Multi-Graphics

*Scalable Graphics using Commodity Graphics Systems*

VIEWS PI Meeting
May 17, 2000

Pat Hanrahan
Stanford University
http://www.graphics.stanford.edu/

---

**Multi-Graphics Project**

*Scalable and Distributed Graphics and Visualization using Commodity Graphics Components*

- Research Team
  - Greg Humphreys, 3rd yr grad, scalable cluster rendering
  - Ian Buck, 1st yr grad, remote rendering
  - Brad Johanson, 4th yr grad, interactive room
  - Matthew Everett, Sr ugrad, fast geometry compression
  - Susan Shepard, staff, Interactive Room
  - Maureen Stone, consultant, Interactive Mural

- Stanford-TriLabs Visualization Cluster
- Research funded by DVC-ARP Program
- Industrial collaboration: Intel, NVIDIA, (SGI, SUN, SONY, 3DFX)
Overview

- Visualization cluster
- Lightning-2 image composition network
- Interactive room (mural and table)
- Parallel WireGL

SGI RealityMonster

"Mural Server"
(8 processor SGI Origin)

Master Thread

Pipe Thread

Pipe Thread

InfiniteReality Graphics

InfiniteReality Graphics

Projector

Projector

Projector

Projector

Projector

Network

Client Thread

Client Thread

Client App

Client App

OpenGL Protocol

G. Humphreys, P. Hanrahan, A distributed graphics system for large tiled displays, Vis99

Interactive Workspaces Overview
Stanford University Computer Science Dept.
G. Humphreys, I. Buck, P. Hanrahan, Distributed rendering for large tiled displays, Submitted to Supercomputing 2000

IR vs. GeForce256 Benchmarks

Millions of triangles per second*

<table>
<thead>
<tr>
<th></th>
<th>InfiniteReality</th>
<th>GeForce DDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unlit</td>
<td>4.858</td>
<td>6.908</td>
</tr>
<tr>
<td>Lit</td>
<td>5.163</td>
<td>7.270</td>
</tr>
<tr>
<td>Unlit textured</td>
<td>5.255</td>
<td>7.270</td>
</tr>
<tr>
<td>Lit textured</td>
<td>3.651</td>
<td>5.715</td>
</tr>
</tbody>
</table>

*Performance measured by Stanford OpenGL benchmarking tool
### IR Vs. GeForce256 Benchmarks

**Millions of pixels per second**

<table>
<thead>
<tr>
<th></th>
<th>InfiniteReality</th>
<th>GeForce DDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pixels</td>
<td>425.638</td>
<td>320.418</td>
</tr>
<tr>
<td>Bilerp textured</td>
<td>297.753</td>
<td>245.612</td>
</tr>
<tr>
<td>Trilerp textured</td>
<td>247.945</td>
<td>154.127</td>
</tr>
<tr>
<td>Trilerp textured blended</td>
<td>247.977</td>
<td>106.165</td>
</tr>
</tbody>
</table>

*Performance measured by Stanford OpenGL benchmarking tool*

### IR Vs. GeForce256 Benchmarks

**Cost**

<table>
<thead>
<tr>
<th></th>
<th>InfiniteReality</th>
<th>GeForce DDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Price</td>
<td>$106,000</td>
<td>$3000($250)</td>
</tr>
</tbody>
</table>

Next gen. PC graphics systems will be much faster

- X-Box 300 million triangles per second, 1 tera-op/s
Chromium – **Cr** (Cluster Renderer)

**Chromium**

- **Cluster**: 32 graphics nodes, 4 server nodes
- **Computer**: Compaq SP750
  - 2 processors (800 Mhz, 133Mhz FSB)
  - i840 core logic (biggest issue for vis-clusters)
  - 256 MB memory
  - 18GB disk (+ 3*36GB on servers)
  - 64-bit, 66 Mhz PCI
  - AGP-4X Pro
- **Graphics**
  - NVIDIA GeForce2 GTS (NVIDIA GeForce256 DDR)
  - NVIDIA NV20 with DVI this summer
- **Network**
  - Myrinet 64-bit, 66 Mhz

Interactive Workspaces Overview
Stanford University Computer Science Dept.
Interactive Workspaces Overview
Stanford University Computer Science Dept.

SP750: i840 implementation

Linux Graphics

- Linux has high quality fast network drivers
  - Servernet II measured at 162 MB/sec
  - Myrinet measured at 142 MB/sec
- Windows fast network drivers disappointing
  - Servernet II only in Windows 2000
  - Myrinet crashes Windows NT in 64/66 slot
- Linux graphics drivers are *rapidly improving* (thanks to DOE!)
  - “98% shared code”
  - Just-released drivers very usable
  - Linux AGP support is the current bottleneck
  - Direct rendering architecture still contentious
  - Ultimately more efficient?
- Our software runs identically on NT, Linux, Irix
Lightning-2 Module

DVI = Digital Visual Interface

DVI inputs

DVI inputs Daisy-chained

Input unit

Framebuffers

Pixel chain

Compositing unit

Scalable Lightning

Steve Hunt, Dan Patterson, Art Webb (Intel)

Gordon Stoll, Matthew Eldridge (Stanford)
Scanline Switching

- Placement information is embedded in the image
- Unit of mapping is a one-pixel-high strip

<table>
<thead>
<tr>
<th>A/B Bit</th>
<th>Display # plus X.Y Coordinates 23 bits</th>
<th>Parity 1 bit</th>
<th>Width 11 bits</th>
<th>Pixel Chain # 4 bits</th>
<th>Opcode 8 bits</th>
</tr>
</thead>
</table>

Strip Header: 2 pixels (48 bits)

Image Assembly and Output

Interactive Workspaces Overview
Stanford University Computer Science Dept.
Image Composition

- Layers or windows
- Chroma-keying
  - Treat local pixels of a specific color as transparent
- Depth compositing
  - Copy depth values into the PC frame buffer
  - Scan depth on a second pixel chain
- Advanced operation
  - Program additional functions in the compositing FPGA
  - Take advantage of the 8-bit opcode
  - Combine data from multiple streams

Interactive Mural

Image and virtual colonoscopy concept courtesy Sandy Napel, Stanford Radiology Department.

