Performance Analysis For Real-Time 3D Computer Graphics

Arthur Lorenz
arthur.lorenz@gmx.net

Munich University Of Applied Sciences
Fachhochschule München
Presentation Overview

- **Overview of important concepts** to achieve real-time 3D computer graphics
- **Considerations** before benchmarking
- **Two** benchmark **examples** using **pbench**
Presentation Overview

- **Overview of important concepts** to achieve real-time 3D computer graphics

- **Considerations** before benchmarking

- **Two** benchmark **examples** using **pbench**
Presentation Overview

- **Overview of important concepts** to achieve real-time 3D computer graphics
- **Considerations** before benchmarking
- **Two** benchmark **examples** using **pbench**
Real-Time 3D Computer Graphics

The immediate generation of images in a computer system through rendering data at an interactive frame rate.

Hz = fps

(e.g., 1, 24, 60, 120 Hz)
Different Settings

- Multi-processing hardware
- Graphics hardware
- Operating system (Linux)
- Application software (SGI OpenGL Performer)
- Scene graph design
Operating System (I)

- Disabling unnecessary system services
- Disabling swapping
- Kernel
  - Improving preemption
  - Additional abstraction layer and interrupt dispatcher
- Scheduling
- Soft and hard affinity
Operating System (I)
Affinity

- Hard affinity requires kernel patch for Linux kernel 2.4.x
- Linux kernel 2.6.x supports hard affinity per se
Application Software (I)

- Culling
  - Back-face (OpenGL)
  - View frustum (SGI OpenGL Performer)
  - Occlusion (Radeon 8500, GeForce 3)
  - Contribution

- SGI OpenGL Performer
Application Software (II) (SGI OpenGL Performer)

- Multiple Stages
  - Synchronous (APP, CULL, DRAW)
  - Asynchronous (ISECT, DBASE, LPOINT, COMPUTE)

- Optimized algorithms
Multiple Stages

- Host/CPU Hardware
- PCI/AGP-BUS
- Graphics Pipeline
- APP
- CULL
- DRAW
- Geometry Engine
- Raster Manager
- Display Generator
- ISECT
- DBASE
- LPOINT
- COMPUTE
Application Software (III) (SGI OpenGL Performer)

- Multiple Stages

Diagram:
- Host/CPU Hardware
- PCI/AGP-BUS
- Graphics Pipeline
- APP ➔ CULL ➔ DRAW ➔ Geometry Engine ➔ Raster Manager ➔ Display Generator
- ISECT
- DBASE
- LPOINT
- COMPUTE
Application Software (III) (SGI OpenGL Performer)

- Multiple Stages

Host/CPU Hardware → PCI/AGP-BUS → Graphics Pipeline

APP → CULL → DRAW → Geometry Engine → Raster Manager → Display Generator

ISECT → DBASE → LPOINT → COMPUTE
Application Software (III) (SGI OpenGL Performer)

- Multiple Stages

Host/CPU Hardware  ➔  PCI/AGP-BUS  ➔  Graphics Pipeline

APP ➔  CULL ➔  DRAW ➔  Geometry Engine ➔  Raster Manager ➔  Display Generator

ISECT
DBASE
LPOINT
COMPUTE
Scene Graph Design

- LOD scaling

LOD0  LOD1  LOD2

distance
Presentation Overview

- **Overview of important concepts** to achieve real-time 3D computer graphics

- **Considerations** before benchmarking

- **Two** benchmark examples using **pbench**
Performance Aspects And Benchmarking

- Some words on benchmarking
- Measurement considerations
- Measurement methods
- Measurement variables
- Traversing the scene
Some Words On Benchmarking

- Corresponds to measuring specific object properties
- Performance comparisons
- Finding bottlenecks
- No scene graph modifications
- Editing application source code not necessary (inefficient code parts)
Measurement Considerations

- Disabling unnecessary system activities
  - System services
  - Swapping
  - Power saving (APM, DPMS)
  - Input devices

- Well-defined benchmark environment
  - System settings
  - Environment variables
Measurement Methods

- Observation
- System utilities (e.g., vmstat, mpstat, sar)
- Querying
  - System (e.g., /proc file system in Linux)
  - Application (e.g., Performer statistics)
- Multiple benchmark runs
Measurement Variables

- Benchmark runtime
- Frame rendering time
- Frame misses
- etc.

