# Network Rendering | 3D Studio VIZ 3.0

# 1 | PROPOSAL

## *Introduction*

The task of producing spatially accurate digital models, complete with convincing materials and lighting, is commonplace within the Architecture, Engineering and Construction (AEC) industry. However, the output produced, so far, has been limited to high-resolution still images and low-resolution video.[1]

Why not offer something closer to a cinematic experience?

> ***Imagine:*** **Your client is seated in the New Media Room as refreshments are served. A video image on the wall shows high cloud arching over south-eastern Australia. The satellite image dissolves as your client is drawn into your design presentation...**

It is not that project teams do not want animated output at a resolution that matches their static output. It therefore follows that there are restrictions acting to discourage production in this direction.

This report presents the results of an investigation into the limitations of current practice within the Faculty of the Built Environment at the University of New South Wales.

In an attempt to map the boundary of existing practice, this report sets out to explore the process of rendering an animated sequence at a higher resolution than is currently considered normal. The focus will be on the logistics that enable the production of a higher standard of presentation quality animation, using render arrays[2], within the AEC industry.

## Definition?

Overhead transparencies, photographic slides, a physical model and a set of hand-rendered arrangement drawings can be considered the historic starting-point for a definition of 'good quality' design presentation. Any need for animated images was by means of a film projector.

From around the late 1970s, the primary artefacts of display were still photographic slides, physical models and hand-rendered arrangement drawings. At this time, the film projector began to be replaced by the videocassette recorder (VCR) and a projected television image. However, this situation was soon to be challenged with the arrival of the personal computer (PC).

The 1980s witnessed the workplace phenomenon of the personal computer and an array of new methods was embraced for the purpose of building design and construction.

---

[1] John Madden, Drafter and Animator, Walter Constructions Pty. Ltd., obtained by interview Aug. 2001.

[2] Though common in the film industry, render arrays (that is, network computers working in unison to produce animated sequences) are less common within the AEC industry.

By the 1990s, the adoption of two-dimensional computer aided drafting (CAD) and, to a lesser extent, three-dimensional modelling was widespread and the personal computer had begun to challenge the VCR as the basis of presentation display.

In 2001, there are many applications not unlike 3D Studio VIZ (3DS VIZ) that have made computer animation a productive tool of the design office.

It is clearly speculative to provide a definitive description of what constitutes a suitable quality for presentation animation. This is because there are many factors that influence the final format of animated digital output. Such things as file format, projection format, hardware and software platforms, and available work hours of the production team act together to influence the suitable quality of the final animated presentation.

The greatest overall determinant is the cost of production itself. Clearly, given a Hollywood-sized budget a good deal more is possible by way of computer-animated graphics.

We are not talking about the film industry here. It is reasonable however, to perceive current film industry use of digital animation techniques as a guide to future directions in presentation quality animation. This is because the software platforms and techniques used in the film industry are often identical or at the very least, computationally similar to those used within the AEC industry. In his paper, *Crossing the Digital Threshold*, Scott McQuire refers to the 'trickle-down' effect of film technology,

"...there is approximately a two year gap between a technique becoming available to high-end producers and its general availability on PCs... The main issue with using PCs, rather than more sophisticated workstations such as the Silicon Graphics Onyx, is the quality of resolution which can be obtained, and the speed with which images can be processed."[3]

In order to constrain the extent to which a definition could be taken, this report will adhere to current practice within the University of New South Wales' Faculty of the Build Environment and its implicit relevance to contemporary AEC practice.

Much in the way that film projections gave way to VHS videotape, which in turn gave way to the personal computer and its VGA SVGA standards, it is believed that adoption of digital television will influence the format of digital output if only for the reason that the common display device will be a digital television set.

The animations produced for this report, though intended for display on a PC monitor with an aspect ration of 4:3, are produced at the digital television aspect ratio of 16:9 [4].

[3] McQuire, S. *Crossing the Digital Threshold.* The Australian Key Centre for Cultural and Media Policy, 1997. p 44

[4] The aspect ratio of 4:3 is the normal PC format, 16:9 is the wide screen digital television format and 2.21:1 is the cinemascope format used in movie theatres.

# 2 | PROCEDURES

## *Software*

3DS VIZ can be installed in one of two ways.

A *typical* installation is required for at least one machine in the network and requires an authorized copy of 3DS VIZ consisting of an authorisation code and a hardware lock.

*Compact* installations are used when setting up rendering servers. These installations are not intended for use as 3DS VIZ editors or workstations. In compact mode, 3DS VIZ requires no authorisation code and hardware lock.

There are three additional items of software required for network rendering. These items are bundled with 3DS VIZ and include; the manager software (manager.exe) that is run on a single machine assigned the manager role, the server software (server.exe) that is set to run on every server machine, and the queue manager software (queueman.exe) that, though not necessary for the execution of a network-rendering task, is used to manage network-rendering projects.[5]

### Manager.exe

"The Network Manager service (Network Manager) is a Windows NT service which must be installed on at least one computer in each group of computers which will participate in network rendering.

The Network Manager communicates with a specified group of Network Rendering Servers to assign jobs and monitor rendering progress. The Network Manager also handles the scheduling of jobs and the existing network infrastructure of servers through the Queue Manager Client."[6] (See figure, below)

[5] It was observed that network rendering could be initiated without use of queue manager. This is however, bad practice as the render-operator has no active control over the network-render servers (their assignment being fixed at the time of render initiation) and he/she receives no render progress feedback. Network rendering without queue manager is akin to working blind-folded while having your hands tied.

[6] 3DS VIZ Help

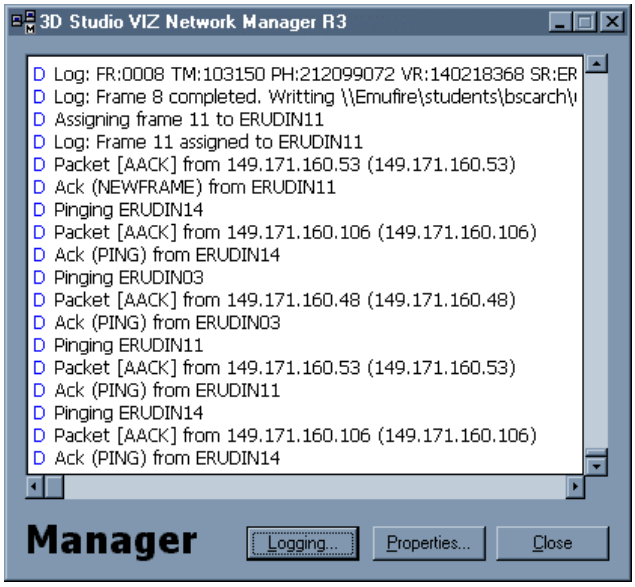



**Figure 1 – Manager application (manger.exe) communicates with a specified group of servers.**

## Server.exe

"The Network Rendering Server must be running on any computer dedicated to rendering a job remotely.

"The Server communicates with the Network Manager and launches 3DS VIZ on that computer to render network processes. The Server has no user interface itself, but you can set various options for it from the Queue Manager Client. For example, you can specify the times of day that the Server is available to do network rendering.

You specify which PCs are to be servers in the Job Assignment dialog, which is a sub dialog of Render Scene."[7] (See figure, below)

[7] 3DS VIZ Help

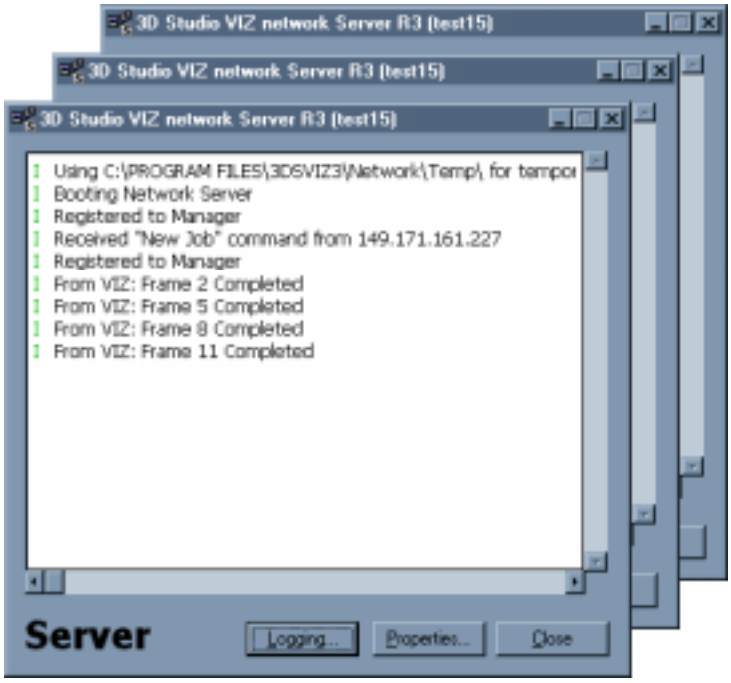



**Figure 2 - Server application (server.exe) is executed on each server machine. These then register with the manager.**

## Queueman.exe

The queue manager (queueman.exe) is a stand-alone network-administration tool that provides a user interface to monitor and control network rendering.

"The Queue Manager can connect to any computer to which you have network access with the appropriate security permissions, and a Network Manager running on it. It can be installed separately from 3D Studio VIZ. It will function correctly on any Intel-based computer running Windows NT with appropriate TCP/IP networking services, including over the Internet. In other words, you can monitor and control network rendering services from any computer connected to the Internet, in addition to using the Internet as a wide-area backbone for a network rendering farm."[8]

During the course of investigation, queue manager functioned intermittently on lab machines running Windows 95 over DHCP.[9] (See figure, below).

[8] *Ibid*.

[9] For more information see, Part 6 | PROBLEMS, Rendering, Queue Manager

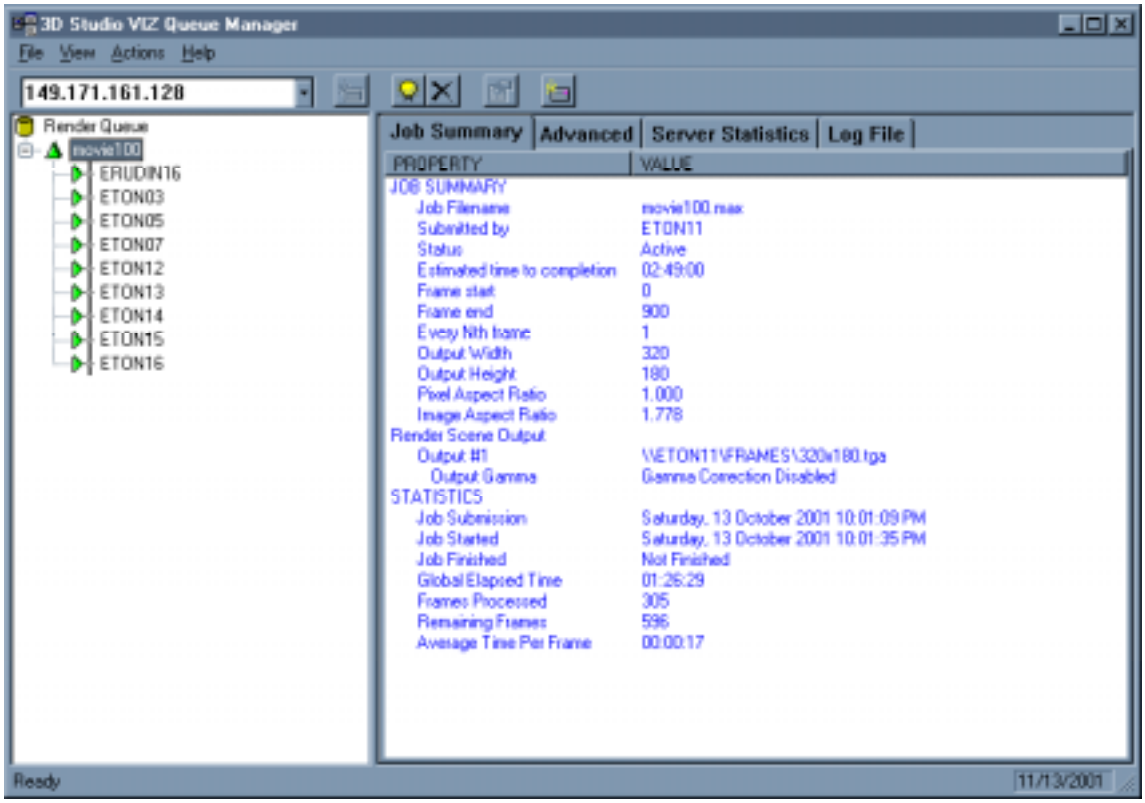



**Figure 3 – Queue manager (queueman.exe). Note the Manager's IP address, in the field, at top left.**

## *Set-up*

The steps required for network-render set-up, within the existing network infrastructure are as follows.

