



UNIVERSITY OF MINNESOTA

Driven to Discover®

Course Overview

CSCI 5980: Game Engine Architecture

Evan Suma Rosenberg | CSCI 5980 | Spring 2026

This course content is offered under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International license.



Welcome!

If you are not registered for the course and would like to request a permission number, please fill out the following form.

<https://z.umn.edu/5980permission>

Introduction



Evan Suma Rosenberg

they/them

Email: suma@umn.edu

Office: 323 Shepherd Labs

Who am I?

Professional history

- I started working with virtual reality technology as an undergraduate in 2003.
- Ph.D. in 2010 with a research focus on locomotion in virtual environments.
- Research professor at the University of Southern California Institute for Creative Technologies until 2018.
- Joined the University of Minnesota in 2018.

Who am I?

Game engine history

- I got my first taste of game engine programming as undergraduate, when I dove into Valve's Source engine code and modified it for a research project.
- Early grad school projects included hacking 3D Game Studio and the Source engine to work with professional-grade VR head-mounted displays and motion tracking systems.
- I did a summer internship at the Naval Research Lab in 2006, where I hacked apart a 1990s-era AR/VR software framework and replaced its outdated graphics code with OpenSceneGraph.
- After that, I wrote my own custom VR engine from the ground up in C++ using OpenSceneGraph, which was used by multiple research groups until Unity 4.0 arrived on the scene (~2012).
- When I started teaching CSCI 4611 a few years ago, I wrote GopherGfx from scratch, a high-level WebGL graphics library in TypeScript. Some of you may have used it!

Who am I?

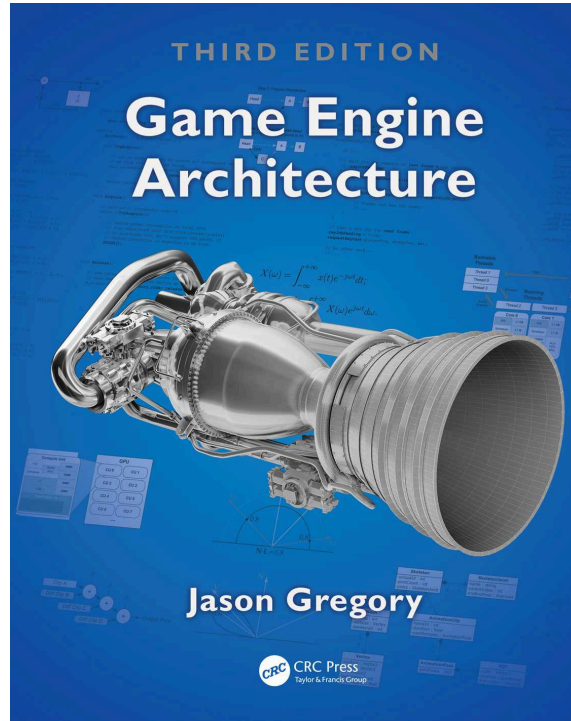
I also play video games. A lot.

- First video game system: Atari 2600 (1980s)
- Most hours logged on any platform: World of Warcraft (2005-2008)
- Most hours logged on Steam: Baldur's Gate 3
- Most recent game completed: Hades 2
- Still in progress: Clair Obscur: Expedition 33

In this course, you will learn to:

- Identify, define, and solve problems related to the design and programming of modern game engines.
- Understand core principles and architectural patterns for real-time interactive systems, such as modularity, control flow, data organization, and resource management.
- Implement a minimalistic game engine in modern C++, demonstrating principled API design, maintainable code structure, and adherence to stated performance goals.
- Use systematic engineering practices, including debugging, instrumentation, and performance analysis.
- Integrate third-party libraries, manage build configuration and dependencies, and design stable engine-facing interfaces.

Textbook



Course Topics

The course is divided into six modules, each of which lasts two weeks.

- Low Level Engine Systems
- Human Interface Devices
- Debugging, Development, and Profiling
- Resources and the File System
- Graphics, Motion, and Sound
- Gameplay Systems

Prerequisites

- CSCI 4611, 5607, or equivalent 3D computer graphics experience
- Familiarity with object-oriented programming (C++, C#, Java, etc.)

2D game programming or scripting in a commercial game engine engine such as Unity and Unreal would be unlikely to provide the foundational knowledge necessary for this course.

Programming Languages and Software

- C++
- Visual Studio Code
- Build Tools: CMake, LLVM, Ninja

We will be developing on Windows. If you need a computer, you can check out a Lenovo GPU laptop with an RTX 4070 graphics card from CSE-IT for the entire semester.

Teaching Modality

- I am planning to teach this course **in-person**.
- All classes will be **recorded** and posted on Canvas.
- I am not currently planning on hybrid instruction.
- If a specific accommodation is necessary, please contact me.

Course Structure

Mondays: Conceptual Lectures

- For one class each week, we will cover lessons in traditional lecture format. These classes will also be supplemented by readings from the textbook.

Wednesdays: Live Programming

- On these days, we will have live, interactive programming sessions with me. You may optionally bring laptops to class to follow along in real-time.

We will generally follow this alternating structure, but the exact schedule may vary based on the content we need to cover.

Programming Assignments

- There will be a total of **five** programming assignments in this class.
- Each assignment will be introduced at the end of a two-week module.
- Please check Canvas for the complete schedule.

