Syllabus for CSC/CPE 473: Advanced Rendering Techniques

Professor: Zoë Wood
Office: Building 14, room 209
Phone: 756-5540
office hours: TTh 10:30-12
lecture: TTh 12:10-1:30pm (14-247) **lab:** 1:40-3:00pm (14-255)
email: zwood@calpoly.edu

General: This course will cover advanced rendering techniques; global illumination via distributed ray tracer using Monte Carlo sampling, exposure to photon mapping and point based color bleeding and a brief exploration of VR (using Unity and Google cardboard). The majority of the course will be focused on students building their own software renderer (distributed ray tracer with Monte Carlo sampling) over an 8 week period. This is an opportunity to explore individual software development as you incrementally add features. Each student's program must be implemented using C++ and will be expected to be reasonably designed using appropriate OO techniques and expected to pass established unit tests to demonstrate correctness. This course requires substantial math and programming skills. ***Ultimately the goal of this course is to allow you to enhance your individual software engineering skills while building a large piece of software intended to make pretty pictures that more accurately simulate light transport (then OpenGL local lighting models).*** *This software development process requires you to consider data structures, performance and practice translating mathematics into C++ code*. During this quarter you are expected to talk to one another and learn from one another as a community of scholars whom never copy code blindly.

Course Objectives: By the end of the quarter students will:

- Understand the basics of ray tracing
- Be able to correctly implement ray-sphere, ray-plane, ray-triangle intersections (with correctness demonstrated via unit testing**)
- Understand and be able to apply basic object oriented design in order to create a well structured larger software project (stochastic sampled ray tracer)
- Be able to program basic data structures to represent geometric objects in a scene (sphere, planes, triangles), including the application of transforms (translate, scale, rotate) and scene objects such as lights, and the camera
- Be able to understand and implement shadow feelers to produce shadows in a software render (ray tracer) (**)
- Be able to understand and implement ray traced rendering of reflective surfaces (**)
- Be able to understand and implement ray traced rendering of refractive surfaces (**)
- Be able to understand and implement a 'virtual camera' (**)
- Be able to understand the basics of Monte Carlo sampling in order to implement an approximation of global illumination via a distributed ray tracer implementation
- Be able to understand and implement a few BRDF (Bi-directional radiance distribution functions) to simulate the reflection of light (**)
- Practice translating mathematics into C++ programs
- Exposure to and implementation of some subset of advanced topics: texture mapping, anti-aliasing, depth of field, motion blur, spatial data structures, parallel programming, point based color bleeding, photon mapping, path tracing, radiosity, scripting to produce animation, real-time ray traced rendering via writing results to framebuffers in OpenGL, basics of VR using Unity SDK for Google cardboard, etc.

Assignments:

- 2 mid-term exams (20% of final grade – 10% each)
- Lab (unit tests) (7% of final grade – 1% each)
- Code reviews (5% of final grade – 2.5% each)
- Ray tracer programming assignment (broken into 5 parts) (55% of final grade)
- One final VR Unity Google cardboard project (10% of final grade)
- Participation (3% of final grade)
    - attend class/ talk in class or office hours interaction

Please see the program description for final details. **There is a strict late policy for all assignments** – if your program is late you will lose: -20% within first 24 hours after deadline, -40% within 48 hours, -100% after 48 hours

Highly recommended Text: "Physically Based Rendering" by Matt Pharr and Greg Humphreys

Class style and logistics
I expect you to participate in class and engage with the class material (studies suggest that taking longhand notes is one of the better ways to guarantee your engagement with the material in class)[1] I also expect you to form a community of scholars for the duration of the quarter (and hopefully longer). My teaching style is very interactive – if you want to know more about why see ([2]). **Laptops have been shown to be distracting in lecture[3] and are not allowed unless specified (or a specific exception is negotiated) -- same for cell phones.**

Cheating: All your **work you hand in must be your own work, unless otherwise specified**. If your program or parts of your program are plagiarized from another student or unapproved source, you will fail the course and a letter will be put in your file with Cal Poly Judicial Affairs.

