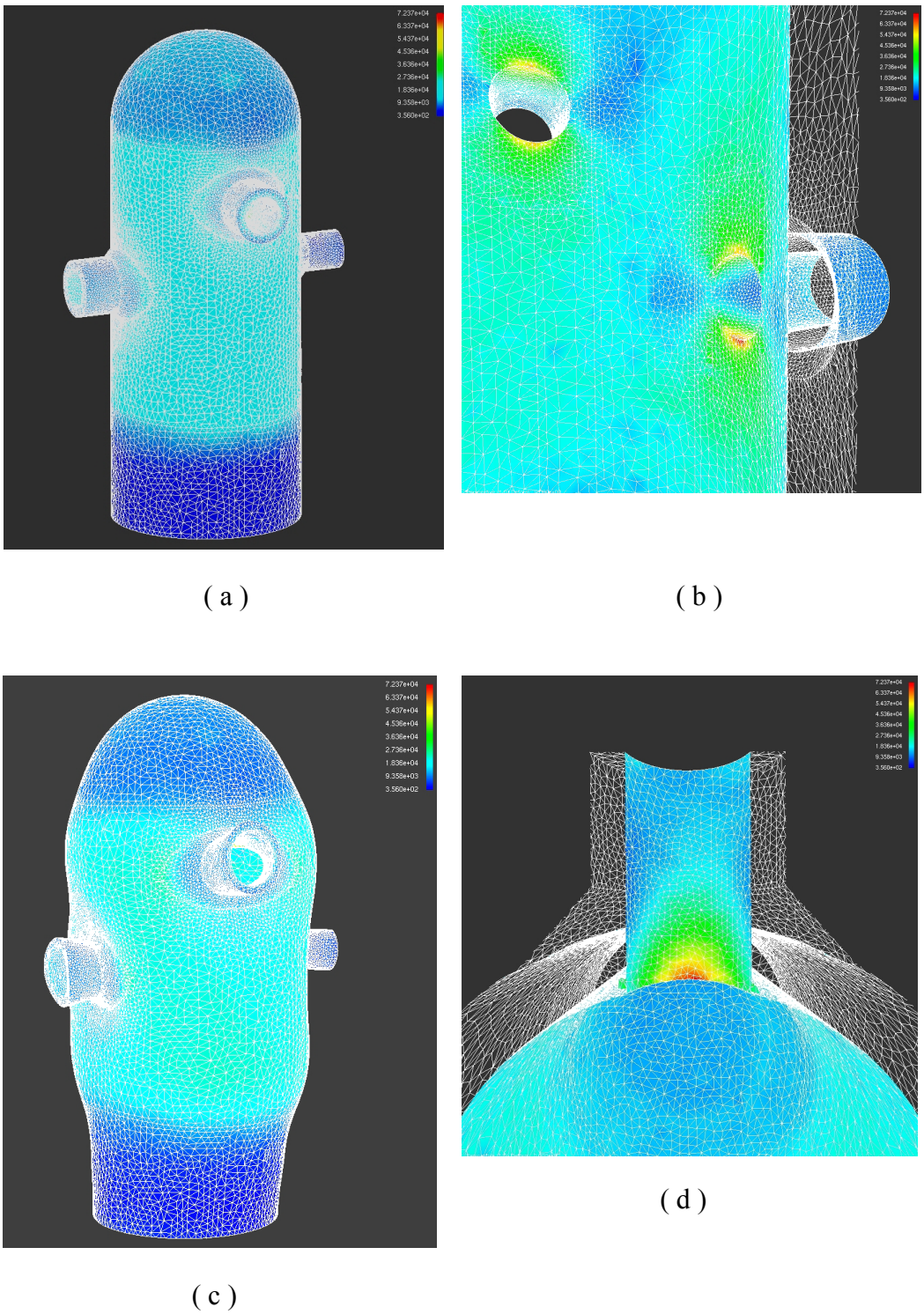
G.Yagawa, H.Kawai and S.Yoshimura (1993), Parallel CAE system for large-scale 3-D finite element, Proc. of SMiRT-12, Stuttgart, Germany, Aug. 15-20, Vol. B, pp.183-194.

Cherri M. Panake (1996), Is Parallelism for You?, IEEE Computational Science & Engineering, Vol.3, No.2, pp.19-37.

