CIS 581 Interactive Computer Graphics

(slides based on Dr. Han-Wei Shen's slides)

Instructor: Rick Parent (parent@cse.osu.edu)

Credit: 4

Class: MWF 10:30 – 11:18 pm DL357

Office hours: MW 11:30 – 12:18 DL 787

Class information on Carmin: www.carmen.osu.edu

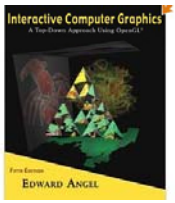
Prerequisite: CSE 222 or 230 or 502

Requirements

- Interested in Computer Graphics
- Capable in C/C++ programming
- Comfortable with basics of linear algebra (vector and matrix calculations) or be able to pick it up easily

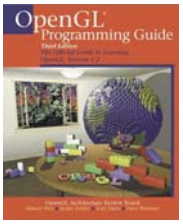
Textbook

• *Interactive Computer Graphics, A Top-Down Approach Using OpenGL* by Edward Angel, 5th edition

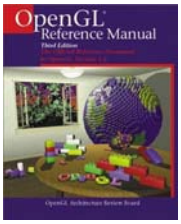


- Very easy to read!
- Help you to understand the lectures and prepare for exams
- Many OpenGL examples in C
- Not 6th Edition; 4th should be OK

Reference Books



The red book
OpenGL programmer's Guide



The blue book
OpenGL reference manual

Not required or even recommended, unless you really expect to program in OpenGL beyond this class (e.g. games)

Grading

- Five Labs: 50 %
  2D and 3D drawing – transformation, lighting, texture mapping, etc.
- Midterm Exam: 20 %
- Final Exam: 30%

What is Computer Graphics?

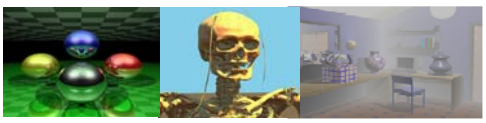
- Computer-generated images or sequences of images (i.e., animations, movies)
- The scientific study of techniques and methods for generating such images
- Not simply trying for photorealism!
  - Painterly effects
  - Sketches, toon shading
  - etc

Some 3D Computer Graphics Applications

- Manufacturing design (CAD)
- Movies, TV, commercials
  - Animations
  - Special effects mixed with live footage
- Visual arts
- Video games
- Scientific visualization
- Simulation of natural

What will I learn from this course?

- A basic understanding of graphics hardware/software technology – algorithms and jargon
- Learn how to use OpenGL to write 2D/3D drawing programs
- Prepare yourself for advanced graphics topics (CIS 681, 781, 782, …)



Some Jargon…

- Graphics Processor (GPU)
- What about it?
  nVIDIA/ATI Graphics Chips

- ✓ 32-bit colors, Z/stencil buffer
- ✓ Advanced Per-pixel lighting
- ✓ 50 million triangles per second

…

OpenGL programming

- An industry standard API

- This is NOT just a API course
  - – You are expect learn the graphics processing pipeline and the theory behind
  - – You don't need to implement the algorithms. Instead, use OpenGL

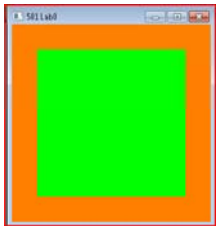
Labs

- 'Official programming environment
- 5 Labs
- 50% of your grade
- Don't get behind on the labs!
- Can be hard to debug – incremental development

Lab Examples

Lab1:

- Make sure you can write, compile, run, and submit simple OpenGL program on class machines



Lab Examples (continued)

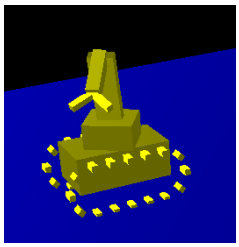
Lab2:

- Learn how to create an OpenGL window (using GLUT)
- Learn how to draw simple 2D primitives (lines, triangles, polygons etc)
- Learn how to process mouse input

Lab Examples (cont'd)

Lab3:

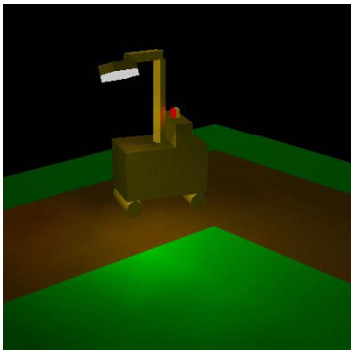
- perspective view
- display lists
- simple illumination
- camera control by mouse movement
- hierarchical animation



Lab Examples (cont'd)

Lab4:

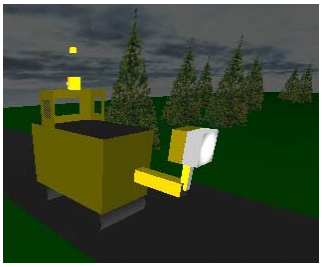
- illumination
- material properties
- decals



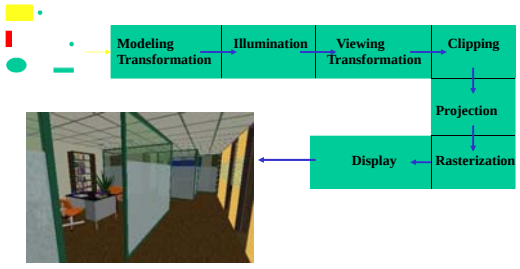
Lab Examples (cont'd)

Lab5:

- texture mapping
- Transparent surfaces
- billboarding
- first person view



Graphics Pipeline



Outline of course

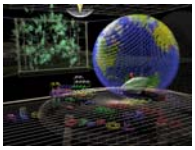
- Geometry
- Rasterization
- Shading
- Hidden surface elimination
- Texture mapping
- Modeling
- Animation
- Ray tracing
- Global illumination

from M. Woo et al., 1997

Outline of course

- Scene Geom
- Rasterization
- Shading
- Hidden surface elimination
- Texture mapping
- Object Modeling
- Animation
- Ray tracing
- Global illumination

How to specify the 3-D positions of the camera and the scene objects and their various parts, how to project these to 2-D image locations, and how to represent trans-formations of these positions



Outline of course

- Geometry
- Rasterization
- Shading
- Hidden surface elimination
- Texture mapping
- Modeling
- Animation
- Ray tracing
- Global illumination

How to set individual image pixels corresponding to projected geometric objects such as points, lines, polygons, and more complicated shapes. Anti-aliasing reduces artifacts (“jaggies”) caused by finite image resolution

Outline of course

- Geometry
- Rasterization
- Shading
- Hidden surface elimination
- Texture mapping
- Modeling
- Animation
- Ray tracing
- Global illumination

How to model light interaction with 3-D surfaces with varying material properties in order to calculate the proper colors perceived by the eye at different image locations

Outline of course

- Geometry
- Rasterization
- Shading
- Hidden surface elimination
- Texture mapping
- Modeling
- Animation
- Ray tracing
- Global illumination

How to efficiently rasterize only the visible parts of scene objects

Outline of course

- Geometry
- Rasterization
- Shading
- Hidden surface elimination
- Texture mapping
- Modeling
- Animation
- Ray tracing
- Global illumination

How to apply “layers” of detail to scene objects to show features, simulate bumps and reflections, or other precomputed shading effects. Procedural texturing is concerned with how some kinds of textures are generated algorithmically

Outline of course

- Geometry
- Rasterization
- Shading
- Hidden surface elimination
- Texture mapping
- Modeling
- Animation
- Ray tracing
- Global illumination

Brown et al, OSU

How to efficiently represent the geometry of scene objects, which may be complex, curved, etc. (CSE 784, CSE682)

Outline of course

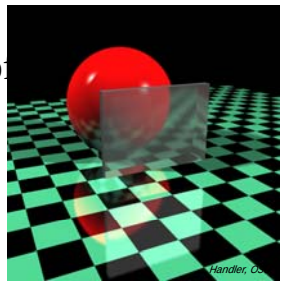
- Geometry
- Rasterization
- Shading
- Hidden surface elimination
- Texture mapping
- Modeling
- Animation
- Ray tracing
- Global illumination

Chen et al, OSU

How to render dynamic scenes, as well as how to simulate dynamic phenomena (CSE 682, CSE 683)

Outline o

- Geometry
- Rasterization
- Shading
- Hidden surface elimination
- Texture mapping
- Modeling
- Animation
- Ray tracing
- Global illumination



Handler, OS

How to realistically simulate the movement of rays from light sources through multiple object reflections and refractions on the way to the eye (CSE 681)

Outline of course

- Geometry
- Rasterization
- Shading
- Hidden surface elimination
- Texture mapping
- Modeling
- Animation
- Ray tracing
- Global illumination

Gao et al, OSU

How to realistically simulate inter-reflections of light between multiple sources and object surfaces (CSE 782)

Outline of course

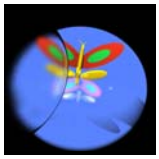
- Geometry
- Rasterization
- Shading
- Hidden surface elimination
- Texture mapping
- Modeling
- Animation
- Ray tracing
- Global illumination

CSE 781 will cover these in more detail with the focus on programmable GPU's and real-time game engine design.

Parmelee and Ruston, OSU

Graphics Courses

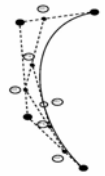

CSE 581 **(Au,Sp)**


CSE 681 **(Au,Wi)**


CSE 682 **(Wi)**

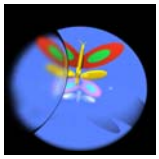

CSE 781 **(Wi)**


CSE 782 **(Au)**


CSE 784 **(Sp)**

Graphics Courses (Autumn)


CSE 581 **(Au,Sp)**


CSE 681 **(Au,Wi)**

CSE 682 **(Wi)**

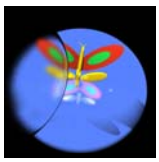
CSE 781 **(Wi)**


CSE 782 **(Au)**

CSE 784 **(Sp)**

Graphics Courses (Winter)

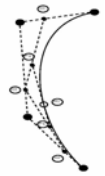

CSE 581 **(Au,Sp)**


CSE 681 **(Au,Wi)**


CSE 682 **(Wi)**


CSE 781 **(Wi)**

CSE 782 **(Au)**


CSE 784 **(Sp)**

Graphics Courses (Spring)


CSE 581 **(Au,Sp)**

CSE 681 **(Au,Wi)**

CSE 682 **(Wi)**

CSE 781 **(Wi)**

CSE 782 **(Au)**


CSE 784 **(Sp)**

Where do I do my labs?

- Graphics PC Lab – CL 112D
  - Each PC has decent graphics card
  - Software: Visual Studio 2010
    OpenGL/Glut

Develop anywhere, but submit source code (.c or .cpp files) that compile and run on CL112D environment.

Image Gallery