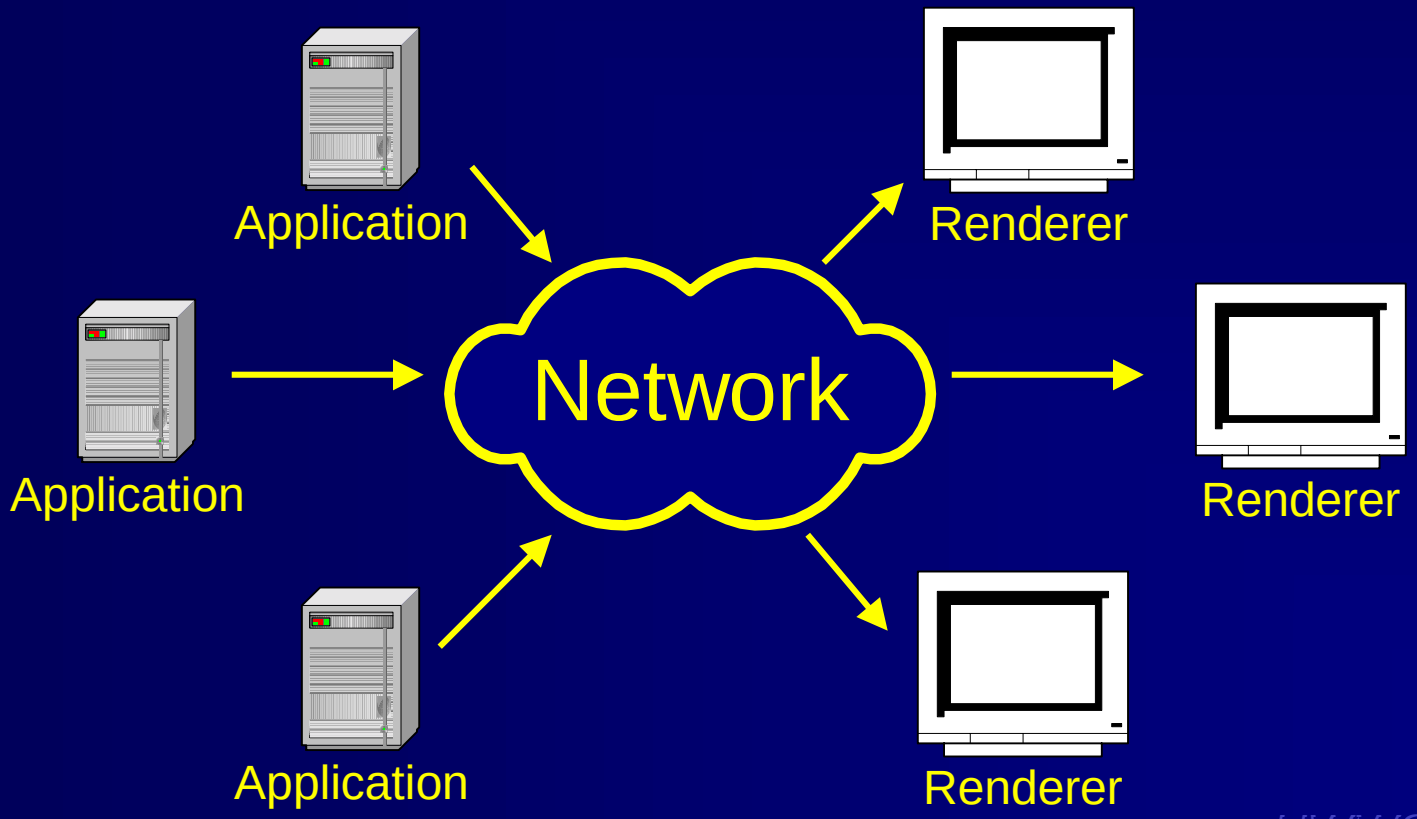
Tracking Graphics State for Network Rendering