(language from CFA): Due to stalled negotiations after seven years of almost no salary increases, the faculty will strike April 13-15 and April 18-19 if no settlement is reached before that date. Please expect that in many, if not all of your courses, faculty will cancel lecture and lab meetings, refrain from answering email, cancel office hours, and stop all other job duties. Faculty care deeply about their students, and we will work with you to ensure that you still meet course requirements during the strike. Your professors, who have been deprived of adequate raises for eight years, will take this action only as a last-chance option; they risk losing salary. This will be the largest faculty strike in U.S. history; if you would like to learn more, please ask me and visit www.calfac.org, CaliforniaFacultyAssociationCalPolySLO on Facebook or @CFA SLO on Twitter.

If you'd like to help, please call or email Chancellor White (562-951-4738 or twhite@calstate.edu) and let him know: "My name is ______; I am a Cal Poly student, and I want you to know that my professors deserve a 5% raise! The CSU has the money; it is a matter of priorities. I cannot get the education I deserve unless my teachers are paid what they deserve." And please feel free to join us on the picket lines if/when the strike occurs.

---

[1] http://www.theatlantic.com/technology/archive/2014/05/to-remember-a-lecture-better-take-notes-by-hand/361478/
[2] Applying the Seven Principles for Good Practice in Undergraduate Education" (1991) Chickering and Gamson
[3] http://www.yorku.ca/ncepeda/laptopFAQ.html

The following schedule for the lectures and assignments is very tentative.

Week 1	3/29/16	Introduction & orientation	
	3/31/16	Academic holiday	*P1: (initial) OO code design & parsing*
Week 2	4/5/16	Light, rays, the camera & spheres	
	4/7/16	plane intersections, shadows & profiling	
		Unit tests due 1 (ray-plane + ray-sphere)	**Ray tracer part 1 due**
Week 3	4/12/16	Shading models and reflection <radiance>	*Unit tests due 2 (reflected ray)*
	4/14/16	***Transforms (geometry and cameras)***	*Unit tests due2 ?*
			Ray tracer part 2 due
Week 4	***4/19/16***	***Transforms (geometry and cameras)<tri intersecti***	*Unit tests due 3 (camera rays+ triangles)*
	4/21/16	Refractions I	*Unit tests due 4 (shading model)*
		Code review 1 (A-G)	
Week 5	*4/26/16*	Refractions II & Anti-aliasing	*Unit tests due 5 (refraction)*
		Code review 1 (G-Y)	
	4/28/16	**Midterm 1**	
		<cancel lab for midterm>	
Week 6	*5/3/16*	Optimizations & advanced (distribution RT + textures)	**Ray tracer part 3 due**
		Unit tests due 5 – anti-aliasing	
	5/5/16	Global Illumination - Monte Carlo (spatial data + boxes)	
		Code review 1 (A-G)	
Week 7	5/10/16	Global Illumination - Photon mapping <spatial + MC? Or gamma correction?>	
		Code review 1 (G-Y)	**Ray tracer part 4 due**
	5/12/16	Global Illumination - Point based color bleeding	
Week 8	5/17	Non-photo realistic rendering	
		Unit tests due 6 – anti-aliasing and performance	
	5/19	Shaders + Environment maps	
		Unit tests due 7 spatial and performance	**Ray tracer part 5 due**
Week 9	5/24	**Midterm 2**	
	5/26	Basics of VR – Unity and Google cardboard	
Week 10	5/31	Challenges in VR rendering	**Final project check-in**
	6/2	Future rendering	**Final project check-in**
Final	Tue 6/7	**Final 1:10-4pm**	**Final Projects demo**

Required unit tests – to be demonstrated in lab will likely include:

1) ray - plane and ray-sphere intersections

2) reflected ray computation (for sphere and plane) <ray direction not color returned>

3) camera rays (for ‘virtual camera’ + ray-triangle intersection)

4) shading model (color computation for given material, ray, and object) <including returned color for a reflected ray - exact BRDF TBD>

5) refraction (for sphere and plane) <both returned color and direction - assuming simple shading model TBD>

6) Anti-aliasing + measure performance as benchmark

7) spatial data structure + measure performance as benchmark