Projectors: 1024x768, 900 ANSI Lumens
Mural: 6’ x 2’, 4096 x 1536, ~60dpi,
Interactive Workspaces Overview
Stanford University Computer Science Dept.
Interactive Workspaces

Advanced display technology meets ubiquitous computing

Planning and Evaluating

Disney Theme Park Model

Interactive Workspaces Overview
Stanford University Computer Science Dept.
Software Goals

1. Efficient “as-possible” remote rendering
2. Scalable to support high resolution tiled displays
3. Parallel graphics API for ultimate scalability
4. Immediate mode for dynamic, time-varying datasets
   (Does not “require” data replication)
5. Support heterogeneous clusters/components/os’es
   (Eventually “network-aware”)
6. Drop-in, dynamically linked OpenGL library

WireGL

- Similar in spirit to GLX OpenGL 1.1 protocol
- Novel features
  - Virtual OpenGL Protocol
  - More efficient handling of \texttt{glvertex} commands
  - Optimized packer
  - State tracking
  - Parallel API
**High Performance Remote Graphics**

GLUT Atlantis Demo
Vertices per second (Frames per second)

<table>
<thead>
<tr>
<th></th>
<th>Direct</th>
<th>Remote</th>
<th>GLX</th>
<th>WireGL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2.8M</td>
<td>100mbit</td>
<td>.108M</td>
<td>.22M</td>
</tr>
<tr>
<td></td>
<td>(1015 fps)</td>
<td>1000mbit</td>
<td>(39)</td>
<td>(78)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>.98M</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(350 fps)</td>
</tr>
</tbody>
</table>

SP750 w/ GeForce w/ Myrinet (32/32) under WindowsNT
Have measured remote rendering at >2M vertices per second
Roughly 35% efficiency – current goal: drive much higher

**State Tracking**

Technique
- Intercept OpenGL commands on client and track state
- Mirror OpenGL state for each connection on server

Enables
- Efficient remote rendering (culling, compression)
- Lazy state update (no state broadcast)
- Fast context switching
- Tiled displays (bucket-sort)
- CAVES and stereo rendering (create multiple viewing transforms)
- Parallel API (reorder streams)
- Mobile contexts and vis-servers

I. Buck, G. Humphreys, P. Hanrahan, Tracking graphics state for networked rendering, to appear Graphics Hardware 2000
Interactive Mural Graphics

- Sort (first) geometry into tiles using bounding box
  - Incrementally update bounds with each vertex
  - Transform only primitive blocks

- Separate rendering stream for each server

Lazy State Updates

- Only send state which is required for rendering

---

Interactive Workspaces Overview
Stanford University Computer Science Dept.
**Mural Performance**

Marching Cubes Application

Prototype 8 Node PC cluster with GeForce and Myrinet

Driving an 8 projector Interactive Mural

Under simulation, performance scales to 32 nodes

**Parallel OpenGL API**

*Key to scalability*

- Multiple applications issuing graphics streams
- Each stream has different graphics context
- Each stream inserts *synchronization* commands
- Graphics streams are *reordered* in the graphics system

Igehy, Stoll, Hanrahan
SIGGRAPH ’98

Interactive Workspaces Overview
Stanford University Computer Science Dept.
Soft Context Switch

- Compute context difference
- Generate OpenGL commands to switch contexts

Fast Context Switching

Graphics context switches per second*

<table>
<thead>
<tr>
<th>Graphics System</th>
<th>Identical</th>
<th>Varying</th>
</tr>
</thead>
<tbody>
<tr>
<td>InfiniteReality</td>
<td>719</td>
<td>697</td>
</tr>
<tr>
<td>Cobalt</td>
<td>2,239</td>
<td>2,101</td>
</tr>
<tr>
<td>GeForce DDR</td>
<td>11,919</td>
<td>5,968</td>
</tr>
<tr>
<td>WireGL</td>
<td>5,817,152</td>
<td>191,699</td>
</tr>
</tbody>
</table>

Identical: graphics state in old and new the same
Varying: graphics state in old and new different
Performance Goals (Summer 2000)

Graphics == Network
- Network: 6 Mtri @ 13 bytes/tri = 78 Mb/s
- Graphics: 6 Mtri/sec, 400 Mpix/sec

32 node system
- 32*5 = 160 Mtri/sec
- 32*400 = 12.8 Gpix/sec

Assumes …
- Perfect load balancing
- Lightning-2 image composition
  Should also work well with pixel readback

Status

Accomplishments
- Interactive room remodeled (Oct 99)
- Conference room -style tabletop display
- Interactive mural (3 smartboards )
- Remote rendering functional
- Tiled rendering functional

Future plans
- Release alpha version of wiregl to DOE, Brown, Princeton, …
- Install cluster by June 1st
- Parallel API by June 1st
- 8 Lightning-2 modules by mid-June
- New 12 projector mural (seamless) by end of summer
- Port DOE app (Williams volume rendering?) by end of summer
- SIGGRAPH Demo w/ Intel/NVIDIA