CS 410 Lecture 01: Introduction

August 25, 2020
CS 410: Computer Graphics

Professor: Ross Beveridge
Room 348 CSB
(970) 491-5877
Office Hours (Zoom):
  4:45 – 5:30 Tue/Thur
  or by appointment
Ross.Beveridge@colostate.edu

Teaching Assistants:
Yongxin Liu
Vishal Kuvar
Office Hours: TBD

Web Site: http://www.cs.colostate.edu/~cs410
1-slide Overview

• This is what you will learn to make

Charumathi Chandrasekaran

Ben Randall

• But you will have to work hard to get there
Another Example – Fall 2012

What can you say about this image?

In other words, what processes are at work – simulated – to create what you are seeing?
More Recent Example 2018

Isaac Law CS 410 Fall 2018
In This Class

• You will build a ‘ray tracer’.
• Input
  – Scene model (objects, surfaces & materials)
  – Lighting models
  – Camera models
• Output
  – An image showing what the camera would see.
What we will do....

• Program a ray tracer from scratch
• General flow of programming assignments
  – Basics of 3D geometric transformations
  – Basics of 3D object modeling
  – Simple ray casting with global geometry
  – Geometric generality – relative placement
  – Illumination and surface reflectance
  – Multi-bounce tracing with reflection/refraction
• Each assignment builds on the one before
  – 20 – 30 hours each
    • Following good software practices is key
In the Process ...

• Master basics of linear alg. as geometry
  – Dot (i.e. inner) products (of vectors)
  – Cross products (of vectors)
  – Homogeneous coordinates
  – Affine & perspective transformations
  – Matrix multiplication
  – Orthonormal matrices

• Expertise of value beyond Graphics!
... 3-D Geometric Intuition ...

• Expertise requires three things.
  – Linear algebraic objects and operations
  – Visual intuition – “seeing” what your specifying
  – Quick and easy shifting back and forth

• Most obvious examples.
  – Move the camera back and a bit left
  – Place object A on top of object B
  – ....
… and of course also …

• Learn about light
  – Color spaces
  – Reflection & refraction

• Surface properties
  – Lambertian (Matte) reflection
  – Specular reflection
  – Hybrid reflection models
Resources Overview

• Public Website / Lectures
  – General concepts
  – Illustrative Examples

• Canvas
  – Assignments, exams, etc.

• SageMath Notebooks (CoCalc)
  – Interactive symbolic math and working code
  – With 3D illustrations of key geometry

• Optional Textbook – a good reference
CS 410 teaches students how to program a computer to generate photo-realistic images. The general idea is that given a scene model, a sensor model and a viewpoint, one should be able to create the same image that a camera would for that scene and viewpoint. Scene models include 3D object models and light sources. Objects models are composed of surfaces, and include both geometry (where is the surface?) and material properties (what is it made of? What color is it?).

This course will emphasize geometric objects and transformations, perspective projections, lighting and reflectance models, shading models, and 3D curves and surfaces. Students will design and implement a ray tracing program from scratch, thereby becoming intimately familiar with the sensor, lighting and object models described above. Perhaps most importantly, students may come to more fully appreciate the power of linear algebra.

Instructor:
Ross Beveridge
Office: 348 CS Building
Office Hours: TBD
Email: Ross.Beveridge@colostate.edu

GTAs:
Yongxin Liu
Office Hours: TBD
Email: Yongxin.Liu@colostate.edu
Vishal Kuvar
Office Hours: TBD
Email: Vishal.Kuvar@colostate.edu

Lecture Time and Place:
3:30-4:45, Tue, Thur, Online (Zoom)

News:
CS410 Lectures will be through Zoom with links posted within CANVAS. (8-21-2020)
Welcome CS410 students for the Fall 2020 run of CS410. (8-20-2020 Ross)
This site is coming online from 8-20-2020 to 8-24-2020 so be patient as parts are incomplete. (8-20-2020 Ross)
Resources (2)

• Private Website – CANVAS
  – Used for Quizzes, Assignments, Grading …

• Office hours
  – Mine:
    • Tuesday 4:45 to 5:30 PM (after lecture)
    • By appointment
  – GTAs
    • TBD
  – Don’t be a stranger!
First 410 Notebook Illustrating SageMath on CoCalc

This is a tiny step down the path we will take this Fall using SageMath through Jupyter - and hosted by CoCalc - to illustrate key concepts during the course of CS410. Today, the very basics of formulas and linear algebra.

Ross Beveridge, August 25, 2020

In [27]:
%display latex
latex.matrix_delimiters(left='[', right=']')
latex.vector_delimiters(left='[', right=']')

To begin, you can designate variables that will be treated as symbolic as opposed to values.

In [28]: var('a','b','c','d','x','yy')

Out[28]:

\( (a, b, c, d, x, yy) \)
Resources (4) Optional Textbook

• Adopted in 2011 at CSU
• Virtues: Excellent
  – Focuses on Ray Tracing
  – Consistent terminology
  – Well staged mathematics
  – E-format available
• Either 3rd or 4th edition

Do not expect lectures to follow textbook specifics nor the textbooks specific mathematical conventions.
Resources (5) Blender

In CS410 – A Tiny Glimpse into a Massive Tool – Still valuable
Course Rules

• Be on time – if attending through Zoom
  – Class starts at 3:30
  – Lectures are recorded, live attendance optional

• Ask & Answer questions
  – Interaction is perhaps more important now!

