3-D MODELS USING INTEGRATION OF GRID COMPUTING

S. Hannah Pauline
Assistant Professor
Electronics and Communication Engineering
SRM Institute of Science and Technology, Chennai, India

Abstract: - *Grid Computing is an advanced technique that uses the concept of distributed computing. A collection of computers, storage and other devices that are joined together by means of communication like the internet is called the grid. This is also used to solve the problems among themselves and manage the information.Supercomputers and parallelismare the opted by various researchers and organizations to help deliver the outputs of slow and computationally-intensive tasks at a faster pace. One of the typical scenarios where high performance computing resources are used to complete the job fast and efficiently is that of 3D rendering. The resources available are combined and utilized by the technique of parallelism like grid computing which helps in the processing of computationally-expensive processes and simulations. Accessibility for the available resources can be improved by integrating Service-Oriented Architecture (SOA) into grid computing. This paper demonstrates how SOA and grid computing can be integrated to implement the rendering of POV-Ray 3D models efficiently.*

Keywords: *Grid computing*

1. INTRODUCTION

Computer graphics is one of the most challenging but interestingbranches in the field of computer science. Both researchers and other techs work towards making the best use of computing resources to produce high-quality outputs that they can directly or indirectly use. Rendering is the process of creating an image using a 2D or 3D model by means of computer programs. The process of converting 3D wire frame models into 2D images automatically with 3D realistic effects on a computer is called3D rendering. This is a slow and intensive process that usually comprisesa large number of complex calculations. The features like lighting, painting, reflection, shadowing, transparency and many others are defined through these calculations.

Thus, this limitation has challenged 3D animation companies to explore for solutions that would decrease the rendering time. For instance, some companies had to spend on the hardware like supercomputers and accelerated graphics processing units (GPUs) ,whereas some other companies had to outsource the rendering process to commercial rendering services online (such as RenderFlow, RebusFarm and ResPower) to reduce the time taken for rendering . These commercial services deploy powerful farms that are composed of hundreds of computers solely dedicated to the rendering process.

Ray tracing is a technique meant for generating high-quality, photorealistic images by using the concept of computer graphics. It traces the path of light through the image plane and simulates the optical effects of the objects in the scene. Parallelism can help concurrently render different slides of complex 3D models on different

Received on : 05.06.2020
Reviced on : 16.07.2020
Accepted on : 08.08.2020
Published on :09.08.2020

Corresponding Author: **S. Hannah Pauline** [hannahpauline.s@ktr.srmuniv.ac.in]

processors/cores that would generate the final images and scenes much faster. Grid computing is a parallelization method that can be utilized to make 3D images and films in less time. This goal can be achieved by networking a number of machines to produce different images of a complete scene simultaneously.Ray tracing produces high-quality images with realistic reflections, shading, andperspective. Computational efficiency of rendering is achieved by fitting the grid service model into a Service-Oriented Architecture (SOA) through design patterns. An architectural pattern in which the application components of a device, provide services to other components via a communication protocol, is called a service-oriented architecture (SOA). The services built in every computer ensure that the service can exchange information with any other service in the network without human interaction or involvement. This is possible without the needing to change the underlying program. The recent development shows the integration of grid into it.

2. GRID-ENABLING THE RENDERING PROCESS:

3D rendering of a large, high-resolution, deep image, which contains many details and visual effects can take several hours to execute on a single computer. In the same way 3D rendering of one short, high resolution scene might take days to complete on a single computer. The ability to aggregate the computing power of the underutilized computers existing at enterprises can reduce the time taken in the rendering phase significantly, while also increasing the return on investment (ROI) of these resources.

