Goals

- **Systems:** Write complex 3D graphics programs (real-time scene in OpenGL, offline raytracer)
- **Theory:** Mathematical aspects and algorithms underlying modern 3D graphics systems
- This course is *not* about the specifics of 3D graphics programs and APIs like Maya, Alias, DirectX but about the concepts underlying them.

---

Instructor

Ravi Ramamoorthi  
[http://www.cs.ucsd.edu/~ravir](http://www.cs.ucsd.edu/~ravir)
- PhD Stanford, 2002. PhD thesis developed "Spherical Harmonic Lighting" widely used in games (e.g. Halo series), movies (e.g. Avatar), etc.
- At Columbia 2002-2008, UC Berkeley 2009-2014
- "Monte Carlo denoising" inspired raytracing offline, real-time
- At UCSD since Jul 2014: Director, [Center for Visual Computing](https://www.cs.ucsd.edu/~ravir/)
- [https://www.youtube.com/watch?v=qpycXqX0e7I](https://www.youtube.com/watch?v=qpycXqX0e7I)
- Have taught Computer Graphics 10+ times
- Computer Graphics online MOOC (CSE 167x) has had 100,000+ registrations, 500,000 video views. Finalist for two inaugural edX Prizes. Will use edX edge, auto-feedback

---

Course Staff

- Ravi Ramamoorthi
- Teaching Assistants:
  - Lifan Wu (will also maintain feedback servers)
  - Tiancheng Sun
  - Alex Kuznetsov

---

Why Study 3D Computer Graphics?

- Applications (discussed next)
- Fundamental Intellectual Challenges

---

MOOC Introductory Video

Some content inspired by Pat Hanrahan from Stanford’s CS148
Entertainment

Movies: Brave, Pixar 2012

Games: Halo 3, Bungie 2007

Lighting Simulation

Interior Design

Automobile Visualization

Computer Aided Design

Mechanical CAD
Architectural CAD
Electronics CAD
Casual Users

Interiors Professional

Google Sketchup

Visualization: Science and Medicine

Visible Human Project: University of Hamburg

Virtual Reality

- VR for design and entertainment
- Simulators: Surgical, Flight, Driving, Spacecraft
Digital Visual Media

- From text to images to video (to 3D?)
- Image and video processing and photography
- Multimedia computers, tablets, phones
- Flickr, YouTube, WebGL
- Real, Virtual Worlds (Google Earth, Second Life)
- Electronic publishing
- Online gaming
- 3D printers and fabrication

Why Study 3D Computer Graphics?

- Applications
- Fundamental Intellectual Challenges
  - Create and interact with realistic virtual world
  - Requires understanding of all aspects of physical world
  - New computing methods, displays, technologies
- Technical Challenges
  - Math of (perspective) projections, curves, surfaces
  - Physics of lighting and shading
  - 3D graphics software programming and hardware

3D Graphics Pipeline

HW 1: Transformations (Jan 23)
Place objects in world, view them
Simple viewer for a teapot

HW 3: Curves (Feb 27)
Beziers and B-Spline curves
To model and draw objects

Curves for Modeling

Rachel Shiner, Final Project Spring 2010

3D Graphics Pipeline

HW 1: Transformations (Jan 23)
Place objects in world, view them
Simple viewer for a teapot

HW 2: Scene Viewer (Feb 15)
View scene, Lighting and Shading
(with GLSL programmable shaders)

HW 3: Curves (Feb 27)
Beziers and B-Spline curves
To model and draw objects

HW 4: RayTracer (Mar 18)
Realistic images with ray tracing
Two basic approaches: rasterize
And raytrace images (HW 2-4)
Image Synthesis Examples

Collage from 2007

Logistics
- Website [http://viscomp.ucsd.edu/classes/cse167/wi19](http://viscomp.ucsd.edu/classes/cse167/wi19) has most of the information (look at it carefully)
- We will be leveraging MOOC infrastructure in a SPOC
  - Please sign up for account at edX edge, join course: DEMO
  - edX edge is compulsory for most assignments, feedback systems
  - Optional for video lectures (class may differ a bit, more), problems
  - Must still submit "official" CSE 167 assignment (see website)
  - Please do ask us if you are confused; we are here to help
- No required texts; OpenGL programming guide, GLSL optional
- Office hours: 12pm – 1pm on Tue, 11am – 12pm on Thu
  - See website for sections, TA office hours. **Sign up for sections!**
- Course newsgroup on Piazza
- Website for late, collaboration policy, etc
- Questions?