Querying variables at runtime may falsify results and influence other measurement values!
Traversing The Scene (I)

- View alignment

Human Mode  Drive Mode  Flight Mode
Traversing The Scene (II)

- Meander path
- User-defined paths possible
Benchmarked Scene Graph ("town")
Benchmarked Scene Graphs ("Drive")
Benchmarked Scene Graphs ("Flight")
## Benchmarked Systems

<table>
<thead>
<tr>
<th>Hardware</th>
<th>up_bimoto</th>
<th>dp_bimoto</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>Intel Pentium 4 Xeon</td>
<td>2 x Intel Pentium 4 Xeon</td>
</tr>
<tr>
<td>Frequency [GHz]</td>
<td>2.4</td>
<td>.</td>
</tr>
<tr>
<td>Main Memory [MB]</td>
<td>512</td>
<td>.</td>
</tr>
<tr>
<td>Graphics Board</td>
<td>NVIDIA GeForce FX 5900</td>
<td>.</td>
</tr>
<tr>
<td>Graphics Driver</td>
<td>44.96</td>
<td>.</td>
</tr>
<tr>
<td>Operating System</td>
<td>SuSE Linux 8.2</td>
<td>.</td>
</tr>
<tr>
<td>Linux Kernel</td>
<td>2.4.25 + affinity patch</td>
<td>.</td>
</tr>
<tr>
<td>Application Software</td>
<td>SGI OpenGL Performer 3.1</td>
<td>.</td>
</tr>
</tbody>
</table>
Presentation Overview

- **Overview of important concepts** to achieve real-time 3D computer graphics

- **Considerations** before benchmarking

- **Two benchmark examples** using pbench
Benchmark Application pbench

- Collects data for **detailed offline analyses**
- **User-defined paths** through the scene
- Allows **collection** of statistical classes and variables available in Performer
- **Little influence** on system performance
- Displays **graphical statistics** using gnuplot
Benchmark Demo ("town")
Other Tools

- **meander** creates a configurable meander path through the scene
- **getaffinity/setaffinity** for getting and setting hard affinity of processes
- **getsched/setsched** for getting and setting the scheduling strategy of processes
- **Several Perl scripts** for automating multiple benchmark runs and averaging collected data
Benchmarks Conducted With \textit{pbench}

- LOD scaling
- Multi-processing with hard affinity
- Many more benchmarks in the thesis
LOD Scaling

Process Time (TOTAL)

Average Process Time (TOTAL) Performance Increase [%]

- Town: 21.23%
- Drive: 311.22%
- Flight: 778.87%

Scene Graph
LOD Scaling
("Drive", DRAW)
LOD Scaling (Results)

- Can enormously boost scene graph performance
- Controls polygonal complexity of models in the scene (CULL stage)
- LOD levels must be available in the scene graph
- Popping up of objects or transitions between LOD levels may be discernable (solution: LOD fading)
Multi-Processing With Hard Affinity (I)

Scene Graph

Average Process Times (TOTAL) [ms]

- town
- Drive
- Flight

Legend:
- up_bimoto
- dp_bimoto
- dp_bimoto + HA
Multi-Processing With Hard Affinity (II)

Average Process Time (TOTAL) Performance Increase [%]

- town: 5.23%
- Drive: -1.40%
- Flight: 6.71%

Scene Graph

CPU0

APP
CULL
DRAW

CPU0

APP
CULL
DRAW

CPU1
Hard Affinity (Results)

- Better exploitation of CPU caches by using hard affinity
- System must be configured for specific applications
- Synchronization between processes costs performance
Summary

- Many hard and software parameters can help achieve real-time 3D computer graphics

- Benchmarking
  - must be well prepared
  - yields results for comparisons
  - helps find bottlenecks

- Scene graph, application, and system need to be tuned in an iterative process for optimal performance
Any Questions?
Add-On Slides
Graphics Hardware