1. Ghost all machines (i.e. re-loading the Window's registry on each machine using Norton Ghost 5.1d by Symantec).

2. Log on to each machine.

3. Uninstall all screen savers.

4. Run winipcfg.exe on the Manager machine and copy the IP address and host name to a text file saved to a student folder. (See Figure, below)

5. Assign a machine to receive the rendered output. The 'target directory' had first to be set up as a 'share' in Windows. The host name of the target machine was then copied to a text file and saved to a network folder.

6. Check the clocks are correct on the Manager (and queue manager if different machines) and adjust where necessary.

7. Initiate and then configure the server application on each Server machine. This involved opening the text file containing the Manager's IP address and pasting this value to the properties box of the server application.

8. Start the manager application on the Manager machine.

9. Start and then configure the queue manager on the Manager machine or another machine on the network.[10] This involved opening the text file containing the Manager's IP address and pasting this value to the field of the queue manager application.

10. Open 3DS VIZ on any machine in the domain and assign the rendered output to a network folder using UNC.

    This step also involved choosing the Targa image file type (See figure, below).

    On higher spec'd machines (eg. those in Lab 01), this involved opening the text file containing the target machine's Host and Share names in UNC format[11] and copying and pasting this value to Render Output > Save File 'files...' field of the render dialogue. On less spec'd machines, this involved typing the value directly into the field because the operating system displayed a 'not enough memory' message.

11. Select the 'Network Render' checkbox within 3DS VIZ and then configure the 'Network Job Assignment' dialogue. (See Figure, below)

12. Select the Server machines from the list of registered servers in the 'Network Job Assignment' dialogue. (See Figure, below). It was *critical* here to select the 'Include Maps' checkbox in order to avoid 'missing map' error messages. For good measure, the 'High Priority' checkbox was also selected.

    At this stage, it was possible to initiate the network rendering process without initiating the queue manager.

13. Select the network job by name in the queue manager and activate the network-rendering task.

14. Save the queue manager report at completion of the render task.

    It was found that the report file could not be saved at the end of the network rendering process if the manager application was shut down prematurely.

15. For the convenience of fellow students, close all programs and re-load all machines.

---

[10] Queue manager would, quite often, not execute on the Manager machine. There then ensued a trial-and-error process to find a machine that would allow queue manager to execute.

[11] Universal Naming Convention (UNC) takes the form \\HostName\ShareName\FileName.ext

## Figures

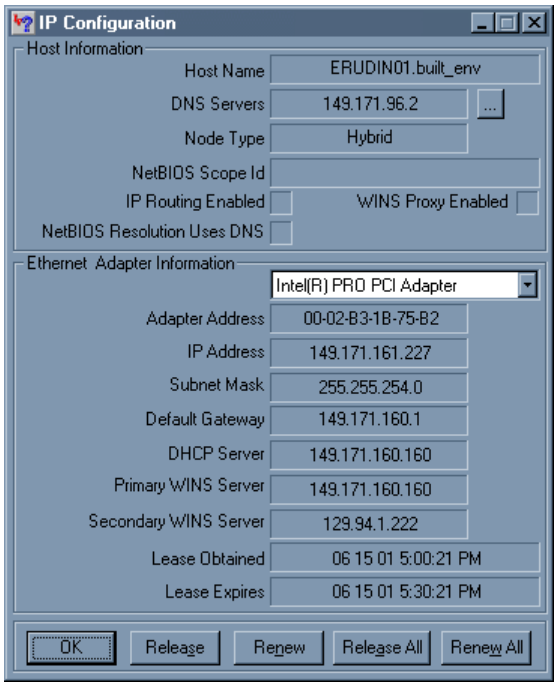



**Figure 4 – IP address configuration (winipcfg.exe). Refer to Set-Up, step 4.**

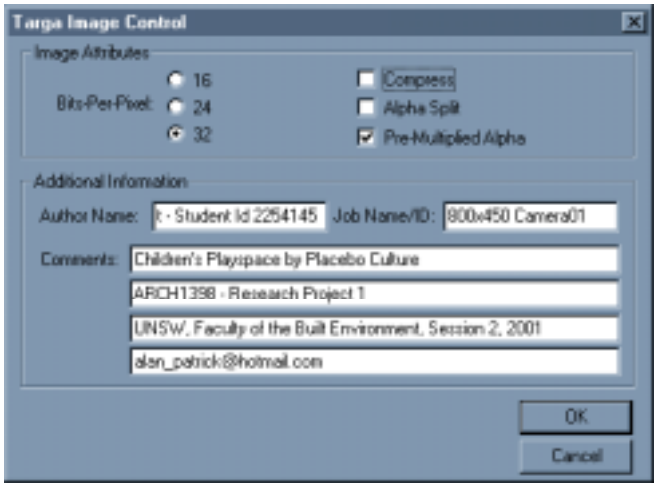



**Figure 5 - Targa Image Control. Refer to Set-Up, step 10.**

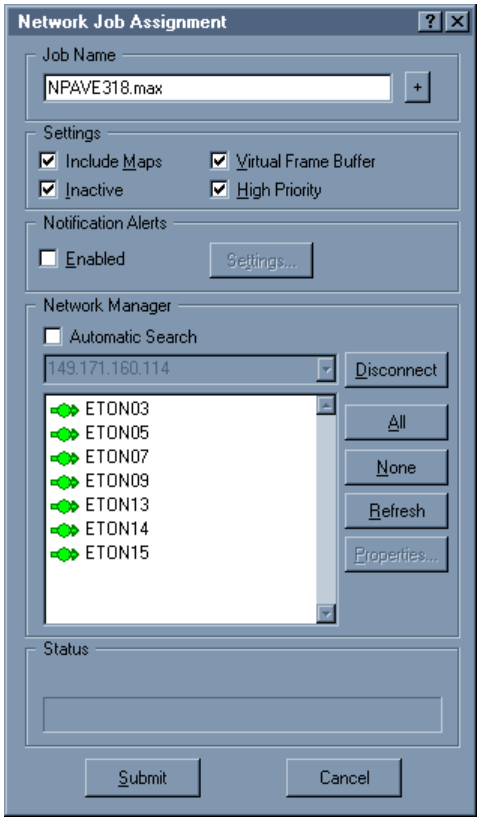



**Figure 6 - Network Job Assignment dialogue. Refer to Set-Up, steps 11 and 12.**

## *Postproduction*

3DS VIZ will accept more than one server machine in network-rendering mode only when the output file type chosen is not of a type used in a 'single user device'. In other words, all rendered output must be to separate image files that are later stitched together to create a single animation file.

The reason for this is clear; a single user device file (eg. an AVI file) is normally created as a continuous process. The network rendering facility bundled with 3DS VIZ is unable to work ahead or recover parts of the file that may have been left out because of server outages.

The initial output file type chosen for this report was Targa.[12]

In network rendering, output files are sequentially numbered automatically by 3DS VIZ and placed in the target folder. The target folder can be placed anywhere on the network and is identified using UNC via a network share.

[12] "The Targa (TGA) format was developed by Truevision for their video boards. The format supports 32-bit true color… and is usually used as a true color format. Targa files are widely used to render still images and to render sequences of still images to videotape. ...3DS VIZ is designed to produce the highest quality output by rendering single frame TGA files or rendering directly to a digital disk recorder..." 3DS VIZ Help.

The postproduction task involved manipulating the output image files and creating a single file type suitable for use in a single user device.[13]

## *Comments*

The 'single user device' limitation has two distinct disadvantages when compared to conventional rendering (that is, rendering on a single machine). Firstly, the storage requirement for the target folder is greater. Secondly, there is an additional process involved, which is the stitching-together of the individual frames to form a single use file type.

The stitching process requires recreation of the animation file's output settings including the total number of frames. These settings may differ from those assigned in the original network-rendering job. Though this could be advantageous in some cases, (for example, in the situation where a change to the output resolution was required) the effect is generally disadvantageous because it adds an additional time component and because it increases the possibility of user error.

For network rendering to be effective, within the existing infrastructure[14], the repetitiveness of the set-up procedure outlined above must be reduced. This could be done through several approaches; The 'ghosting' procedure could be tailored to suit the needs of a network server; There may be benefits in creating separate domains for the network rendering task and other student tasks: Rendered output could be managed at the network administrative level instead of the current student user level.

[13] The animation file type chosen was AVI (Audio-Video Interleaved). This file type is currently regarded as the Windows standard for movie files.

[14] That is, the infrastructure as at the time of writing and as maintained by the Faculty of the Built Environment, UNSW.

# 3 | SCENE

The 3DS VIZ scene used in this report is based on a proposed children's playground in the Sydney suburb of Bankstown and was designed by Placebo Culture, Sydney. The scene consists of a tree-lined boundary surrounding twelve animal characters positioned around a central water feature. Chinese and Vietnamese horoscopes have informed the animal characters while the characterisations are designed to appeal to the children of Bankstown's diverse culture.

Output resolutions aside, there are many factors that affect a scene's render time. Put simply, the more complex the scene the more time it takes to render. The primary source of complexity is the number of object faces. Secondary sources of complexity are the setting of shadow parameters, the number of light sources, the file size of bitmaps used in materials and the material types.

## *Construction*

Each animal character was constructed in 3DS VIZ from bitmaps that were supplied by the designers. Adoption of the encapsulated postscript file type (EPS) for design delivery permitted unconstrained re-sizing and editing, using Photoshop, which was of great benefit to the 3D modelling task.

