

The Graphics Pipeline Revisited

(thanks to David Luebke, University of Virginia)

Johns Hopkins Department of Computer Science
 Course 600.460: Interactive Graphics and Games, Spring 2005, Professor: Jonathan Cohen

Classic Rendering Pipeline
 (as taught in Intro Graphics courses)

Classic Rendering Pipeline

What's wrong with this model (for an OpenGL system)?

- Model/view transforms combined
- Really "vertices" not "primitives"
 - Making this the *vertex pipeline*
- There's a lot going on in the "scan conversion" stage!
 - Primitive assembly
 - Rasterization
 - Texture mapping
 - Per-pixel lighting
 - Visibility (Z-buffer)
- We refer to these collectively as the *pixel or fragment pipeline*

High-Level Pipeline

Back up & think about the larger picture:

• What sort of tasks does each stage perform?

High-Level Pipeline

| Application | Geometry (a.k.a. "vertex pipeline") | Rasterization (a.k.a. "pixel pipeline" or "fragment pipeline") |
|-----------------|--|---|
| Handle input | Transform | Rasterize (fill pixels) |
| Simulation & AI | Lighting | Interpolate vertex parameters Look up/filter textures |
| Culling | Skinning | Z- and stencil tests |
| LOD selection | Calculate texture coords | Blending |
| Prefetching | | |

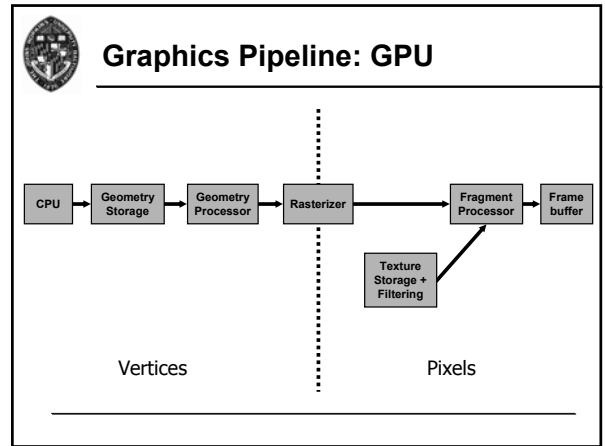
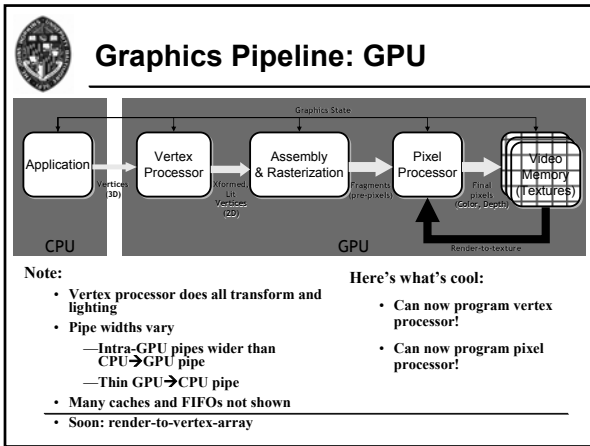
Graphics Pipeline: CPU

SIM

CULL

DRAW

DISPLAY



Graphics Pipeline: GPU

Don't forget! Not your father's (or professor's) GPU

