University of Minnesota
Driven to Discover®

Introduction to Computer Graphics

CSCI 4611: Programming Interactive Computer Graphics and Games

Evan Suma Rosenberg | CSCI 4611 | Fall 2022

Instruction Team

Instructor

Evan Suma Rosenberg

they/them

Grad TA

Frank Bender

they/them

Grad TA

Bridger Herman

he/him

Undergrad TA

Yanai Sun

Who am I?

I started working on virtual reality research as an undergraduate in 2003.

Received Ph.D. in 2010 with a focus on locomotion in virtual environments.

Researcher at the University of Southern California Institute for Creative Technologies from 2010-2018.

Joined the University of Minnesota in 2018.

Who am I?

I'm also a lifelong video gamer.

First games: Breakout, Space Invaders (Atari 2600)

Last game completed: Elden Ring (PS5)

Current game: Pathfinder: Wrath of the Righteous (PC)

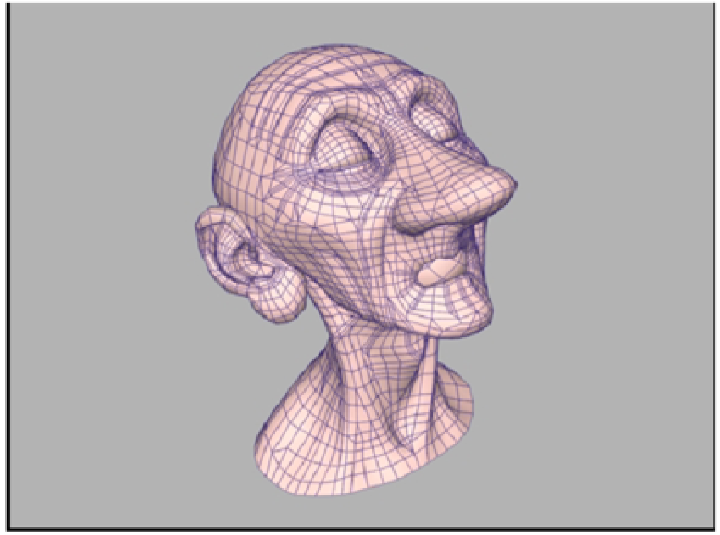
Most anticipated game: Horizon Forbidden West (PS5)

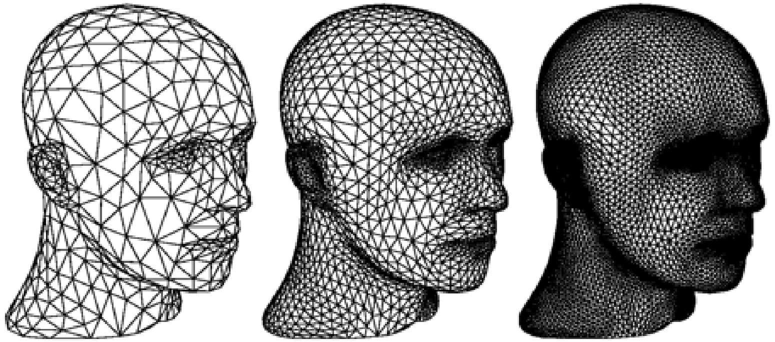
No, I won't tell you my Nintendo, Playstation, or Steam IDs. :)

==

Andy van Dam, Brown University
https://cs.brown.edu/people/avandam/

(Geri's Game, Pixar 1998)





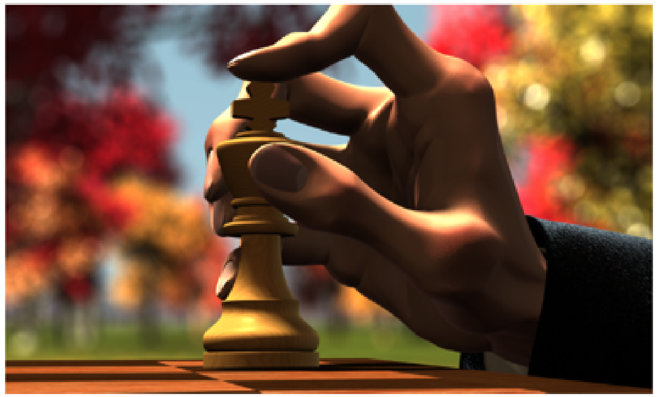


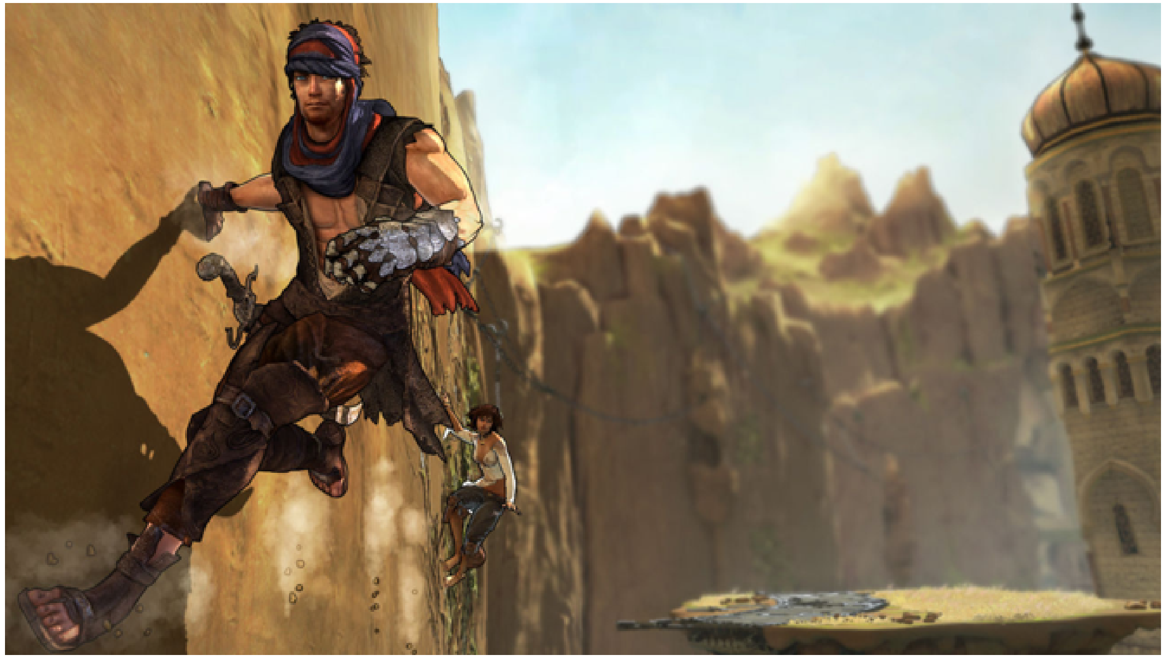
Figure 5: Geri's hand as a piecewise smooth Catmull-Clark surface. Infinitely sharp creases are used between the skin and the finger nails.