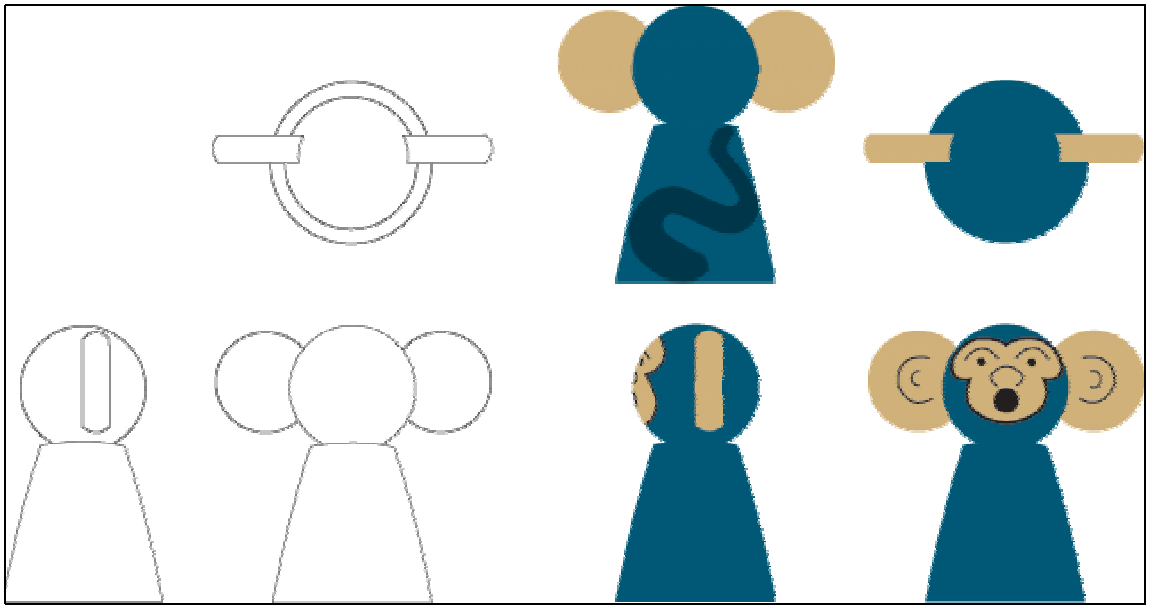


**Figure 7 – The Monkey design as supplied by Placebo Culture in EPS file format.**

The majority of the animal characters began as 3DS VIZ box objects that were then fashioned into shape using a succession of 3DS VIZ free form deformation (FFD) modifiers.[15]

To minimise the scene's file size, individual animal objects were collapsed into editable meshes.[16]

[15] Lofting was used for the Rooster.

## Materials

Standard 3DS VIZ material types where used on all but three objects in the scene. The other material type used was Multi/Sub-Object on the Pig's snout, the Dragon's head and the Dog's ears.

Consideration was given to the use of raytraced materials. The adoption of raytraced materials would have enhanced the digital savvy of the scene but would have also added rendering time to a scene that was already considered 'slow to render'.

### *Auto Reflect/Refract Maps*

"Rendering Iterations: Sets the number of inter-object reflections in non-flat automatic reflection maps. Although increasing this value can sometimes enhance image quality it also increases rendering time for reflections."[17] This material type was omitted from the scene to reduce render time.

## Shadows

Of the 17 light sources in the scene, shadows are used with only one (Sun01). The remaining sixteen lights are split into subterranean (four groups of three) and overhead (four single) and these have their shadow parameters switched off.

The subterranean light-groups are each made up of three lights representing red, green and blue tones. They are placed below the terrain to mimic the effect of the sunlight's natural reflective quality. This mimics an effect known as radiosity. Together, the colours produce a white-light source while their individual colours act to enhance the vibrancy of the scene by picking-out similar hues. The overhead lights act to soften the harshness that would otherwise result in the use of a single overhead light source.

Raytraced shadows, like those used for sun01, include a setting that significantly influences a scene's rendering time. The setting is know as 'quadtree depth' and is explained thus, "The default is 7 levels, and if you reduce this, it takes up less RAM (but more time) and if you increase it, it takes up much more RAM (but less time).

"The highest practical range is 7 to 8, and the lowest practical range is 4 to 5. In some cases, if raytraced shadows provide an incorrect result, it helps to decrease the quadtree depth by a tick or two, but otherwise you probably won't want to adjust this parameter." [18]

---

[16] Scene statistics, produced by the 3DS VIZ Archive command, are listed in Appendix C | SPECIFICATIONS.

[17] 3DS VIZ Help.

[18] *Ibid*

## Object Motion Blur

Early modelling experiments tested the effect of object motion blur produced by the movement of the camera. The results were unsatisfactory and the use of object motion blur was rejected. However, its effect on rendering time is worth noting.

With object motion blur, the value that effects a scene's render time is referred to as 'samples' and is explained thus, "When Samples = Duration, there is no randomness (and if both numbers are at their maximum value (16), you get a dense result (which costs between 3-4 times the normal rendering time for that specific object).

## *Folder Structure*

The folder structure used in this report is worth mentioning for its use of Window's shortcuts. Their use was necessary because of the repetitive nature of the network render set-up procedure, which is imposed upon the user by the existing network infrastructure.

Shortcuts to common items were placed in a student folder for ease of access. (See below)

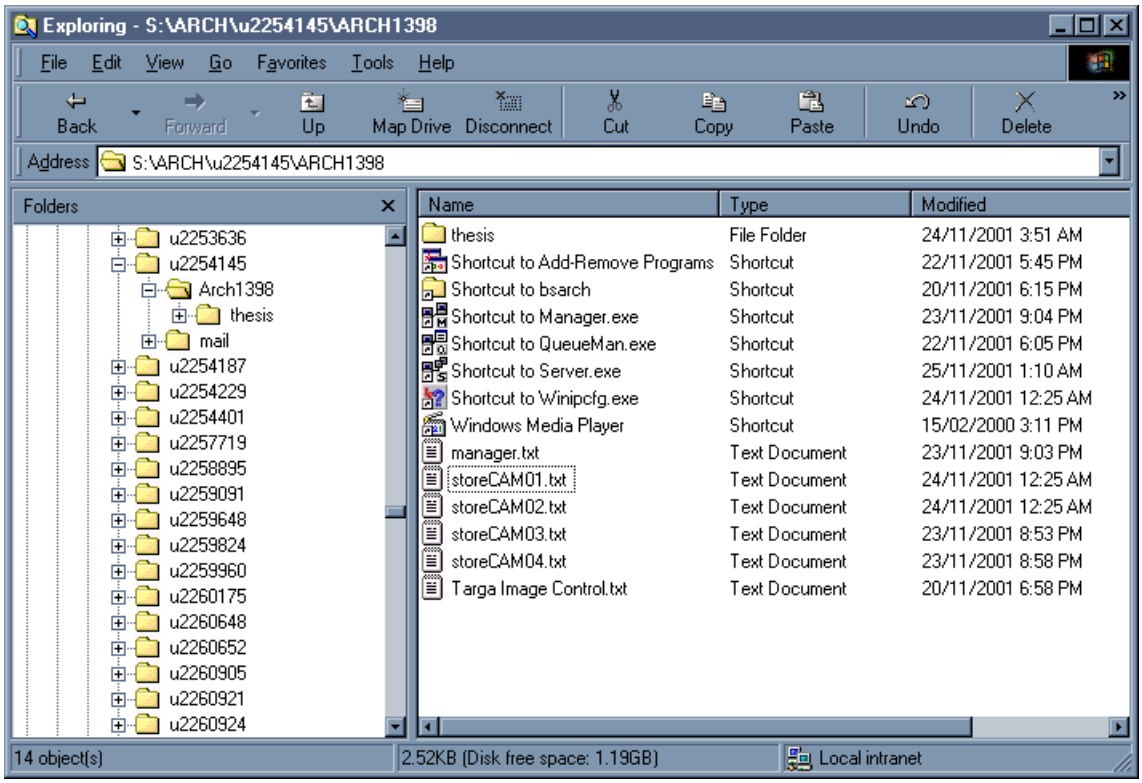



**Figure 8 – The directory/folder structure. The use of Window's shortcuts managed the repetition involved in network render set-up.**

Set-up for the manager machine, involved obtaining the machine's IP address from a Window's shortcut to 'Winipcfg.exe'. The value of the IP address was then copied to a text file called 'manager.txt'.

Set-up for the server machines involved executing each server's local version of the server application. Activation of the server application was achieved using a Window's shortcut. The manager's IP address was then copied from the manager text file and pasted into the server application's configuration settings.

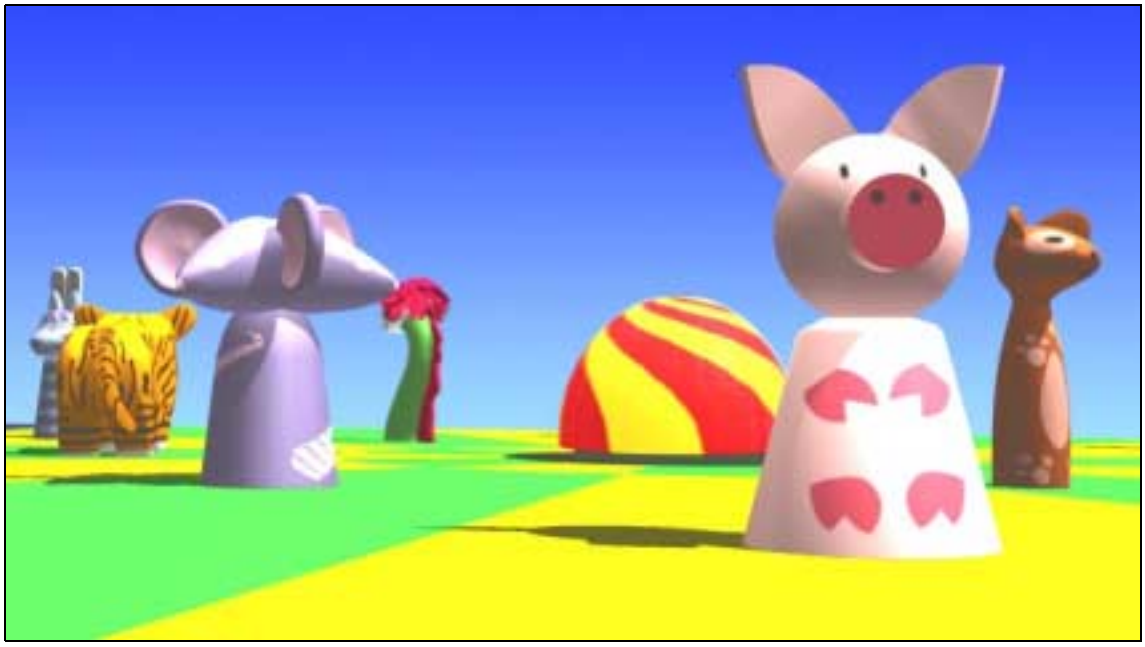
For a step-by-step description of the set-up procedure see, Section 2, Procedures, Set-up.

# 4 | PICTURES

The scene consists of a tree-lined boundary surrounding twelve animal characters positioned around a central water feature. Chinese and Vietnamese horoscopes have informed the animal characters while the characterisations are designed to appeal to the children of Bankstown's diverse culture.

Camera 01, frame 450

Camera 02, frame 406



Camera 03, frame 596

Camera 04, frame 388 (Lens flare added using Photoshop)

# 5 | PROGRAM

The program of investigation was divided into four tasks. These were an animation task, a single frame-rendering task, an output task and a storage task.

## *Hardware*

Machines in computer labs 01, 04 and 05 were used in the investigation because they varied in only one respect. The machines in Lab 01 and 04 both have 256 megabytes of RAM though they differ in CPU while machines in Lab 04 and 05 possess identical CPUs with different amounts of RAM. The table below lists the machines used.

**Table 1 - The hardware.**

| Lab | Room | Render Array |
|---|---|---|
| 01 | RC2033 | 16 x Pentium IIIs rated at 933 megahertz with 256 RAM |
| 04 | RC3035 | 16 x Pentium IIIs rated at 600 megahertz with 256 RAM |
| 05 | RC3036 | 16 x Pentium IIIs rated at 600 megahertz with 128 RAM |

## *Output*

In this task, the animation consisted of 101 frames at a resolution of 800 x 480 pixels. The final AVI file was 5,024 KB whereas the destination folder that held the 101 separate Targa files held a total of 51.3 megabytes.[19]

The task illustrated an approximate tenfold increase in storage space compared to rendering on a single machine.[20]

## *Storage*

An early task set out to experience the queue manager's response when the 50-megabyte capacity of a student folder was exceeded. The result mimics that which would happen if the hard disk on which the target directory resides reached its maximum capacity.

