

# Rendering 3D Computer Graphics on a Parallel Computer

J. Tindle. K. Ginty. S.J. Tindle

*Department of Computing, Engineering and Technology, Faculty of Applied Sciences,  
University of Sunderland, St. Peter's Campus, Sunderland, SR6 0DD, U.K.  
(e-mail: john.tindle: kevin.ginty@sunderland.ac.uk)*

**Abstract:** Using the cluster computing facility at the University of Sunderland, the performance of four commercial applications for 3D computer graphics rendering are compared. While these applications employ different approaches to the problem of computer graphics rendering, some common issues are identified. The rendering process is described and various ways to use system resources investigated and compared. Common themes concerning Operating System performance and hardware configuration are reviewed. Results of tests conducted using the embedded rendering engines supplied with each application are presented.

**Keywords:** High performance cluster computer, 3D computer graphics, model, animate, render, networking, grid.

## 1. INTRODUCTION

This paper describes experience gained when experimenting with the use of a parallel computer to render 3D computer graphics. Nowadays, 3D computer graphics modelling and animation finds application in many different areas, particularly in engineering and media.

In mechanical engineering, for example, 3D modelling is used to visualise mechanical components such as gears and bearings. Designers are able to check the fitness of the interface between components prior to actual manufacture. This approach can help to avoid costly rework caused by design errors.

Similarly, it is now common practice for architects to create a 3D model of a proposed development to be viewed by the potential client. This allows the client to "walk through" the 3D model and make recommendations for changes to meet their requirements. Expenditure on the model is usually relatively small in comparison to the actual build cost, for example, of a riverside development with hotels, housing, public parks and avenues.

DCET (Department of Computing, Engineering and Technology) recently created an animation showing the inside of a performance hall. By observing the animation, it was possible to evaluate the view of the stage from various locations in the hall.

## 2. RENDERING AND 3D MODELLING

This paper primarily considers 3D computer graphics modelling and animation. To create a 3D model, wire frame objects are placed in a 3D world and light sources are positioned in the space. Fig. 1 shows a typical graphical user interface; in this case from the Blender 3D computer graphics modelling system. Cameras are placed in the world to observe the scene. The cameras are able to capture a series of

scenes from the 3D world that has been created. To animate the action within a scene, it is possible to move the location of the objects, cameras and light sources.

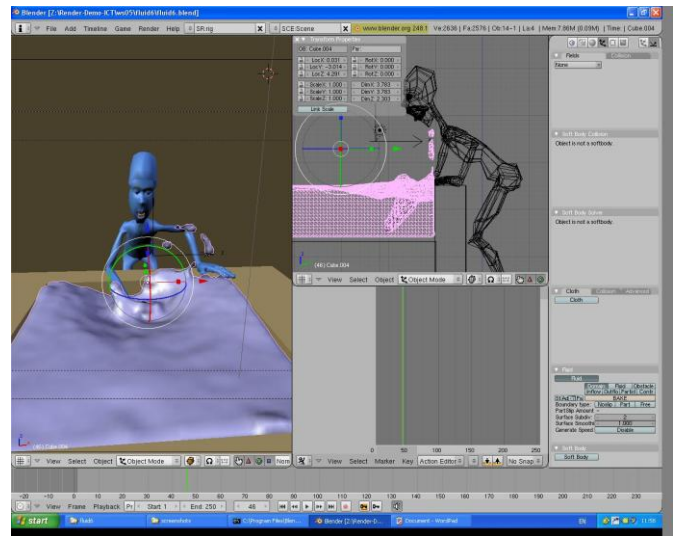


Fig. 1. Graphical User Interface (Blender, 2008)

Rendering is the process of painting colour on to the object surfaces within the scene to produce a visually realistic image. The rendering process can be very time-consuming because of the large amount of data involved. A typical animation would require 25-30 frames per second to produce a smooth animation. When developing a project, the designer uses a workstation to create a series of frames. Rendering can be carried out as a serial or parallel process. The typical time necessary to render a single scene for an animation is 30-60 seconds; however, a complex scene may require 15 minutes, while a very simple scene may take only 10 seconds. Fig. 2 is a screenshot of a rendered 3D model taken from the Blender project entitled "fluid6\_testmanc.blend" (Blender, 2008).