Ian Buck Greg Humphreys Pat Hanrahan

Computer Science Department
Stanford University

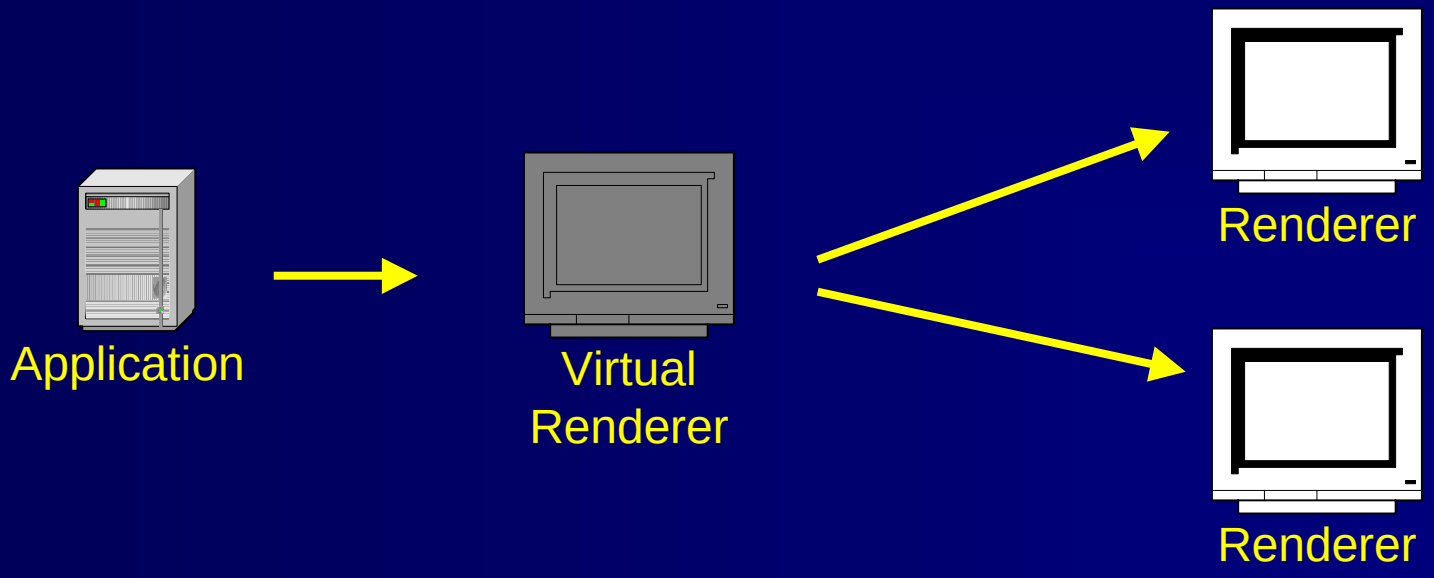
Distributed Graphics

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



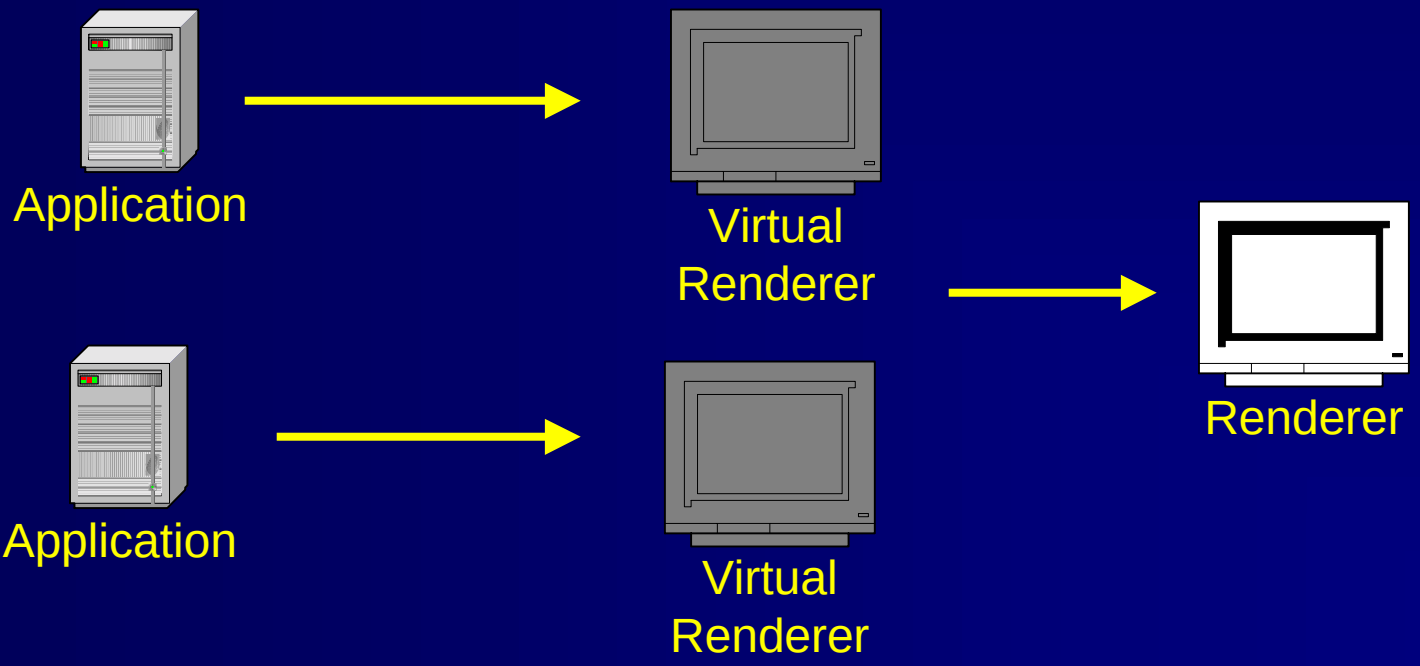
Virtual Graphics

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

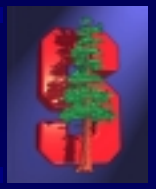


Virtual Graphics