The result of this task is shown in the figure below.

[19] Minimum file size = 437 KB, maximum file size = 605 KB

[20] The AVI file type was adopted.

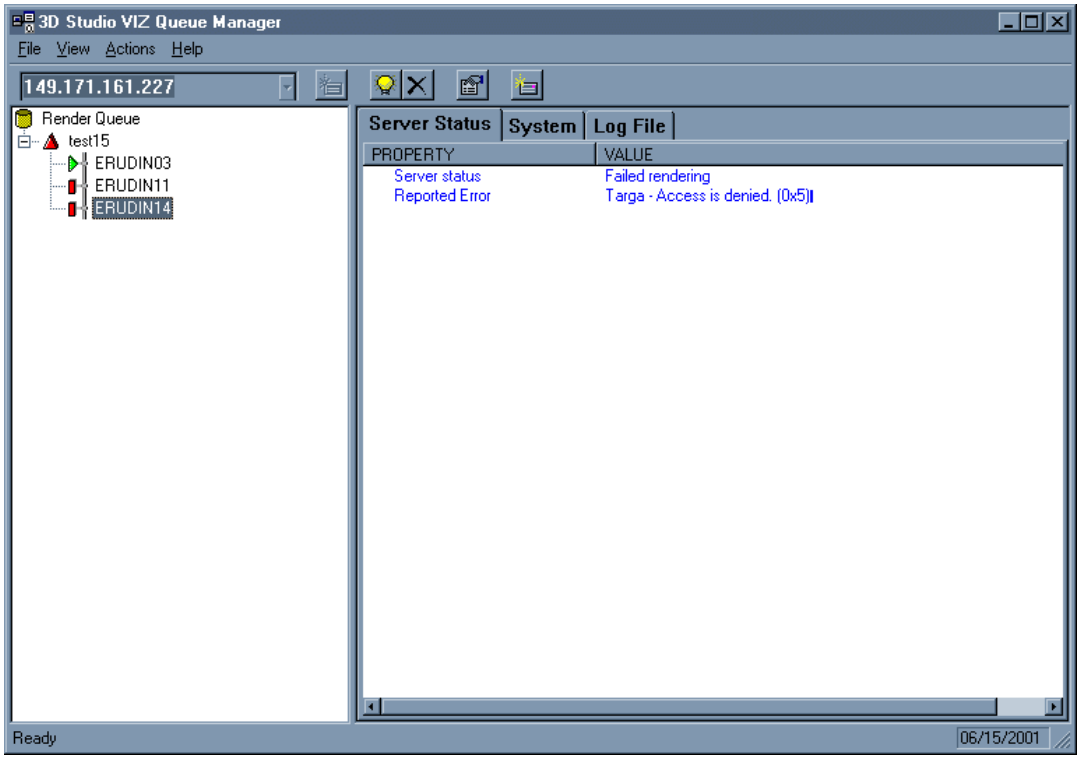



**Figure 9 – Queue manager error when output limit reached. ERUNIN03 has yet to report the failure.**

The stitching process requires the user to select the same render output settings, apart from the file type and network rendering options, that were originally used in the network rendering job and ensure the same number of frames.

It is clear that unless network rendering is managed correctly, user mistakes will occur in the rush to stitch the frames together.

## *Animation*

The maximum obtainable resolution and frame number, for the given scene, was obtained through a trial-and-error process that also functioned as a learning process.

It was originally conceived that a frame rate of 15 frames per second (FPS) would produce one minute of playback over 900 frames of animation. When the output generated from the four camera views was combined, the total playback was expected to be around four minutes of edited video.

After an extended process of trials, it was determined that the resolution was to be set at 640 x 360. This resolution resulted in network-rendering task that utilised up to 30 computers at one time for periods of up to 7 hours.[21]

[21] All queue manager reports, relevant to this investigation, are found on the accompanying CD in the 'Reports' folder.

The output from the animation stage was used to produce the animated sequence found on the accompanying CD.

## *Single Frame*

The single frame task was carried out to determine the effect of RAM and CPU on render time.

The lowest specified machines (Lab 05) determined the maximum resolution of a single frame and a process of trial-and-error was used to achieve this.

The first attempt was made at a resolution of 5120 x 3620. This was successful in Lab 01 but failed in Labs 04 and 05. The next attempt reduced the output frame to a resolution of 1024 x 724 and this was successful in all three labs.

### Comparisons

Data collected from the single frame-rendering task offered the following comparisons.

RAM Question; what performance difference is there between a Pentium III rated at 600 megahertz and 128 megabytes of RAM and an almost identical machine with 256 megabytes of RAM?

CPU question; what differences exist in render performance between a Pentium III rated at 933 megahertz and a Pentium III rated at 600 megahertz?

### Reporting

Queue manager allowed the generation of ASCII text reports containing detailed statistics about a particular render job.

An alternative method for accessing render statistics that is functional in standalone mode is to use *File > Summary Info.*

An example of a queue manager report is shown below.[22]

[22] Single frame investigations have shown that the job duration, which is calculated by subtracting the "Job Finished" value from the "Job Started" value, is not equivalent to the "elapsed" value. This suggests that the time taken for 3DS VIZ to prepare for rendering the first frame (which includes calculating the lights) is excluded from the time taken to produce a rendered image of that first frame.

| | | | | | |
|---|---|---|---|---|---|
| 3D Studio Max Report File | | | | | |
| Job: | LAB01 | | | | |
| Submitted by: | ERUDIN01 | | | | |
| | | | | | |
| State: | Complete | | | | |
| Frame start: | 450 | | | | |
| Frame end: | 450 | | | | |
| Every Nth frame: | 1 | | | | |
| Output Width: | 1024 | | | | |
| Output Height: | 724 | | | | |
| Pixel Aspect Ratio: | 1.000000 | | | | |
| Image Aspect Ratio: | 1.414365 | | | | |
| | | | | | |
| Job Submission: | 2001/11/16 15:24:21 | | | | |
| Job Started: | 2001/11/18 15:22:04 | | | | |
| Job Finished: | 2001/11/18 15:41:30 | | | | |
| | | | | | |
| DATE | TIME | FRAME | ELAPSED | MEMORY | SERVER |
| 2001/11/18 | 15:41:30 | 450 | 967900 | 251822080 | ERUDIN03 |

In the 3D Studio MAX (sic) report above, the value for "elapsed" time is measured in tics.[23]

## Analysis

The three single frame tasks were executed in a controlled manner (that is, as close to identically as the network infrastructure allowed) to measure the time it took to complete a single frame of the scene.

The CPU comparison revealed that an increase in CPU of 64% produces a decrease in render time of 24%.

The RAM comparison revealed that an increase in RAM of 50% produces a decrease in render time of 14%.

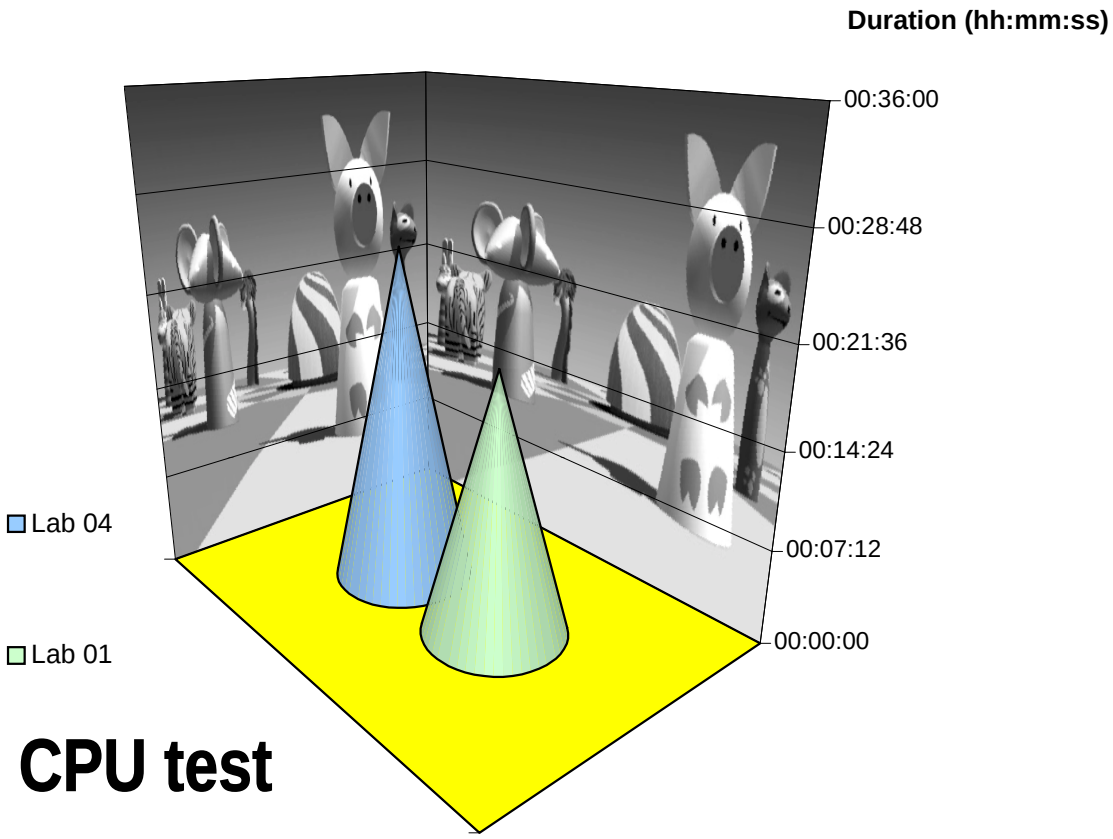
The results prove that, in order to increase the speed of rendering, CPU is more important than RAM but the amount of RAM is a contributing factor.

## Results

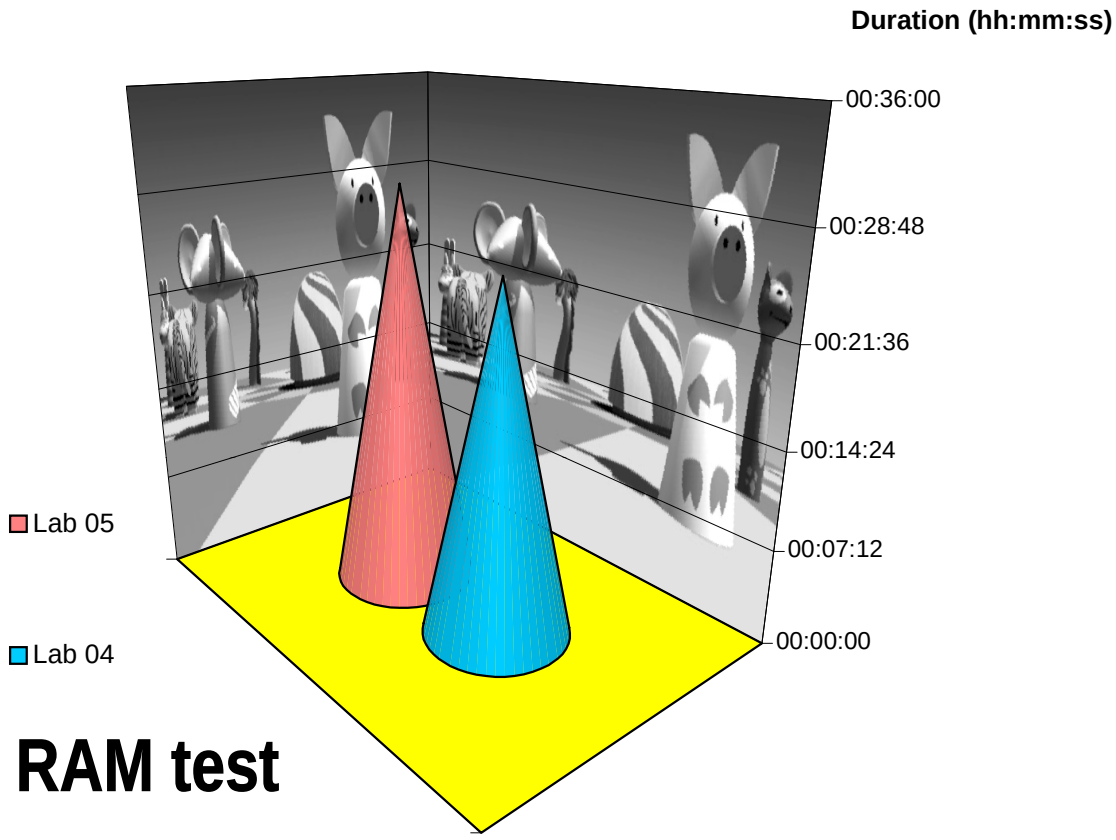
Results of the three tests prepared in the single frame-rendering task are shown on the following pages.

