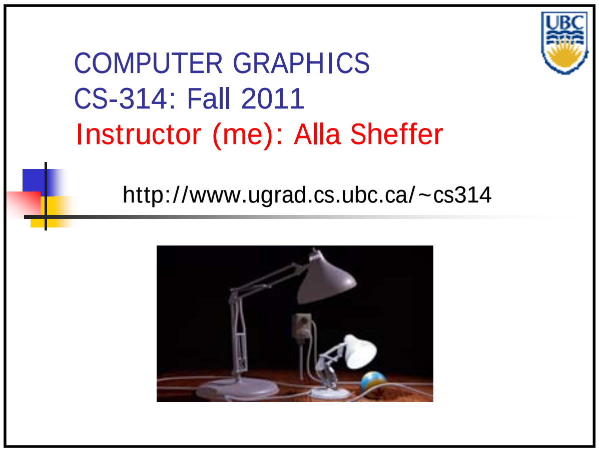
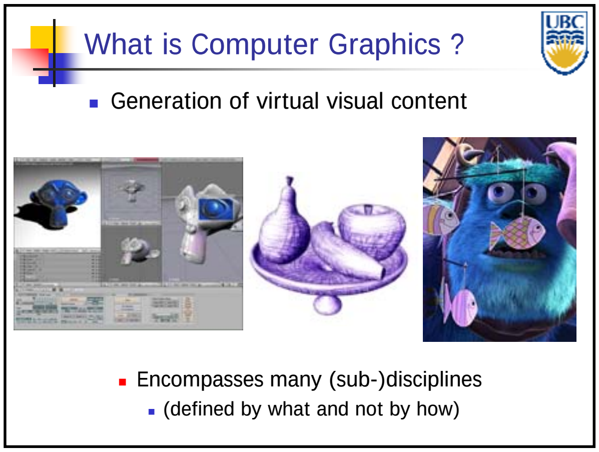


COMPUTER GRAPHICS
CS-314: Fall 2011
Instructor (me): Alla Sheffer

http://www.ugrad.cs.ubc.ca/~cs314



What is Computer Graphics ?

- Generation of virtual visual content

- Encompasses many (sub-)disciplines
  - (defined by what and not by how)

What is CG used for: Movies

Animation

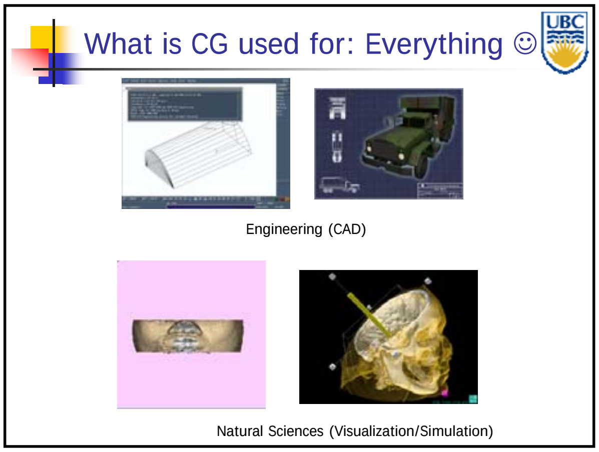
Special Effects (mixed reality)

What is CG used for: Games

PlayStation.2
QUAKE III
REVOLUTION
PlayStation.2
XGIII
EXTREME G RACING
Acclaim

What is CG used for: Digital Media

ALWAYS COOL



What is CG used for: Everything ☺

Engineering (CAD)

Natural Sciences (Visualization/Simulation)

The Science of CG

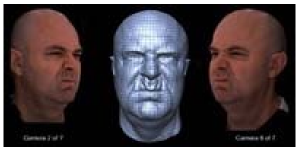
- Content Creation (3D):
  - Modeling - Representing object properties
    - Geometry: polygons, smooth surfaces etc.
    - Materials: reflection models etc.
  - Animation - Making geometric models move and deform
- Rendering - Generation of 2D images from (3D) models
  - Interactive rendering
  - Global (offline) methods: Ray-tracing, etc...
- Imaging – manipulation of 2D images