Stylized Rendering in Games

MONDAY, 26 JULY | 9:00 AM - 12:15 PM | ROOM 502 A
SIGGRAPH 2010 Course

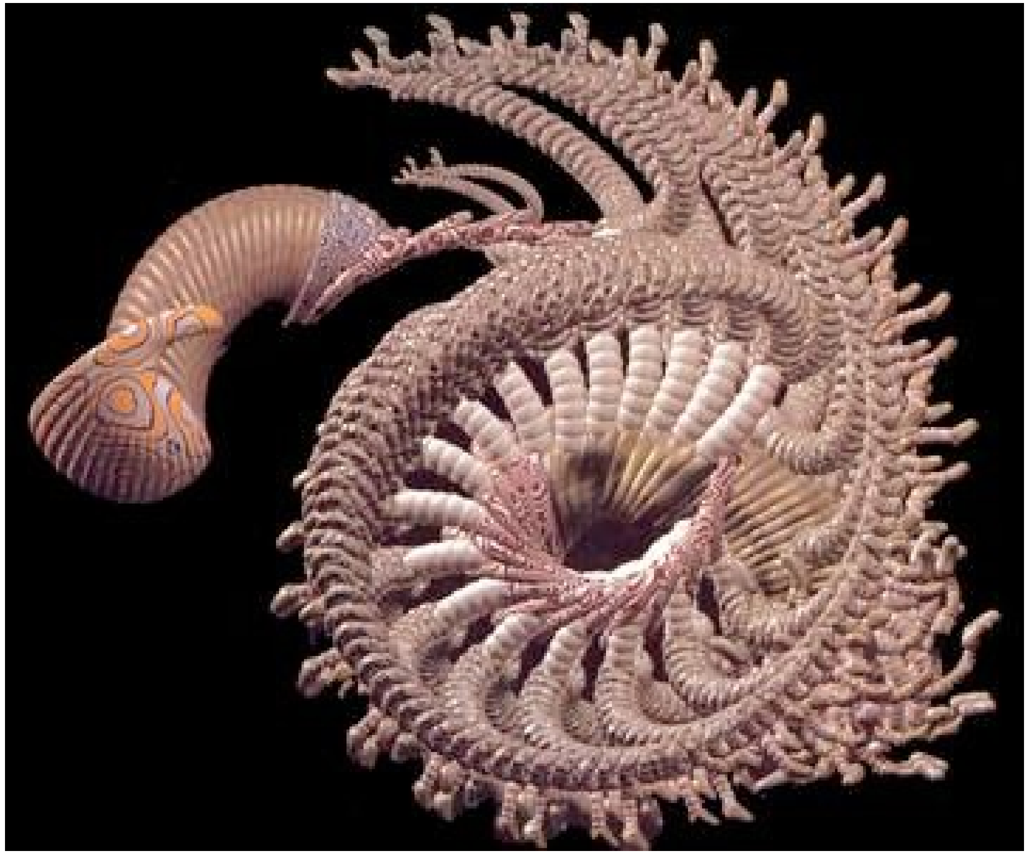
The Illustrative Rendering of Prince of Persia

Lecturer:
Jean-François St-Amour
Affiliation:
Ubisoft Montreal



What is Computer Graphics?

- Computer graphics generally means creation, storage and manipulation of models and images
- Models come from diverse and expanding set of fields including physical, mathematical, artistic, biological, and even conceptual (abstract) structures

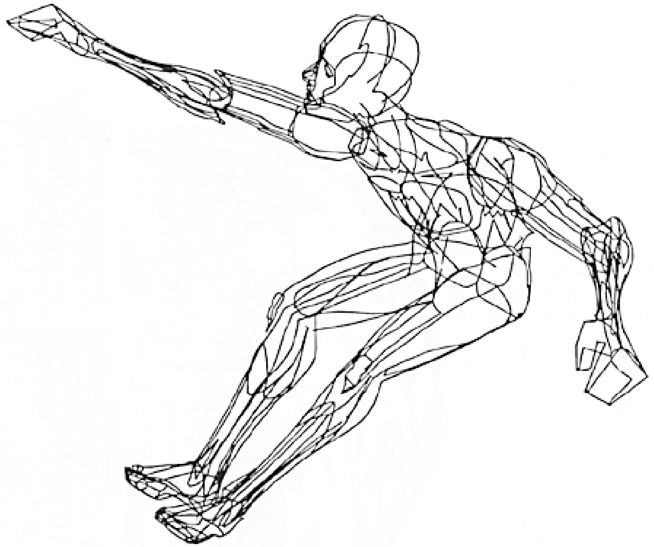


Frame from animation by William Latham, shown at SIGGRAPH 1992. Latham uses rules that govern patterns of natural forms to create his artwork.

What is Computer Graphics?

- William Fetter coined term “computer graphics” in 1960 to describe new design methods he was pursuing at Boeing

- Created a series of widely reproduced images on pen plotter exploring cockpit design, using a 3D model of the human body.



What is Computer Graphics?

“Perhaps the best way to define computer graphics is to find out what it is not. It is not a machine. It is not a computer, nor a group of computer programs. It is not the know-how of a graphic designer, a programmer, a writer, a motion picture specialist, or a reproduction specialist.