Pei-Wen Liu, Lih-Shyang Chen, Su-Chou Chen, Jong-Ping Chen, Fang-Yi Lin, Shy-Shang Hwang(1996), Distributed Computing: New Power for Scientific Visualization, IEEE Computer Graphics and Applications, Vol.16, No.3, pp.42-51.

Jaswinder Pal Singh, Anoop Gupta, and Marc Levoy (1994), Parallel Visualization Algorithms: Performance and Architectural Implications, IEEE Computer, Vol.27, No.7, pp.45-55.

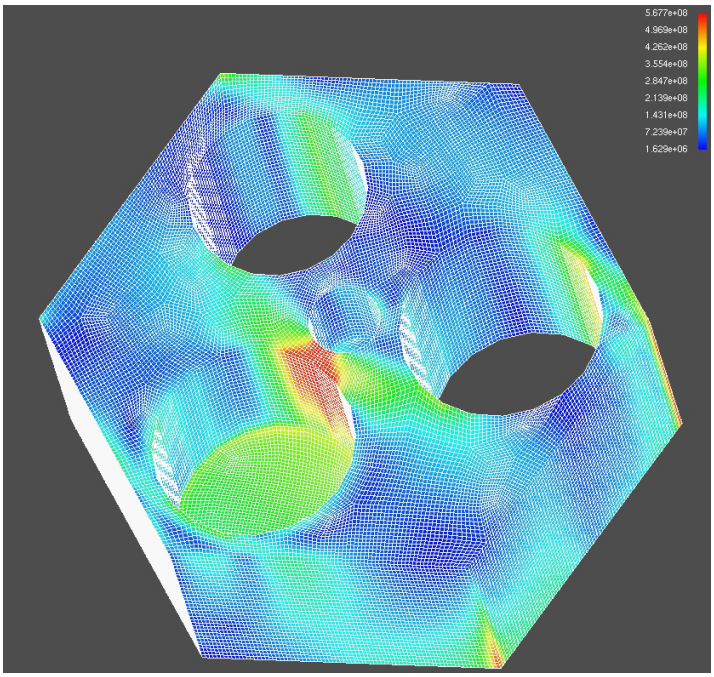
Michel Garland and Paul Heckbert (1997), Surface Simplification Using Quadric Error Metrics, Proceedings of SIGGRAPH 97 (Los Angeles, California, August 3-8,1997), In Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, pp.209-216,



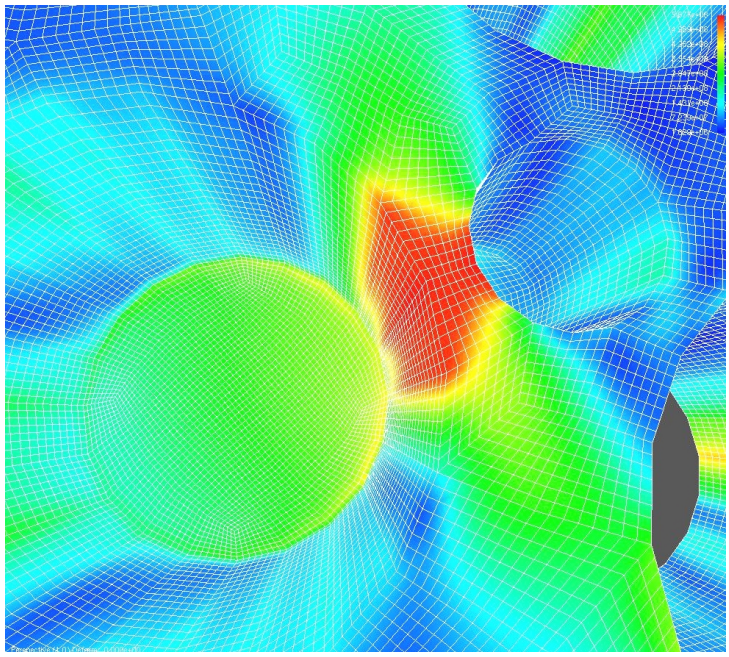
(a) (b)

(c) (d)

Figure 8 (a,b,c,d) Scalar field displays of the equivalent stress from the elastic analysis of a reactor pressure vessel model with a million DOFs.



(a)



(b)

Figure 9 (a,b) Scalar field displays of the equivalent stress from the elastic-plastic analysis of a HTTR model with 1.2 million DOFs.