What This Course Is About

- Fundamental algorithms of computer graphics
- Course Focus: Rendering
  - Why?
    - Critical core graphics component
    - Lots of ideas/methods shared with other CG components
    - Content creation addressed in detail by 4th year courses
      - Modeling – 424 (next year)
      - Animation – 426 (now)

The Science of CG : Brief history

Then (1980s)



Now (trailer from SIGGRAPH'11)

What This Course Is About

- Practice graphical programming (OpenGL)
  - Why? Theory != Practice ☺
  - Learning by doing
  - Graphics is about visuals – testing/applying your knowledge on paper is BORING

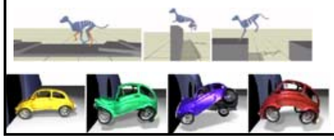
The science of CG (circa 2011)

- Modeling

- Rendering

- Imaging

- Animation



What This Course is **NOT** About

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

Why study CG?

- It is fun – create visually appealing results
- Opens doors to lots of job opportunities
- Gain programming experience
  - Not "just" programming – lots of math, theory, intuition
- Warning: Not at easy course
  - heavy math
  - heavy programming

Policies (boring stuff): http:www.ugrad.cs.ubc.ca/~cs314

Why at UBC?

- Top graphics research group

Teaching Staff

- Instructor: Alla Sheffer
- Office hrs: X651, Mon 2-3pm, Tue 3:30-4:00pm, Wed 2-3pm
  - Contact info:
    - sheffa@cs.ubc.ca
      - **Use discussion board for questions relevant to other students**
- TAs: Mike Boers (mail@mikeboers.com), Mikhail Bessmeltsev (bmpix@cs.ubc.ca)

What next?

- CPSC 424: Geometric Modeling
  - Next year
- CPSC 426: Computer Animation
  - Now (next time in 2 years ☹)

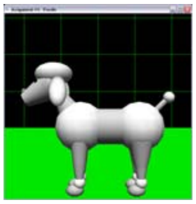
- CPSC 514: Computer Graphics: Rendering
- CPSC 524: Computer Graphics: Modelling
- CPSC 526: Computer Animation
- CPSC 533B: Animation Physics
- CPSC 533C: Information Visualization

Course Information

- Up-to-date information:
  - http://www.ugrad.cs.ubc.ca/~cs314
    - updated often, reload frequently
  - Discussion board (follow link from course home page) – **send request to moderator to get access**
  - I assume that once information is posted on board or web-page students know it
    - within 2 workdays

More Info

- Programming prereq
  - CPSC 221 or CPSC 260+EECE 320
  - Good knowledge of C++
- Math prereq
  - MATH 200 (Calculus III)
  - MATH 221 (Matrix Algebra)



Important Dates

- Assignment 1 due: Sep 23
- Assignment 2 due: Oct 14
- Assignment 3 due: Nov 4
- Assignment 4 due: Dec 2

- Midterm 1: Oct 20
- Midterm 2: Nov 10

Lectures/Labs

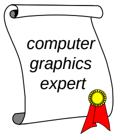
- Lectures: Tue/Thu 2-3:30
- Labs:
  - Wed 12-13, Thu 15:30-16:30
  - Example problems in spirit of written assignments and exams + help with programming assignments
- Attendance not a MUST ...but Strongly recommend that you attend both

Course Organization

- Programming assignments:
  - C++, Windows or Linux
    - **Tested on department Linux machines**
  - OpenGL graphics library / GLUT for user interface
- **Face to face grading in lab**
  - Opportunity to show all the “cool” extra stuff
  - Test that you do know what every piece of your code does
- Hall of fame – coolest projects from 2002 on

Grading

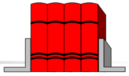
- **Programming Assignments:** 40%
  - OpenGL “Hello World” (5%) – **out now**
  - 3D Transformations – modeling/animation (10%)
  - Rendering pipeline (10%)
  - Ray tracing ++ (15%)
- **Theory Homework:** (5%)
  - 5-6 assignments with pass/fail grade
- **Two Midterms:** (25%)
  - 12% +13%
- **Final Exam:** (30%)



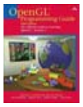
Late Work

- 3 grace days
  - for unforeseen circumstances
  - strong recommendation: don’t use early in term
  - handing in late uses up automatically unless you tell us
- Exception: severe illness or crisis, as per UBC rules
  - MUST
    - Get approval from me ASAP (in person or email)
    - Turn in proper documentation