1. A user (programmer or artist) submits the created ".pov" file to the desktop grid either through the GUI layer laid on top of the grid-enabled application or by manually copying it to a known path. Regardless of whether the ".pov" file is passed to the grid by the grid-enabled application or manually copied, it must be set in a location that can be accessed by all grid nodes (i.e; network folder or database).
2. The grid-enabled application creates the grid jobs according to the settings specified by the user and then forwards them to the grid.
3. The manager schedules the jobs and sends them to the free executors.
4. POV-Ray must be installed on each executor prior to the implementation of this model.
When an executor receives a rendering job from the manager, it launches POV-Ray in a separate process and passes the arguments accompanied by the job.
5. POV-Ray renders the job, and when the output result is ready, the launched process exits itself and the executor returns the results back to the manager.
6. The manager then forwards back the rendered outputs to the grid-enabled application and fires "ThreadFinished" event if successfully completed or "ThreadFailed" event if an error has occurred. The implementers should save the rendered jobs on shared media that all executors can access.
7. When all jobs are completed, the manager fires the "ApplicationFinished" event. This event enables programmers to post-process results and release resources when the whole application completes.

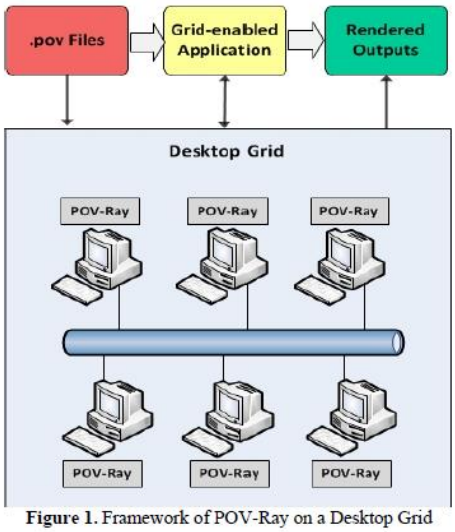

Figure 1. Framework of POV-Ray on a Desktop Grid

Each image is composed of a set of rows and columns of pixels – known as the image resolution. The ability to slice an image up into a set of meaningful segments in order to concurrently render them on different computers, can drastically decrease the time needed to generate the final output.During the implementation, the authors decided to subdivide the rendering task of one image into a number of subtasks where each subtask encompasses a set of rows. That is, our algorithm has three consequent phases: Render, Crop and Combine (RCC).
The fig.2 shows an example of parallel image rendering.

1. The end user specifies the height and width of the target image along with the passed ".pov" file. In addition, the user can define the number of rows per job and pass it to the client application.
2. The client application creates the grid jobs and submits them to the manager node. Each job has a rendering command for a portion of the target image.
3. The manager node passes the queued jobs to the connected executors.
4. Each executor launches the POV-Ray application by starting an external process in Windows using the/ a "Process" class provided by .NET Framework.
5. POV-Ray generates the (partial) image according to the specified options and then saves it to the specified output path. Implementers should note that all grid nodes should have a shared output path so that the executors can save the rendered outputs. This can be accomplished by using a network drive or a database where all nodes have access to.
6. Each of the completed jobs generates a full-sized image which has an empty (black) portion in addition to the segment that contains the rendered data.
7. After the completion of a job, the client application crops the rendered image to create a smaller version, in order to keep the non-empty segment.
8. If any of the jobs fail, the client application will queue them on a temporary list to re-process again, either on the grid or the client machine.
9. After the successful completion of all jobs, the client application merges the cropped images according to their original arrangement to produce the final image.

3. ARCHITECTURE

User Interface: This layer enables the end user to pass the rendering parameters (e.g., the ".pov" file, output resolution, and the output path) and to display the list of rendered images. ASP.NET along with JavaScript and AJAX technologies are used to create a user interface layer.

Web Service: ASP.NET and WCF are used to build a web service layer. A RESTful web service is responsible for accepting the rendering parameters and giving them to the rendering APIs. This layer is a major component in this scenario as it enables authorized users to access the grid re-sources from any location, simply by invoking the right operation and passing the relevant parameters.

Grid Resources: This layer includes the computational power that will be used for parallel rendering of the images. Grid resources may include dedicated (or idle) desktop machines, laptops and servers.
Storage: This is where the rendered images (and any related information) will be stored. Storage layer may include databases, file servers, or both.
The fig.4 depicts the grid architecture.