[23] One tic is 4800th of a second.

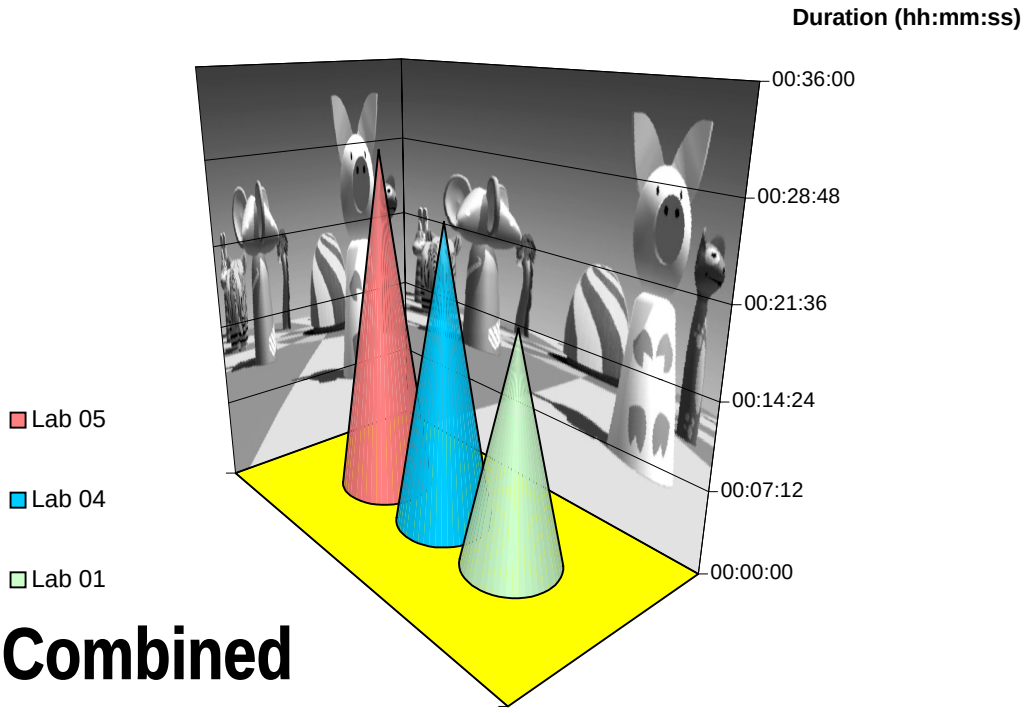
| Job | Lab 01 | Lab 04 | Boost |
|---|---|---|---|
| Submitted by | ERUDIN01 | ELLIS06 | |
| RAM | 256 | 256 | |
| CPU | 933 | 600 | 64% |
| Width | 1024 | 1024 | |
| Height | 724 | 724 | |
| Image Aspect Ratio | 1.414365 | 1.414365 | |
| Started | 15:22:04 | 14:29:25 | |
| Finished | 15:41:30 | 14:54:55 | |
| Duration | 00:19:26 | 00:25:30 | 24% |



| Job | Lab 04 | Lab 05 | Boost |
|---|---|---|---|
| Submitted by | ELLIS06 | ESROM11 | |
| RAM | 256 | 128 | 50% |
| CPU | 600 | 600 | |
| Width | 1024 | 1024 | |
| Height | 724 | 724 | |
| Image Aspect Ratio | 1.414365 | 1.414365 | |
| Started | 14:29:25 | 14:48:06 | |
| Finished | 14:54:55 | 15:17:55 | |
| Duration | 00:25:30 | 00:29:49 | 14% |



| Job | Boost | Lab 01 | Lab 04 | Lab 05 | Boost | Job: |
|---|---|---|---|---|---|---|
| Submitted by | | ERUDIN01 | ELLIS06 | ESROM11 | | Submitted by |
| RAM | | 256 | 256 | 128 | 50% | RAM |
| CPU | 64% | 933 | 600 | 600 | | CPU |
| Width | | 1024 | 1024 | 1024 | | Width |
| Height | | 724 | 724 | 724 | | Height |
| Image Aspect Ratio | | 1.414365 | 1.414365 | 1.414365 | | Image Aspect Ratio |
| Started | | 15:22:04 | 14:29:25 | 14:48:06 | | Started |
| Finished | | 15:41:30 | 14:54:55 | 15:17:55 | | Finished |
| Duration: | 24% | 00:19:26 | 00:25:30 | 00:29:49 | 14% | Duration |



# 6 | PROBLEMS

Problems encountered during the course of this project are of two types. There were problems associated with editing the 3DS VIZ scene and problems associated with the network rendering process.

## *Editing*

There were problems with editing the scene as they related to opening the file, using the viewports, adjusting raytraced shadow parameters to improve shadow edges and aligning objects.

### File Open

It was often observed that when opening the file from within 3DS VIZ (using *File > Open*) the application would crash. This was true even immediately after re-loading (re-ghosting) the machine.

The solution was to locate the file in Windows Explorer and launch 3DS VIZ by double-clicking the file's icon.

The reason for this problem is unknown.

### Viewports

3DS VIZ viewports would often refuse to render correctly. This problem was momentarily corrected by adopting a trial-and-error process of re-assigning one or another of the two video driver options (HEIDI, OpenGL or Direct3D – See figure, below).

Each re-assignment of the video-driver required 3DS VIZ to restart. The frustration caused by this was made worse by the 10-megabyte size of the file.

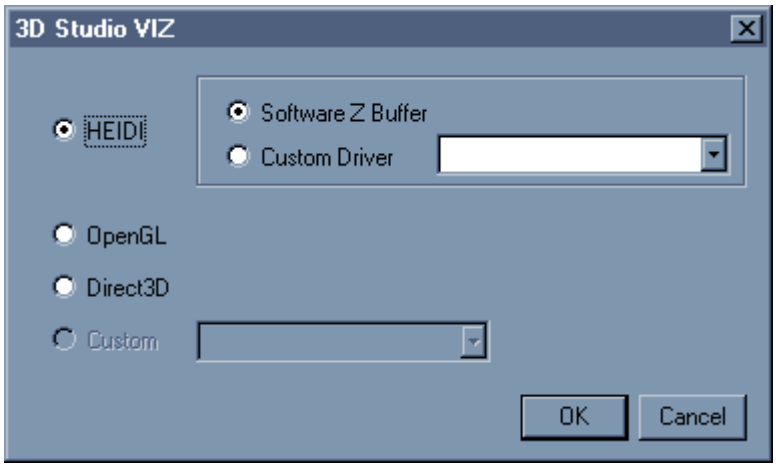



**Figure 10 - Display Driver Configuration dialogue**

### Raytraced Shadows

A problem was discovered with the edges of raytraced shadows. The shadows appeared without antialising where they fell against the terrain and not where they fell against the animal characters.

Attempts to soften the edges of the shadow failed. Eventually, after consideration of the editing history of the scene (the model was originally built in 3DS VIZ 2.0 then 3DS MAX 2.5 and finally 3DS VIZ 3.0) it was decided to merge the items into a new 3DS VIZ scene without the sun. A new sun was then added in 3DS VIZ. At around the same time, the large bitmap used for the pavement, with a file size around 100 megabytes, was revised to a series of smaller maps

One or both of these changes solved the problem.

### Object Alignment

Difficulty was experienced aligning the base of the characters over the sloping surface of the design's terrain. Attempts to use the Align tool failed because the z coordinate of the base of each character did not conform to the "max/min" paradigm of the command. All attempts in adjusting the z coordinate visually failed because the Viewport (Interactive) Renderer[24] would often render incorrectly.

## *Rendering*

There were several problems encountered with network rendering and these are set out below.

### Dynamic Host Configuration Protocol (DHCP)

The use of DHCP may produce a network-rendering vulnerability. "When you have more than one Manager running on the same network, each with its own set of dedicated servers, you need to specify which Manager to use. DHCP cannot be used in this case."[25]

At first, this 3DS VIZ advisory appeared to be resolved by adopting an easy work-around. All that seemed necessary was to identify the manager's DHCP assigned IP address and include it in the server's 'properties' dialogue. This appeared to work on all occasions.

However, given that dynamically allocated IP addresses have a limited lease, is it possible that one or more machines involved in any of the four concurrent processes (i.e. 3DS VIZ, server, manager and queue manager) could release its IP address before network-rendering is complete?

---

[24] "The interactive renderer, used for the viewports, is designed for speed so you can easily manipulate your objects in a shaded environment. It's not the same as the 3DS VIZ production renderer, which is used for your final images. Therefore, a number of effects that are available to the production renderer will not show up in the viewports." 3DS VIZ Help.

[25] 3DS VIZ Help

## File Type Limitation

It was discovered that the network-rendering feature of 3DS VIZ would only work with more than one “Server” machine when the file type chosen was not a “single user device”.

“If the output of a network-rendering job is an AVI, FLI, FLC, or CEL file or a single user device, the job can be assigned to a single server only. The Network Job Assignment dialog changes, depending on the file output type of a job. For example, if you are network rendering to one of the file formats above, the All and None buttons do not appear and the dialog title bar contains the word “Single.”

“If a job that has an AVI, FLI, FLC, or CEL file output type is stopped for any reason (to deactivate it, or because a machine goes down), re-rendering the file restarts at the first frame. Frames cannot be appended later to these file types.

“To take advantage of the distribution power of network rendering we suggest you first render to a series of Targa files. Then use the Targa files as an animated background in a blank 3D Studio VIZ design and render the sequence out to the desired output type (i.e. AVI).”[26]

## Queue Manager

The queue manager would often hang on initiation. In fact, it was generally a hit-and-miss procedure to get it running at all.[27]

Speculation has it that queue manager times-out on opening when there is a large amount of network traffic. Evidence for this is circumstantial but the problem appeared to coincide with a rise in network traffic.

Later rendering tasks lessen the likelihood of network traffic being the cause of the problem. Final tasks were carried out at night, in empty computer labs, yet the problem was still experienced.

## Shared Drives

It was necessary to set up a shared directory/folder when the destination folder was the hard drive on a local machine. This was not necessary when the destination folder was the student folder because the existing network infrastructure established the “share”. The problem was solved by the use of shared destination folders identified over the network using UNC.

