PixelFlow: A Sort-Last Parallel Graphics Machine

Design Criteria

- Tens of millions of triangles/sec throughput
- Linearly scalable performance
- Programmable shading

Two-Rack PixelFlow Machine

Screen Subdivision (Pxpl5)

Image Composition

PixelFlow Architecture
PixelFlow Board
Boards have same hardware components
• 2 PA-RISC processors
  transform, generate SIMD commands
• Shared processor memory
  128x64 SIMD Array (8-bit ALU)
  perform pixel operations
• Texture Memory (64 MB per board)
  not cost effective?
Board function chosen by application
  Renderer
  Shader
  Frame buffer (requires daughter card)

Board Diagram

Actual PixelFlow Board

Shader Board
Operates on one particular screen region
PA-RISC
  Generate/cache shading commands for EMCs
  Loop through shader functions
    Pre-light, light, post-light
EMCs
  Perform shading computation
    Image texture lookup
    Lighting
    Programmable shading operations

Renderer Board
Operates on subset of geometry
PA-RISC
  Stores display lists of static geometry
  Transforms geometry
  Generates/binitizes SIMD commands
EMC (Enhanced Memory Chip)
  Enable primitives pixels
    including setting Z
  Set shader id
  Load/interpolate parameters
    colors, normals, texcoords
    other arbitrary shader parameters

Image Composition
One region at a time on renderer boards
Composite each region, sending to one shader board
Shading boards send results to frame buffer
Composition network
  100 Gbit/sec bandwidth
  Bidirectional signaling hardware
Compositor Operating Modes

Programmable Shading

*Procedural shading*
High-level language for programming
Modified RenderMan language
Shading compiler generates C-code for storing EMC commands on PA-RISC
256 bytes of local memory per pixel
(show Olano SIGGRAPH 98 video)

API

Modified OpenGL
Added support for programmable shading
Added frame synchronization commands
Restrictions apply

OpenGL on PixelFlow

Application runs on host machine
Global state changes broadcast to all boards
lights, matrices, etc.
Primitives (glBegin/glEnd blocks) distributed round-robin among renderer boards
Textures loaded/replicated across all shader boards

OpenGL Extensions

Load/instance shader function
Set current shader
glMaterial extended to arbitrary shader parameters
  - global attribute state stores arbitrary parameters as well as built-ins (color, coord, etc.)
  - named shader parameters may be shared among different shader functions
Frame synchronization commands

OpenGL Restrictions

No global state changes within glBegin/glEnd
changes within glBegin/glEnd sent to a single Render, not broadcast
Cannot read back frame buffer during frame
  - Frame buffer not complete until composited and shaded at end of frame
Primitive ordering not currently guaranteed
  - bad for geometry-based decals (e.g. runway stripes)