Fig. 2. 3D Rendered Model (Blender, 2008)

DCET researchers have completed more than sixty animation projects having various levels of complexity. In all cases a series of numbered frames are created that may be combined to produce the final animation. It is normal practice to use an uncompressed image file format to retain image detail. Creation of the final animation is normally a post process. The file size produced by the rendering process is typically of the order 1MB. To date the maximum file size handled by DCET is 6MB for a High Definition single frame image. A typical video animation project generates the data presented in Table 1 below.

**Table 1. Render Data**

Single frame size	1MB
Frames per second	30
Number of frames in 10min video	$30 * 60 * 10 = 18K$
Frame data for a 10min video	$18K * 1MB = 18GB$
Ray tracing enabled	yes
Image resolution	standard definition
Time to render a single frame	6.4 minutes
Render time using a single processor	$18K * 6.4 / 60 * 24 = 80$ days or 2.66 months
Render time using a parallel computer with 160 cores	$80 / 160 = 0.5$ days or 12 hours

### 2.1 Library Modules

There are a number of special library modules that may be used to enhance the quality of the rendered image providing support for the following effects (Autodesk, 2009):

Fur	animal fur
Particle	dust and snow, explosion
Fire	flames and smoke
Water	transparent liquid with light refraction

### 2.2 Ray Tracing

When rendering a transparent object, such as glass or water, ray tracing may be used to produce a high quality image. The rays passing through the object are reflected, transmitted and

refracted. The user may set the number of reflections to a high value to produce a high quality image. However, when ray tracing is employed, the associated processing time may increase significantly to 5 to 10 minutes per frame. This may prove too long to consider rendering for a video using a single processor computer. Furthermore in some cases a single high quality frame may take a number of days to render using a single processor computer.

### 2.3 Comparison of Rendering Systems

The authors have undertaken research to identify other groups that have published papers or data relating to the performance of rendering systems. It has proven to be very difficult to identify relevant information. A small amount of information, limited in scope, has been found in online discussion fora, where issues relating to system implementation and fault finding are discussed.

The authors believe that the operators of render farms will probably have completed developments similar to those undertaken by the authors. It is possible that the lack of published papers may be attributed to pressure to meet deadlines or a reluctance to divulge commercially sensitive technical information.

This section will compare the performance of four different 3D modelling packages and their associated rendering systems. In particular, the efficiency of the rendering process will be compared when running on a large cluster computer. Rendering efficiency in this context is defined in the following way (1):

$$E = R/N * 100\% \quad (1)$$

where

$E$  - is the rendering efficiency

$R$  - is the average number of active render nodes

$N$  - the total number of Compute Nodes available for rendering in the cluster

In this analysis the time taken to load and store an image is assumed to be negligible compared to the render time. Typically a complex image file of 1MByte would take about 8mS to transmit via Gigabit Ethernet and about one minute or more to render.

Rendering may be carried out by executing a batch file process on the command line. The process of rendering from the command line does not normally provide the user with a useful overview of render task progress on a cluster computer. The format of a typical command is given below:

*Render-program Input-file Output-file Output-file-type*

In most cases, the alternative graphical user interface (GUI) embedded within the rendering system is usually preferred because it is also possible to schedule tasks and to monitor overall render progress from a single workstation.

A significant number of computer graphics projects have been completed using the DCET cluster computer. Most have involved rendering a large number of frames to produce an animated video. The results and experience gained by the

authors has enabled them to compare in detail the performance of four different rendering systems.

