THE UNIVERSITY of NEW MEXICO

Introduction to Computer Graphics with WebGL

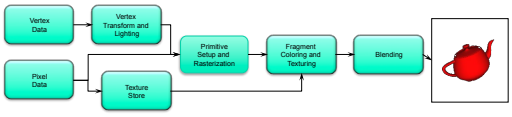
Ed Angel

OpenGL and WebGL

THE UNIVERSITY of NEW MEXICO

OpenGL

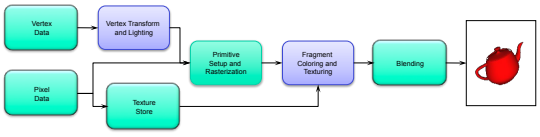
- Developed at SGI based on pipeline model
- Used their VLSI pipeline
- Released 1994
- Originally fixed function
- Immediate mode graphics



THE UNIVERSITY of NEW MEXICO

OpenGL Evolution

- OpenGL 2.0 added programmable shaders
  - *vertex shading* augmented the fixed-function transform and lighting stage
  - *fragment shading* augmented the fragment coloring stage
- However, the fixed-function pipeline was still available



Immediate Mode Graphics

- Geometry specified by vertices
  - Locations in space(2 or 3 dimensional)
  - Points, lines, circles, polygons, curves, surfaces
- Immediate mode
  - Each time a vertex is specified in the application, its location is sent to the GPU
  - Old OpenGL style uses `glBegin`, `glVertex`,`glEnd`
  - Creates bottleneck between CPU and GPU
  - Removed from OpenGL 3.1 and OpenGL ES 2.0

Modern OpenGL

- Performance is achieved by using GPU rather than CPU
- Control GPU through programs called shaders
- Application's job is to send data to GPU
- GPU does all rendering
- Immediate mode replaced by retained mode
- As of OpenGL 3.1, all applications must provide shaders

Retained Mode Graphics

- Put all vertex attribute data in array
- Send array to GPU to be rendered immediately
- Almost OK but problem is we would have to send array over each time we need another render of it
- Better to send array over and store on GPU for multiple renderings

OpenGL 3.1

- Totally shader-based
  - No default shaders
  - Each application must provide both a vertex and a fragment shader written in OpenGL Shading Language (GLSL)
- No immediate mode
- Few state variables
- Most 2.5 functions deprecated
- Backward compatibility not required
  - Exists a compatibility extension

Other Versions

- OpenGL ES
  - Embedded systems
  - Version 1.0 simplified OpenGL 2.1
  - Version 2.0 simplified OpenGL 3.1
    - Shader based
- WebGL
  - Javascript implementation of ES 2.0
  - Supported on all recent browsers
- OpenGL 4.1, 4.2, …..
  - Add geometry, tessellation, compute shaders

Why WebGL?

- Advantages of desktop OpenGL
  - rendering code independent of platform
  - simple but close to hardware
  - makes use of GPU features
  - relatively stable
- Disadvantages of desktop OpenGL
  - code must be recompiled for each platform
  - no interaction or input functions
  - can't run from a remote site

THE UNIVERSITY of NEW MEXICO

Why WebGL?

- All recent browsers run HTML5 and JavaScript
  - no system dependencies
  - code interpreted
- Code located remotely but run locally
  - uses local GPU
- Availability of many packages
  - WebGL compatible with CSS, jQuery and others for interaction and style

THE UNIVERSITY of NEW MEXICO

Potential Issues with WebGL

- JS is an interpreted language
  - slower (maybe)
- Lacks some of the latest features of desktop OpenGL
  - geometry shaders
  - tessellation shaders
  - compute shaders
  - no core profile
- Potential security issues