- Color Depth (16 bpp, 24/32 bpp)

- Resolution (320x200, ..., 2048x1536)

- Utilizing hardware-optimized functions
  - Fast primitives (e.g., triangle strips)
  - Vertex and pixel shaders
Multi-Processing Hardware

- UMA (Unified Memory Architecture)
- SMP (Symmetrical Multi-Processing)
- ASMP (Asymmetrical Multi-Processing)
Operating System (II)

- Scheduling

<table>
<thead>
<tr>
<th>Feature</th>
<th>Linux Kernel 2.4</th>
<th>RTLinux Kernel 2.4</th>
<th>RTAI Kernel 2.4</th>
<th>Linux Kernel 2.6</th>
<th>Windows 2000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Soft Real-Time</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Hard Real-Time</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Dynamic Priority Scheduling</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>First-In First-Out Scheduling (FIFO)</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Round-Robin Scheduling (RR)</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Rate Monotonic Scheduling (RMS)</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Earliest Deadline Scheduling (EDF)</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Application Software (I)
Back-Face Culling

- OpenGL
Application Software (I)
Back-Face Culling

- OpenGL
Application Software (I) View Frustum Culling

- SGI OpenGL Performer

\[ R = \text{Root Node} \]
Application Software (I) View Frustum Culling

- SGI OpenGL Performer

\( \mathbb{R} = \text{Root Node} \)
Application Software (I)
Occlusion Culling

- SGI OpenGL Performer (supplementary) and in graphics hardware (GeForce 3, Radeon 8500)
Application Software (I)  
Occlusion Culling

- SGI OpenGL Performer (supplementary) and in graphics hardware (GeForce 3, Radeon 8500)
Application Software (I)
Contribution Culling
Application Software (I)
Contribution Culling
Combinations

APP → ISECT → Scene → DBASE

Pipeline 0: LPOINT → CULL → DRAW
Pipeline 1: LPOINT → CULL → DRAW

Frame Buffer
Hard Disk
Measurement Methods (III)

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allocate memory for statistical structures and variables</td>
<td></td>
</tr>
<tr>
<td>check and enable statistical class and mode</td>
<td></td>
</tr>
<tr>
<td>set statistical attributes for collection</td>
<td></td>
</tr>
<tr>
<td>get system time (overall start time)</td>
<td></td>
</tr>
<tr>
<td>start rendering a frame</td>
<td></td>
</tr>
<tr>
<td><strong>execute APP, CULL stage</strong></td>
<td></td>
</tr>
<tr>
<td>start timer for DRAW stage</td>
<td></td>
</tr>
<tr>
<td><strong>execute DRAW stage</strong></td>
<td></td>
</tr>
<tr>
<td>stop timer for DRAW stage</td>
<td></td>
</tr>
<tr>
<td>write elapsed time for DRAW to statistical structure</td>
<td></td>
</tr>
<tr>
<td>finish rendering current frame</td>
<td></td>
</tr>
<tr>
<td>calculate internal statistics (average, accumulation, etc.)</td>
<td></td>
</tr>
<tr>
<td>query and write statistics to file</td>
<td></td>
</tr>
<tr>
<td>while (!exit)</td>
<td></td>
</tr>
<tr>
<td>get system time (end time of overall benchmark duration)</td>
<td></td>
</tr>
<tr>
<td>calculate overall benchmark time</td>
<td></td>
</tr>
<tr>
<td>write overall benchmark time to file</td>
<td></td>
</tr>
</tbody>
</table>

While executing the processes, the system times are captured at critical points to calculate internal statistics, such as average and accumulation, and overall benchmark times for accurate performance analysis.
# Benchmarked Systems

