# GEVP: GRID ENABLED VISUALIZATION PIPELINE

ABOAMAMA ATAHAR AHMED[1], MUHAMMAD SHAFE ABD LATIFF [1], KAMALRULNIZAM ABU BAKAR [1], ABDUL HANAN ABDULLAH [1], ZAINUL AHMAD RAJION [2] AND SIRILUCK LORPUNMANEE [1]

[1] Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia,
Skudai, Johor, 81300, Malaysia
aboamama@utm.my, shafie@utm.my, knizam@utm.my, hanan@utm.my , lorepunmanee@hotmail.com
[2] School of Dental Sciences, Health Campus Universiti Sains Malaysia 16150 Kubang Kerian, Kelantan, Malaysia
zainul@kck.usm.my

***ABSTRACT***
*The current grid computing practices and techniques built on static resource management technique associate with several drawbacks. These drawbacks forced the current grid systems of taking the direction of remote viewing and zooming techniques of already processed images. There is an urgent need for algorithms and techniques to support automatic resources discovery and selection of grid resources. This research investigates how the grid services could be used to support remote visualization. The implemented grid enabled visualization architecture is supported with an automatic resources selection mechanism to enable the visualization pipeline components to be automatically orchestrated without human interference. The presented findings are supported with practical implementation of grid enabled visualization prototype.*

***KEYWORDS***
*Grid computing, Medical Datasets, Visualization Techniques, Resource Management, Resource Discovery, Resource Selection*

## 1. INTRODUCTION

Scientific visualization is a process of transforming numerical datasets into pictorial format understandable by human. This datasets is normally large in its size and algorithmically complex. Therefore, processing these datasets with conventional desktop computer is not sufficient, where the machine will be overwhelmed with intensive processing of large datasets even with the latest development of visualization techniques. Additionally, the rapid increase of complexity and size of datasets make the task of performing real time visualization cumbersome. Moreover, designing visualization system to run on a single machine always results in specialist high cost supercomputers. These high-end resources are expensive and often based in secure location with limited access privileges. Several methods and techniques were introduced in the past few years to tackle the problem of providing computational power required for visualization operations. The introduced techniques result most often in building a cluster of nodes to provide the necessary computational power [1], [2], [3], [4], [5], [6], [6]. Other implementation focuses on remote visualization such as [7], [8], [9], [10], [11], [12].

Most of these implementations suffer from several drawbacks, such as the failure to meet the requirements of real time formation of visualization pipeline and lack of providing the required

DOI: 10.5121/ijgca.2011.2101 

flexible interactivity with visualization scene. These drawbacks are due to the heterogeneous nature of grid nodes and diversity of the environment hosting these grid nodes. Moreover, the different capability of nodes is also an important contributing factor to the load imbalance and difficulties in the pipeline formation.

The implementations of current grid enabled applications raised several issues specifically for real time resource discovery and selection in the grid environment. However, studies focused on implementing the selection of static resource and satisfied with implementing MPI (Message Passing Interface) to direct the jobs to the selected resource such as [13], [14], [6], [15], and [16]. while others rely on the dynamic attributes of the hosts which quered before the lanch of the visualization pipeline . These implementations based on the designing visualization pipeline without considering the run time changes that occur during pipeline execution [17], [18] and [19]. Considering the run time changes of the visualization pipeline attributes is important for obtaining best performance possible for the pipeline. Furthermore, fast request and response operations should be considered during pipeline execution and not only at the initial launch of the visualization pipeline.

Earlier simulation studies such as [20] and [21] were undertaken to provide grid like environment where the proposed platform implemented optimal conditions which could occur in any grid platform, but these implementations failed to meet the heterogeneous nature of real grid environment, that is due to difficulties in simulating unpredicted environment of the grid.

Other implementations focused on scheduling techniques that are known to have significant impact on any time critical system. In addition to other ongoing implementations that focuses on providing prediction strategies and methods, such as [22], [23] and [24]. However, [10] presented a self-adapt framework over wide-area networks. Their work was based on performance estimations and predicting pipeline decomposition on the network. Although their work focused on volume visualizations, but their real time methodology for the pipeline did not consider the automatic selection process. Instead focused on producing analytical framework for the pipeline and neglect load imbalance resulted from the visualization nodes.

The current grid computing practices introduce new challenges for remote real time visualization such as resource discovery and real time automatic resource selection. These challenges result in an urgent need for mechanisms to be associated with the current grid computing practices. The mechanisms should provide automation ability for resource discovery and selection to orchestrate the grid nodes. This paper investigates how the automatic resource selection mechanism could be used to support real time remote visualization of large medical datasets on the grid environment. The presented results supported with Grid Enabled Visualization Pipeline (GEVP) prototype to visualize large medical datasets in real time.