Computer graphics is all these – a consciously managed and documented technology directed toward communicating information accurately and descriptively.”

Computer Graphics, by William A. Fetter, 1966

What is Interactive Computer Graphics?

- User controls contents, structure, and appearance of objects and their displayed images via rapid visual feedback

- Basic components of an interactive graphics system

  - Input (e.g., mouse, tablet and stylus, force feedback device, scanner, live video streams…)

  - Processing (and storage)

  - Display/output (e.g., screen, paper-based printer, video recorder, non-linear editor…)

  First truly interactive graphics system, Sketchpad, pioneered at MIT by Ivan Sutherland for his 1963 Ph.D.

- thesis

Sketchpad in 1963. Note use of a CRT monitor, light pen, and function-key panel.

What is Interactive Computer Graphics?

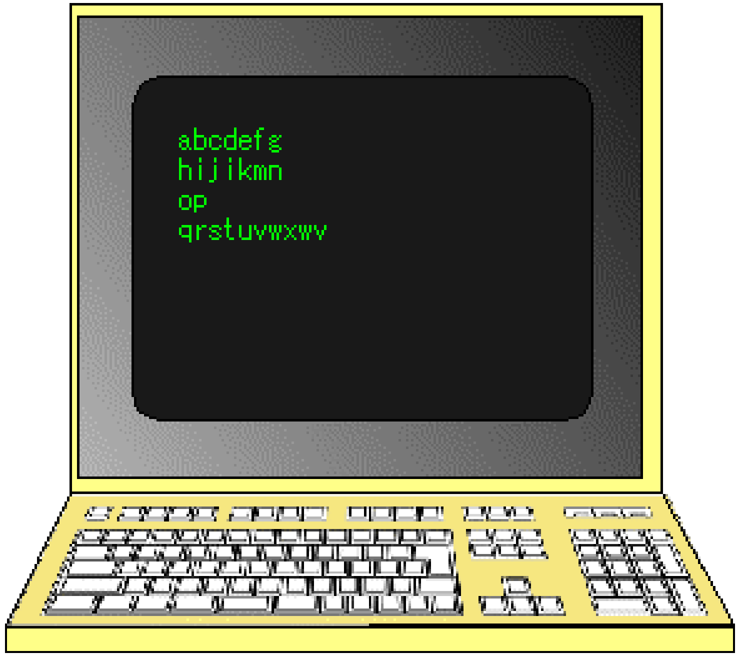
Almost all the key elements of an interactive graphics system are expressed in first paragraph of Sutherland's 1963 Ph.D. thesis:

The Sketchpad system uses drawing as a novel communication medium for a computer. The system contains input, output, and computation programs which enable it to interpret information drawn directly on a computer display. Sketchpad has shown the most usefulness as an aid to the understanding of processes, such as the motion of linkages, which can be described with pictures. Sketchpad also makes it easy to draw highly repetitive or highly accurate drawings and to change drawings previously drawn with it…

Evolution of Computer Graphics

Character Displays (1960s – now)

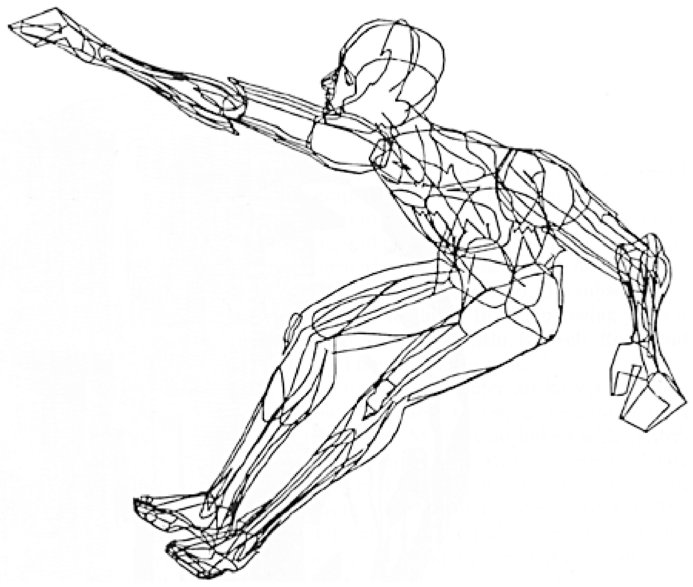
- Display: text plus alphamosaic pseudo-graphics
- Object and command specification: command-line typing
- Control over appearance: coding for text formatting (.p = paragraph, .i 5 = indent 5)



Evolution of Computer Graphics

Vector Displays (1963 - 1980s)

- Display: line drawings and stroke text; 2D and 3D transformation hardware
- Object and command specification: command-line typing, function keys, menus
- Control over appearance: pseudo-WYSIWYG
- Term "vector" graphics survives as "scalable vector graphics" library from Adobe and W3C – shapes as transformable objects rather than just bitmaps



