Advanced Computer Graphics
CSCI B581 – Spring 2017

Instructor:
Mitja Hmeljak,
http://pages.iu.edu/~mitja
mitja@indiana.edu
Advanced Computer Graphics
CSCI B581 Spring 2017

Instructor:
Mitja Hmeljak, mitja@indiana.edu

Lectures: AC C116 (Cedar Hall-Union Street Center),
MW 2:30PM – 3:45PM

Office hours: Tue, Thu 2:30-3:30pm (office: LH 425)

Assistant Instructors:
Leif Christiansen, leifchri@iu.edu
Steve Babcock, wsbabcoc@iu.edu

Labs: LH030 (Lindley Hall),
Fri 1:00PM - 2:15PM
Office hours: TBA
Objectives

Broad introduction to Computer Graphics
  Software
  Hardware
  Applications

Shader-Based OpenGL:
  → → **OpenGL ES** ← ←

( primarily OpenGL ES 2.0, compatible with "desktop" OpenGL 3.1 (and later) and "browser" WebGL 1.0 (and later) )
Prerequisites

*Good* programming skills (*C* & descendants, *Python*, ...)

*Basic Data Structures*
- Linked lists
- Arrays

*Geometry*

*Simple Linear Algebra* (vector and matrix operations)
Assignments

Programming Assignments
  Simple
  Interactive
  2D and 3D

Written (non-programming) Homeworks

B581 syllabus on main page at:
  https://www.cs.indiana.edu/classes/b581/2017/
References

Textbooks

Required:
OpenGL ES 2.0 Programming Guide (1st Edition)
by A. Munshi, D. Ginsburg and D. Shreiner
Addison-Wesley Professional 2009

Required:
Required: "Interactive Computer Graphics" 6th (or 7th) Edition
by Edward Angel and Dave Shreiner - Pearson
https://www.cs.unm.edu/~angel/BOOK/INTERACTIVE_COMPUTER_GRAPHICS/
Additional References

http://www.khronos.org/
standard-defining consortium: OpenGL &c.
official documentation and specifications
Khronos OpenGL ES Registry - https://www.khronos.org/registry/gles/#specs32

OpenGL ES Programming Guide for iOS

http://www.opengl.org
used to be the official repository for standards documents
links to man pages, OpenGL wiki, etc.

a little dated: OpenGL Shading Language, 3rd Edition

also of interest: The OpenGL Programmer’s Guide (the Redbook) 7th Edition
The definitive reference for desktop OpenGL
Shader-based

Many computer graphics courses use OpenGL but still use fixed-function pipeline...
...which does not require shaders
Does not make use of the full capabilities of the graphics processing unit (GPU)

OpenGL ES

OpenGL "for Embedded Systems"
runs on the vast majority of mobile devices
makes use of local hardware
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>About You (index card)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>B581</strong></td>
<td><strong>First Name, Last Name</strong></td>
<td><strong>2017-01-09</strong></td>
</tr>
</tbody>
</table>

1. Name, Major, Class Year
2. Why are you taking this course?
3. Do you have any previous computer graphics programming experience? In what APIs and programming language(s)?
   When did you take any previous graphics classes?
4. Do you have any previous experience with *native* mobile programming?
   If so, was it for Android (Java), iOS (Objective C, Swift), other?
5. What would you like to be able to do with computer graphics?
6. What are your concerns about this course (if any)?
7. Do you have access to a laptop, and would you be willing to bring it to *lectures and labs*? (if yes, please state operating system type and version version)
8. Do you have access to a smartphone device running iOS or Android?
   (if yes, please state operating system type and version version)
What is Computer Graphics?

Computer graphics simulates the physics of the interaction of light with matter.

The computational problem in computer graphics is to carry out that simulation to a level of accuracy consistent with the available resources and the requirements of the problem.
Computer Graphics

*Computer graphics* deals with all aspects of creating images with a computer

- Hardware
- Software
- Applications
Example

Where could this image be coming from?

What hardware / software do we need to produce it?
Preliminary Answer

**Application:** The object is an artist’s rendition of the sun for an animation to be shown in a domed environment (planetarium)

**Software:** Maya, Blender, etc. for modeling and rendering but those are built on top of OpenGL

**Hardware:** PC/Mac/mobile device with sufficient graphics capabilities for modeling and rendering
Basic Graphics System

Input devices

Processor (CPU)

Graphics processor

Frame buffer

Output device

Image formed in frame buffer
what we used to have as output: a CRT

Can be used either as a line-drawing device (calligraphic) or to display contents of frame buffer (raster mode)