• Professional behavior at all times
  – Be polite & respect others
  – Be sensitive to Zoom manners

• See the Syllabus!
Major Activities / Grading

- **Quizzes / Exercises**
  - 10% of your grade – there to help you study
- **Programming Projects (~5)**
  - 50% of your grade total
  - Generally 10% each
- **Midterms (2)**
  - 20% of your grade (total)
  - 10% of your grade (each)
- **Final exam**
  - 20% of your grade

Quizzes and exams using CANVAS.
Possible exercises using CoCalc.
Policies

• Assigned work is done alone
  – No joint projects
  – No open note exams
  – No taking code from the internet
  – Follow the department academic integrity policy

• All work done on time
  – No late period – deadlines are deadlines
    • Multiple submissions OK – last is graded
  – When given two (or three) weeks, start right away!

• Exceptions: unforeseeable circumstances
Systems and Tools

- You may be surprised …
- You will not learn a complex and/or arcane API for a giant graphics package
- Instead, you will build your ray tracer from scratch
  - C++ is recommended
  - Java acceptable
  - No Python (sorry)
- Re-discover lessons from CS253 and CS314
This course is NOT...

• A course in OpenGL
  – OpenGL shaders are complex & detailed
  – Ray Tracing will become the dominant paradigm
    • Thanks to GPUs and parallel architectures
• A course in using any other Graphics API

This is a course about the mathematics and algorithms underlying Ray Tracing. It should make you a better programmer, improve your linear algebra, and prepare you to study computer vision (which is where the jobs are…)

8/25/20
Systems - Requirements

• Ray Tracer Mechanics
  – Must run on CS Linux Machines
  – Tools for viewing models will be provided
  – Tools for viewing images will be provided

• Test your code in the Department!

• Speed a low priority until ….
  – Long run-times interfere with grading.
Let us walk quickly through some of the key concepts of this course as a means of setting up a context for what is to come.
What is an image?
What is an image? (cont)

• An array of values
  – Intensities (if gray scale)
  – Triples of red, green & blue (color)

• This image is a part of the previous image
  – Where?
How is an image formed?
Step 1: Light Source

- Lights Sources have
  - Intensities
  - Color
  - Positions

- Bonus: atmospheric effects
Step 2: Reflection

[Image of a light ray reflecting off a mirror]
Step 3: Projection
How (artificial) images are formed

- Light Source
- Coordinate System
- Geometric Objects
- Sensor

*pinhole camera*

$d = \text{distance from hole to sensor}$
Now mimic light...

From source to object to sensor
... or ...

...from source to object to object to sensor
... or ...

...you get the point.
True – But Backwards!

• Light sources emit a LOT of light rays
  – Most of them never strike your camera
• And you might have multiple light sources
  – In fact, you usually do, and the light intermingles
• So following every light ray is expensive

Instead, work backwards: start at the surface of the lens, and figure out where the light came from … all the way back to the light source.
In other words
What can this approach do?

A photorealistic rendered image created by using POV-Ray 3.6
CS 410
Project done by Kyle Olson in Fall of 2014.
Real-time Ray Tracing 2012

Nvidia’s Kepler real-time raytracing and destruction (1080p) - The Verge

45,000 views

https://www.youtube.com/watch?v=w9SH8xlgzol
Ray tracing stole the show from Nvidia’s new RTX graphics cards

By Kevin Lee  an hour ago  Software

Setting the tone for the next decade of computer graphics
Real-Time Ray Tracing 2019

All 11 Games That Support NVIDIA’s RTX Ray Tracing Technology

24,522 views

AllAboutPerspective
Published on Feb 14, 2019

©J. Ross Beveridge 2020
Here are all the games that support Nvidia’s RTX ray tracing

By Jon Martindale
July 30, 2020

1. Amid Evil
2. Battlefield V
3. Call of Duty: Modern Warfare
4. Control
5. Deliver Us the Moon
6. Justice Online
7. JX3
8. Mechwarrior 5: Mercenaries
9. Metro Exodus (and The Two Colonels DLC)
10. Minecraft
11. Quake II RTX
12. Shadow of the Tomb Raider
13. Wolfenstein: Youngblood
What skills will we need?

• Lots of Math
  – Linear Algebra (Matrices, Vectors, Dot Products)
  – Discrete representations (Images)

• A little Physics
  – Lighting/Energy Transfer
  – Color reflections/refraction

• A little Mechanical Engineering
  – CAD/CAM, Solid Models

• A whole bunch of Computer Science
What do I expect?

• Familiarity with vectors and matrices
  – Mechanics, e.g. multiplication
  – Basic ideas (perhaps rusty)

• Programming ability
  – You know how to design, implement & test thousand line plus programs
    • Test-first development
    • Design patterns
    • Object-oriented designs

• Willingness to work (hard)
Quick Who Does What?

• I will describe graphics principles
  – Mathematics (e.g. projection)
  – Physics (e.g. reflections)
• I will describe some algorithms
  – E.g. ray/polygon intersection
• We will discuss/practice concepts.
• You will convert ideas into code.
More About SageMath/CoCalc

• Yes – a student subscription is required
• Yes – that is a $14 cost to each student
• Yes – used extensively in CS 410
• Yes – rests on top of Python and Jupyter
• Yes – I will show you just a bit today

Expect an email invitation to Join the CS410 Course on the CoCalc site in the next day.
The End