## 2. RELATED WORK

Ever since the first practices of scientific visualization, scientists are keen to take advantage of high performance computing technologies. The scientist enthusiasm sparked several areas and created unprecedented challenges for developers which have never been explored. The sparkles of the visualization systems from early line plots to very advanced software volume rendering and surface extraction techniques, have enormously affected the speed and the development of high performance computing capabilities. However, there are a few historic standpoints where the development had to take a major turning in its directions. This steering development mechanism has no control and theoretical ending point, but has very complex freezing points. Recent implementations show complexity even with very advanced visualization algorithms where it exceeds the capabilities of modern hardware. The only direction for software

developers is to take advantage of recently developed high speed networks and internet. Unfortunately, the utilization of these communication technologies have several issues such as the distribution of the system components throughout the entire architecture. A good example of this complexity is in the field of bioinformatics where visualizing large datasets demand high computational powers and high speed data transfer techniques. Recent development of visualization applications was sparked by landmark NSF report Visualization in Scientific Computing by [25]. The report was based on the proof of concept which highlighted the fact of great advantages behind breaking the visualization components into smaller interconnected visualization processes. The usefulness of decomposing the visualization tasks into smaller processes allows the processes to be distributed and placed in different computing nodes. Additionally, the nodes could be located in geographically dispersed environments. However, this concept presents an interesting solution which seems to take full advantage of the existing network capabilities, but keeping in mind different network topologies with different hardware and software architectures on the running nodes presents numerous challenges. These challenges are categorized into two major categories. The first category related to optimum utilization of resources with different hardware and software architectures. The second category related to the used of visualization components and its flexibility to adopt the changing attributes in the hosting environment. Although a few papers have been published in the area of resource discovery and selection of grid resource, but the interest of this research area enforces the practical advantages of the solution which can be adopted in a very large scale. However, the important publication in the area was practically started by [26]. The work was focused on decentralized resource discovery. The mechanism for the discovery was based on a flat, decentralized, self-configuring architecture. The discovery is done when the requests are transferred from one node to another in a hierarchical fashion. The node responds to the requests either by matching resource or passes the request to the next node in the network. This process continues to the deadline of time-to-live (TTL) of the requests. Other implementation was carried out by [23]. Their work was focused on the introduction to the concept of Grid potential. Their implementation was based on adaptive control the extent of data dissemination in a Grid. Their main aim was to encapsulate the relative processing capabilities of the different machines and networks that constitute the Grid. However, the existing grid enabled visualization systems are in the direction of translating the existing data-flow concept presented by [25] as described in Figure 1.

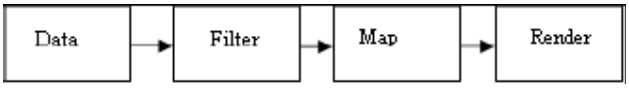


Figure 1. Haber and McNabb Visualization Pipeline

## 3. PARALLEL VISUALIZATION PIPELINE AND GRID

The parallelism in GEVP is achieved through providing multiple sequential pipeline work simultaneously in parallel to process the assigned partition of the datasets. Figure 2 shows the parallel visualization pipeline configuration on the grid. The differences between this implementation and conventional cluster implementations are in a form of extra parameters needed for adjusting the speed of program execution in parallel environment. Additionally, individual machines distributed in the grid are not identical as in the cluster where there are major differences in the processing power and memory and other performance related specification. Based on that, there is an obvious need for a mechanism which able to deal with

the numerous parameters related to the nodes attributes and the processed datasets In GEVP this mechanism is provided by resources discovery and selection techniques.