Computer graphics goes back to the earliest days of computing
Strip charts
Pen plotters
Simple displays using A/D converters to go from computer to calligraphic CRT
Cost of refresh for CRT too high
Computers slow, expensive, unreliable
Wireframe graphics
Draw only lines
Sketchpad
Display Processors
Storage tube

wireframe representation of object
Sketchpad

Ivan Sutherland’s PhD thesis at MIT
Recognized the potential of man-machine interaction

Loop
Display something
User moves light pen
Computer generates new display

Sutherland also created many of the now common algorithms for computer graphics
Display Processor

Rather than have the host computer try to refresh display use a special purpose computer called a display processor (DPU)

Graphics stored in display list (display file) on display processor
Host compiles display list and sends to DPU
Created by Tektronix

Did not require constant refresh

Standard interface to computers

Allowed for standard software

Plot3D in Fortran

Relatively inexpensive

Opened door to use of computer graphics for CAD community

Raster Graphics
Beginning of graphics standards

IFIPS
  GKS: European effort
    Becomes ISO 2D standard
  Core: North American effort
    3D but fails to become ISO standard

Workstations and PCs
Raster Graphics

Image produced as an array (the *raster*) of picture elements (*pixels*) in the *frame buffer*
Raster Graphics

Allows us to go from lines and wire frame images to filled polygons
Although we no longer make the distinction between workstations and PCs, historically they evolved from different roots.

Early workstations characterized by:
- Networked connection: client-server model
- High-level of interactivity

Early PCs included frame buffer as part of user memory:
- Easy to change contents and create images

Realism comes to computer graphics

smooth shading  environment mapping  bump mapping

Special purpose hardware
  Silicon Graphics geometry engine
    VLSI implementation of graphics pipeline
Industry-based standards
  PHIGS
  RenderMan
Networked graphics: X Window System
Human-Computer Interface (HCI)

OpenGL API
Completely computer-generated feature-length movies (Toy Story) are successful
New hardware capabilities
  Texture mapping
  Blending
  Accumulation, stencil buffers
Photorealism
Graphics cards for PCs dominate market
   Nvidia, ATI
Game boxes and game players determine direction of market
Computer graphics routine in movie industry: Maya, Lightwave
Programmable pipelines
Computer Graphics and Image Formation

Computer graphics systems focus on "simulating the physics of the interaction of light with matter".

The computational problem that Computer Graphics aims to solve is to "carry out that simulation to a level of accuracy consistent with the available resources and the requirements of the problem".

We also refer to this simulation as "Image Formation".
Elements of Image Formation

Objects
Viewer
Light source(s)

Attributes that govern how light interacts with the materials in the scene
Note the independence of the objects, the viewer, and the light source(s)
Light

*Light* is the part of the electromagnetic spectrum that causes a reaction in our visual systems. Generally these are wavelengths in the range of about 350-750 nm (nanometers). Long wavelengths appear as reds and short wavelengths as blues.
One way to form an image is to follow rays of light from a point source finding which rays enter the lens of the camera. However, each ray of light may have multiple interactions with objects before being absorbed or going to infinity.
Luminance and Color Images

Luminance Image
Monochromatic
Values are gray levels
Analogous to working with black and white film or television

Color Image
Has perceptual attributes of hue, saturation, and lightness
Do we have to match every frequency in visible spectrum? No!
Three-Color Theory

Human visual system has two types of sensors
  Rods: monochromatic, night vision
  Cones
    Color sensitive
    Three types of cones
    Only three values (the tristimulus values) are sent to the brain
Need only match these three values
Need only three primary colors
Old-style: Shadow Mask CRT
Additive and Subtractive Color

Additive color
Form a color by adding amounts of three primaries
CRTs, projection systems, positive film
Primaries are Red (R), Green (G), Blue (B)

Subtractive color
Form a color by filtering white light with cyan (C), Magenta (M), and Yellow (Y) filters
Light-material interactions
Printing
Negative film
Use trigonometry to find projection of point at \((x,y,z)\)

\[
x_p = -\frac{x}{z/d} \quad y_p = -\frac{y}{z/d} \quad z_p = d
\]

These are equations of simple perspective.
Models and Architectures

Learn the basic design of a graphics system
Introduce pipeline architecture
Examine software components for an interactive graphics system
So... how do we go about Image Formation?

Can we mimic the synthetic camera model to design graphics hardware software?

Application Programmer Interface (API)

Need only specify

- Objects
- Materials
- Viewer
- Lights

But how is the API implemented?
Physical Approaches

**Ray tracing**: follow rays of light from center of projection until they either are absorbed by objects or go off to infinity

- Can handle global effects
  - Multiple reflections
  - Translucent objects
- Slow
- Must have whole data base available at all times

**Radiosity**: Energy based approach

- Very slow
Reading Assignment for Week 1:

• Edward Angel and Dave Shreiner, Chapter 1 sections 1.2, 1.3, 1.4