Intro

Week 1, Mon Jan 8

Outline

• Defining Computer Graphics
• Course Structure
• Course Content Overview
What is Computer Graphics?

• create or manipulate images with computer
  • this course: algorithms for image generation
What is CG used for?

- movies
  - animation
  - special effects
What is CG used for?

• computer games
What is CG used for?

• images
  • design
  • advertising
• art
What is CG used for?

- virtual reality / immersive displays
What is CG used for?

- graphical user interfaces
  - modeling systems
  - applications
- simulation & visualization
Real or CG?

http://www.alias.com/eng/etc/fakeorfoto/quiz.html
Real or CG?
Real or CG?
Real or CG?
Expectations

- hard course!
  - heavy programming and heavy math
- fun course!
  - graphics programming addictive, create great demos
- programming prereq
  - CPSC 216 (Program Design and Data Structures)
  - course language is C++/C
- math prereq
  - MATH 200 (Calculus III)
  - MATH 221/223 (Matrix Algebra/Linear Algebra)
Course Structure

• 39% programming projects
  • 8% project 1 (building beasties with cubes and math)
  • 8% project 2 (flying)
  • 8% project 3 (TBA)
  • 15% project 4 (create your own graphics game)
• 25% final
• 24% midterm (week 5 Fri 2/9, week 11 Wed 3/28)
• 12% written assignments
  • 3% each HW 1/2/3/4
• programming projects and homeworks synchronized
Programming Projects

- **structure**
  - C++, Linux
    - OK to cross-platform develop on Windows, Mac
  - OpenGL graphics library
  - GLUT for platform-independent windows/UI
  - face to face grading in lab
- **Hall of Fame**
  - first project: building beasties
    - previous years: giraffes, frogs, elephants, birds, poodles, dinos, cats…
  - last project: create your own graphics game
Late Work

- 3 grace days
  - for unforeseen circumstances
  - strong recommendation: don’t use early in term
  - handing in late uses up automatically unless you tell us
- otherwise: 50% if one day (24 hrs) late, 0% afterwards
- only exception: severe illness or crisis
  - as per UBC rules
  - must let me know ASAP (in person or email)
    - at latest, 7 days after return to school
  - must also turn in form with documentation (doctor note)

Regrading

- to request assignment or exam regrade
  - give me paper to be regraded, and also in writing
    - what problem you're disputing
    - detailed explanation why you think grader was wrong
  - I will not accept until next class after solutions handed out
- I may regrade entire assignment
  - thus even if I agree with your original request, your score may nevertheless end up higher or lower
Course Information

- course web page is main resource
  - updated often, reload frequently
- newsgroup is ubc.courses.cpsc.414
  - note old course number still used
  - readable on or off campus
- (no WebCT)
Teaching Staff

- instructor: Tamara Munzner
  - tmm@cs.ubc.ca
  - office hrs in ICICS/CS 011
    - Wed/Fri 11-12
  - or by appointment in X661
- TAs: Matt Baumann, Gordon Wetzstein
  - mabauman@cs.ubc.ca
  - wetzste1@cs.ubc.ca
- use newsgroup, not email, for all questions that other students might care about
Labs

- attend one lab per week
  - Tue 1-2, Thu 10-11 (Matt Baumann)
  - Fri 12-1 (Gordon Wetzstein)
- mix of activities
  - example problems in spirit of written assignments and exams
  - help with programming projects
- no deliverables (unlike intro classes)
- strongly recommend that you attend
Required Reading

- Fundamentals of Computer Graphics
  - Peter Shirley, AK Peters, 2nd edition

- OpenGL Programming Guide, v 1.4
  - OpenGL Architecture Review Board
    - v 1.1 available for free online

- readings posted on schedule page
Learning OpenGL

- this is a graphics course using OpenGL
  - not a course *on* OpenGL
- upper-level class: learning APIs mostly on your own
  - only minimal lecture coverage
    - basics, some of the tricky bits
- OpenGL Red Book
- many tutorial sites on the web
  - nehe.gamedev.net
Plagiarism and Cheating

• don’t cheat, I will prosecute
  • insult to your fellow students and to me
• programming and assignment writeups must be individual work
  • can discuss ideas, browse Web
  • cannot just copy code or answers
  • cannot do team coding
  • exception: final project can be team of two or three
• you must be able to explain algorithms during face-to-face demo
  • or no credit for that part of assignment
  • and possibly prosecution
Citation

- cite all sources of information
  - what to cite
    - study group members, books, web sites
  - where to cite it
    - README for programming projects
    - end of writeup for written assignments
  - http://www.ugrad.cs.ubc.ca/~cs314/Vmay2005/policies.html# plag
Course Content Overview
This Course

• we cover
  • basic **algorithms** for
    • rendering – displaying models
    • (modeling – generating models)
    • (animation – generating motion)
  • programming in OpenGL, C++

• we do not cover
  • art/design issues
  • commercial software packages
Other Graphics Courses

- CPSC 424: Geometric Modeling
  - offered this year
- CPSC 426: Computer Animation
  - offered next year
- CPSC 514: Image-based Modeling and Rendering
- CPSC 526: Computer Animation
- CPSC 533A: Digital Geometry
- CPSC 533B: Animation Physics
- CPSC 533C: Information Visualization
- CPSC 530P: Sensorimotor Computation
Rendering

- creating images from models
  - geometric objects
    - lines, polygons, curves, curved surfaces
  - camera
    - pinhole camera, lens systems, orthogonal
  - shading
    - light interacting with material
- illustration of rendering capabilities
  - Shutterbug series by Williams and Siegel using Pixar's Renderman
    - www.siggraph.org/education/materials/HyperGraph/shutbug.htm
Modelling Transformation: Object Placement
Viewing Transformation: Camera Placement
Perspective Projection
Depth Cueing
Depth Clipping
Colored Wireframes
Hidden Line Removal
Hidden Surface Removal
Per-Polygon Shading
Gouraud Shading
Specular Reflection
Phong Shading
Curved Surfaces
Complex Lighting and Shading
Texture Mapping
Displacement Mapping
Reflection Mapping
Modelling

- generating models
  - lines, curves, polygons, smooth surfaces
  - digital geometry
Animation

- generating motion
  - interpolating between frames, states

http://www.cs.ubc.ca/~van/papers/doodle.html
Readings

- today
  - FCG Chap 1
- Wed
  - FCG Chap 2
    - except 2.5.1, 2.5.3, 2.7.1, 2.7.3, 2.8, 2.9, 2.11.
  - FCG Chap 5.1-5.2.5
    - except 5.2.3, 5.2.4
- Fri
  - RB Chap Introduction to OpenGL
  - RB Chap State Management and Drawing Geometric Objects
  - RB App Basics of GLUT (Aux in v 1.1)