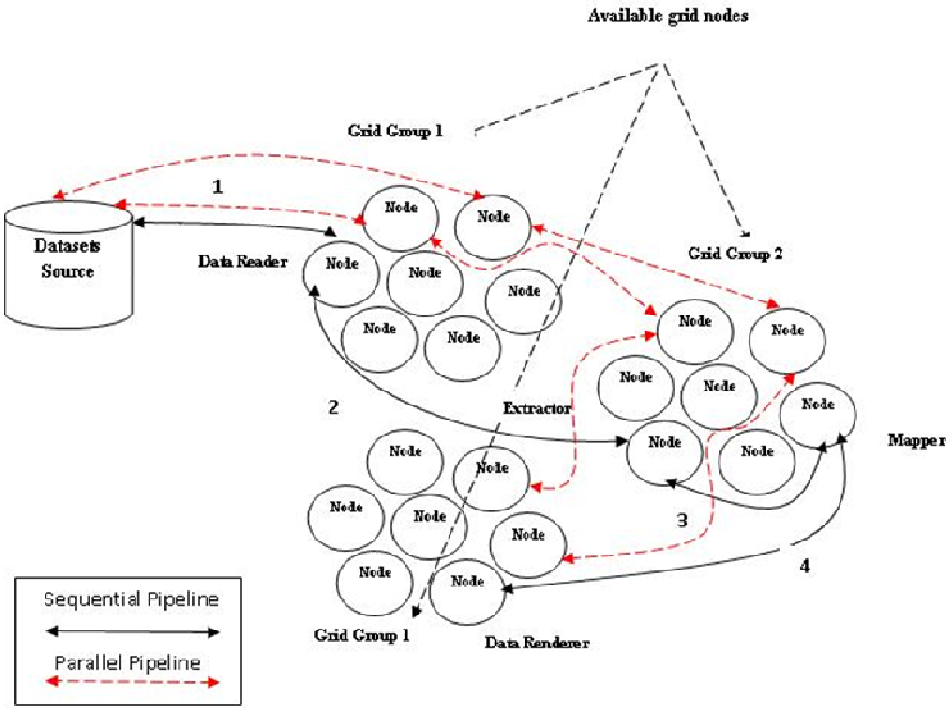


Figure 2. Parallel Visualization Pipeline Selection

Figure 2 shows the operation involved in parallel visualization pipeline. The pipeline starts with parallel data reader to read the datasets from the dataset service. The datasets then passed to distributed extractors in the grid. These extractors could be based in different location. The datasets then passed to data mapping services to assemble the datasets then passed to the rendering and then to the display on the client.

## 3.1. Distribution of Parallel Visualization Pipeline

Figure 3 shows the initial Isosurface drawing requests containing dataset address locations and Isovalues. The steps involved in parallel visualization support reading the datasets in parallel. This permits the distribution of the grid services in this implementation to be more efficient than the sequential visualization pipeline. This is due to the ability of parallel data reader service in assigning more extractors than the sequential implementation.

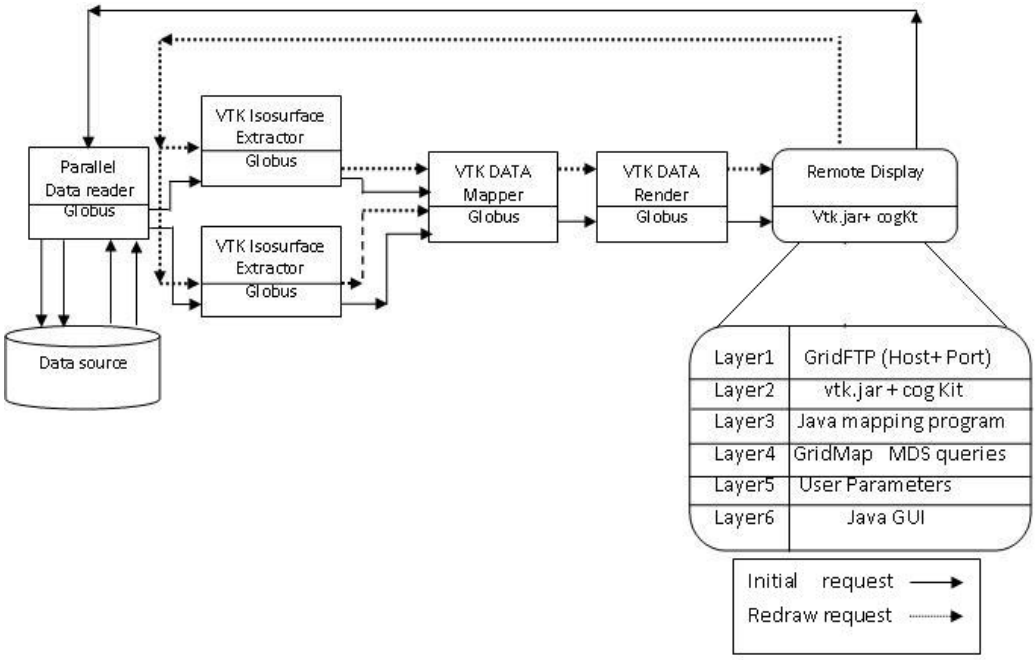


Figure 3. Grid Visualization Pipeline with Parallel Data Reader and Redraw Process Requests of an Isosurface

The remaining services such as Mapping and rendering and the display client perform similar operation. Although mapping service in this case receiving more than one input from a number of extractors, but the rendering services are similar to the sequential pipeline.

### 3.2. Test-bed Implementation for Parallel Visualization Pipeline

The used resources for testbed implementation include 2 HP workstations equipped with NVidia GeForce 4MX Go graphics, 512 MB of RAM and 2.87 GHz CPU running on Linux RedHat 9, and one with NVidia nv10 GeForce 256 SDR graphics card , 256 MB of RAM running Linux Fedora core 3. At the client user, HP Notebook is used and it is equipped with Intel(R)Pentium(R)4 CPU 2.80GH, Graphic Adapter ATI Mobility IGP 340M/345M , 512 MB of RAM and ST94011A 40 GB disk drives that ran on Windows XP Professional. All the machines were linked with LAN cable 100MB Ethernet LAN. Extra nodes for the extraction operations. The surface extraction is performed on two nodes equipped with NVidia nv10 GeForce 256 SDR graphics card, 256 MB of RAM running Linux Fedora core 3 Same node for the display client.