Unfortunately, it has not been possible to make direct comparison between identical projects running on these different rendering systems. The authors have found that large complex projects cannot be imported directly into other 3D computer graphics systems. However, it has been possible to compare many different projects with similar levels of complexity.

### 2.3.1 Autodesk 3Ds Max

The 3Ds Max rendering system deploys a client server architecture based upon network applications. There are three main software components in the 3Ds Max rendering system; the render server, the render client process (Backburner) and a monitor process (Autodesk, 2009). When the rendering process is observed via the monitor it can be seen that all nodes are active virtually all of the time. As soon as a render node has completed a task another task is activated without any discernible delay. In this case the render efficiency  $E = 100\%$ .

### 2.3.2 Blender and Yadra

Blender is a 3D modelling system for computer graphics. Yadra (yet another distributed render agent) and Blender are both open source software systems. Yadra is a Java based, platform-independent tool, for rendering animations of Blender on a network. Yadra and Blender may be combined to create a distributed rendering system that is based upon web technology (Blender, 2004a, b; SourceForge, 2008).

A Yadra node may be programmed via a script to act either as a render agent (Compute Node) or as a render server. The user may connect a browser to the render server to monitor progress and to add new tasks to the scheduler.

The Yadra scheduler maintains a linear list of Compute Nodes (render agents) and the scheduler repeatedly works slowly down the list from the top seeking to identify a node that can be allocated and receive a new render task.

When a small number of Compute Nodes (CN) are active (for example ten, CN1-10) it was observed that all nodes are concurrently active. In a typical example, the frame render time is in the range 30 seconds to 1 minute. As all nodes are fully active, the render efficiency  $E = 100\%$ .

However, when more nodes are deployed (for example forty, CN1-CN40), the render efficiency is reduced and in the range 70 to 80%. This is because some Compute Nodes (CN30-CN40) very rarely become active in this scheme.

### 2.3.3 Maxon Cinema4D

The Cinema4D rendering system deploys a client server architecture based upon network applications. There are three main software components in the Cinema4D rendering system; the render server, a render client process that runs on all Compute Nodes and a browser that may be connected to the server to monitor progress [Maxon, 2009]. To configure the system the net render client is installed on all Compute

Nodes and all other library files are simply copied.

The efficiency of this system is similar to that of 3Ds Max as normally all render nodes are active in the system,  $E = 100\%$ . In tests the authors have noticed that the delay between the end of a task and the start of a new render task is similar to that of 3Ds Max.

### 2.3.4 Newtek Lightwave

The Lightwave rendering system (ScreamerNet) employs a rendering system based upon mapped drives. Render tasks are sent out to the Compute Nodes (render agents) from a central controller and a browser is used to monitor progress (NewTek, 2009). Lightwave does not employ the standard client server architecture.

The Lightwave render scheduler maintains a circular list of potential Compute Nodes. This scheduler steps between Compute Nodes every 2.5 seconds on average and allocates new tasks to inactive nodes.

In this system, to ensure that all Compute Nodes are active, the average render time for a single frame can be calculated.

Case 1

where  $T \geq N * S$  and

$T$  - average frame render time  
 $N$  - number of Compute Nodes  
 $S$  - scheduler step time  
 $R$  - number of active nodes

The efficiency of the render process is given by (2):

$$E = R / N * 100\% \quad (2)$$

As an example, consider the DCET cluster which has 40 Compute Nodes. The average render frame time to ensure that all nodes are active is determined by:

$$T \geq 40 * 2.5 \text{ seconds} = 100 \text{ seconds} \\ = 1 \text{ minute } 40 \text{ seconds}$$

$$E = 40/40 * 100 \%$$

As all nodes are active in this case the efficiency  $E = 100\%$ . The authors have noted from their experience that normally a quite complex scene will require this amount of time to render a single frame.