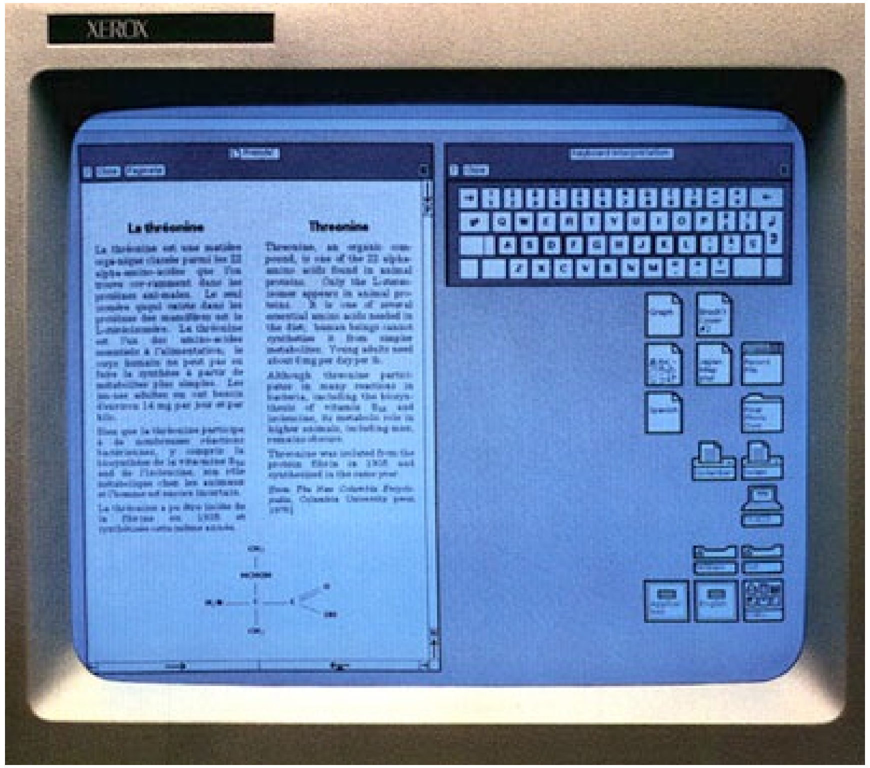
Evolution of Computer Graphics

2D Bitmap Raster Displays (1972 at Xerox PARC - now)

- Display: windows, icons, legible text, "flat earth" graphics. Late 1960’s saw first use of raster graphics, especially for flight simulators.

- Object and command specification: minimal typing via WIMP (Windows, Icons, Menus, Pointer) GUI: point-and-click selection of menu items and objects, widgets and direct manipulation (e.g., drag and drop), desktop metaphor

- Control over appearance: WYSIWYG (What You See Is What You Get)



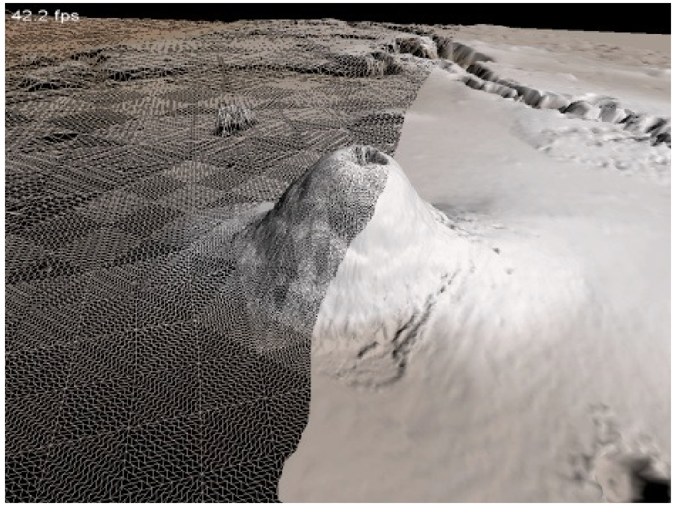
Evolution of Computer Graphics

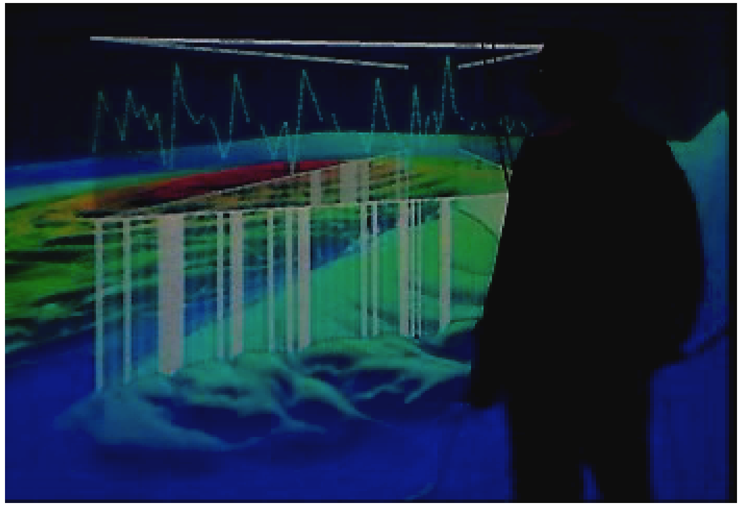
3D Graphics Workstations (1984 at SGI – now)

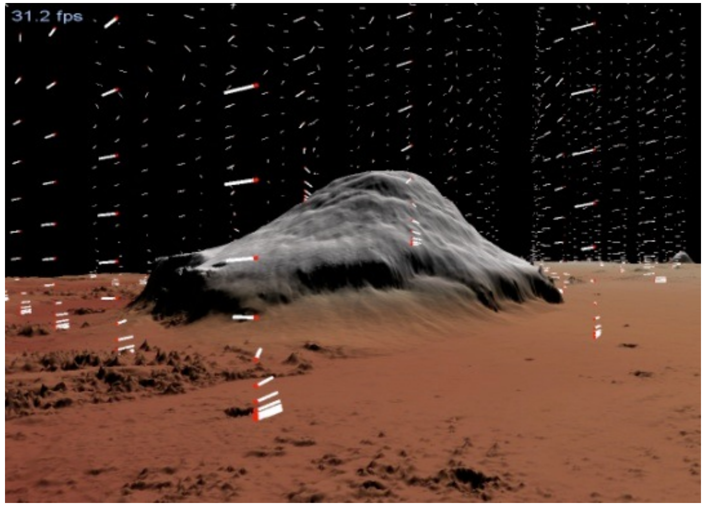
- Display: real-time, pseudo-realistic images of 3D scenes
- Object and command specification: 2D, 3D and nD input devices (controlling 3+ degrees of freedom) and force feedback haptic devices for point-and-click, widgets, and direct manipulation
- Control over appearance: WYSIWYG
- High-end PCs with discrete graphics cards (nVidia GeForce™, ATI Radeon™) have supplanted graphics workstations