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

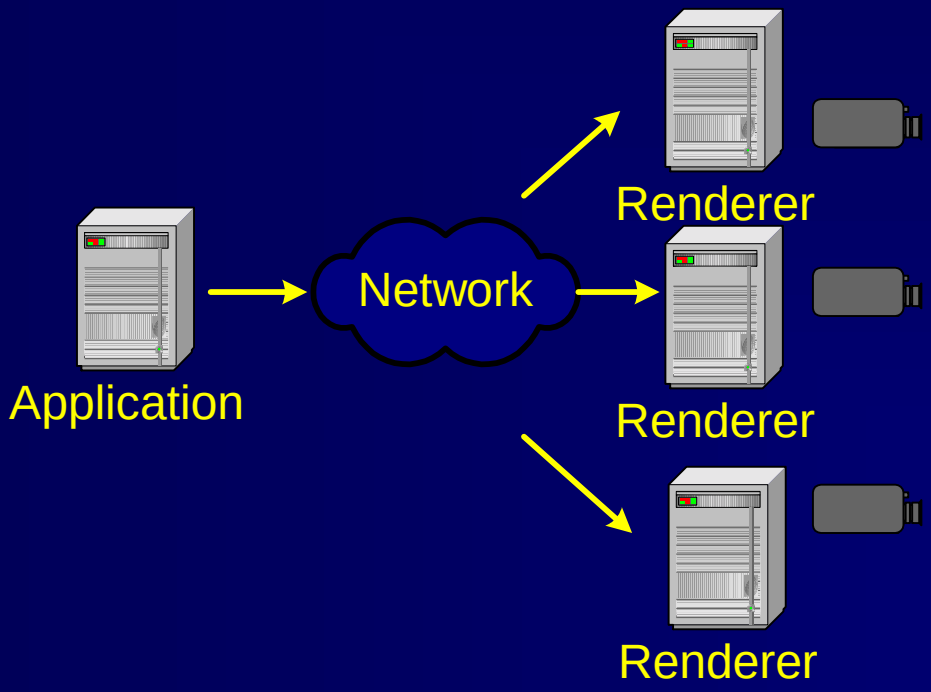


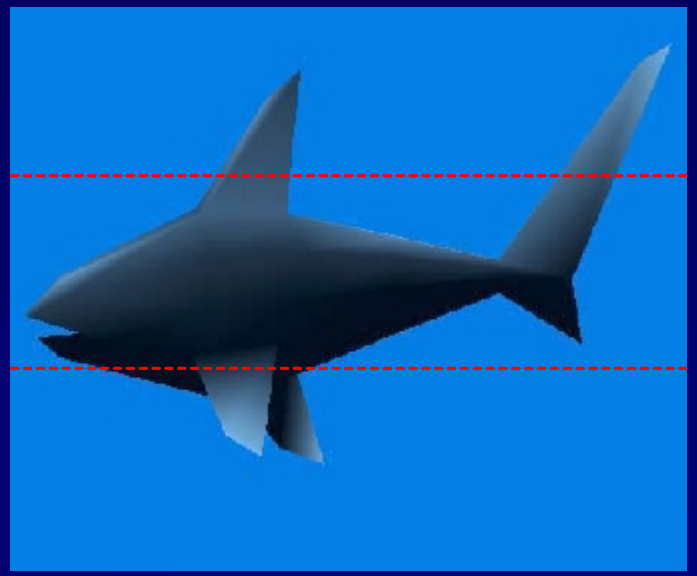
Virtual Graphics



- 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

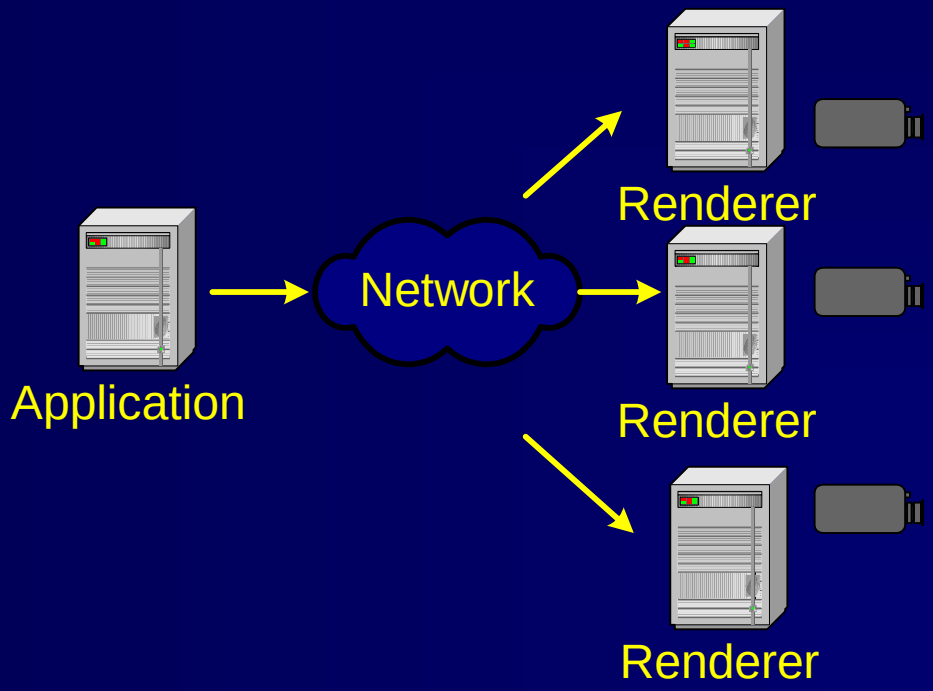
Tiled Rendering

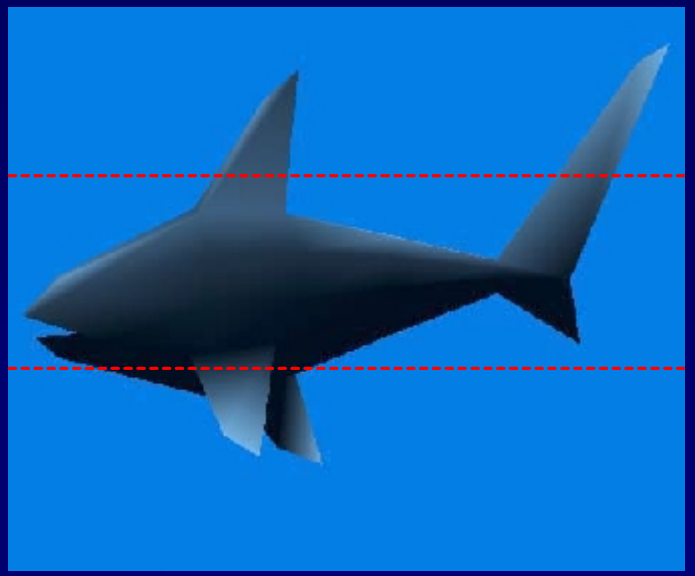




- Minimize network traffic
  - Sort first geometry commands
  - Broadcast state commands?