### 3.3. Experimental Results for Parallel Visualization Pipeline

Table 1. Models Used in Benchmarks.

| Model Name | Number of Polygons | Size of Data File |
|---|---|---|
| Skeleton head | 4.28 million | 15.1MB |
| 3D Dental Scan | 14.62 million | 58 MB |

The first raw skeleton datasets consist of 121 slices of 256 * 256 * 256 producing the file size of 15.1MB. The second model is 3D dental scans consist of 167 slices of 256 * 256 * 256 producing the file size of 58 MB. The datasets are firstly read in parallel by the Parallel Data reader services. For 3D model of dental scans datasets during the experiment. The data reader services partition the datasets into two portions. The slices 1 to 81 read concurrently with the slices from 82 to 167 data reader service. The implemented Parallel Data Reader Service is based on vtkVolume16Reader Java class and modified to read the datasets in parallel and work as grid services located at skudai.fsksm.utm.my node, which serves as the starting node of the pipeline. The portions of the datasets are then sent to the assigned Isosurface Extractors immediately after completing the reading stage. Isosurface Extractors use marching cubes and a polygon decimation algorithm for geometry generation the vtkmarchingcubes algorithm was used to extract the Isosurface from the supplied datasets and vtkDecimatePro to reduce the number of produced polygons from the previous step. For better illustration purpose, Figure 4 Dental Scans Datasets shows the visualization client with two visualized objects. The object on the left shows the lower part of the dental scan processed on mewah.fsksm.utm.my with Isovalue 1600. Additionally, the object on the right of the same figure is processed on kulai.fsksm.utm.my on a machine with the same Isovalue. Moreover, its is the same condition for figure 5 and the only difference is that it is processed with Isovalue 600. Figure 6 shows the visualization client with combined portions of the portioned datasets on the client machine. However, the Mapping Service is needed before the datasets are sent to the visualization client to reduce the load of appending datasets to form one single object. vtkAppendFilter is used at the mapping service to append the datasets together.