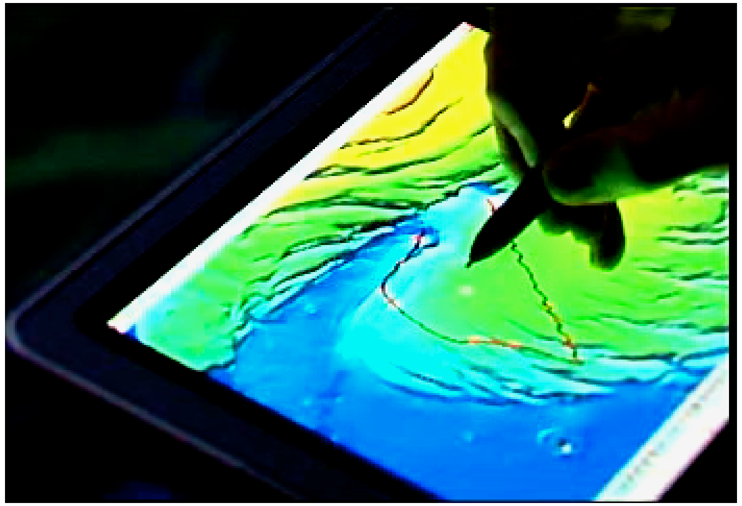
HORIZON
II
FORBIDDEN WEST™
YouTube
STORY TRAILER

New Forms of Computing



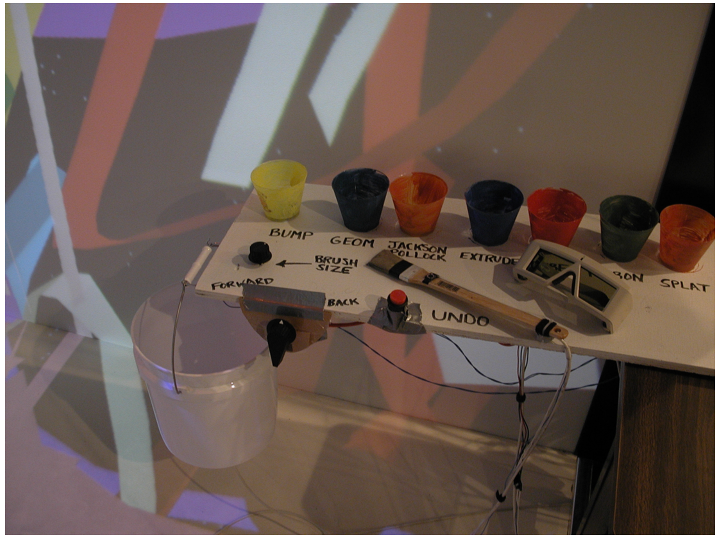


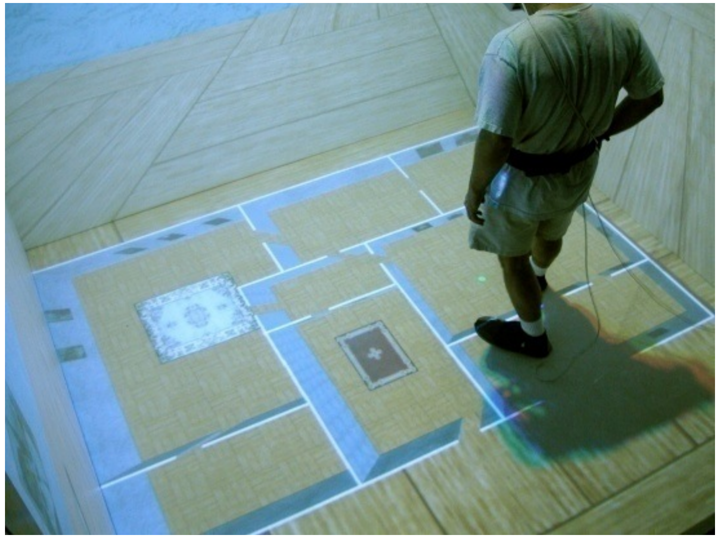




New Forms of Computing

Cave Painting





Use feet for navigation,

freeing hands for other uses

YouTube

Tilt Brush
by Google
YouTube

In this course, you will...

... understand basic concepts and algorithms relevant to computer graphics programming.

... identify, define, and solve problems involving 2D and 3D computer graphics.

... implement real-time computer graphics using a high-level scene graph library.

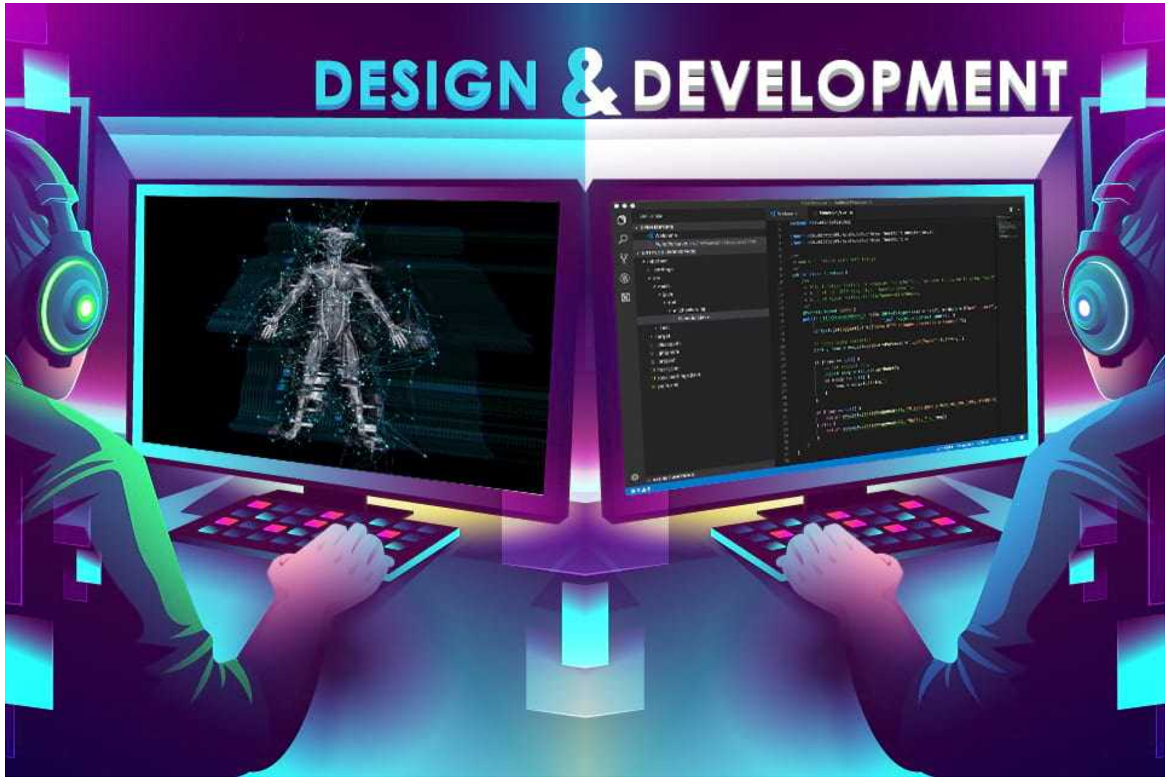
... write shader programs that run on the graphics processing unit (GPU).