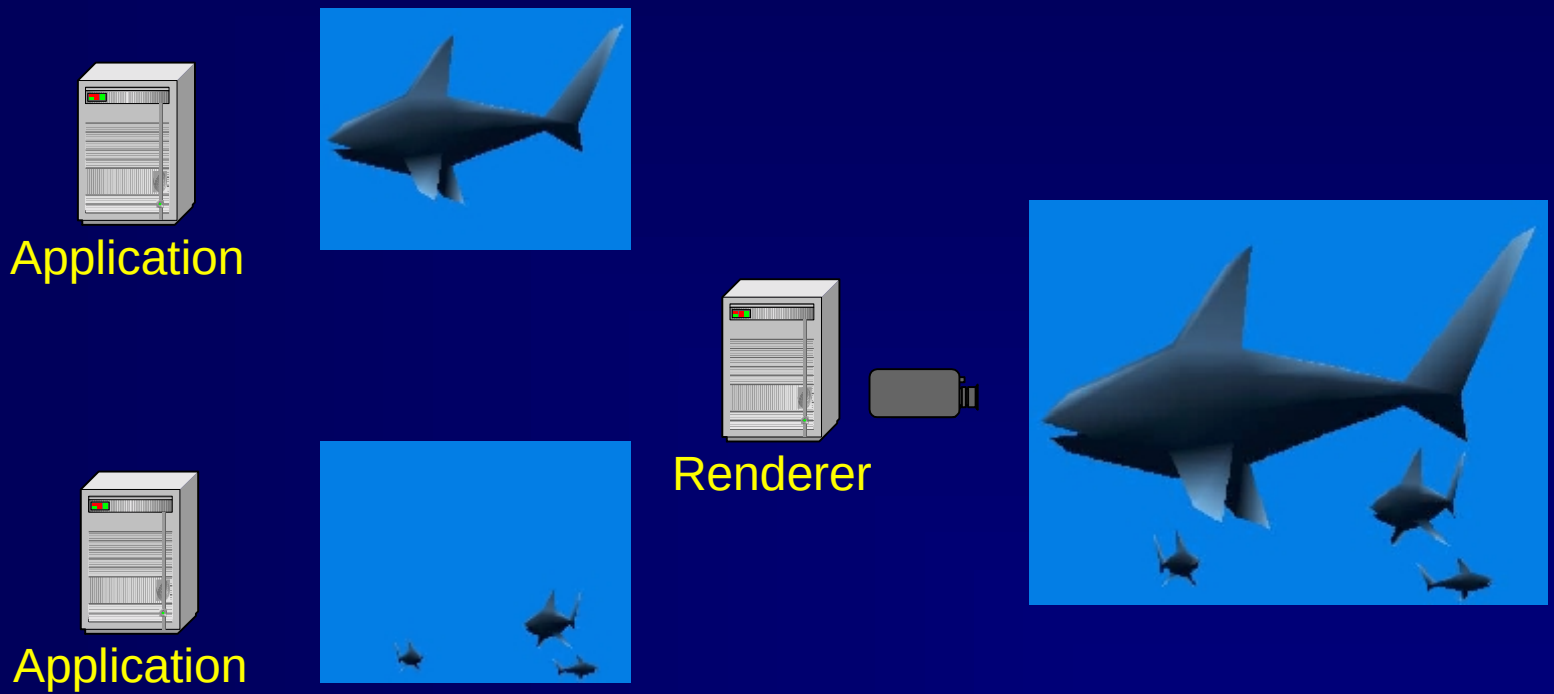
Tiled Rendering





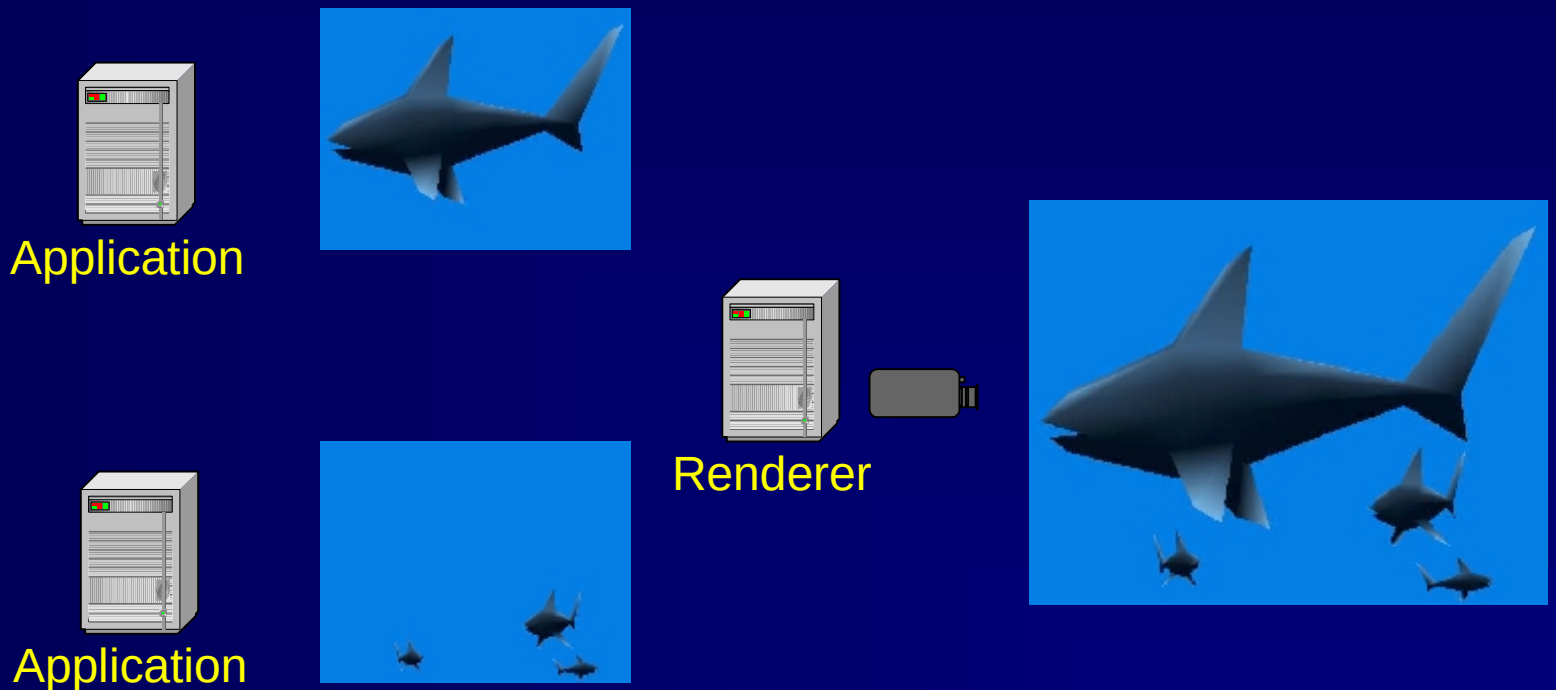
- Lazy State Update
  - Issue minimal state commands to sync render

Parallel Rendering



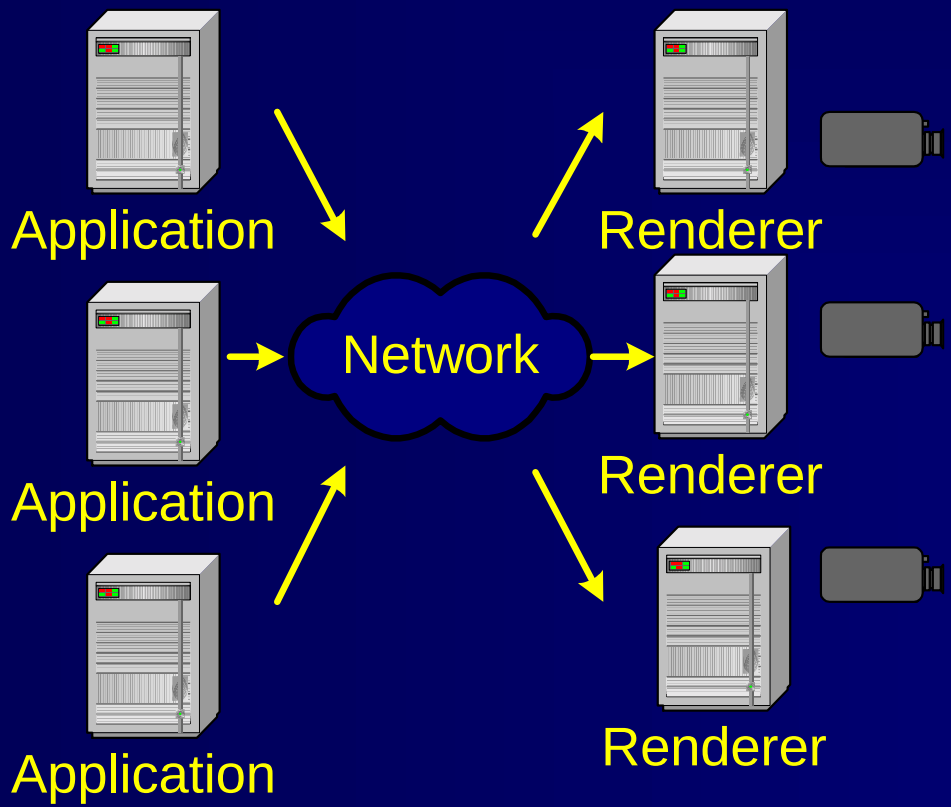
- Hardware context switching too slow
  - .17 mS / switch NVIDIA GeForce
  - 32 streams, 60 fps = 30% Frame Time