## Load Timeout

A problem appeared with machines in Labs 04 and 05. It was experienced while running a single frame-rendering task in network rendering mode. Notification of the error was given in the server’s dialogue and read, “Rebooting MAX by force due to load timeout”. This error appeared

---

[26] *Ibid*.

[27] The success rate was about 1 in 2

after approximately 4 hours of successful rendering. The error caused queue manager to recover the render job assignment and start it all over again.

At the time of writing, this error had not been resolved. It is worth noting that the same task performed, in exactly the same circumstances, in Lab 01 failed to produce an error.

## Monitor Blackout

A machine's monitor [28] blacked-out while it was acting as the rendering initiator (i.e. the 3DS VIZ session) in a three-part network-rendering task. Usually, such an event would be dismissed as a platform error (i.e. an unknown hardware or software problem) and, as it was not a recurrent problem, would not warrant mention. However, it is worth noting because either;

The fault failed to induce an error in the network-rendering task. All attempts to revive the monitor, short of re-booting the machine, failed to bring it back to 'life'.

Or

The fault caused the network-rendering task to fail without feedback to the user. In this case, queue manager continued to show that the network-rendering server was 'assigned an active'. However, the manager, which was resident on the same machine as the queue manager, reported the message "Socket has been invalidated' See figure, below.

The 'Monitor Blackout' problem demonstrates the need to consider hardware features, such as power savers, and their effect on the network rendering process.

## Pic Saver

It was common for the Pic Saver application to fail.[29] An operating system alert box, which offered a 'close program' option, confirmed this. Choosing the 'close' option succeeded in closing the faulty application without causing any negative impact on the network-rendering task.

## IP address Conflict

It was noticed that during a three-part network rendering task the sever machine responded with "The system has detected a conflict for IP address 149.171.161.124 with the system having hardware address 00:C0:0C:03:7F:62

[28] BECU, Lab 04

[29] One such failure produced an alert box, in German (!), that read "Anwendungsfehler"

# 7 | FURTHER INVESTIGATION

There are four areas in which I recommend further investigation.

1. The first area of interest is the Windows Registry. Its usefulness became apparent in an article that appeared in the Sydney Morning Herald that outlined how to prevent the Windows screen saver from interrupting the disk defragmentation process.

   In the article[30], Ian Niedzwiecki wrote,

   "If you don't disable your screen saver before running Defrag, it'll interrupt the defragmentation process unless you move your mouse every few minutes to keep it from starting.

   "An easier solution, if you're running Windows 98, is to set up a registry entry to prevent it from running when you're defragmenting."

   The procedure was tested on a home PC and proven. Its success suggests that similar alterations to the Windows Registry would enhance the efficiency of network rendering by optimising the system's configuration.

2. The second area of interest is to investigate the pros and cons of using each of the three implementations of DHCP in a 3DS VIZ network-rendering environment.

   "DHCP supports three mechanisms for IP address allocation. In "automatic allocation", DHCP assigns a permanent IP address to a client. In "dynamic allocation", DHCP assigns an IP address to a client for a limited period of time (or until the client explicitly relinquishes the address). In "manual allocation", the network administrator assigns a client's IP address, and DHCP is used simply to convey the assigned address to the client. A particular network will use one or more of these mechanisms, depending on the policies of the network administrator."[31]

   It has been noted that even with the required user access privileges the use of DHCP must be restrained.

   "When the Server (or queue manager for that matter) is outside the local network (as in the case of a WAN or a multi-segmented network connected through a router) servers connected to the same network can still use DHCP, provided the Manager has a fixed name and IP address.

   "If your network is set peer-to-peer without an NT server (as is the case with most home networks), it is easier to set the machines with permanent, fixed IP addresses." [32]

---

[30] *Tip Exchange with Helen Bradley - Stop Screen Saver Havoc*. SMH IT, July 17, 2001, p5.

[31] Richards, H. *Understanding DHCP*. In *PC Network Advisor*, Issue 87 (September1997) pp. 11-14

[32] 3DS VIZ Help

3. The third area of interest is batch rendering using a Loopback adapter. “You may still use Network Rendering without being connected to a network as a means to process Batch Rendering. In this case, a fixed TCP/IP address is set up for use with Microsoft’s Loopback adapter.”[33]

4. The fourth area of interest are commercial services provided over the Internet via a 3DS VIZ plug-in similar to Intel’s Screamline Rendering Service. (See, Appendix B)

[33] *Ibid*.

# 8 | CONCLUSION

For the purposes of this report I have asked the question, why don't AEC professionals offer something closer to a cinematic experience in their animated content? In attempting to answer that question, I have come to know the boundary of existing practice, as found within the University of New South Wales' Faculty of the Built Environment.

It is clear that the restrictions acting to discourage developments in this direction are threefold. The first is the limitation inherent in the existing hardware. We will have to wait for that "two year gap" to close before desktop computers match the speed of Silicon Graphics Onyx systems that are currently used in the film industry.

The second restriction, which is closely related to the first, is the cost of production itself. There is a need for a project fee structure that reflects the costs involved. However, these costs are high when client demand is low. It will be the bolder design firms who seize the opportunity through acquiring expertise in this area that will create that demand.

A third and important restriction acting to discourage developments in the direction of high-quality animation in the AEC market is the existing nature of network technology and its current management. Existing local area networks (LANs) must perform a multitude of tasks for a multitude of users and no reasonable AEC Company is going to sacrifice its business-critical LAN for the sole purpose of rendering cinematic quality animations. It will take skill at the network administrative level to adjust existing LANs for the needs of network rendering.

At the time of writing, the network-rendering task still remains outside the realm of normal practice within the Faculty of the Built Environment, UNSW.

# A | LIMITATIONS

## *System*

The computer labs within the FBE are not configured for network rendering but they are vulnerable to it.

The first thing to note is that network rendering with 3DS VIZ is *not* supported by the existing network infrastructure. For network rendering "All machines need to be operational under the shipping version of Windows NT 4.0. Network rendering is not supported under the Windows 95 operating system. Although the network renderer can be set up and may appear functional, network rendering will not work reliably under Windows 95."[34]

Secondly, many network protocols and procedures are inaccessible to undergraduate students.

## *Accessible*

Accessible options within the existing network infrastructure are reliant upon a network communication system known as Dynamic Host Configuration Protocol (DHCP) that avoids the need for network administrative access privileges by adopting the DHCP assigned IP address.

There are two methods of allocating IP addresses using DHCP. Dynamic allocation assigns an IP address to a client for a limited period of time or until the client explicitly relinquishes the address while automatic allocation assigns a permanent IP address to a client.

Graham Hannah, the Built Environment Computer Unit (BECU) manager has confirmed that 'dynamic allocation' is implemented within the FBE computer labs.[35]

Use of winipcfg.exe has subsequently determined an IP address lease period of thirty-minutes.

## Directories

An early method adopted was to send completed frames to a folder within the student account. This approach worked without the need to set up a 'network share'. However, its limitation is the 50-megabyte limit imposed by the existing network infrastructure.

The alternative method is to set a network share and identify it using Universal Naming Convention (UNC). UNC names begin with a double backslash and do NOT include a drive letter. The convention is,

\\machine_name\directory\subdirectory\filename

---

[34] 3DS VIZ Help

[35] Confirmed by e-mail, June 2001.

There are two limitations with this method. One is that the network share is lost when the machine is re-loaded. The other is the insecure nature of the output or 'target' folder which would normally be placed somewhere on the C-drive. All output would be lost when the machine is re-loaded.[36]

## *Inaccessible*

Inaccessible options include direct manipulation of TCP/IP addresses in conjunction with the use of DCHP and the network-rendering process known as batch rendering.

### Batch Rendering

No investigation was made into this aspect of 3DS VIZ rendering due to access limitations. However, it is worth considering in gaining a broader understanding of the rendering options available.

"When you have no network card and no modem, you can still use network rendering to process batch rendering, in which case you configure your TCP/IP settings to use a software adapter supplied by Microsoft, called the Loopback adapter. This is a program that allows packets of data to be sent out and returned ("looped back") to a single machine."

Using Microsoft's Loopback Adapter you can set-up the Network Renderer to function on a single non-networked computer. Although a Network Card is not required for Network Rendering, you still must set up the TCP/IP protocol to use the batch rendering capabilities of the Network Renderer. To do this, you must first install the MS Loopback Adapter, which allows packets of data to be sent out and returned ("looped back") to a single machine that does not have a network adapter card.

When the MS Loop Back Adapter and the TCP/IP protocol are correctly installed, set up the machine as you would any other machine for Network Rendering. Both the Manager and Server programs must be running and correctly communicating with one another before launching 3D Studio VIZ. Then you can use the machine to submit multiple jobs to itself to execute batch renders."[37]

It is worth noting that batch rendering does not work with Windows 9x because at least two sessions of 3DS VIZ need to be opened simultaneously - one for the render initiation process and one for the server process. Simultaneous sessions of 3DS VIZ are not possible using Windows 9x.

### Windows Registry

In the execution of a network-rendering task, it is worth considering the full range of system processes in order to maximise the network rendering efficiency. Non-essential processes such

---

[36] This was later made more 'secure' when the output folder was assigned to the amazingly hidden D-drive.

[37] 3DS VIZ Help

as screensavers and shared resources (eg. the situation where a render server may act as a print server) should be controlled.

As an aid in achieving better control over the full range of operating system processes, the Windows Registry is an obvious starting point.

# B | SCREAMLINE

Screamline Rendering Service promised a seamless integration between the 3DS VIZ editing environment and a remote high-speed network via the Internet. It was hoped that the offer of a FREE 1000 FRAMES would have formed the basis of an exploration into a new approach to rendering animations using 3DS VIZ/MAX. However, 3D Studio MAX 3.1 was required in order to use the service and an FBE license for this software was not found in time for inclusion in this project.

The following press release[38] touts for new customers of the service.

## Screamline Rendering Services Launches High-Performance Web-Based Graphics Rendering Service with Free Trial Offer

**Free JobAnalyst. Software Delivers Time and Cost Estimates**

January 29, 2001. Screamline Rendering Services (www.screamlinerendering.com) is offering the 3D animation and visual effects community a special trial offer to introduce its Web-based rendering services and project management software.

Animation and visual effects professionals can download Screamline service's patent-pending JobAnalyst management tool at no cost, and then upload their rendering projects. Then, for a limited-time, they can enjoy free rendering services for the first 1000 frames rendered by the Screamline service to test personally the speed, security and reliability of the Screamline service and the benefits of the JobAnalyst software.

Robert J. McLaurin, General Manager for Internet Computing Services, highlighted the simplicity of the Screamline service model: "We make it very easy to send jobs in for rendering and retrieve them using existing Internet connections. There are no long-term contracts, so customers can submit jobs to the render farm on an as-needed basis and know up-front what the job will cost."

Unlike other rendering services, the Screamline service charges by frames and not its estimated run time, an advance made possible by the JobAnalyst project management application. The JobAnalyst software is designed to run on customers own workstations.

The JobAnalyst tool incorporates sophisticated algorithms that quantify processing requirements to generate a complexity factor for each project. This ensures an accurate quote for both cost and delivery time before the submission of a job. It alerts customers to potential problems with animation files before they submit the project.

Fundamental to the JobAnalyst tool is control of all aspects of the job. Key features include the ability to set processing options; to monitor and manage jobs anytime day or night though the Internet; and to check each project in advance, determining quickly if any of the files needed are unavailable or inaccessible.

In addition, the JobAnalyst tool supports secure communication with the server array. The JobAnalyst program is free to any animation and visual effects professionals who register at the

[38] Online (02/11/01) see, http://www.3dlinks.com/press/Screamline-29-Jan-2001.cfm

Screamline Rendering Services web site, www.screamlinerendering.com.