... implement user interfaces for interaction with 3D graphics applications.

What won't you learn in this class?

How to create art assets
(3D models, textures, etc.)

How to design video games



Textbook

REAL-TIME RENDERING

FOURTH EDITION

Akenine-Möller • Haines • Hoffman
Pesce • Iwanicki • Hillaire

Thoroughly updated, this fourth edition of *Real-Time Rendering* focuses on modern techniques used to generate synthetic three-dimensional images in a fraction of a second. It provides a guide to the many algorithms that have been developed and applied over the past decade, and includes over 1100 new references. Chapters on physically based materials and lighting, global illumination, and non-photorealistic rendering are extensively revised. Some previous sections are now expanded into chapters on color, shadows, volumetric techniques, and efficient shading. A new chapter has been added about virtual and augmented reality. This edition discusses current, practical rendering methods used in games and other applications. It also presents a solid theoretical framework and the relevant mathematics for the field of interactive computers graphics, all in an approachable style.

Reviews

"This is the book I recommend to everyone starting out in the industry. Not only is it a great reference on so many topics, each topic is covered in impressive depth with great references for further exploration."
— Alex Vlachos, Valve

"*Real-Time Rendering* condenses literally thousands of cutting-edge papers, talks, and blogs into a single, easy-to-read volume presenting today's best practices, open problems, and promising state-of-the-art research. A key reference for beginners and experts!"
— Chris Wyman, Principal Research Scientist, NVIDIA

"Since it was first published, *Real-Time Rendering* has been an invaluable companion to anyone who wants to keep up with this dynamic field. It combines rigorous coverage of the fundamentals with up-to-date discussion of the latest techniques. The fourth edition is required reading for anyone serious about computer graphics."
— Matt Pharr, co-author of *Physically Based Rendering: From Theory to Implementation*

"I built our rendering engine and my career on what I learned in previous editions of Real-Time Rendering. This new edition is carefully updated to represent the current state of our field, and will remain the first resource I check when tackling a new challenge."
— Patrick Cozzi, Principal Graphics Architect, Cesium, and co-editor of *OpenGL Insights*

"Real-Time Rendering is the first book I recommend reading to anyone who wants to learn real-time graphics. All the relevant knowledge in one place, and a joy to read, too!"
— Aras Pranckevičius, Unity Technologies

About the Cover: The cover shows an example of advanced real-time rendering technology used in the 2018 *Star Wars*™ short *Reflections*, which includes real-time ray tracing of reflections and area light shadows computed on the GPU. The short was produced by Epic Games and built in Unreal Engine in collaboration with ILMxLAB and NVIDIA.

Games and Animation

CRC Press
Taylor & Francis Group
an informa business
www.crcpress.com
CRC Press titles are available as eBook editions in a range of digital formats

ISBN 978-1-138-62700-0

REAL-TIME RENDERING

FOURTH EDITION

Akenine-Möller • Haines • Hoffman
Pesce • Iwanicki • Hillaire

CRC

REAL-TIME RENDERING

FOURTH EDITION

Tomas Akenine-Möller
Eric Haines
Naty Hoffman
Angelo Pesce
Michał Iwanicki
Sébastien Hillaire

CRC Press
Taylor & Francis Group
AN A K PETERS BOOK

https://www.realtimerendering.com

Programming Languages and Software

- **TypeScript**

  Open-source language that builds on JavaScript by adding static type definitions.

- **GopherGfx**

  High-level software library for 3D computer graphics using WebGL.