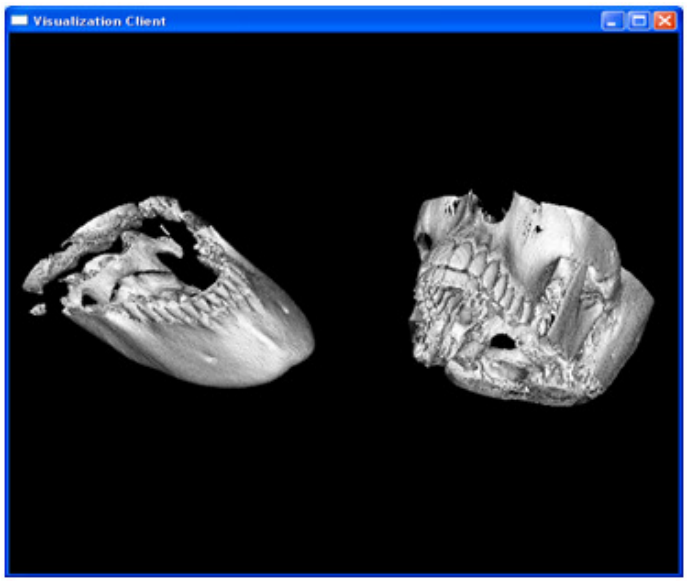


Figure 4. Dental Scan with Isovalue 1600

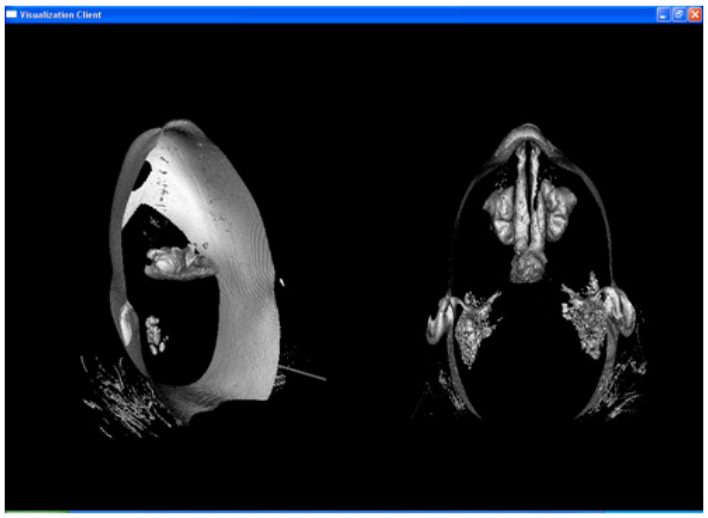


Figure 5. Dental Scan with Isovalue 1600

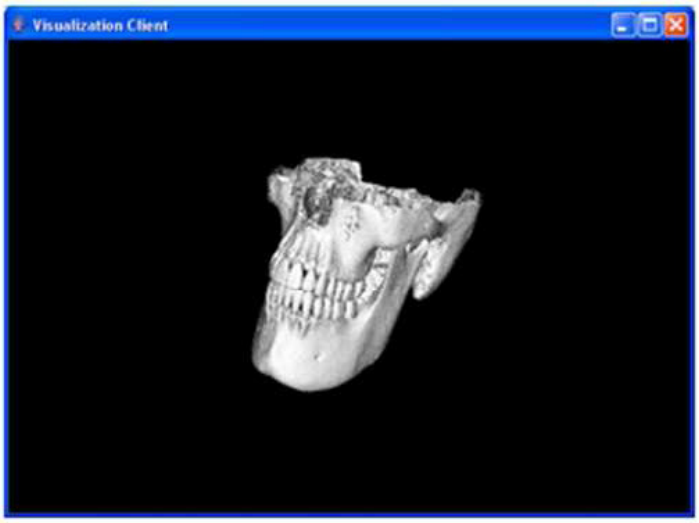


Figure 6. Dental Scan with Isovalue 1600

This stage is also responsible for mapping the produced polygons to the selected Render Services and then to the display client. Figure 7 shows the performance of the selected Isosurface Extractors for dental scan datasets and the number of produced polygons for each Isovalue. In The Figure the node skudai.fsksm.utm.my (A) performs the reading operation where the load is not noticeable due to the fast reading operations in parallel during the pipeline execution. Mewah fsksm.utm.my (B) and Kulai.fsksm.utm.my(C) used for extractions operation where the load is noticeable and the differences of the load in the nodes is due to different attributes of the nodes. The overall grid nodes performance is shown in the figure in form of memory and number of processes.

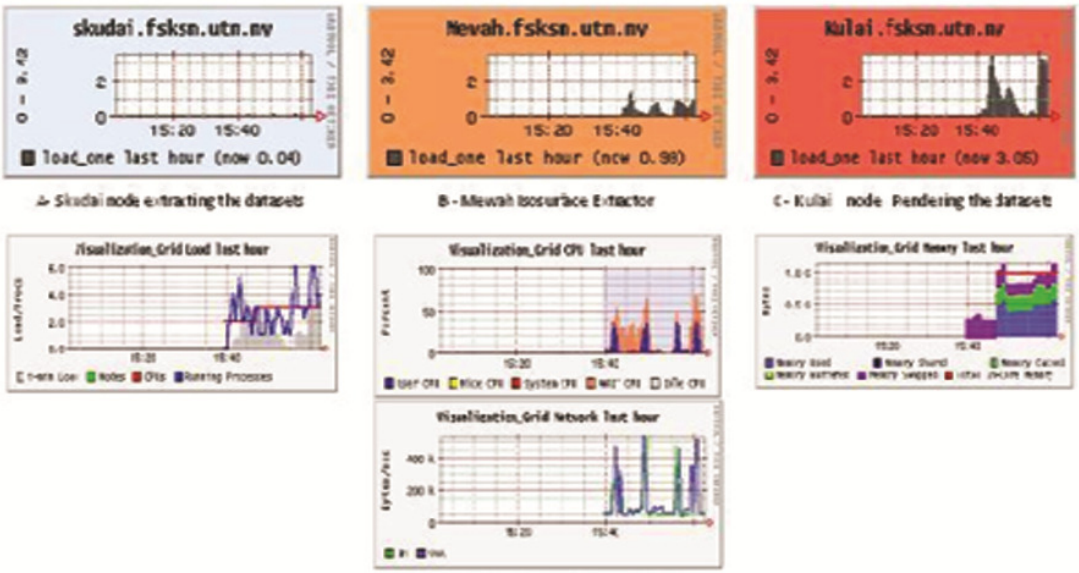


Figure 7. Pipeline Performances for Parallel Visualization Pipeline

## 4. MANUAL AND AUTOMATIC VISUALIZATION PIPELINE FORMATION AND RESOURCE SELECTIONS

The requirements of resource selection process from visualization pipeline include how the visualization pipeline components are constructed to cooperate and answer other architecture components requests. The design and placement of the visualization pipeline components should be based on best performance of architecture and not to the randomly placed components in the distributed nodes. Current manual formation of visualization pipeline perform manual configuration of the pipeline components. Therefore, the entire resource selections process is carried out manually. Unfortunately, this manual selection process of the resources causes several drawbacks to the overall performance of the grid architecture. These drawbacks could cause further disadvantages that will give unstable performance to the overall architecture performance.