<table>
<thead>
<tr>
<th>Hardware</th>
<th>snorre</th>
<th>ii06</th>
<th>up_bimoto</th>
<th>dp_bimoto</th>
<th>dpht_bimoto</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>Pentium 3</td>
<td>Pentium 4</td>
<td>Pentium 4 Xeon</td>
<td>2 x Pentium 4 Xeon</td>
<td>2 x Pentium 4 Xeon (HT)</td>
</tr>
<tr>
<td>Frequency [MHz]</td>
<td>850</td>
<td>1800</td>
<td>2400</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>Main Memory [MB]</td>
<td>512</td>
<td>1024</td>
<td>512</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>Graphics Board</td>
<td>GeForce 2 Go</td>
<td>GeForce 4 Ti4200</td>
<td>GeForce FX 5900</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>Operating System</td>
<td>SuSE 9.0</td>
<td>RedHat 9</td>
<td>SuSE 8.2</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>
Software

- SuSE Linux 8.2 (Kernel 2.4.25 + Affinity Patch)
  - NVIDIA 44.96
  - GNU g++ Compiler 3.3.1
  - SGI OpenGL Performer 3.1

- Microsoft Windows 2000 (Service Pack 4)
  - NVIDIA ForceWare 53.03
  - Microsoft Visual C++ 6.0
  - SGI OpenGL Performer 3.1
## Scene Graph Data

<table>
<thead>
<tr>
<th>Scene Graph</th>
<th>town</th>
<th>Drive</th>
<th>Flight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Triangles</td>
<td>44315</td>
<td>175421</td>
<td>199610</td>
</tr>
<tr>
<td>Points</td>
<td>569</td>
<td>0</td>
<td>7745</td>
</tr>
<tr>
<td>Lines</td>
<td>0</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>GeoStates</td>
<td>79</td>
<td>302</td>
<td>295</td>
</tr>
<tr>
<td>Materials</td>
<td>5</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>Texture Envs</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Textures</td>
<td>61</td>
<td>219</td>
<td>229</td>
</tr>
<tr>
<td>LPStates</td>
<td>1</td>
<td>0</td>
<td>25</td>
</tr>
<tr>
<td>Total texture usage [MB]</td>
<td>2.77</td>
<td>49.24</td>
<td>31.53</td>
</tr>
<tr>
<td>Main memory usage [MB]</td>
<td>260.58</td>
<td>317.68</td>
<td>259.49</td>
</tr>
<tr>
<td>Main memory usage after benchmark run [MB]</td>
<td>320.80</td>
<td>357.60</td>
<td>319.90</td>
</tr>
</tbody>
</table>
# LOD Scaling (Settings)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;pbench&quot; version</td>
<td>008.61</td>
</tr>
<tr>
<td>System</td>
<td>up_bimoto</td>
</tr>
<tr>
<td>Vertical display frequency [Hz]</td>
<td>77.21</td>
</tr>
<tr>
<td>Statistics frequency [Hz]</td>
<td>77</td>
</tr>
<tr>
<td>Graphics card driver (Linux)</td>
<td>NVIDIA 44.96</td>
</tr>
<tr>
<td>Resolution [pixel]</td>
<td>1024x768</td>
</tr>
<tr>
<td>Color depth [bpp]</td>
<td>24</td>
</tr>
<tr>
<td>System variables</td>
<td>__GL_FSAA_MODE=0, __GL_DEFAULT_LOG_ANISO=0, __GL_SYNC_TO_VBLANK=1</td>
</tr>
<tr>
<td>XFree86Config</td>
<td>Option &quot;NvAqp&quot; &quot;3&quot;</td>
</tr>
<tr>
<td>Scheduler</td>
<td>SCHED_OTHER</td>
</tr>
<tr>
<td>Performer multi-processing mode</td>
<td>PFMP_APPCULLDRAW</td>
</tr>
<tr>
<td>Performer frame synchronization</td>
<td>PFPHEASE_FREE_RUN</td>
</tr>
<tr>
<td>View mode</td>
<td>&quot;Human Mode&quot;</td>
</tr>
<tr>
<td>Number of passes</td>
<td>10</td>
</tr>
<tr>
<td>Scene graphs</td>
<td>town, Drive, Flight</td>
</tr>
</tbody>
</table>
# Multi-Processing With Hard Affinity (Settings)

