Introduction

Programmable what?
- Simple functions
- Run at each pixel/sample
- Compute surface shading, lighting, displacement maps, atmospheric effects, ...
  - Whitted 82, Cook 84, Perlin 85, Hanrahan 90
  - Rhoades 92

Surface shader
- For one sample
- Inputs:
  - Intrinsic color, normal, texture coordinates, width, length, bumpiness, swirliness, ...
- Outputs:
  - Color, (opacity)
Resource requirements

- Programmability
- Memory
- Computational power

Programmability

- Programmable processors at sample level
- High level language (i.e. RenderMan)
  - Hanrahan 90, Upstill 90

Memory

- Table memory
- Local memory

Parallelism

- Pixel-Planes 5 (Fuchs 89)
  - 2–50 Graphics processors (i860)
  - 1–20 Renderers (custom)
  - 16k Pixel processors / renderer (custom)
- Reality Engine (Akeley 93)
  - 8–12 Geometry engines (i860XP)
  - 5–20 Fragment generators (custom)
  - 80–320 Image engines (custom)
<table>
<thead>
<tr>
<th>Resource requirements</th>
<th>Deferred shading</th>
<th>Uniform/varying</th>
<th>Fixed point/floating point</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programmability</td>
<td>Keep shading parameters at each pixel</td>
<td>Uniform = constant across pixels/samples</td>
<td>Pixel processors</td>
</tr>
<tr>
<td>Memory</td>
<td>Shade after visibility is determined</td>
<td>Varying = different in each pixel/sample</td>
<td>– Many processors</td>
</tr>
<tr>
<td>Computation power</td>
<td>Pros:</td>
<td>Don’t compute uniform values at every pixel — compute once and broadcast</td>
<td>– Simple instruction set</td>
</tr>
<tr>
<td>– Parallelism</td>
<td>– Doesn’t shade hidden pixels</td>
<td></td>
<td>– Floating point acceleration is unlikely</td>
</tr>
<tr>
<td>– Deferred shading</td>
<td>– Shading independent of geometric complexity!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>– Uniform/varying</td>
<td>– Better utilization on SIMD</td>
<td></td>
<td>– Fixed point</td>
</tr>
<tr>
<td>– Fixed point/floating point</td>
<td>– Can’t affect visibility (No transparency, no displacement maps!)</td>
<td></td>
<td>– When required precision is known</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>– More efficient in time and memory</td>
</tr>
</tbody>
</table>
**PixelFlow**

- Node description
- Memory
- Timings
- System

**PixelFlow node**

- Geometry network
- Composition network

**Memory**

- 16MB table memory
- 256 bytes local memory
- 128 bytes local memory/communication

**Timings (μs)**

- 4 byte
- 4 byte
- 2 byte
- + 3.94 μs
- 0.13
- 0.07
- * 2.53
- 2.00
- 0.50
- / 7.04
- 6.40
- 1.60
- sqrt 6.98
- 3.33
- 1.22

**Timings (μs)**
PixelFlow

- Node description
- Memory
- Timings
- System

PixelFlow system

- Rasterizer Nodes
- Shader Nodes
- Frame Buffer Node

An example

- Video
- Shading functions
- Time

Show video

Shading functions

- Pins
  - Crown, label, scuffs, dirt, Phong
- Alley
  - Wood, reflection map
- Ball
  - Phong
- Light
  - Shadow map
An example

- Video
- Shading functions
- Time
  - Breakdown of 33ms frame time
  - Breakdown of 150µs to run all shaders (excluding table lookups)
  - Time for table lookups
  - Use of multiple processors

Time: 7 ms - shadow map

An example

- Video
- Shading functions
- Time
  - Breakdown of 33ms frame time
  - Breakdown of 150µs to run all shaders (excluding table lookups)
  - Time for table lookups
  - Use of multiple processors

Time: 7 ms - reflection map

An example

- Video
- Shading functions
- Time
  - Breakdown of 33ms frame time
  - Breakdown of 150µs to run all shaders (excluding table lookups)
  - Time for table lookups
  - Use of multiple processors

Time: 15.7 ms - final image
An example

- Video
- Shading functions
- Time
  - Breakdown of 33ms frame time
  - Breakdown of 150µs to run all shaders (excluding table lookups)
  - Time for table lookups
  - Using multiple processors

Shading: 2µs - crown

Shading: 15µs - label

Shading: 39µs - scuffs & dirt
An example

- Video
- Shading functions
- Time
  - Breakdown of 33ms frame time
  - Breakdown of 150μs to run all shaders (excluding table lookups)
  - Time for table lookups
  - Using multiple processors

Shading: 15μs - wood

Shading: 28 μs - light/shadows

Shading: 12μs - Phong (pins)
An example

- Video
- Shading functions
- Time
  - Breakdown of 33ms frame time
  - Breakdown of 150µs to run all shaders (excluding table lookups)
  - Time for table lookups
  - Using multiple processors

Shading: 27µs - reflection

Shading: 12µs - Phong (ball)

Time for table lookups

- About 23ns per pixel
- Worst case
  - Bowling pin (4 lookups) in all pixels
    - Label image
    - Scuff bump map
    - Dirt image
    - Shadow map
  - Total 760µs per region
Using multiple processors

Requirements
PixelFlow
Example

Computations
Communications

Rasterizer Nodes
Shader Nodes
Frame Buffer Node

Example

Screen region

Rasterizer 1
Rasterizer 2
Shader 1
Shader 2
Frame buffer

Using multiple processors

Requirements
PixelFlow
Example

Computations
Communications

Rasterizer Nodes
Shader Nodes
Frame Buffer Node

Example

Screen region

Rasterizer 1
Rasterizer 2
Shader 1
Shader 2
Frame buffer

Using multiple processors

Requirements
PixelFlow
Example

Computations
Communications

Rasterizer Nodes
Shader Nodes
Frame Buffer Node

Example

Screen region

Rasterizer 1
Rasterizer 2
Shader 1
Shader 2
Frame buffer

Using multiple processors

Requirements
PixelFlow
Example

Computations
Communications

Rasterizer Nodes
Shader Nodes
Frame Buffer Node

Example

Screen region

Rasterizer 1
Rasterizer 2
Shader 1
Shader 2
Frame buffer
Using multiple processors

Rasterizer Nodes

Shader Nodes

Frame Buffer Node

Computation

Communication

Future work

- RenderMan-like shading compiler
- Allow programming other places
- Convince commercial vendors that they can and should do programmable shading too

Acknowledgements

- Lawrence Kesteloot, Fredrik Fatemi
- UNC PixelFlow Group
- Tony Apodaca, Pixar
- Hewlett-Packard
- ARPA ISTO Order No. A410
- NSF Grant No. MIP-9306208