The disadvantages of manual selection of the resource to construct the visualization pipeline are related to the components configuration and size of the used datasets. The first disadvantage is in the difficulty of selecting the right nodes manually when there are too many discovered nodes in the grid. The users have no prior knowledge of the nodes specifications. Therefore, the users have no clue of which node that has better performance than the other. Thus, manual selection of the nodes is impractical when the number of nodes is large.

The second disadvantage relates to the performance of the selection process. The speed of selection process has great impact of the overall architecture performance. In addition to that, the uncertainty of which resources to select extends the configuration time and this affects the pipeline initiation time. Additionally, the desired selection process is not only based on the distribution of visualization pipeline components, but on the automation of selecting available resources. Therefore, a mechanism is required to automate the selection process. This mechanism should be based on some realistic assumptions, such as the dynamic available

memory, CPU cycle and storage. Although similar grid enabled visualization application claimed to have successful implementations of these techniques, but the drawbacks of their implementations is that the selection of the resources are completely performed manually. On the other hand, the distribution of the visualization pipeline components is hard to be predicted due to the heterogeneity and uncertainty of the number of the available nodes. This is caused by the anonymity of the number of nodes that make the grid visualization pipeline. The only way to tackle this anonymity is to force the grid nodes to response to applications or systems requests rather than adjusting every resource available in the grid pool to suit some application or specific system conditions.

Furthermore, the involved node in the selection process are associated and operated with different loads and characteristics. This makes the process of adjusting and predicting the load of computing nodes is hard before the runtime of the applications or the systems. Therefore, the computing nodes should always have the state of visualization capable that means the required components to perform the visualization already enabled in the nodes. As an example in this research the preparation in node to perform the surface extraction or rendering or mapping needed is configured in advance and given the visualization ready state (VRS).

Therefore, the choice of selecting a process on the node is totally dependent on the used selection technique. In addition to that, the technique should always be based on the processed datasets first and the number of machines being discovered with the enabled visualization components. Additionally, the speed of the selection is also based on the used technique and the number of the accepted parameters by the used algorithm. The implementation of GEVP described in this paper utilizes several parameters involved in the selection process for real time visualization process. The parameters are presented in a form of the attributes of the datasets. These attributes are categorized into fixed and dynamic attributes. Fixed attributes consist of size of datasets, the number of slices, the prefixed names and location. The dynamic attributes which are related to the changing parameters during the system runtime is the changing Isovalue that redraws the surface for the same datasets. This changing parameters result in different performance for static manual implementations of the selection process in comparing with the dynamic automated selection process.

Figure 8 shows the result of automatic selection in comparison with manual selection for the available nodes. However, the selection result does not include all active discovered nodes but only the selected nodes to participate in the Isosurface extraction for the visualization pipeline. Therefore, the used algorithm should be accurate in classifying the node according to the needed specifications to process the datasets. Additionally, the selection might be different from one particular dataset to another unless the specified dataset requests the same computational load that might lead to the selection of different number of nodes. The reason for that is the load for participating nodes in the visualization pipeline may change at any moment during the pipeline execution. Figure 8 shows the different time needed to process the datasets with automatic selection which is obviously faster than the manual selection process. The performance is based on the time needed to process the polygons for specific datasets.