**Screamline Pricing**

The Screamline service charges on a per-frame basis, with the total cost determined by a job's complexity. Factors that determine complexity include the number of objects, polygons, light sources, shaders, and textures, as well as job settings such as anti-aliasing, motion blur, ray tracing and resolution.

The JobAnalyst software reports the complexity of every job to the customer prior to processing. The complexity factor lets the customer see directly the affects of changing rendering settings. It also supports a comparison of the cost of processing different projects on an "apples to apples" basis.

**System Requirements**

The Screamline Rendering Services render farm, populated by Intel-based servers, hosts many top computer graphic and rendering packages including Maya, by Alias Wavefront. Use of the JobAnalyst program for Maya requires only an Internet connection; a workstation with either Microsoft. Windows. NT 4 SP6 or later or Windows 2000, or SGIs IRIX 5.6 or later; and a licensed copy of Maya version 2.5 or later. JobAnalyst for Discreet's 3D Studio MAX and Pixar's Photo-realistic Renderman are scheduled for release this quarter.

---

**About Screamline Rendering Service**

The Screamline Rendering Services are provided by Intel's Internet Computing Services, an Application Service Provider serving and supporting applications requiring the highest level of computing performance for highly parallel, time-critical applications. Internet Computing Services is a part of the Intel New Business Group, which was formed to nurture and grow new market opportunities.

## *Update*

Future opportunities to investigate Screamline Rendering Service are now in doubt because the service has been withdrawn. The reasons were explained in an e-mail sent to registered members.

From: Rigby, David A [SMTP:david.a.rigby@intel.com]
To: 'alang@fbe.unsw.edu.au'
Cc:
Subject: Thanks from Screamline Rendering Services
Sent: 20/10/01 4:23
Importance: Normal

October 19, 2001

Screamline Rendering Services, provided by Internet Computing Services, has enjoyed working with our customers in the computer graphics animation industry to provide outsourced rendering services.

Due to a deteriorated funding climate for Internet-based businesses and unfavorable market conditions, Internet Computing Services, part of the Intel New Business Group, has decided to exit the outsourced rendering services business.

**This service will be unavailable effective October 31, 2001**

We value our clients and will work with you to complete existing projects

If you have any questions or concerns about your scheduled projects, contact Customer Support at 1-866-RENDERS (1-866-736-3377) or via email at

support@screamlinerendering.com.

Again, thank you for your patronage and support of Screamline Rendering Services.

David Rigby

Customer Support Manager
Screamline Rendering Services

# C | SPECIFICATIONS

## *Objects:*

| Name (Type) | Verts | Faces | Shadows | Shadows | Blur | Hidden | Frozen | Material (Type) |
|---|---|---|---|---|---|---|---|---|
| RegPoint01 (Editable Mesh) | 28 | 26 | Yes | Yes | No | No | No | Blank (Standard) |
| GoatBase01 (Editable Mesh) | 1247 | 2452 | Yes | Yes | No | No | No | GoatBody01 (Standard) |
| GoatHornLeft01 (Editable Mesh) | 422 | 840 | Yes | Yes | No | No | No | GoatHorn01 (Standard) |
| GoatHornRight01 (Editable Mesh) | 422 | 840 | Yes | Yes | No | No | No | GoatHorn01 (Standard) |
| HorseBodyLeft01 (Editable Mesh) | 718 | 1258 | Yes | Yes | No | No | No | HorseBodyLeft02 (Standard) |
| HorseBodyRight01 (Editable Mesh) | 718 | 1258 | Yes | Yes | No | No | No | HorseBodyRight02 (Standard) |
| HorseEar01 (Editable Mesh) | 662 | 1320 | Yes | Yes | No | No | No | HorseEar01 (Standard) |
| HorseEar02 (Editable Mesh) | 662 | 1320 | Yes | Yes | No | No | No | HorseEar01 (Standard) |
| MonkeyBase01 (Editable Mesh) | 442 | 805 | Yes | Yes | No | No | No | MonkeyBodyFront (Standard) |
| MonkeyBase02 (Editable Mesh) | 441 | 803 | Yes | Yes | No | No | No | MonkeyBodyRear (Standard) |
| MonkeyEar01 (Editable Mesh) | 176 | 326 | Yes | Yes | No | No | No | MonkeyEarFront (Standard) |
| MonkeyEar02 (Editable Mesh) | 176 | 326 | Yes | Yes | No | No | No | MonkeyEarFront (Standard) |
| MonkeyEar03 (Editable Mesh) | 169 | 312 | Yes | Yes | No | No | No | MonkeyEarRear (Standard) |
| MonkeyEar04 (Editable Mesh) | 169 | 312 | Yes | Yes | No | No | No | MonkeyEarRear (Standard) |
| MonkeyHead01 (Editable Mesh) | 231 | 426 | Yes | Yes | No | No | No | MonkeyHeadFront (Standard) |
| MonkeyHead02 (Editable Mesh) | 232 | 428 | Yes | Yes | No | No | No | MonkeyHeadRear01 (Standard) |
| PigBase01 (Editable Mesh) | 101 | 176 | Yes | Yes | No | No | No | PigBaseFront01 (Standard) |
| PigBase02 (Editable Mesh) | 101 | 176 | Yes | Yes | No | No | No | PigBaseRear01 (Standard) |
| PigEar01 (Editable Mesh) | 58 | 112 | Yes | Yes | No | No | No | PigHeadRear (Standard) |
| PigEar02 (Editable Mesh) | 58 | 112 | Yes | Yes | No | No | No | PigHeadRear (Standard) |
| PigHead01 (Editable Mesh) | 231 | 426 | Yes | Yes | No | No | No | PigHeadFront (Standard) |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| PigHead02 (Editable Mesh) | 232 | 428 | Yes | Yes | No | No | No | PigHeadRear01 (Standard) |
| PigSnout02 (Editable Mesh | 50 | 96 | Yes | Yes | No | No | No | Material #345 (Multi/Sub-Object) |
| TigerBody01 (Editable Mesh) | 1172 | 2340 | Yes | Yes | No | No | No | TigerComposite01 (Standard) |
| TigerEarLeft02 (Editable Mesh) | 1297 | 2590 | Yes | Yes | No | No | No | TigerEar01 (Standard) |
| TigerEarLeft03 (Editable Mesh) | 1297 | 2590 | Yes | Yes | No | No | No | TigerEar01 (Standard) |
| TigerTail01 (Editable Mesh) | 223 | 442 | Yes | Yes | No | No | No | TigerTail (Standard) |
| OxBodyLeft01 (Editable Mesh) | 1331 | 2436 | Yes | Yes | No | No | No | OxBodyLeft01 (Standard) |
| OxBodyRight01 (Editable Mesh) | 794 | 1400 | Yes | Yes | No | No | No | OxBodyRight01 (Standard) |
| OxHornLeft01 (Editable Mesh) | 602 | 1200 | Yes | Yes | No | No | No | OxHorn01 (Standard) |
| OxHornRight01 (Editable Mesh) | 602 | 1200 | Yes | Yes | No | No | No | OxHorn01 (Standard) |
| RoosterBody01 (Editable Mesh) | 12825 | 25108 | Yes | Yes | No | No | No | RoosterBody01 (Standard) |
| Foliage01 (Foliage) | 647 | 1133 | Yes | Yes | No | No | No | Ghost (Standard) |
| Foliage02 (Foliage) | 682 | 1193 | Yes | Yes | No | No | No | Ghost (Standard) |
| Foliage03 (Foliage) | 654 | 1145 | Yes | Yes | No | No | No | Ghost (Standard) |
| Foliage04 (Foliage) | 677 | 1188 | Yes | Yes | No | No | No | Ghost (Standard) |
| Foliage05 (Foliage) | 649 | 1134 | Yes | Yes | No | No | No | Ghost (Standard) |
| Foliage06 (Foliage) | 654 | 1149 | Yes | Yes | No | No | No | Ghost (Standard) |
| SnakeHead01 (Editable Mesh) | 2822 | 5640 | Yes | Yes | No | No | No | SnakeSkin01 (Standard) |
| DogBody01 (Editable Mesh) | 1013 | 1821 | Yes | Yes | No | No | No | DogBodyFront01 (Standard) |
| DogBody02 (Editable Mesh) | 1013 | 1821 | Yes | Yes | No | No | No | DogBodyRear01 (Standard) |
| DogEar01 (Editable Mesh) | 56 | 108 | Yes | Yes | No | No | No | DogBrownOnly (Standard) |
| DogEar02 (Editable Mesh) | 56 | 108 | Yes | Yes | No | No | No | DogPinkEar01 (Multi/Sub-Object) |
| DogHeadFront (Editable Mesh) | 359 | 588 | Yes | Yes | No | No | No | DogHead01 (Standard) |
| RabcatCatBody01 (Editable Mesh) | 222 | 360 | Yes | Yes | No | No | No | Material #16 (Standard) |
| RabcatCatEar01 (Editable Mesh) | 482 | 960 | Yes | Yes | No | No | No | RabCatCat01 (Standard) |
| RabcatCatHead01 (Editable Mesh) | 1980 | 3731 | Yes | Yes | No | No | No | RabCatCat01 (Standard) |
| RabcatRabbitHead01 (Editable Mesh) | 1909 | 3629 | Yes | Yes | No | No | No | RabCatRabbit01 (Standard) |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Ratbody01 (Editable Mesh) | 243 | 447 | Yes | Yes | No | No | No | RatBodyCompDiffuse (Standard) |
| RatFace01 (Editable Mesh) | 338 | 672 | Yes | Yes | No | No | No | RatFace (Standard) |
| RatFace02 (Editable Mesh) | 338 | 672 | Yes | Yes | No | No | No | RatFace (Standard) |
| RatFace03 (Editable Mesh) | 338 | 672 | Yes | Yes | No | No | No | RatFace (Standard) |
| RatHalfEar01 (Editable Mesh) | 1333 | 2628 | Yes | Yes | No | No | No | RatEar01 (Standard) |
| RatHalfEar02 (Editable Mesh) | 1333 | 2628 | Yes | Yes | No | No | No | RatEar01 (Standard) |
| RatHalfEar03 (Editable Mesh) | 1333 | 2628 | Yes | Yes | No | No | No | RatEar01 (Standard) |
| RatHalfEar04 (Editable Mesh) | 1333 | 2628 | Yes | Yes | No | No | No | RatEar01 (Standard) |
| RatHalfEar05 (Editable Mesh) | 1333 | 2628 | Yes | Yes | No | No | No | RatEar01 (Standard) |
| RatHalfEar06 (Editable Mesh) | 1333 | 2628 | Yes | Yes | No | No | No | RatEar01 (Standard) |
| DragonScallop01 (Editable Mesh) | 3459 | 6914 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHeadTooth04 (Cone) | 146 | 288 | Yes | Yes | No | No | No | DragonTeeth01 (Standard) |
| DragonHeadTooth03 (Cone) | 146 | 288 | Yes | Yes | No | No | No | DragonTeeth01 (Standard) |
| DragonHeadTooth02 (Cone) | 146 | 288 | Yes | Yes | No | No | No | DragonTeeth01 (Standard) |
| DragonHeadTooth01 (Cone) | 146 | 288 | Yes | Yes | No | No | No | DragonTeeth01 (Standard) |
| DragonHeadNeckLoft01 (Editable Mesh) | 978 | 1880 | Yes | Yes | No | No | No | DragonBodyGreen01 (Standard) |
| DragonHeadEye02 (Editable Mesh) | 482 | 960 | Yes | Yes | No | No | No | DragonEye01 (Standard) |
| DragonHeadEye01 (Editable Mesh) | 482 | 960 | Yes | Yes | No | No | No | DragonEye01 (Standard) |
| DragonHead01 (Editable Mesh) | 4230 | 8456 | Yes | Yes | No | No | No | DragonHead (Multi/Sub-Object) |
| DragonHair44 (Editable Mesh) | 2306 | 4608 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair43 (Editable Mesh) | 2306 | 4608 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair42 (Editable Mesh) | 40 | 76 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair41 (Editable Mesh) | 40 | 76 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair39 (Editable Mesh) | 40 | 76 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair36 (Editable Mesh) | 40 | 76 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair35 (Editable Mesh) | 40 | 76 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair34 (Editable Mesh) | 49 | 94 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| DragonHair33 (Editable Mesh) | 45 | 86 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair32 (Editable Mesh) | 42 | 80 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair31 (Editable Mesh) | 46 | 88 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair30 (Editable Mesh) | 40 | 76 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair29 (Editable Mesh) | 64 | 124 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair28 (Editable Mesh) | 39 | 74 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair27 (Editable Mesh) | 37 | 70 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair26 (Editable Mesh) | 43 | 82 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair25 (Editable Mesh) | 44 | 84 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair24 (Editable Mesh) | 34 | 64 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair23 (Editable Mesh) | 65 | 126 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair22 (Editable Mesh) | 59 | 114 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair20 (Editable Mesh) | 204 | 404 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair19 (Editable Mesh) | 129 | 254 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair18 (Editable Mesh) | 157 | 310 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair17 (Editable Mesh) | 97 | 190 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair16 (Editable Mesh) | 138 | 272 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair15 (Editable Mesh) | 43 | 82 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair09 (Editable Mesh) | 58 | 112 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair08 (Editable Mesh) | 51 | 98 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair07 (Editable Mesh) | 28 | 52 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair06 (Editable Mesh) | 38 | 72 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair05 (Editable Mesh) | 23 | 42 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair03 (Editable Mesh) | 45 | 86 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair02 (Editable Mesh) | 25 | 46 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| DragonHair01 (Editable Mesh) | 55 | 106 | Yes | Yes | No | No | No | DragonFinRed01 (Standard) |
| Terrain01 (Editable Mesh) | 4 | 2 | Yes | Yes | No | No | No | Terrain (Standard) |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| PavementLine01 (Editable Mesh) | 149 | 147 | Yes | Yes | No | No | No | HueA (Standard) |
| PavementLine02 (Editable Mesh) | 100 | 98 | Yes | Yes | No | No | No | HueB (Standard) |
| PavementLine03 (Editable Mesh) | 52 | 50 | Yes | Yes | No | No | No | HueA (Standard) |
| PavementLine04 (Editable Mesh) | 100 | 98 | Yes | Yes | No | No | No | HueB (Standard) |
| PavementLine05 (Editable Mesh) | 100 | 98 | Yes | Yes | No | No | No | HueA (Standard) |
| PavementLine06 (Editable Mesh) | 3 | 1 | Yes | Yes | No | No | No | HueB (Standard) |
| PavementLine07 (Editable Mesh) | 101 | 99 | Yes | Yes | No | No | No | HueA (Standard) |
| PavementLine08 (Editable Mesh) | 101 | 99 | Yes | Yes | No | No | No | HueB (Standard) |
| PavementLine09 (Editable Mesh) | 100 | 98 | Yes | Yes | No | No | No | HueA (Standard) |
| PavementLine10 (Editable Mesh) | 100 | 98 | Yes | Yes | No | No | No | HueB (Standard) |
| PavementLine11 (Editable Mesh) | 100 | 98 | Yes | Yes | No | No | No | HueA (Standard) |
| PavementLine12 (Editable Mesh) | 52 | 50 | Yes | Yes | No | No | No | HueB (Standard) |
| FountainKnob01 (Editable Mesh) | 338 | 672 | Yes | Yes | No | No | No | FountainSpiral (Standard) |
| FountainKnobCylinder01 (Editable Mesh) | 470 | 936 | Yes | Yes | No | No | No | Default (Standard) |
| FountainKnobWater01 (Editable Mesh) | 1376 | 2748 | Yes | Yes | No | No | No | WaterBlue (Standard) |

