Tracking Graphics State for Network Rendering

Ian Buck    Greg Humphreys    Pat Hanrahan

Computer Science Department
Stanford University
Distributed Graphics

How to manage distributed graphics applications, renderers, and displays?
Virtual Graphics

- Virtualize the graphics output
  - Serial input to parallel graphics
  - Application assumes single large resource
Virtual Graphics

- Virtualize the graphics destination.
  - Driver manages shared resource.
  - Application assumes owns graphics.
Virtual Graphics

- Tiled Rendering
  - Single application rendering to many outputs

- Parallel Rendering
  - Many applications rendering to a single output

- Previous Work
  - Window Systems
    - X11
    - SunRay
  - Visualization Servers
    - GLR
    - GLX
Tiled Rendering

- Minimize network traffic
  - Sort first geometry commands
  - Broadcast state commands?
**Lazy State Update**

- Issue minimal state commands to sync render
Parallel Rendering

- Hardware context switching too slow
  - 0.17 ms / switch NVIDIA GeForce
  - 32 streams, 60 fps = 30% Frame Time
• Software context switch
  ▪ Generate state commands for switch
  ▪ Single hardware context
Cluster Rendering

Application → Network → Renderer → Application

Application → Network → Renderer → Application

Application → Network → Renderer → Application
Overview

- Data structure for generating context comparisons.
- Tiled Rendering
  - Lazy State Updates
- Parallel Rendering
  - Soft Context Switching
- WireGL
  - OpenGL driver for cluster rendering.
Context Data Structure

- Challenge: Generate state commands of context differences.
- Direct comparison too slow.
- Acceleration data structure:
  - Track difference information during execution
  - Quick search for comparison
Context Data Structure

- Hierarchical dirty bits
  - Indicate which elements need comparison

```
Texture -> Lighting -> Fragment
  |                     |
  v                     v
ShadeModel            Light0
  |                        |
  v                        v
Position                Enable
```
Hierarchical dirty bits

Context A: `glEnable(GL_LIGHT0)`

Context B:

- Texture
- Lighting
- Fragment
  - ShadeModel
  - Light0
    - Position
    - Enable
Context Data Structure

- Context Diff

Context A → Context B

- Texture
- Lighting
  - ShadeModel
  - Light0
    - Position
    - Enable
  - Fragment

glDisable(GL_LIGHT0)
Context Data Structure

- State command invalidates all other contexts
- Wide dirty bit vector
- Single write invalidates all contexts
Tiled Rendering

- Geometry Bucketing
  - Track object space bounding box
  - Transform object box to screen space
  - Send geometry commands to outputs which overlap screen space extent
• Lazy State Update
  • Defer sending
  • Custom state commands for each render
Lazy State Update

Load transform state $M_1$
Render Geometry
Lazy State Update

Load transform state $M_1$
Render Geometry
Load transform state $M_2$
Render Geometry

Virtual Context
Matrix: $M_2$

Render Contexts
Matrix: $M_1$
Matrix: $M_2$

```
glLoadMatrix(M2)
```
Tiled Rendering Results

- Volume Rendering
  - 1.5 Mtri Surface
  - 1024x768 Outputs
  - 8x4 = 25 Mpixel display
Tiled Rendering Results

- **Quake III**
  - OpenGL State Intensive
  - Fine Granularity
  - 8x4 dominated by overlap
Parallel Rendering

- Requires fast context switching between streams
Soft Context Switching

- Generate State Commands
  - Context compare operation to generate state commands

- Benefits
  - Prevent hardware pipeline flushes
  - Switch time dependent on context differences

Matrix: $M_1$

`glLoadMatrix(M_2)`

Matrix: $M_2$
Soft Context Switching

- Results:
  - Varying current color and transformation state.
  - Context switches per second:

<table>
<thead>
<tr>
<th>System</th>
<th>Context Switches per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGI Infinite Reality</td>
<td>697</td>
</tr>
<tr>
<td>SGI Cobalt</td>
<td>2,101</td>
</tr>
<tr>
<td>NVIDIA GeForce</td>
<td>5,968</td>
</tr>
<tr>
<td>WireGL</td>
<td>191,699</td>
</tr>
</tbody>
</table>
Conclusions

- State tracking hierarchical dirty bit
  - Allows for fast context comparison operations

- Enables Virtual Graphics
  - Tiled Rendering
  - Parallel Rendering

- WireGL
  - http://graphics.stanford.edu/software/wiregl
Acknowledgements

- Matthew Eldridge
- Chris Niederauer
- Department of Energy contract B504665
Tracking Graphics State for Network Rendering

Ian Buck    Greg Humphreys    Pat Hanrahan

Computer Science Department
Stanford University