Parallel Rendering

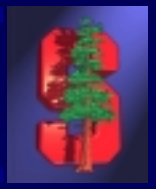


- Software context switch
  - Generate state commands for switch
  - Single hardware context

Cluster Rendering



Overview



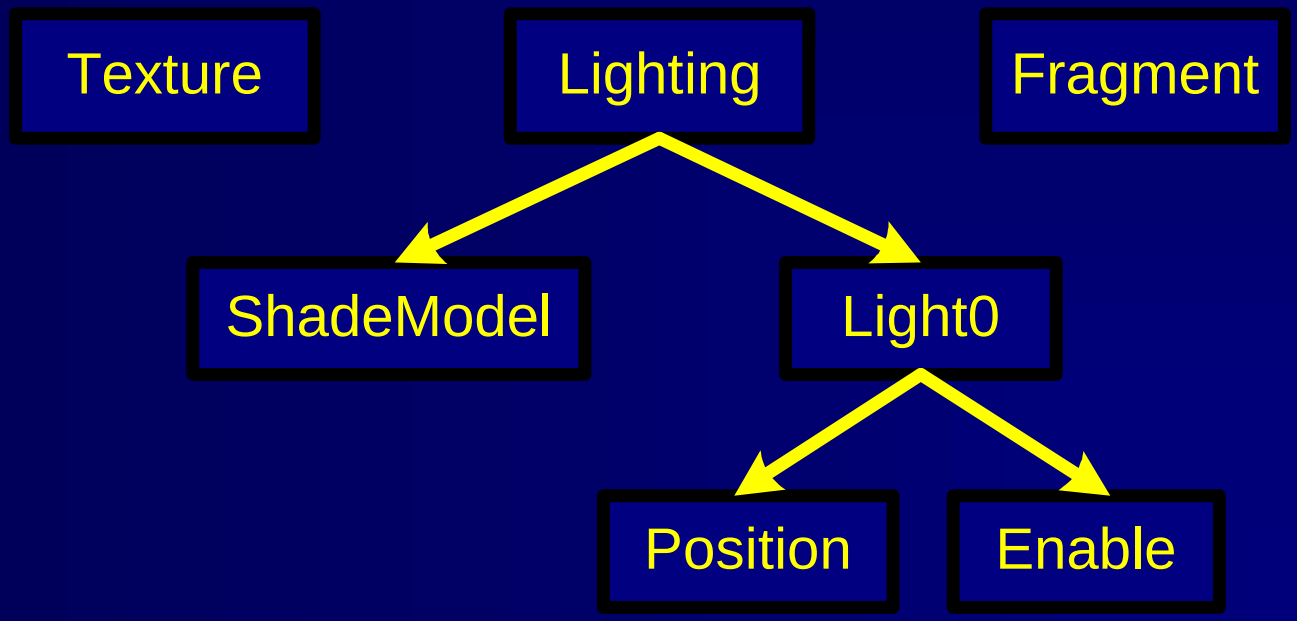
- Data structure for generating context comparisons.
- Tiled Rendering
  - Lazy State Updates
- Parallel Rendering
  - Soft Context Switching
- WireGL
  - OpenGL driver for cluster rendering.