- **Visual Studio Code**

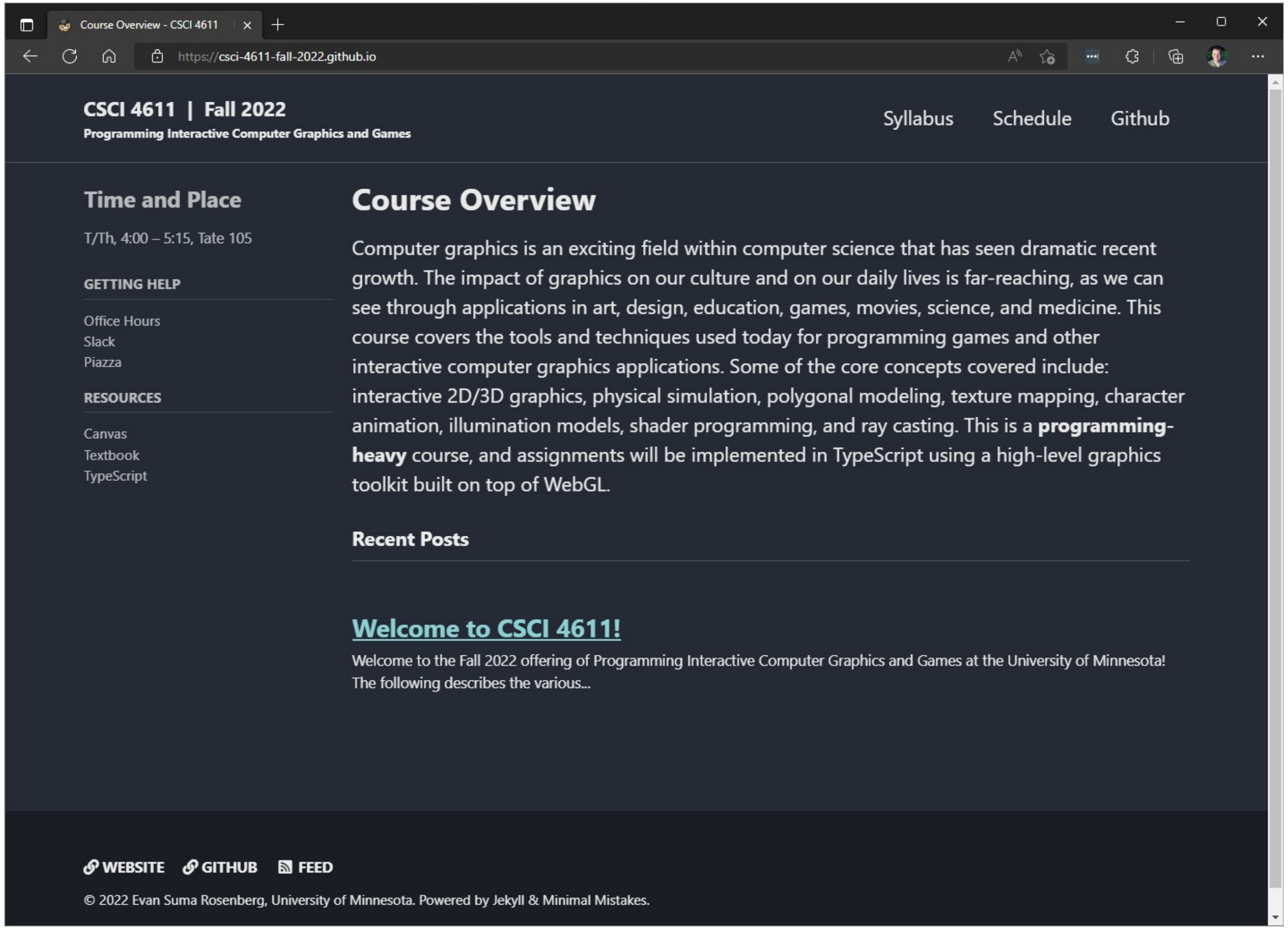
  In the Stack Overflow 2021 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool, with 70% of 82,000 respondents reporting that they use it.

UNIVERSITY OF MINNESOTA
Driven to Discover®

Course Overview

CSCI 4611: Programming Interactive Computer Graphics and Games

Evan Suma Rosenberg | CSCI 4611 | Fall 2022



Course Overview - CSCI 4611
https://csci-4611-fall-2022.github.io

CSCI 4611 | Fall 2022

Programming Interactive Computer Graphics and Games

Syllabus Schedule Github

Time and Place

T/Th, 4:00 – 5:15, Tate 105

GETTING HELP

Office Hours
Slack
Piazza

RESOURCES

Canvas
Textbook
TypeScript

Course Overview

Computer graphics is an exciting field within computer science that has seen dramatic recent growth. The impact of graphics on our culture and on our daily lives is far-reaching, as we can see through applications in art, design, education, games, movies, science, and medicine. This course covers the tools and techniques used today for programming games and other interactive computer graphics applications. Some of the core concepts covered include: interactive 2D/3D graphics, physical simulation, polygonal modeling, texture mapping, character animation, illumination models, shader programming, and ray casting. This is a **programming-heavy** course, and assignments will be implemented in TypeScript using a high-level graphics toolkit built on top of WebGL.

Recent Posts

Welcome to CSCI 4611!

Welcome to the Fall 2022 offering of Programming Interactive Computer Graphics and Games at the University of Minnesota! The following describes the various...

WEBSITE GITHUB FEED

© 2022 Evan Suma Rosenberg, University of Minnesota. Powered by Jekyll & Minimal Mistakes.

Course Website

https://csci-4611-fall-2022.github.io

Prerequisites

To succeed in this class, you should…

… have completed CSCI 2021 or equivalent experience.

… be interested in having fun with computer graphics!

… willing to commit sufficient time

3 credits = 9 hours / week for an “average” student to get a C

Course Modality

I am planning to teach this course in-person.

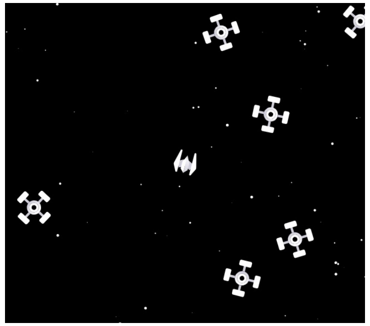
The class will **not** be taught remotely unless necessary (e.g., personal illness).

All classes will be recorded and posted online.

The University expects all community members to respect those who choose to wear a mask, as well as those who choose not to wear one. I do not intend to wear a mask in class myself, and I fully support your individual choices around masking.

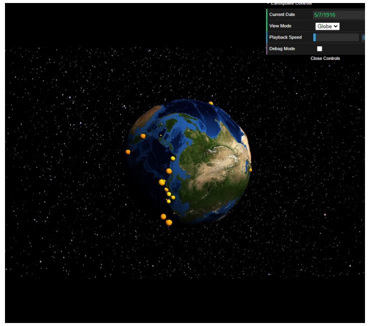
Course Structure

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



Interactive 2D Graphics

Physical Simulation



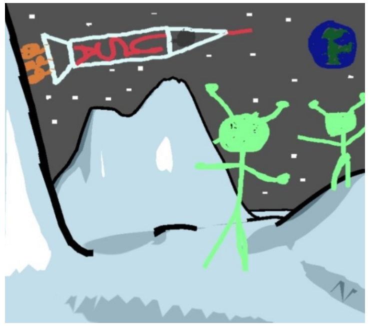
Current Date 5/7/1916
View Mode Globe
Playback Speed
Debug Mode □

Meshes, Textures, and Morphing

Character Animation

Rendering Style: Toon Shading
Model: bunny.obj
Light: Point Light

Lighting and Shading



Cameras, Rays, and 3D Interfaces

Course Structure

Tuesdays: Graphics Concepts

For one class each week, we will cover lessons on graphics concepts, math, and techniques. These classes will generally follow a standard lecture format.