Literature (optional)



- Fundamentals of Computer Graphics
  - *Third edition (second is OK too – but note syllabus changes)*
  - Peter Shirley, A.K. Peters



- OpenGL Programming Guide
  - J. Neider, T. Davis and W. Mason, Addison-Wesley

Basics of Computer Graphics: Rendering Pipeline

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
    - http://www.xmission.com/~nate/opengl.html

Rendering

Goal:

- Transform (3D) computer models into images
- Photo-realistic (or not)

Interactive rendering:

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

Offline rendering:

- Ray-tracing
- Global illumination

Plagiarism and Cheating

- Short Summary: Don't cheat
  - Homework and programming assignments 1 to 3 are individual work
  - Programming assignment 4 can be done in pairs
  - Can discuss ideas (including on DB), browse Web
  - But cannot copy code or answers
    - If you REALLY think using a source is OK cite it
- **Must** be able to explain algorithms during face-to-face demo
  - or no credit for that assignment, possible prosecution

Rendering Tasks (no particular order)

- Project 3D geometry onto image plane
  - Geometric transformations
- Determine which primitives/parts of primitives are visible
  - Hidden surface removal
- Determine which pixels geometric primitive covers
  - Scan conversion
- Compute color of every visible surface point
  - Lighting, shading, texture mapping

Rendering Pipeline

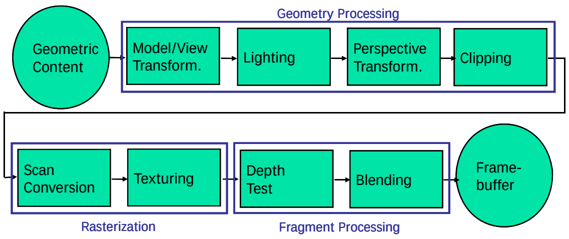
- What is it? All of this:
  - Abstract model - sequence of operations to transform geometric model into digital image
  - Abstraction of how graphics hardware works
  - Underlying API (application programming interface) model for programming graphics hardware
    - OpenGL
    - Direct 3D
- Actual implementations vary

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

The Rendering Pipeline



(Tentative) Lecture Syllabus

- Introduction + Rendering Pipeline (week 1)
- Transformations (week 2/3)
- Scan Conversion (week 4/5)
- Clipping (week 5)
- Hidden Surface Removal (week 6/7)
- Review & Midterm (week 7)
  - Midterm: Oct 20
- Lighting Models (week 8)
- Texture mapping (week 9/10)
- Review & Midterm (week 10)
  - Midterm: Nov 10
- Ray Tracing (week 11)
- Shadows (week 11/12)
- Geometric Modeling (week 12/13)
- Review (last lecture)

Discussion

Advantages of pipeline structure

- Logical separation of 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

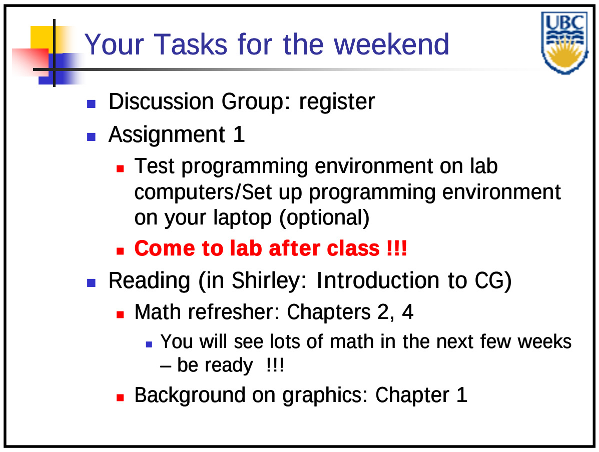
Coming Up...:

Tue:

- More details on rendering pipeline

Next Week:

- Geometric transformations



Your Tasks for the weekend

- Discussion Group: register
- Assignment 1
  - Test programming environment on lab computers/Set up programming environment on your laptop (optional)
  - **Come to lab after class !!!**
- Reading (in Shirley: Introduction to CG)
  - Math refresher: Chapters 2, 4
    - You will see lots of math in the next few weeks – be ready !!!
  - Background on graphics: Chapter 1