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

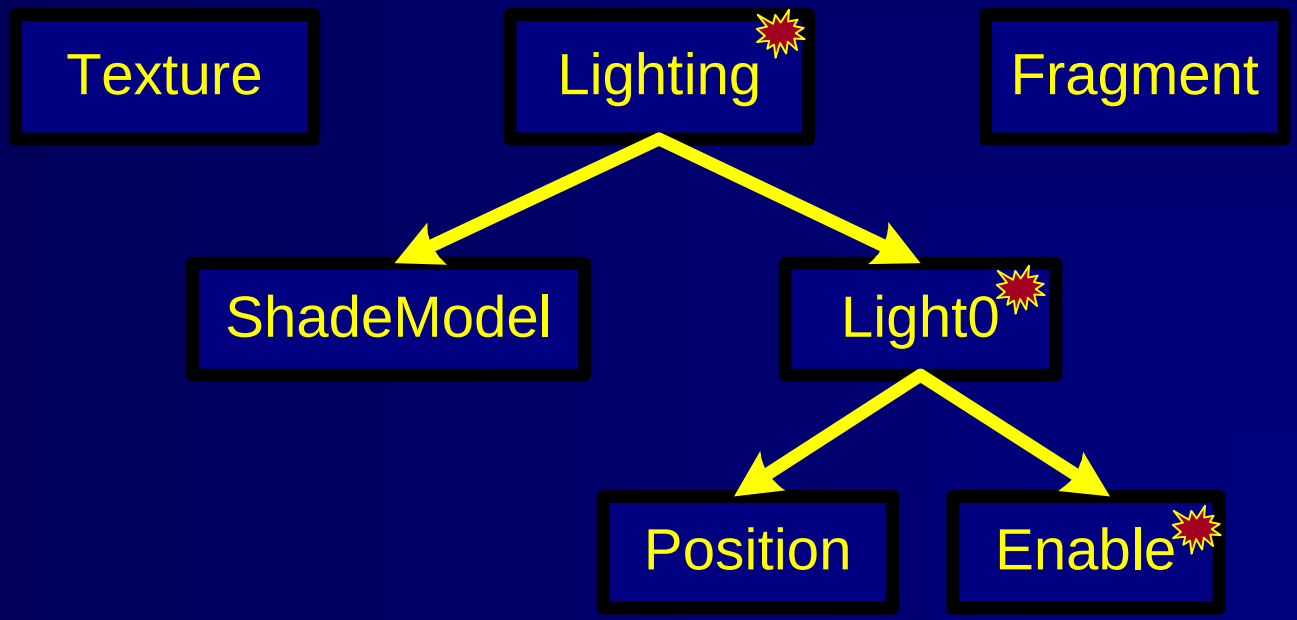


Context Data Structure

Hierarchical dirty bits

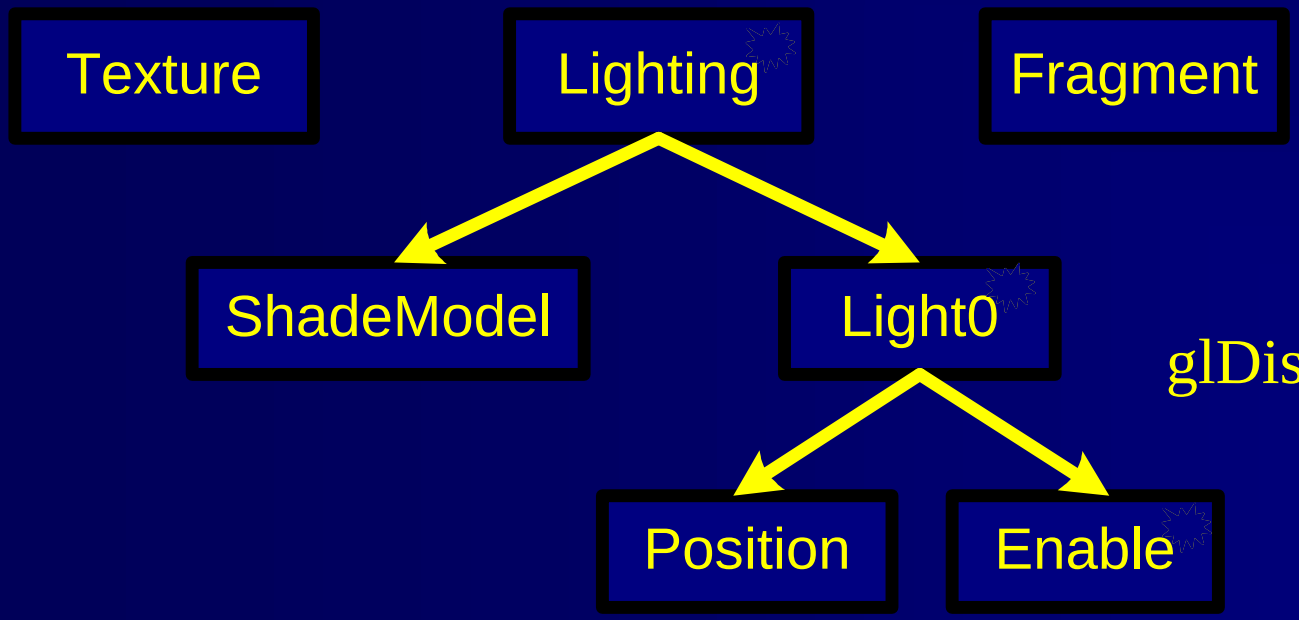
Context A: glEnable(GL_LIGHT0)

Context B:



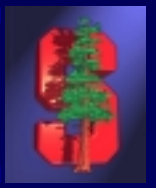
Context Data Structure

- Context Diff



glDisable(GL_LIGHT0)