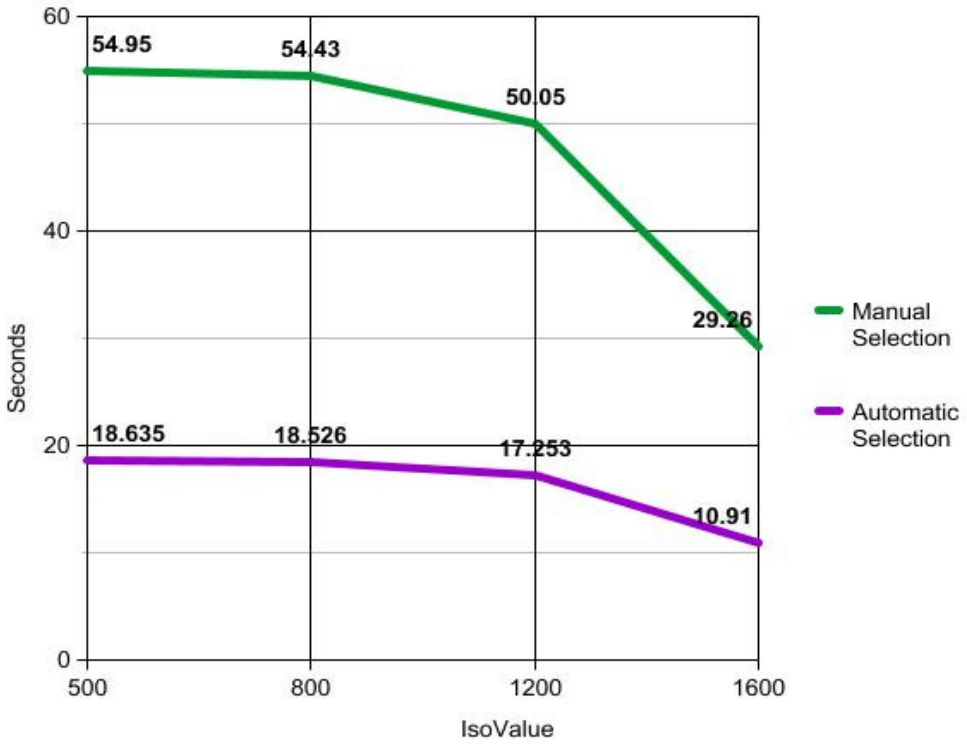


Figure 8. Automatic Resource Selection And Manual Resource Selection

Additionally, the timing difference between the Isovalue is related to the density of the datasets where the density is high with the Isovalue 500 and low for the Isovalue 1600. The reason for that is the amount of produced polygons is high with Isovalue 500 due to extraction of the polygons responsible for visualizing a skin tissue. Therefore, the time needed is 18.6 seconds to visualize these polygons. Consequently, for the same datasets with the Isovalue 1600 the time needed was 10.9 seconds and the produced number of polygons (circa 11 million polygons). This decrease of extraction time and is because of the exclusion of the skin tissues in the visualization of this datasets. This is specifically due to not extract the polygons responsible for skin tissues as in the case for Isovalue 500.

In addition to the recorded time spent for every Isovalue to extract and visualize the data there is also time spent for reading the datasets and mapping the generated polygons which is also included in the recorded time. Furthermore, the recorded time is also relies on the number of used resources for extraction where the parallel extraction time is unlike reading time which in some cases is intangible.

## 5. CONCLUSIONS

The current grid enabled systems suffer from numerous limitations such as providing zooming and viewing techniques of remotely stored static images. Additionally, some of implemented grid enabled visualization systems and applications tuned to take advantages of the participating client nodes in the grid which is already suffers from lack of resources while other implementation tuned to extend the existing implementations and provide grid services to allow access to systems functions. On the other hand, some implementations provides conventional

client server methods under grid enabled systems where they allowed third party systems and client to take advantages of servers for processing large datasets and clients for providing control mechanism for operations carried in the server. the architecture proved to have significant decrease of the time required to form the automatic visualization pipeline and datasets extraction. The results from automatically selected visualization pipeline nodes are compared with the manual selected pipeline nodes which specifically designed to show the major difference in the performance between the two pipelines. This increase in performance for the time need to extract the medical datasets proves the significance behind applying the mechanism of resource discovery and selection technique. In addition to the adopted design for the architecture which allow the various measurements to be conducted where it was practically not possible in grid environment.

## REFERENCES


[1] F. P. Gregory, *In search of clusters: the coming battle in lowly parallel computing*: Prentice-Hall, Inc., 1995.

[2] C. P. B.Wylie, V. Lewis and K. Moreland, "Scalable rendering on PC clusters," *IEEE CG&A 21(4),* pp. 62–69, 2001.

[3] E. B. L. Muraki S. , K.-L. Ma, M. Ogata and X. Liu, "A PC cluster system for simultaneous interactive volumetric modeling and visualization.," *In Proc. IEEE Symp. Parallel and Large-Data Visualization and Graphics,,* pp. 95–102, 2003.

[4] I. Merelli, L. Milanesi, D. D. Agostino, A. Clematis, M. Vanneschi, and M. Danelutto, "Using parallel isosurface extraction in superficial molecular modeling," in *Distributed Frameworks for Multimedia Applications, 2005. DFMA '05. First International Conference on*, 2005, pp. 288-294.

[5] J. L. Williams and R. E. Hiromoto, "Sort-middle multi-projector immediate-mode rendering in Chromium," in *Visualization, 2005. VIS 05. IEEE*, 2005, pp. 103-110.

[6] K. Engel, O. Sommer, C. Ernst, and T. Ertl, "Remote 3D Visualization using Image-Streaming Techniques " *Proceedings of the International Symposium on Intelligent Multimedia and Distance Education,* 1999.

[7] W. Bethel, B. Tierney, J. Lee, D. A. G. D. Gunter, and S. A. L. S. Lau, "Using High-Speed WANs and Network Data Caches to Enable Remote and Distributed Visualization," in *Supercomputing, ACM/IEEE 2000 Conference*, 2000, pp. 28-28.

