CPSC 314
Computer Graphics

Wolfgang Heidrich

People

**Instructor:**
- Wolfgang Heidrich

**TAs:**
- Benjamin Cecchetto
- Landon Boyd
- Sarah Motiee (hourly)
What is Computer Graphics?

Create or manipulate images with computer
- this course: algorithms for image generation

What is CG used for?

Graphical user interfaces
- Modeling systems
- Applications

Simulation & visualization
**What is CG used for?**

**Movies**
- Animation
- Special effects

**Computer games**
What is CG used for?

Images

- Design
- Advertising
- Art

Real or CG?

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

http://www.autodesk.com/eng/etc/fakeorfoto/quiz.html

CG!

Real or CG?

http://www.autodesk.com/eng/etc/fakeorfoto/quiz.html

Real!
Real or CG?

http://www.autodesk.com/eng/etc/fakeorfoto/quiz.html

CG!

Real or CG?

http://www.autodesk.com/eng/etc/fakeorfoto/quiz.html

Real!
Real or CG?

http://www.autodesk.com/eng/etc/fakeorfoto/quiz.html

CG!

Real or CG?

http://www.autodesk.com/eng/etc/fakeorfoto/quiz.html

Real!
Real or CG?
http://www.autodesk.com/eng/etc/fakeorfoto/quiz.html

Real!

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

Real!
Real or CG?

http://www.autodesk.com/eng/etc/fakeorfoto/quiz.html

Real!

What This Course Is About

Topics covered

- Fundamental algorithms of computer graphics
- Interactive graphics:
  - The rendering pipeline
    - Abstract model for the functioning of graphics hardware and interactive graphics systems
  - Color spaces and reflection models
  - Shadow algorithms
- Ray-tracing, global illumination
What This Course is NOT About

*Topics NOT covered:*
- Artistic and design issues
- Usage of commercial software packages
- Applications (i.e. game design)

*Topics covered with little detail:*
- Animation, Geometric Modeling
  - *These have separate undergrad classes*
  - *CPSC 424 (Geometric Modeling)*

Learning OpenGL

*This is a graphics course using OpenGL*
- Not a course on OpenGL

*Learning API mostly on your own*
- Only minimal lecture coverage
  - *Basics, some of the tricky bits*
- Also: ask in the labs
- OpenGL Red Book
- many tutorial sites on the web
  - *nehe.gamedev.net*
Course Organization

**Components:**
- Lectures
- Homework problems, reading
- Labs
- Programming assignments (3+1)
- Quizzes (2)
- Final

**Required skills:**
- Assignments: demanding programming problems
- Exams: math heavy, lots of linear algebra, some calculus, algorithms

---

Course Organization

**Grades and Grading**
- Programming assignments: 35%
  - 5% for Assignment 0, 10% each for A1-3
- Quiz 1: 10%  Quiz 2: 15%
- Final: 40%
- You MUST pass the **final** and the **assignment** portion to pass the course

- Assignment 0 is out today, due next Monday
  - *Mostly math refresher, setting up programming environment*
Course Organization

**Homework problems**
- NOT graded
- BUT: essential preparation for quizzes/final

**Labs**
- Opportunity to work on assignments with TAs present
- TAs discuss solutions to homeworks, quizzes, etc.
  - *If you have problems solving the homeworks, go to the labs!*
- Labs start today (go there for help setting up the compute environment for A0)

---

Course Organization

**Programming assignments:**
- C++, Linux (or Windows, Mac)
- OpenGL graphics library / GLUT for user interface
- Labs: ICICS 005
  - *Linux machines*
  - *All assignments need to run on these machines*

**Collaboration policy:**
- No collaboration on programming assignments
- Reference all external resources
Course Organization

Up-to-date information:
- http://www.ugrad.cs.ubc.ca/~cs314
- WebCT (follow link from course home page)
  - Bulletin board
  - Reporting of grades

Books

Textbook:
  - Contains required reading
- Woo, Neider: OpenGL Programming Guide
  - Very useful as reference for assignments
  - This book is online: see link from course web page

Other Books:
- Foley, vanDam, Feiner, Hughes: Computer Graphics, Principles and Practice
  2nd Edition in C, Addison Wesley
Syllabus

Overview

The Rendering Pipeline (1)
- Geometry transformations, linear, affine, and perspective transformations
- Lighting/illumination
- Clipping of lines and polygons
- Vertex arrays, triangle strips, display lists

Syllabus

The Rendering Pipeline (2)
- Scan conversion of lines and polygons
- Shading and interpolation
- Texture mapping

The Rendering Pipeline (3)
- Modern hardware features
- Vertex shaders / pixel shaders, etc.
Syllabus

Color and reflection
- Color spaces and tristimulus theory
- Physical reflection models

Shadow Algorithms
- Shadow volumes and shadow maps

Ray-tracing, Global illumination
- Only if there is time

The Rendering Pipeline – An Overview

Wolfgang Heidrich
3D Graphics

**Modeling:**
- Representing object properties
  - *Geometry*: polygons, smooth surfaces etc.
  - *Materials*: reflection models etc.

**Rendering:**
- Generation of images from models
  - *Interactive rendering*
  - *Ray-tracing*

**Animation:**
- Making geometric models move and deform

---

**Rendering**

**Goal:**
- Transform computer models into images
- May or may not be photo-realistic

**Interactive rendering:**
- Fast, but until recently low quality
- Roughly follows a fixed patterns of operations
  - Rendering Pipeline

**Offline rendering:**
- Ray-tracing
- Global illumination
Rendering

Tasks that need to be performed (in no particular order):
- Project all 3D geometry onto the image plane
  - Geometric transformations
- Determine which primitives or parts of primitives are visible
  - Hidden surface removal
- Determine which pixels a geometric primitive covers
  - Scan conversion
- Compute the color of every visible surface point
  - Lighting, shading, texture mapping

The Rendering Pipeline

What is it? All of this:
- Abstract model for sequence of operations to transform a geometric model into a digital image
- An abstraction of the way graphics hardware works
- The underlying model for application programming interfaces (APIs) that allow the programming of graphics hardware
  - OpenGL
  - Direct 3D

Actual implementations of the rendering pipeline will vary in the details
The Rendering Pipeline

Discussion

**Advantages of a pipeline structure**

- Logical separation of the different components, modularity
- Easy to parallelize:
  - Earlier stages can already work on new data while later stages still work with previous data
  - Similar to pipelining in modern CPUs
  - But much more aggressive parallelization possible (special purpose hardware!)
  - Important for hardware implementations!
- Only local knowledge of the scene is necessary
Discussion

Disadvantages:

- Limited flexibility
- Some algorithms would require different ordering of pipeline stages
  - Hard to achieve while still preserving compatibility
- Only local knowledge of scene is available
  - Shadows
  - Global illumination

Coming Up...:

Wednesday, Friday:
- More details on the on the rendering pipeline

Next Week:
- Geometric transformations
Your Tasks Until Next Monday

**Assignment 0**
- Refresher of linear algebra
- Set up programming environment on lab computers

**Labs start today!**
- TAs can help with computer setup for A0

**Reading (in Shirley: Introduction to CG)**
- Math refresher: Chapters 2, 4
  - Optional (for now): 2.5-2.9
- Background on graphics: Chapter 1

Wolfgang Heidrich