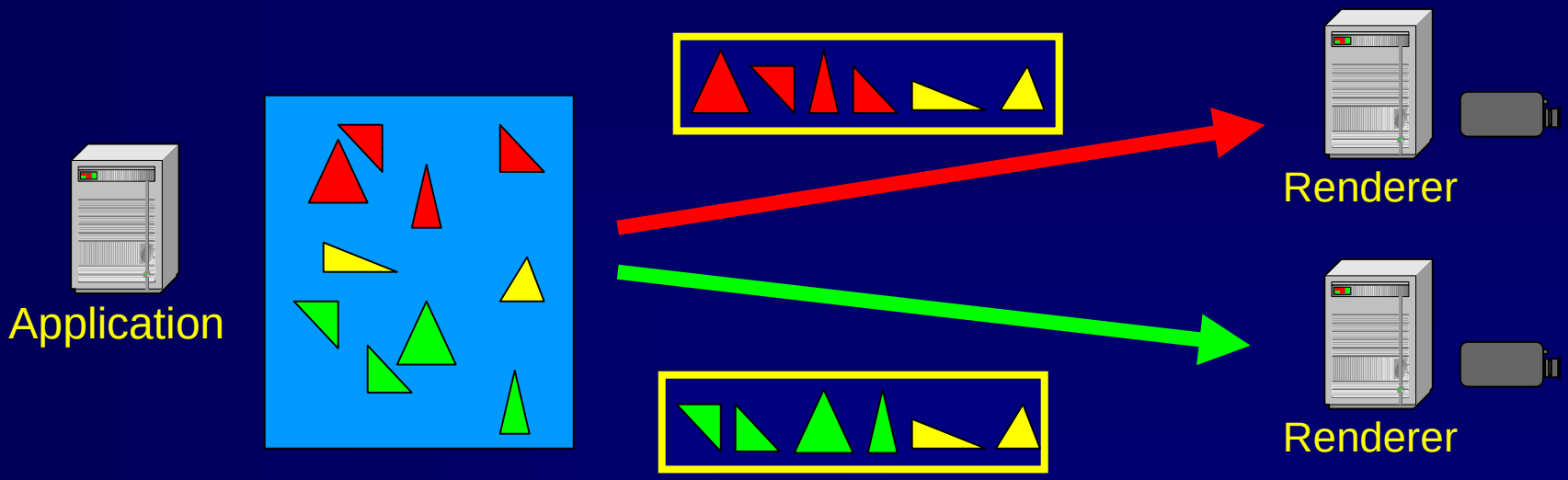
Context Data Structure



- State command invalidates all other contexts
- Wide dirty bit vector

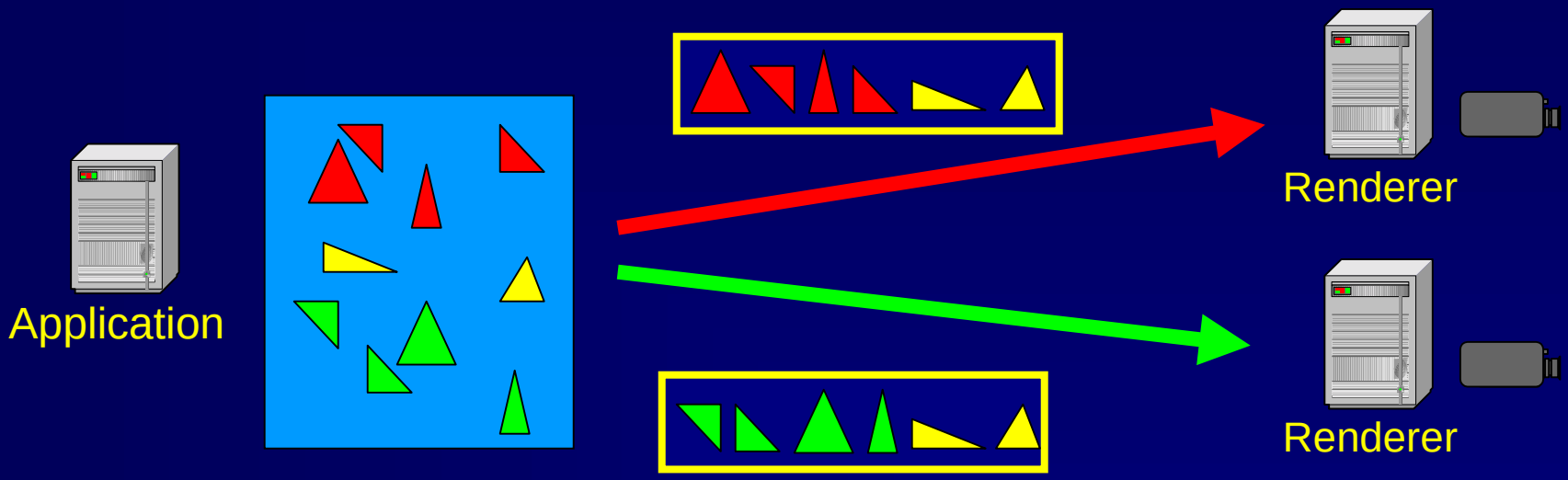
- Single write invalidates all contexts

Tiled Rendering



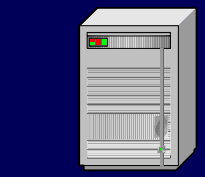
- Geometry Bucketing
  - Track object space bounding box
  - Transform object box to screen space
  - Send geometry commands to outputs which overlap screen space extent

Tiled Rendering



- Lazy State Update
  - Defer sending
  - Custom state commands for each render

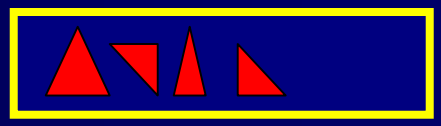
Lazy State Update



Application

Virtual Context

Matrix: M_1



Render Contexts

Matrix: M_1

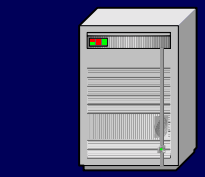
Matrix: I

glLoadMatrix(M_1)

Load transform state M_1

Render Geometry

Lazy State Update



Virtual Context

Render Contexts

Matrix: M_2

Matrix: M_1

Matrix: M_2

glLoadMatrix(M_2)