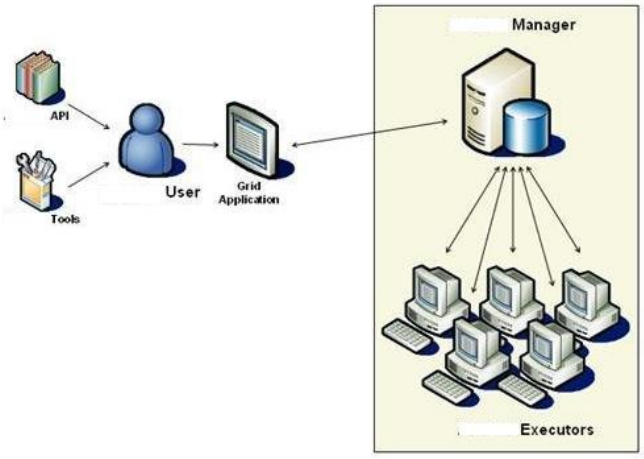

Figure 4:grid architecture

4. PERFORMANCE EVALUATION:

The performance is enhanced when grid is used for the process of image rendering. When small images have to be rendered, the performance does not vary much. But, as the image grows larger, the time taken to render the image without grid is more. Thus, grid computing is more efficient and less time consuming for image rendering than without it.

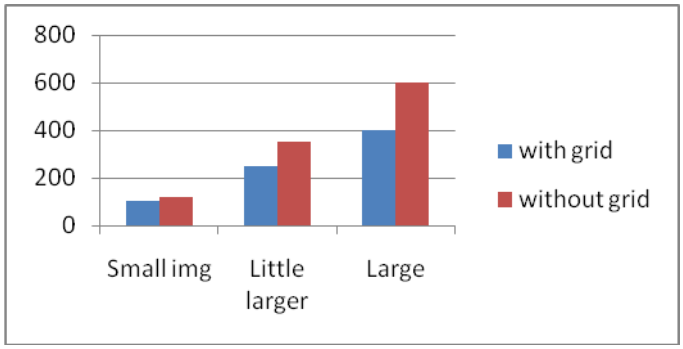

Figure 5: graph showing the variation in performance

5. CONCLUSION AND FUTURE WORK:

Results show that the speed of the 3D rendering task of images increases proportionally with the number of connected workers. This results in the implementers gaining the ability to render the presented images (and animation frames) in a few seconds if implementers can build a grid of 50 or 100 computers. Building such a grid is trivial since most organizations, universities and research labs usually have more than 50 computers. With a semi-real time rendering for complex models, organizations can do more work in much less time, and without increasing costs.

REFERENCES

1.Distributed.net [2002]http://distributed.net/

2. De Roure, M. A. Baker, N. R. Jennings and N. R. Shadbolt .The Evolution of the Grid [2002]http://www.ecs.soton.ac.uk/~nrj/download-files/evolution-of-grid.pdf.

3. Foster. ”Internet Computing and the Emerging Grid”.*Nature*.7, December.[2000]http://www.nature.com/nature/webmatters/grid/grid.html.

4.Globus projectweb siteUniversity of Chicago [2002]http://www.globus.org

5. L. Ferreira and V. Berstis.“Fundamentals of Grid Computing” IBMRedbooks.IBM Publications Center. [2002].http://publibb.boulder.ibm.com/Redbooks.nsf/RedpaperAbstracts/redp3613.html

6.Sun MicrosystemsSun Cluster Grid Architecture [2002].http://wwws.sun.com/software/grid/SunClusterGridArchitecture.pdf

7.S. Tanenbaum. Modern Operating Systems. Upper Saddle River,New Jersey. Prentice-Hall. [2003.]

8.S. Tanenbaum. Computer Networks.Fourth Edition. Upper SaddleRiver, New Jersey. Prentice-Hall. [2002.]