[8] M. P. B. Randy W. Heiland, and Danesh K. Tafti, "VisBench: A Framework for Remote Data Visualization and Analysis," *Springer-Verlag Berlin Heidelberg,* pp. 718 -727, 2001.

[9] N. J. Avis, I. J. Grimstead, and D. W. Walker, "Grid Enabled Remote Visualization of Medical Datasets," *Medicine Meets Virtual Reality The Magical Next Becomes the Medical Now,* vol. Vol 111, pp. 22 - 28, 2005.

[10] Q. Wu, J. Gao, M. Zhu, N. S. V. A. R. N. S. V. Rao, J. A. H. J. Huang, and S. A. I. S. Iyengar, "Self-Adaptive Configuration of Visualization Pipeline Over Wide-Area Networks," *Computers, IEEE Transactions on,* vol. 57, pp. 55-68, 2008.

[11] B. H. McCormick, "Visualization in scientific computing," *SIGBIO Newsl.,* vol. 10, pp. 15-21, 1988.

[12] I. Adriana and T. F. Ian, "On Fully Decentralized Resource Discovery in Grid Environments," in *Proceedings of the Second International Workshop on Grid Computing*: Springer-Verlag, 2001.

[13] I. Foster and N. T. Karonis, "A Grid-Enabled MPI: Message Passing in Heterogeneous Distributed Computing Systems," in *Supercomputing, 1998. SC98. IEEE/ACM Conference on*, 1998, pp. 46-46.

[14] I. Bowman, J. Shalf, K.-L. Ma, and W. Bethel, "Performance Modeling for 3D Visualization in a Heterogeneous Computing Environment," LBNL--56977; R&D Project: K11107; TRN: US200513%%589, 2004.

[15] S. Hastings, T. Kurc, S. Langella, U. A. C. U. Catalyurek, T. A. P. T. Pan, and J. A. S. J. Saltz, "Image processing for the grid: a toolkit for building grid-enabled image processing applications," in *Cluster Computing and the Grid, 2003. Proceedings. CCGrid 2003. 3rd IEEE/ACM International Symposium on*, 2003, pp. 36-43.

[16] J. Krystynak, "High speed network issues in a distributed visualization application," in *Proceedings of the 1992 ACM/IEEE conference on Supercomputing* Minneapolis, Minnesota, United States: IEEE Computer Society Press, 1992.

[17] J. G. Ian, J. A. Nick, and W. W. David, "Automatic Distribution of Rendering Workloads in a Grid Enabled Collaborative Visualization Environment," in *Proceedings of the 2004 ACM/IEEE conference on Supercomputing*: IEEE Computer Society, 2004.

[18] G. Andreas, A. Mark, K. Torsten, B. Christian, L. Stefan, and B. Thomas, "Conceptual design and implementation of a pipeline-based VR-system parallelized by CORBA, and comparison with existing approaches," in *Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry* Singapore: ACM, 2004.

[19] A. Wierse, "Performance of the Covise Visualization System under Different Conditions," *SPIE 2410,* pp. 218-229, 1995.

[20] J. G. B. H. R. Sosic, "Nimrod: A Tool for Performing Parametised Simulations using Distributed Workstations," unknown, 1995.

[21] H. B. William, G. C. David, C. Luigi, A. P. Millar, S. Kurt, and Z. Floriano, "Simulation of Dynamic Grid Replication Strategies in OptorSim," in *Proceedings of the Third International Workshop on Grid Computing*: Springer-Verlag, 2002.

[22] E. Elmroth and J. Tordsson, "Grid resource brokering algorithms enabling advance reservations and resource selection based on performance predictions," *Future Generation Computer Systems,* vol. 24, pp. 585-593, 2008.

[23] Y. H. Libing Wu, Jianqun Cui , Simeng Wang , Laurence T. Yang and Naixue Xiong "A Resource Discovery Algorithm with Probe Feedback Mechanism in Multi-domain Grid Environment," *Springer Berlin / Heidelberg,* vol. Volume 5036/2008, pp. 178-186, 2008.

[24] J. Gao, J. Huang, C. R. Johnson, and S. A. A. S. Atchley, "Distributed data management for large volume visualization," in *Visualization, 2005. VIS 05. IEEE*, 2005, pp. 183-189.

[25] R. B. Haber and D. A. McNabb, "Visualization Idioms: A Conceptual Model for Scientific Visualization Systems "Visualization in Scientific Computing"," *IEEE Computer Society Press Tutorial,* pp. 74--93, 1990.

[26] T. Sanya, K. Takahiro, K. Kenji, H. Hiroki, and Y. Toshitsugu, "A time-to-live based reservation algorithm on fully decentralized resource discovery in grid computing," *Parallel Comput.,* vol. 31, pp. 529-543, 2005.