Quizzes

- There will be a total of **five** quizzes in this class.
- Each quiz will be scheduled in the middle of a two-week module, starting with Module 2.
- Quizzes will be **take home, open book**, and **written on paper**.
- They can be completed anytime within a three-day window (no time limit).

Exams

- My teaching philosophy is based on the practice of offering more frequent assessments, rather than larger, high-stakes exams.
- There is no midterm or final exam.
- During finals week, there will be an optional quiz that you can take to replace the lowest score on a previous quiz (including a zero if you missed one).

Class Participation

- Some classes will include a participation exercise that need to be completed within one week.
- These exercises are very short and meant to provide deeper engagement with concepts.
- All of the exercises will be submitted online, so you can complete them even if you watch the lecture video asynchronously.
- Completing the participation exercises will count for 10% of your final grade.
- **The lowest two** will be dropped to allow for the occasional missed class.

Grading

- Programs - 50%
- Quizzes - 40%
- Participation - 10%

Grades in this class are based on objective criteria, using clearly defined and transparent rubrics. I will not make grade adjustments based on subjective judgments that may be influenced by personal or implicit biases.

Flexible Late Policy

- Each assignment will have a fixed due date, but everyone will be granted **5 late points** at the beginning of the semester.
- Each late point is worth a 24-hour extension for any programming assignment.
- You are free to allocate them however you want throughout the course.
- For example, you could use all your points for a 5-day extension on a single assignment or spread them around among multiple assignments.
- If you use all your late points, the assignment will be graded with a penalty of 5% for each day it is late.

Frequently Asked Question

If I don't use all my late points by the end of the semester, will I receive bonus points on my final grade?

- **No.** The late policy intended to provide flexibility for missed work.
- This includes legitimate absences such as illness, family crisis, or other factors beyond your control.
- Sometimes, life happens, and we all need a safety net.

Collaboration Policy

- All work submitted for this course is required to be your original work, except where specifically noted in the assignment description.
- In general, you are expected to do your own thinking, your own design, and your own implementation.
- Note that this policy applies to all direct communication and collaboration with other humans, regardless of whether they are enrolled in the class.

The last bullet point originally used the term "people," but I don't want to tempt Roko's basilisk.

Use of Web Resources and AI Tools

- The internet is, obviously, the most extensive source of documentation for working with C++ and software libraries used in the class.
- However, advancements in AI have made it possible to get "answers" without having to search for them.
- Sometimes, the responses from AI are astoundingly deep, comprehensive, and accurate.
- Other times, the output sounds plausible, but it is actually riddled with errors or, even worse, completely misguided.
- These situations may be possible for an expert to detect and correct, but it can undermine student learning.

Use of Web Resources and AI Tools

- In this class, I will not restrict your use of AI tools on programming assignments.
- However, **with great power, comes great responsibility.**
- I expect everyone to approach the output from AI with a critical lens, and fact check their output.
- I also expect everyone to be honest and transparent about how they used AI and other web resources.

Why is this important?

Imagine you are a professional game developer.

The game you worked on is critically acclaimed and nominated for game awards.

Someone on the internet accuses the game developers of using AI tools.

The CEO and/or design lead publicly defends their workflows as human-generated.

Soon after, it turns out this isn't entirely true. There are uses of AI throughout their workflow. It's consistent with what the rest of the industry is doing, but it's poorly documented and ill-defined.

The public conversation and press turns negative, and the game is stripped of some its awards.

AI use threatened the success of the game, the company, and the developers' livelihoods, in what should have been a moment of celebration.

Engadget

The Indie Game Awards snatches back two trophies from Clair Obscur over its use of generative AI

Dec 22 • By Lawrence Bonk



Kotaku

Clair Obscur: Expedition 33 Loses Indie Game Awards After Being Accused Of Lying About AI Use

Dec 22 • By John Walker



Mashable

'Clair Obscur: Expedition 33' stripped of Indie Game Awards due to generative AI use

Dec 20 • By Amanda Yeo



Let's take a look at the syllabus...

<https://github.com/CSCI-5980-Spring-2026/Syllabus>

Getting Help

- **Slack:** This will be our primary method of communication in this course. There are two ways to use it:
 - Post in the general channel. This is fine for general questions about the class, clarification questions about assignments, etc. *You should never post code or solutions in the general channel!*
 - Send me a private message. This will often be the most efficient way to get help with programming questions. If you push your code to GitHub, I will be able to check it out and test it on my local machine.
- **Appointments:** If you would like to meet with me, please drop me a message on Slack and we can schedule a Zoom call or in-person appointment.
- **Drop-in hours:** I will be available for drop-ins in my office and Zoom on Thursdays from 3-4pm.

Respectful Conduct

- At this point in your careers, you may be aware that computing as a field has a culture problem.
- Studies show, our interactions with women and other students who belong to groups with longstanding underrepresentation in our field often turn them away.
- I'm talking about negative experiences, even small ones, that turn students off to the point that they leave computing.
- To those who may already feel as though they do not belong here, these experiences add up, and may lead them switch to a different field entirely.
- If you are in the majority in our field, I want to make a special appeal for you to help change this.
- I hope that no matter who where you came from and what you believe, treating your classmates with kindness and respect is something we can all agree on.

Course Feedback

- There will be an anonymous course feedback form available on Canvas throughout this semester.
- If something is working well, this is helpful for me to know.
- Suggestions for improvement are also welcome!



Questions?