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;pbench&quot; version</td>
<td>008.62</td>
</tr>
<tr>
<td>System</td>
<td>up_bimoto, dp_bimoto, dp_bimoto + HA</td>
</tr>
<tr>
<td>Vertical display frequency [Hz]</td>
<td>60.352268</td>
</tr>
<tr>
<td>Statistics frequency [Hz]</td>
<td>60</td>
</tr>
<tr>
<td>Graphics card driver (Linux)</td>
<td>NVIDIA 44.96</td>
</tr>
<tr>
<td>Resolution [pixel]</td>
<td>1024x768</td>
</tr>
<tr>
<td>Color depth [bpp]</td>
<td>24</td>
</tr>
<tr>
<td>System variables</td>
<td>__GL_FSAA_MODE=0, __GL_DEFAULT_LOG_ANISO=0, __GL_SYNC_TO_VBLANK=1</td>
</tr>
<tr>
<td>XFree86Config</td>
<td>Option &quot;NvAgp&quot; &quot;3&quot;</td>
</tr>
<tr>
<td>Scheduler</td>
<td>SCHED_OTHER</td>
</tr>
<tr>
<td>Performer multi-processing mode (up_bimoto)</td>
<td>PFMP_APPCULLDRAW</td>
</tr>
<tr>
<td>Performer multi-processing mode (dp_bimoto)</td>
<td>PFMP_APP_CULLDRAW</td>
</tr>
<tr>
<td>Performer multi-processing mode (dp_bimoto + HA)</td>
<td>PFMP_APP_CULLDRAW (CPU0: OS, APP; CPU1: CULL+DRAW)</td>
</tr>
<tr>
<td>Performer frame synchronization</td>
<td>PFPHASE_FREE_RUN</td>
</tr>
<tr>
<td>View mode</td>
<td>&quot;Human Mode&quot;</td>
</tr>
<tr>
<td>Number of passes</td>
<td>10</td>
</tr>
</tbody>
</table>
Software

- SuSE Linux 8.2 (Kernel 2.4.25 + Affinity Patch)
  - NVIDIA 44.96
  - GNU g++ Compiler 3.3.1
  - SGI OpenGL Performer 3.1

- Microsoft Windows 2000 (Service Pack 4)
  - NVIDIA ForceWare 53.03
  - Microsoft Visual C++ 6.0
  - SGI OpenGL Performer 3.1
Benchmarks Conducted In The Thesis (I)

- Color Depth
- Resolution
- Culling
- LOD Scaling
- Display Lists
- Intersection Lines
- Light Sources
- Querying Statistical PROC Variables Separately
- Querying Statistical Variables
- Drawing Statistics
Benchmarks Conducted In The Thesis (II)

- IO-APIC
- Context Switch Rate
- Multi-Processing Modes
- Hard Affinity
- Multi-Processing with Hard Affinity
- Scheduling (Linux, W2K)
- System Stressing with Active Waiting
- System Stressing with Calculations (Linux, W2K)
- Scheduling with Calculations (Linux, W2K)
- Compiler Optimizations (Linux, W2K)
# Querying Statistical Variables

<table>
<thead>
<tr>
<th>Statistical Variable</th>
<th>Average Query Time Per Variable [µs]</th>
<th>Average Benchmark Runtime Performance Decrease [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>town</td>
<td>Drive</td>
</tr>
<tr>
<td>no stat</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>APP</td>
<td>9.22</td>
<td>4.21</td>
</tr>
<tr>
<td>CULL</td>
<td>9.59</td>
<td>7.77</td>
</tr>
<tr>
<td>DRAW</td>
<td>8.44</td>
<td>8.74</td>
</tr>
<tr>
<td>ISECT</td>
<td>8.28</td>
<td>8.52</td>
</tr>
<tr>
<td>DBASE</td>
<td>13.35</td>
<td>4.22</td>
</tr>
<tr>
<td>LPOINT</td>
<td>9.71</td>
<td>4.03</td>
</tr>
<tr>
<td>PROC (sep)</td>
<td>37.00</td>
<td>56.66</td>
</tr>
<tr>
<td>PROC (all)</td>
<td>15.47</td>
<td>23.32</td>
</tr>
</tbody>
</table>
Querying Statistical Variables (Results)

