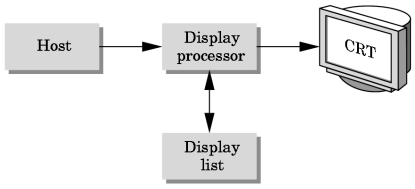
Graphics Hardware

Overview of Pipeline Architecture Alternatives

Hardware



First “graphics” processors just did display management, not rendering per se.
bitblit for block transfer of bits

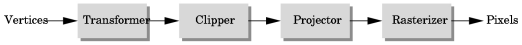
Goal

Very fast frame rate on scenes with lots of interesting visual complexity

Complexity from polygon count and/or texture mapping

Pipeline Architecture

- Pioneered by Silicon Graphics, picked up by graphics chips companies (Nvidia, 3dfx, S3, ATI,...).
- OpenGL library was designed for this architecture (and vice versa)
- Good for opaque, textured polygons and lines



Why a Pipeline Architecture?

Higher throughput

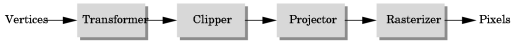
But potentially long latency

Parallel pipeline architecture

each stage can employ multiple specialized processors, working in parallel, busses between stages

#processors per stage, bus bandwidths carefully tuned for typical graphics use

Pipeline Stages



Immediate mode rendering

application generates stream of geometric

primitives (polygons, lines)

system draws each one into buffer

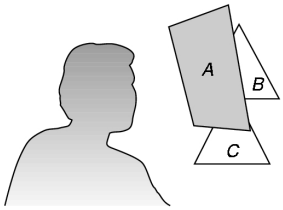
entire scene redrawn anew every frame

- transform
- light
- clip
- perspective divide
- rasterize (scan convert)
- texture & fog
- z-buffer test
- alpha blend, dither

Implementing Algorithms in Hardware

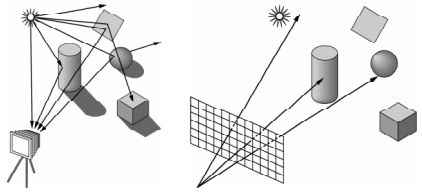
Some work well, others are harder

- **Z-buffer**
  computations are bounded, predictable



Implementing Algorithms in Hardware

- **Ray tracing**
  poor memory locality
  computational cost difficult to predict (esp. if adaptive)
  SIMD (single instruction, multiple data) parallel approach
  keep copy of entire scene on each processor



Current chip design may not be the long term answer

- **Observation: # triangles == # of pixels**
- **Could focus on interactivity**
  Latency becomes a problem
- **Could focus on animation**
  Avoid repeating computations
  Image-based rendering?

Pixel Planes and Pixel Flow (UNC)

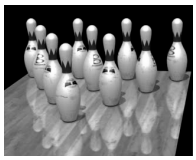
http://www.cs.unc.edu/~pxfl/

programmable processor per pixel
good for programmable shading, image processing
can be used for rasterization
Pixel-Planes 4: 512x512 processors with 72bits of memory
But most processors idle for most triangles
Pixel-Planes 5: divide screen into ~20 tiles each with a bank of processors. Network is limit. 2Million tri/sec.

Pixel Planes and Pixel Flow (UNC)

Pixel-Flow: Image composition. Subdivide geometry to processors and recombine by depth using special hardware

Rendered on simulator and predicted to run in real time on physical hardware



Talisman (Microsoft)

http://research.microsoft.com/MSRSIGGRAPH/96/Talisman/

Observation: an image is usually much like the one that preceded it in an animation.

Goal: a $200-300 board

image-based rendering
cache images of rendered geometry
re-use with affine image warping (sophisticated sprites)
re-render only when necessary to reduce bandwidth and computational cost

Current & Future Issues

- **interaction**
- **geometry compression**
- **progressive transmission**
- **alternative modeling schemes (not polygon soup)**
  - parametric surfaces, implicit surfaces, subdivision surfaces
  - generalized texture mapping: displacement mapping, light mapping
  - programmable shaders
- **beyond just geometry:**
  - dynamics, collision detection, AI?

Assignment 1

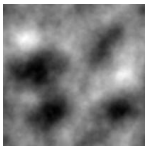
Height Fields

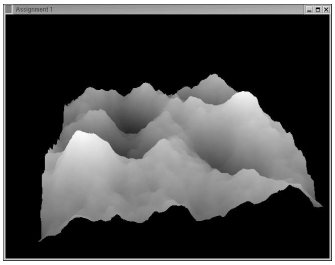
Height Fields

- **Why?**
  - Get started with OpenGL
  - Some room for creativity
- **Where?**
  - Wean 5336 or your machine at your risk!
- **How?**
  - Cross-realm authentication via andrew
  - Send problems to me or to the TA's (soon)
  - Make sure that you made directory with correct permissions—most common problem

Height Fields

- **What?**





- **When? -- Due midnight September 12th**