Case 2

where  $T < N * S$

In this case the number of active nodes may be determined by (3):

$$R = T / S \quad (3)$$

Consider a typical example where  $T = 25$  seconds,  $S = 2.5$  seconds and  $N = 40$ :

$$R = 25 / 2.5 = 10 \text{ (number of active nodes)}$$

$$E = 10 / 40 * 100\% = 25\%$$

In this case, the efficiency of the render process is poor as only 25% of the render nodes are active at any point in time. The authors have found that ScreamerNet is inefficient when used to render simple scenes but is more efficient when used to render scenes with greater complexity.

In the majority of projects that have been rendered by the authors, the average render time required is typically 30 seconds to 1 minute to render a single frame. Consequently, the Lightwave render system does not normally make good use of the compute resources available on the DCET cluster.

In addition, the Lightwave scheduler system does not allow the user to add a new render task to the queue after the render process has been started. As a result, unmanned batch processing is more difficult to organise using this system.

A different approach has been adopted by the designers of the Lightwave rendering system (ScreamerNet). Rather than use the more traditional client server or web based architecture, the LW system makes use of mapped network drives (NewTek, 2009). In their documentation, Newtek recommend that Lightwave is installed in Windows XP Professional (WinXP) and state that their system will support up to one thousand distributed worker nodes. Unfortunately, WinXP will only support a maximum of ten concurrent network connections. To overcome this restriction, it is necessary to install the main Lightwave application on a system that will support a large number of concurrent network connections. In the DCET Cluster Computer, Lightwave uses the MD1000 DAS (direct attach storage) system to overcome this limitation. The authors consider that the Lightwave system is quite difficult to configure correctly.

### 3. DCET CLUSTER COMPUTER SPECIFICATION

A Case Study by Dell UK (2008) describes the design philosophy for the DCET Cluster Computer. An outline specification for the system is listed in Table 2 below.

**Table 2. DCET Cluster Computer Specification**

Compute Node processor type Intel Xeon	64 bit
Number of Compute Nodes	40
Number of dual core processors per Compute Node	2
Number of CPU cores per Compute Node	4
Ram memory per Compute Node	8 Gigabyte
Number of cluster networks: Data, IPC and IPMI	3
Network type Gigabit Ethernet	1Gbps
Disk storage distributed	10 TByte
Disk storage centralised - MD1000	7.5 TByte
Operating System	Win2003 CCE* 64 Bit

Key\*: CCE Compute Cluster Edition

The main switch installed in the DCET cluster is a Cisco Catalyst 6509. The Cisco Catalyst 6500/Cisco 7600 Series Supervisor Engine 720 integrates a high-performance 720Gbps crossbar switch fabric with a forwarding engine in a

single module, delivering 40 Gbps of switching capacity per slot enabling 4\*48-port Gigabit Ethernet density line cards. With hardware-enabled forwarding for IPv4, the system performance is capable of 400 Mpps for IPv4.

To date the authors have noted that the traffic level on a single link in the network does not exceed 30% of the full capacity (1Gbps) during rendering. The authors have observed in practice that the network is able to manage the traffic levels associated with parallel rendering.

### 4. CONCLUSIONS

After completing many computer graphics rendering projects, the authors are able to conclude that significant resources are required to properly support this activity; namely a large number of identical Compute Nodes, a very high bandwidth network and a large centralised disk storage system.

Recently the authors rendered some projects for the students of a local Further Education college. These projects made use of advanced ray tracing features. The authors were informed that it was not practical to render these projects in the college because they each required an estimated render time in the range 70 to 90 hours using a single processor computer. The DCET cluster was used to successfully render these projects in a time of typically less than 30 minutes each.