## *Cameras:*

| Name (Type) | Hidden | Frozen |
|---|---|---|
| Camera01 (Free Camera) | No | No |
| Camera03 (Free Camera) | No | No |
| Camera02 (Free Camera) | No | No |
| Camera04 (Free Camera) | No | No |

## *Lights:*

| Name (Type) | Hidden | Frozen |
|---|---|---|
| OmniSub01 (Omni Light) | No | No |
| OmniSub02 (Omni Light) | No | No |
| OmniSub03 (Omni Light) | No | No |
| OmniSub04 (Omni Light) | No | No |
| OmniSub05 (Omni Light) | No | No |
| OmniSub06 (Omni Light) | No | No |
| OmniSub07 (Omni Light) | No | No |
| OmniSub08 (Omni Light) | No | No |
| OmniSub09 (Omni Light) | No | No |
| OmniSub10 (Omni Light) | No | No |
| OmniSub11 (Omni Light) | No | No |
| OmniSub12 (Omni Light) | No | No |
| Omni01 (Omni Light) | No | No |
| Omni02 (Omni Light) | No | No |
| Omni03 (Omni Light) | No | No |

| | | |
|---|---|---|
| Omni04 (Omni Light) | No | No |
| Sun01 (Free Directional Light) | No | No |

## *Helpers:*

| Name (Type) | Hidden | Frozen |
|---|---|---|
| Goat01 (Dummy) | No | No |
| Horse01 (Dummy) | No | No |
| Monkey01 (Dummy) | No | No |
| Pig01 (Dummy) | No | No |
| Tiger01 (Dummy) | No | No |
| Ox01 (Dummy) | No | No |
| Rooster01 (Dummy) | No | No |
| Dog02 (Dummy) | No | No |
| Rabcat01 (Dummy) | No | No |
| Rat02 (Dummy) | No | No |
| Dragon03 (Dummy) | No | No |
| Compass01 (Object) | No | No |
| Foutain01 (Dummy) | No | No |

## *Scene Totals:*

Objects: 117
Shapes: 0
Cameras: 4
Lights: 17
Helpers: 13
Total: 151

## *Mesh Totals:*

Verts: 70990
Faces: 136421

## *Materials:*

GoatBody01 (Standard)
C:\2254145\viz\map\GoatBody01.GIF

GoatHorn01 (Standard)
C:\2254145\viz\map\GoatHorn01.GIF

HorseBodyLeft02 (Standard)
C:\2254145\viz\map\HorseBodyLeft03.GIF

HorseBodyRight02 (Standard)
C:\2254145\viz\map\HorseBodyRight03.GIF

HorseEar01 (Standard)
C:\2254145\viz\map\HorseEar01.GIF

MonkeyBodyFront (Standard)
C:\2254145\viz\map\MonkeyBodyFront01.GIF

MonkeyBodyRear (Standard)
C:\2254145\viz\map\MonkeyBodyRear01.GIF

MonkeyEarFront (Standard)
C:\2254145\viz\map\MonkeyEarFront01.GIF

MonkeyEarRear (Standard)
C:\2254145\viz\map\MonkeyEarRear01.GIF

MonkeyHeadFront (Standard)
C:\2254145\viz\map\MonkeyHeadFront01.GIF

MonkeyHeadRear01 (Standard)
C:\2254145\viz\map\MonkeyBodyFront01.GIF

PigBaseFront01 (Standard)
C:\2254145\viz\map\PigBodyFront01.GIF

PigBaseRear01 (Standard)
C:\2254145\viz\map\PigBodyRear01.GIF

PigHeadRear (Standard)
C:\2254145\viz\map\PigHeadRear01.GIF

PigHeadFront (Standard)
C:\2254145\viz\map\PigHeadFront01.GIF

PigHeadRear01 (Standard)
C:\2254145\viz\map\PigHeadRear01.GIF

**Material #345** (Multi/Sub-Object)
SnoutSide (Standard)
Snout (Standard)
C:\2254145\viz\map\PigNose01.GIF

TigerComposite01 (Standard)
C:\2254145\viz\map\TigerSmile02.GIF
C:\2254145\viz\map\TigerSide01.GIF
C:\2254145\viz\map\TigerBack01.GIF
C:\2254145\viz\map\TigerFeet01.GIF
C:\2254145\viz\map\TigerBum01.GIF

TigerEar01 (Standard)
C:\2254145\viz\map\TigerEarRear.GIF
C:\2254145\viz\map\TigerEarFront.GIF

TigerTail (Standard)
C:\2254145\viz\map\TigerBack01.GIF

OxBodyLeft01 (Standard)
C:\2254145\viz\map\OxBodyLeft02.GIF

OxBodyRight01 (Standard)
C:\2254145\viz\map\OxBodyRight02.GIF

OxHorn01 (Standard)

RoosterBody01 (Standard)
C:\2254145\viz\map\RoosterBody02.GIF

Ghost (Standard)
C:\program files\3DSVIZ3\maps\sky.jpg

SnakeSkin01 (Standard)
C:\2254145\viz\map\SnakeBodyWrap02.GIF

DogBodyFront01 (Standard)
C:\2254145\viz\map\DogBodyFront01.GIF

DogBodyRear01 (Standard)
C:\2254145\viz\map\DogBodyRear01.GIF

DogBrownOnly (Standard)
C:\2254145\viz\map\DogHeadRear01.GIF

DogPinkEar01 (Multi/Sub-Object)
**Material #4** (Standard)
C:\2254145\viz\map\DogHeadRear01.GIF
**Material #5** (Standard)
C:\2254145\viz\map\DogEarRear01.GIF

DogHead01 (Standard)
C:\2254145\viz\map\DogHeadFront01.GIF

**Material #16** (Standard)
C:\2254145\viz\map\RabcatBase01GIF.gif

RabCatCat01 (Standard)
C:\2254145\viz\map\RabcatCat01.GIF

RabCatRabbit01 (Standard)
C:\2254145\viz\map\RabcatRabbit01.GIF

RatBodyCompDiffuse (Standard)
C:\2254145\viz\map\RatPurple01.GIF
C:\2254145\viz\map\RatTail02.GIF

RatFace (Standard)
C:\2254145\viz\map\RatPurple01.GIF
C:\2254145\viz\map\RatFace01.GIF

RatEar01 (Standard)
C:\2254145\viz\map\RatPurple01.GIF
C:\2254145\viz\map\RatEar01.GIF

DragonFinRed01 (Standard)
C:\2254145\viz\map\DragonBodyRed01.GIF

DragonTeeth01 (Standard)

DragonBodyGreen01 (Standard)
C:\2254145\viz\map\DragonBodyGreen01.GIF

DragonEye01 (Standard)
C:\2254145\viz\map\DragonEye01.GIF

DragonHead (Multi/Sub-Object)
DragonNose (Standard)
C:\2254145\viz\map\DragonNose04.TIF
DragonMouth01 (Standard)
DragonMouthHole (Standard)
DragonWhite (Standard)
C:\2254145\viz\map\DragonFaceWhite.TIF

Terrain (Standard)

HueA (Standard)
C:\2254145\viz\map\centerB1.tif

HueB (Standard)
C:\2254145\viz\map\centerA1.tif

FountainSpiral (Standard)
C:\2254145\viz\map\fountain01.gif

WaterBlue (Standard)
C:\2254145\viz\map\trickle01.avi

**Environment Map:**
**Map #1** (skyblue01.tif)
C:\2254145\viz\map\skyblue01.tif