Thursdays: Programming with the Professor

On these days, we will have live, interactive programming sessions with me. Please feel free to 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.

Assignments and Quizzes

At the beginning of each two-week module, a quiz and programming assignment will be posted on Canvas.

The quiz is due at the end of the first week.

They are open-book and have no time limit. You can take the quiz anytime within the first week.

Note that the quizzes are designed to get you thinking about the assignment early, so it is generally a good idea to complete them before starting work on the program.

The program is due at the end of the second week.

If you struggled with the quiz, then seek help from TAs as needed, and make sure that you understand the key concepts before starting on the program.

No Midterm or Final Exam

Grading

60% - Programs

40% - Quizzes

Wizard Bonus Challenge

- **All programming assignments in the course will include great opportunities to go beyond the requirements and do cool work.**

- **On each assignment, you can earn one bonus point for implementing a meaningful new feature to your program.**

  A single point may not sound like a lot, but assignments use a 20-point scale, so this is equivalent to a 5% bonus!

- **While grading the assignments, the TAs will identify the best four or five examples of people doing cool stuff with computer graphics.**

  We call these students our wizards, and after each assignment, the students selected as wizards will have their programs demonstrated to the class.

Flexible Late Policy

Each student is granted a total of 10 late points at the beginning of the semester.

Each 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 10-day extension on a single assignment or spread them around among multiple assignments.

Flexible Late Policy

The late policy intended to provide flexibility for missed work.

This includes legitimate absences such as illness, family crisis, etc.

If you have points remaining, you do not need to contact me for an extension on programming assignments.

Your current late points will be visible in Canvas and will be updated when your assignments are graded.

Frequently Asked Questions

Do I have to tell you when I want to use my late points?

No, we will apply credit automatically based on the time of your submission.

Do these apply to the quizzes as well as the programs?

No, the quizzes are much shorter and worth less than the programming assignments, so you should plan to complete them by the deadline. They can be completed anytime within the first week of the module, so you would need a very convincing justification for an additional extension.

Frequently Asked Questions

Is there a penalty if I run out of late points and submit an assignment after the deadline?

If you use all your late points, we will still accept it and grade the assignment with a late penalty of **5% for each day** it is late.

What if I get really sick and need another extension?

This is possible in rare circumstances, but you would need to talk with me about your situation.

Frequently Asked Questions

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 points are meant to increase **equity** for students that are disadvantaged in this course.

If they were redeemable for bonus points, then this would effectively penalize the students who need extensions for legitimate reasons beyond their control.

The term “equity” refers to fairness and justice and is distinguished from equality: Whereas ***equality*** *means providing the same to all,* ***equity*** *means recognizing that we do not all start from the same place and must acknowledge and make adjustments to imbalances.*

All work in this class is individual!

You are encouraged to discuss the content of the lectures and the texts with your peers

You are permitted to discuss and make posts about programming in general

(e.g., a syntax error you are stuck on, missing include file)

Communication with others must stop before discussing a solution to the quiz or assignment

If you have any question about whether discussing something with peers might go beyond what is permitted, then stop and ask us first!

Use of Web Resources

The web will be one of your best learning tools for working with TypeScript and computer graphics toolkits.

What you can do:

You are encouraged to use the web to lookup questions about the APIs used and even to look at example programs written with TypeScript and computer graphics toolkits.

For example, if your assignment is to develop your own texture mapping routine for a sphere, you can lookup the API documentation to learn the right function to call and the right syntax to use to set the texture coordinate for each vertex in a mesh.

What you cannot do:

You are **not** allowed to search for solutions to assignments.

For example, you should not google for “texture map a sphere” and find out how to setup a loop and use cos() and sin() equations to determine the correct texture coordinates for each vertex.

Third Party Content

You are permitted to use art assets from the internet in programming assignments.

This includes 3D models, textures, sounds, etc.

You must cite all third-party content in a readme file.

This is important! Students often forget this.

Getting Help

Piazza will be used for posting questions about the course and will be closely monitored by the TAs.

If you need help with an assignment, this is the first place you should go.

Slack will be the primary platform for synchronous communication.

This should generally be used when your question is not appropriate for posting on Piazza or you need to privately ask me a question.

You can also send me an email, but the response time will likely be slower.

The Zoom integration has been installed in Slack.

When you click the call button, it will initiate a Zoom meeting.

Office Hours

Office hours may be held either in-person or virtually.

Although we will do our best to accommodate walk-ins, priority will be given to students that book an appointment slot.

You can always find an up-to-date calendar on the course website.

We are also happy to answer questions asynchronously.

If it is a quick question, we can answer directly on Slack.

If it requires more explanation, we may initiate a Zoom call.

Wait List Policy

- **I am aware that this course is in high-demand and there are many students on the wait list.**

  I receive a large number of permission number requests, and unfortunately it is simply not possible to accommodate everyone.

- **I will issue permission numbers for graduating seniors or students with another convincing justification for taking the class right now.**

  Examples include starting research projects or summer internships involving computer graphics or video games. In these cases, the class should provide a direct benefit and taking it in a future semester would be late to help with the specific opportunity.

Inclusive Conduct

At this point in your careers, I trust you are already aware that computing as a field has a huge problem with a lack of diversity.

At all levels, numerous groups are underrepresented, including women, African Americans, Hispanics, American Indians, Alaska Natives, Native Hawaiians, Native Pacific Islanders, and persons with disabilities.

If you are in the majority in our field, I want to make a special appeal for you to join me in helping to change this.

Studies show that our interactions with women and other students who belong to groups with longstanding underrepresentation in our field often turn them away.

Course Feedback

There is an anonymous course feedback form available on Canvas throughout this semester.

Please feel free to let me know how things are going!

“Homework”

Make sure to join the course Slack and Piazza.

Links are posted on the course website.

The first live programming class will be on Thursday.

If you want to follow along with my on your own computer, you should install **Visual Studio Code** and **Node.js**.

Questions?