After comparing the performance of four different rendering systems the authors have drawn the following conclusions:

- The most popular, stable and efficient system of those tested is Autodesk 3Ds Max. The 3Ds Max render system makes good use of available computing resources.
- The open source Yadra and Blender combination provides the user with a reliable low cost alternative to the commercial products that are on offer. The Yadra system does not always make good use of available computing resources.
- The efficiency of the Cinema4D system is similar to that of 3Ds Max as normally all render nodes are active during the rendering process. The Cinema4D render system makes good use of available computing resources.
- The Newtek Lightwave rendering system when deployed on a large cluster computer has been proven to be inefficient and often does not make good use of available resources. For production scheduling, users may wish to consider using an alternative standalone third party scheduler.
- The Newtek Lightwave scheduler is based upon the use of mapped drives. The authors believe that this system is quite difficult to configure and set up. As a result, the authors argue that a better solution would be to employ a web based approach or a client server architecture that is not based upon mapped drives.

The authors have noted that it is not normally necessary for the user to allocate render tasks to individual CPU cores. A number of experiments were carried out, for example, using 3Ds Max and Windows 2003 CCE 64bit. Typically during the render process a block of eight frames is automatically allocated to each Compute Node for rendering. A Compute Node uses the four CPUs (two dual core processors) available to render each block of frames.

In addition, the authors have directly compared the performance on a single Compute Node with that of ten Compute Nodes. The theoretical maximum processing speedup factor in this case is 10 and the actual value was measured to be in the range 9.1 to 9.5. The authors consider this result to be acceptable. Experiments have shown that the reduction in speedup from 10 to 9.5 is largely due to the communications overhead associated with large file transfer.

Significant computing resources are required to successfully complete a 3D graphics project. Typically, it is necessary to provide large shared disk storage areas to accommodate the massive graphics programs and the rendered graphics output files. In addition, parallel processing methods using many Compute Nodes (CPUs) are required to minimise the render execution time. High bandwidth networks are also necessary to ensure that there is no communications bottleneck.

For interest, a companion ICSE 2009 paper entitled "Cluster Systems - An Open-Access Design Solution" provides more detail relating to the hardware design of the DCET Cluster Computer.

#### ACKNOWLEDGEMENTS

The Science Research Investment Fund (SRIF) is a joint initiative by the UK Office of Science and Technology (OST) and the Department for Education and Skills (DfES). The purpose of SRIF is to contribute to higher education institutions' (HEIs) long-term sustainable research strategies and address past under-investment in research infrastructure. The University of Sunderland Cluster Computer was purchased with the support of the SRIF III fund.

#### REFERENCES

- Autodesk (2009). Autodesk 3Ds Max Modelling and Animation Software Features.  
<http://www.autodesk.co.uk/adsk/servlet/index?siteID=452932&id=12353508>(Last accessed 6 July 2009)
- Blender (2004a). *Blender Documentation Volume I - User Guide*. <http://www.blender.org/documentation/htmlI/> (Last accessed 6 July 2009)
- Blender (2004b). *Blender Documentation Volume II - Reference Guide*.  
<http://www.blender.org/documentation/htmlII/> (Last accessed 6 July 2009)
- Blender (2008). Blender project: fluid6\_testmanc.blend  
<http://www.blender.org/development/release-logs/blender-243/fluid-enhancements/> (Last accessed 7 July 2009.)
- Dell (2008). Eco Friendly Super Computing – Power efficient “green” super computing solution puts British

- university at the forefront of research. *A Case Study by Dell UK*  
[http://www1.euro.dell.com/content/topics/global.aspx/casestudies/en/emea/uk/fy2008\\_q4\\_id798?c=uk&cs=RC1050265&l=en&s=pad](http://www1.euro.dell.com/content/topics/global.aspx/casestudies/en/emea/uk/fy2008_q4_id798?c=uk&cs=RC1050265&l=en&s=pad) (Last accessed 6 July 2009)
- NewTek (2009). Lightwave 3D – model, animate and render. Innovative solution for graphics film and television production. <http://www.newtek.com/lightwave/index.php> (Last accessed 6 July 2009)
- SourceForge (2008). Yadra (yet another distributed rendering application). A Network-Render-Tool for Blender. A Java based, platform-independent tool, for rendering animations of blender in a network. <http://sourceforge.net/projects/yadra> (Last accessed 6 July 2009).