This is a Modernized Course

- Modern 3D Graphics Programming with GPUs
  - Modern OpenGL (3+), GLSL 330 core
  - Real-time feedback servers for all homeworks
- GLSL + Programmable Shaders from HW 1
- Should be very portable, but need to set up your environment, compilation framework (HW 0)

Innovation: Feedback Servers

- Feedback/Grading servers for all homeworks
- Submit images and/or code, compare to original
  - Program generates difference images, report url
  - Can get feedback multiple times, submit final url
- All (except curves homework 3) run on edX edge
  - “Feedback” not necessarily grading
    - Can run extra test cases, look at code, grade fairly
    - But use of feedback servers/edX edge is mandatory
- Will test out immediately with HW 0 images
  - HW 1 - 2 will have both code and image feedbacks
  - Can use any (laptop/desktop) computer. We also try to have the basement labs fully set up

Online Lectures

- Online lectures and screencasts for most course:
  - (with English and Chinese/ Subtitles [courtesy XuetangX])
  - Review for CSE 167 (but still have regular classes)
  - For general interest (share with non-CS 167 students)

- Originally recorded in 2012 for MOOC offering
  - CAVEAT: Does not include all material (curves)
  - Was updated in 2017 for more recent OpenGL
  - Same as video lectures on edX edge (some errata)

- Currently view lectures as complementary
  - Hence, viewing them optional (e.g. miss a class)
  - Please note caveats; "official" CSE 167 is in class

- May separately have UCSD screencasts
### Workload
- Lots of fun, rewarding but may involve significant work
- 4 programming projects (+HW 0); almost all are time-consuming (individual except HW 4). START EARLY !!
- Course will involve understanding of mathematical, geometrical concepts taught (tested on midterm)
  - No final; will do a take-home small assignment instead
- Grade mostly programming, weights on website
  - Ignore weighting on edX site; we weight as on CSE 167 site
- Prerequisites: Solid C/C++/Java/Python programming background. Linear algebra (review on Thu) and general math skills. No knowledge of graphics/OpenGL needed.
  - Should be able to pick up C/C++, and look up some OpenGL
- Should be a difficult, but fun and rewarding course

### CSE 167 is only a first step
- If you enjoy CSE 167 and do well:
  - In Spring: CSE 190 (VR course; Schulze), CSE 291 (Computational Photography; Ochoa)
  - Next winter: CSE 165 (3DUI), 169 (Animation)
  - Graduate: CSE 274 (Topics), many 291s

### To Do
- Look at website
- Various policies for course. E-mail if confused.
- Sign up for edX edge, Piazza, etc.
- Skim assignments if you want. All are ready
- Assignment 0, Due Jan 16 next week (see website). [both parts needed, total 10 points]
- Set up compilation framework in HW 0, feedback
- Any questions?

### History
- Brief history of significant developments in field
  - End with a video showcasing graphics

The term Computer Graphics was coined by William Fetter of Boeing in 1960
First graphic system in mid 1950s USAF SAGE radar data (developed MIT)

### From Text to GUIs
- Invented at PARC circa 1975. Used in the Apple Macintosh, and now prevalent everywhere.
### Drawing: Sketchpad (1963)
- Sketchpad (Sutherland, MIT 1963)
- First interactive graphics system (VIDEO)
- Many of concepts for drawing in current systems
  - Pop up menus
  - Constraint-based drawing
  - Hierarchical Modeling

### Paint Systems
- Nowadays, image processing programs like Photoshop can draw, paint, edit, etc.

### Image Processing
- Digitally alter images, crop, scale, composite
- Add or remove objects
- Sports broadcasts for TV (combine 2D and 3D processing)

### Modeling
- Spline curves, surfaces: 70s – 80s
- Utah teapot: Famous 3D model
- More recently: Triangle meshes often acquired from real objects

### Rendering: 1960s (visibility)
- Roberts (1963), Appel (1967) - hidden-line algorithms
- Sutherland (1974) - visibility = sorting

### Rendering: 1970s (lighting)
- 1970s - raster graphics
  - Blinn (1974) - curved surfaces, texture
  - Catmull (1974) - 2-buffer hidden-surface algorithm
Rendering (1980s, 90s: Global Illumination)

- early 1980s - global illumination
  - Whitted (1980) - ray tracing
  - Goral, Torrance et al. (1984) radiosity
  - Kajiya (1986) - the rendering equation

History of Computer Animation

- 10 min clip from video on history of animation
  - http://www.youtube.com/watch?v=LzZwiLUVaKg
- Covers sketchpad, animation, basic modeling, rendering
- A synopsis of what this course is about
- (watch offline if short on time)