Load transform state M_1

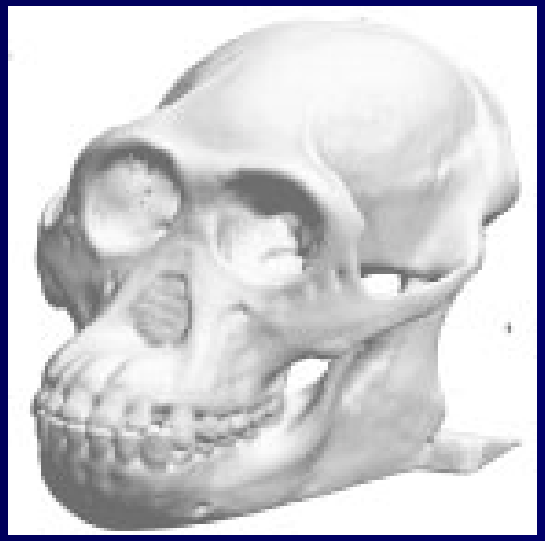
Render Geometry

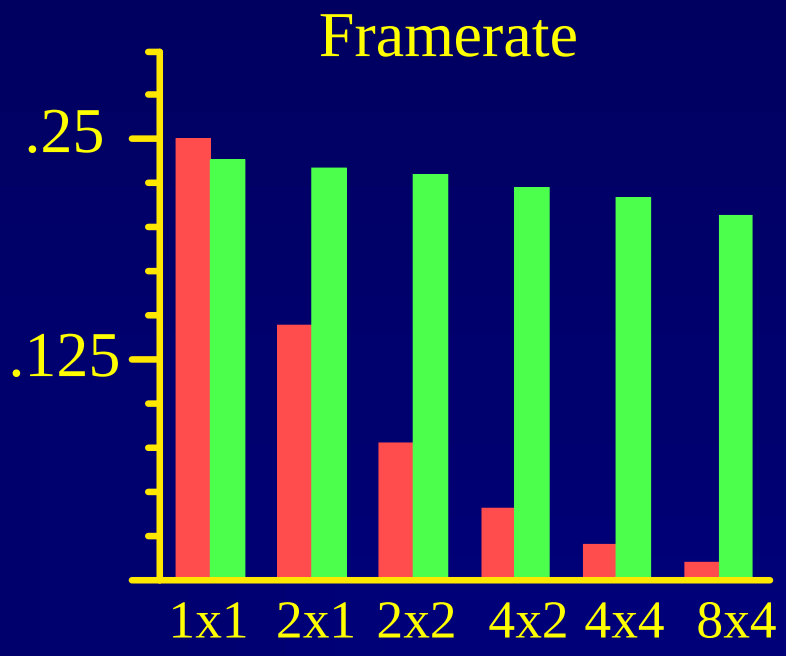
Load transform state M_2

Render Geometry

Tiled Rendering Results

Marching Cubes

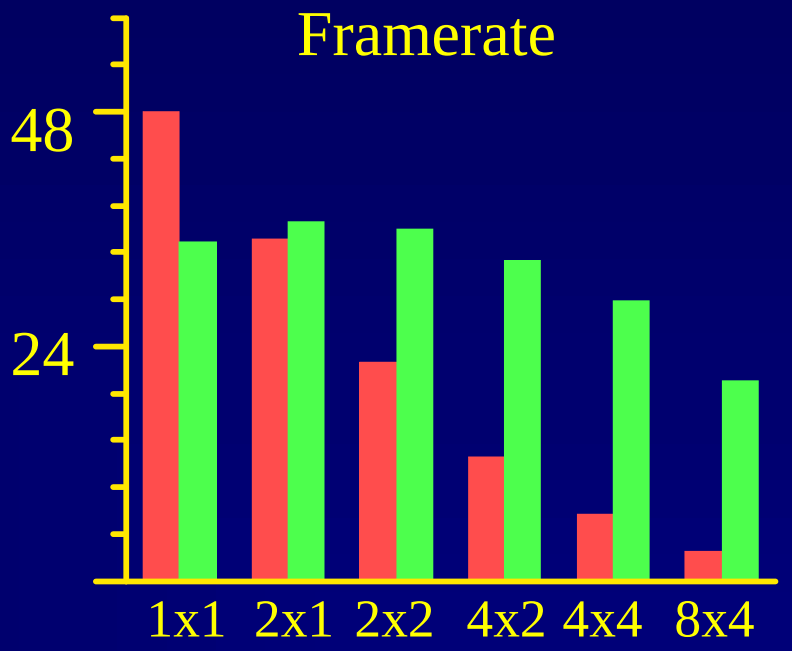




- Volume Rendering
  - **1.5 Mtri Surface**
  - **1024x768 Outputs**
  - **8x4 = 25 Mpixel display**

Tiled Rendering Results

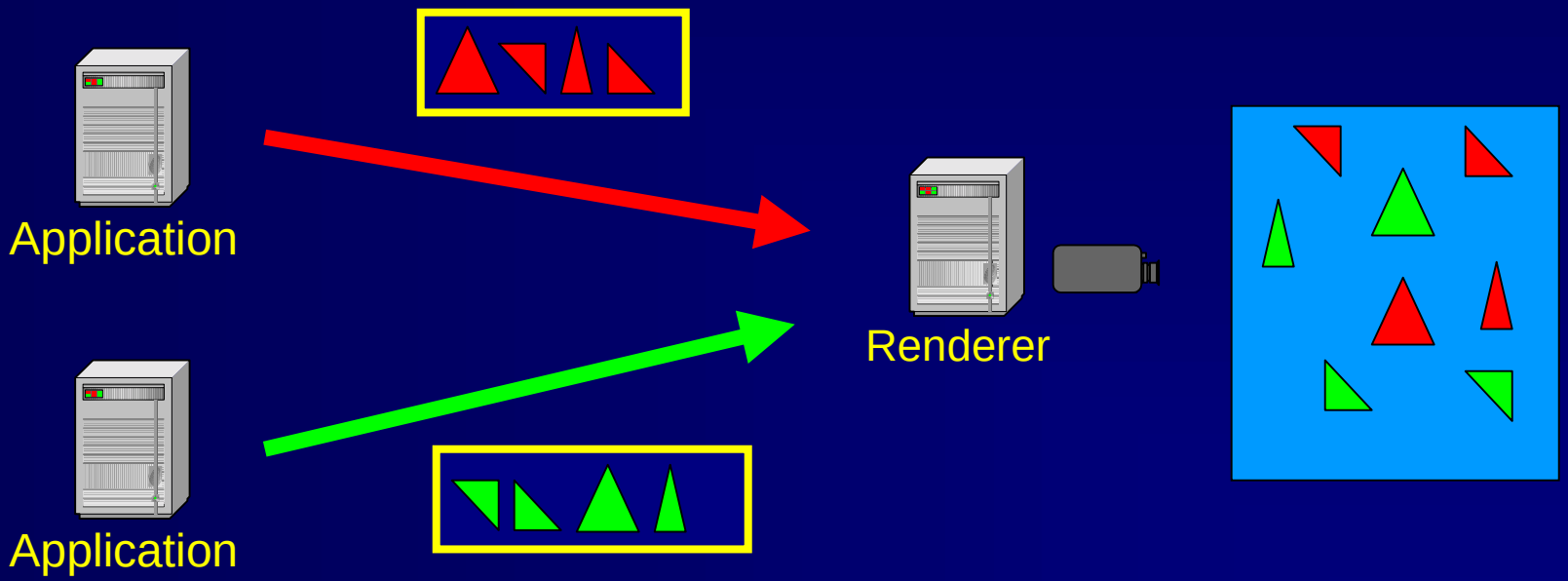
Quake III



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

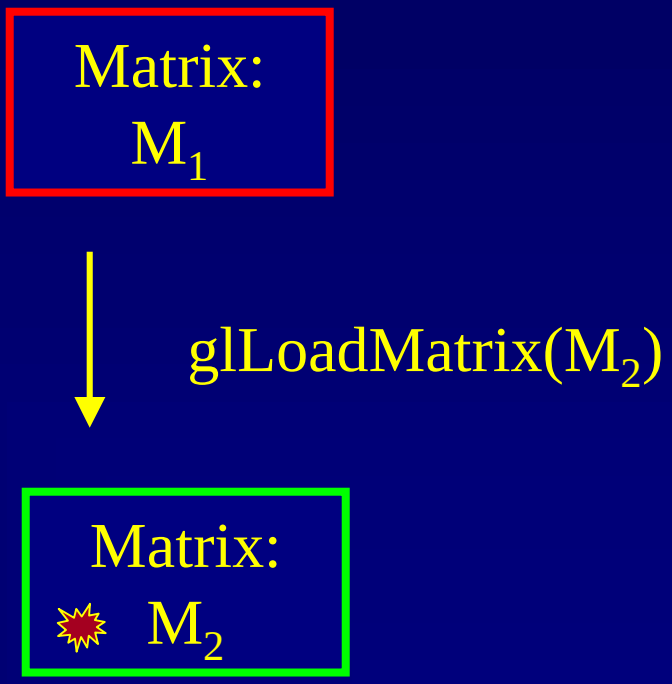
Parallel Rendering

- Requires fast context switching between streams

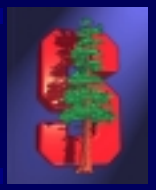


Soft Context Switching

- Generate State Commands
  - Context compare operation to generate state commands
- Benefits
  - Prevent hardware pipeline flushes
  - Switch time dependent on context differences



Soft Context Switching



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

SGI Infinite Reality	697
SGI Cobalt	2,101
NVIDIA GeForce	5,968
WireGL	191,699

Conclusions

- State tracking heiarchical 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