- Querying statistical variables takes little time
- Retrieval of statistical classes is faster than fetching variables separately
- Query time varies between scene graphs (scene graph complexity, multi-processing mode)
Scheduling (dp_bimoto, "town")

<table>
<thead>
<tr>
<th>Scheduling</th>
<th>TOTAL</th>
<th>APP</th>
<th>CULL</th>
<th>DRAW</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCHED_OTHER (Linux)</td>
<td>1.76</td>
<td>0.24</td>
<td>0.07</td>
<td>1.36</td>
</tr>
<tr>
<td>SCHED_FIFO (Linux)</td>
<td>1.74</td>
<td>0.23</td>
<td>0.06</td>
<td>1.36</td>
</tr>
<tr>
<td>SCHED_RR (Linux)</td>
<td>1.77</td>
<td>0.24</td>
<td>0.07</td>
<td>1.37</td>
</tr>
<tr>
<td>normal (W2K)</td>
<td>3.24</td>
<td>0.25</td>
<td>0.05</td>
<td>2.83</td>
</tr>
<tr>
<td>high (W2K)</td>
<td>3.24</td>
<td>0.25</td>
<td>0.05</td>
<td>2.83</td>
</tr>
<tr>
<td>real-time (W2K)</td>
<td>3.24</td>
<td>0.25</td>
<td>0.05</td>
<td>2.83</td>
</tr>
</tbody>
</table>
Scheduling (dp_bimoto)

![Graph showing overall frame misses by scene graph and scheduling policies]

- **Scene Graph**: Town, Drive, Flight
- **Scheduling Policies**:
  - SCHED_OTHER (Linux)
  - SCHED_FIFO (Linux)
  - SCHED_RR (Linux)
  - normal (W2K)
  - high (W2K)
  - real-time (W2K)
Scheduling (dp_bimoto, "town", APP)
Scheduling (dp_bimoto, "town", CULL)
Scheduling (dp_bimoto, "town", DRAW)
Scheduling (Results)

- Scheduling strategies in Linux 2.4.x offer minor tuning possibilities

- Task switching in Windows 2000 happens more uniformly than in Linux 2.4.x

- The DRAW stage does not perform well in Windows 2000
LOD Scaling

Process Time (TOTAL)

Benchmark Runtime

Average Process Time (TOTAL) Performance Increase [%]

Average Benchmark Runtime Performance Increase [%]

Scene Graph

on

town  Drive  Flight

21.23  311.22  778.87

town  Drive  Flight

42.77  289.97  643.62

71
LOD Scaling
("town", DRAW)

Steps/PFFSTATSVAL PFTIMES PROC DRAW Plot of
'town x100 y100 a10 isect PROC P2 f120 L81 i5.txt'

Legend
PFFSTATSVAL PFTIMES PROC DRAW (LOD Scaling Off)
PFFSTATSVAL PFTIMES PROC DRAW (LOD Scaling On)

Steps [ ]
0 500 1000 1500 2000

PFFSTATSVAL PFTIMES PROC DRAW [ms]
0 10 20 30 40 50 60 70 80

77 Hz
110 Hz
View Frustum Culling

Benchmark Runtime

Process Time (TOTAL)

Scene Graph

Average Benchmark Runtime Performance Decrease [%]

Average Process Time (TOTAL) Performance Decrease [%]

town    Drive    Flight

off
View Frustum Culling ("town", DRAW)
View Frustum Culling
("Drive", DRAW)
View Frustum Culling (Results)

- Culling dramatically improves rendering speed (2 to 7 times)

- Unnecessary geometry is removed as soon as possible (CULL stage) without being further processed

- Current graphics cards implement occlusion culling as z-culling (starting with GeForce 3, Radeon 8500)
Color Depth

Benchmark Runtime

Process Time (TOTAL)

Scene Graph

Scene Graph

16 to 24 bpp
Color Depth
("Flight", DRAW)
Color Depth (Results)

- Color depth of 16 bpp is usually sufficient
- 4% - 12% less performance with 24 bpp due to additional consumption of memory bandwidth and read/write accesses of z buffer and frame buffer memory