CS488/688

Fall 2024

Course Information

University of Waterloo

Department of Computer Science

Instructor(s): Stephen Mann

July 31, 2024

Contents

Chapter 1

CS488/688 F24 General Information

University of Waterloo

Term and Year of Offering: Fall 2024

Course Number and Title: CS488/688, Introduction to Computer Graphics

Instructor: Name, Office, Phone, email, Office Hours

Stephen Mann DC2317 x34526 `smann@uwaterloo.ca` `By appointment`

TA(s): Name, Office, email, Office Hours

Zachary Leger TBA `zcjleger@uwaterloo.ca` TBA
Scott Steinfield TBA `sssteinf@uwaterloo.ca` TBA
Weijie Zhou TBA `w239zhou@uwaterloo.ca` TBA

Course Newsgroup: `piazza`

Course URL: `http://www.student.cs.uwaterloo.ca/~cs488/`

Laboratory, Phone: MC3007 x36560

Programming Environment: C++, GNU/Linux, OpenGL, Lua

Required Texts:

Hearn, Baker, Carithers, *Computer Graphics with OpenGL*, Prentice Hall.

1.1 Course Description

Software and hardware for interactive computer graphics. Implementation of 3-D transformations, clipping, perspective, and input routines. Data structures, hidden surface removal, colour shading techniques, and some additional topics will be covered.

1.2 Course Objectives

At the end of the course you should be able to:

- write interactive 3D computer graphics programs;
- understand how linear and perspective transformations are used in modeling and rendering in 3D computer graphics;
- understand the process of rendering, lighting, hidden surface removal, and other computer graphics techniques;
- write a simple ray tracer.

1.3 Important Dates

- Assignment 0 due: Thursday, September 12th [Week 2]
- Assignment 1 due: Thursday, September 19th [Week 3]
- Assignment 2 due: Thursday, October 3rd [Week 5]
- Assignment 3 due: Thursday, October 24th [Week 7]
- **Midterm Exam: NA**
- Assignment 4 due: Tuesday, November 12th [Week 9]
- Assignment 5, First Goal due: Tuesday, November 5th [Week 8]
- Assignment 5, Second Goal due: Friday, November 15th [Week 10]
- Assignment 5, Third Goal due: Tuesday, December 3rd [Week 12]
- Demonstrations: video submitted with project
- **Final Exam: TBA**

1.4 Important Skills

- Reading.
- Read piazza.
- Read the course notes.
- Last but not least, read and follow instructions.

1.5 Text, References, and Documentation

Our lecture notes are fairly extensive. Even though copies of these notes are included with the course notes (which you are required to purchase), you are still required to purchase the course text.

In previous terms the text book has been either *Computer Graphics* by Foley, van Dam, Feiner, and Hughes; a "textbook" version of this text, *Introduction to Computer Graphics*, by Foley, van Dam, Feiner, Hughes, and Phillips; or *Computer Graphics* by Hearn and Baker. The first of these is a thick tome with a white cover, the second has a red cover. The last has either a white or a grey cover, depending on the edition. Any of these text books would be sufficient for the course. Check the used book store. If you plan to work in the computer graphics field, we would strongly recommend investing in a current edition of the first text, *Computer Graphics* by Foley, van Dam, Feiner, and Hughes—although it is a bit encyclopedic for the purposes of this course.

One additional reference book is recommended. The *OpenGL Programming Guide* documents the 3D graphics programming interface you will be using.

A great deal online documentation for Lua and other course software is also available through the course web page, and from the web.

We are also recommending Jim Blinn's book. This book is a collection of his columns from CG&A. Most of what he discusses is at a lower level than we are concerned with. However, there are a few good project ideas in the book, and he gives all the dirt on many low level graphics operations. If this sort of thing interests you, consider buying the book. However, you are not expected to know any material from this book (unless it is covered elsewhere or in lecture).

1.6 General overview of topics to be covered

- **The Graphics Environment** (4 hours)

  Overview of a representative processing sequence that connects application programs with the images they display on screen. Outline of the graphics library to be used and the hardware of the graphics workstation.

- **Mathematical Underpinnings** (4 hours)

  A review of concepts and tools: points, vectors, lines, planes, matrices, dot and cross products, vector space, affine space, projective space, etc.

- **Transformations** (4 hours)

  2- and 3-dimensional translation, rotation, and scaling as matrix operations. Homogeneous coordinates. Clipping, windowing, and viewing with perspective.

- **Interrupting, Picking, Polling, Callbacks** (3 hours)

  The management of picking, selecting, and control tasks through the use of event queues, interrupts, and device polling. Windowing systems and user interface toolkits.

- **Hidden Surfaces and Shading** (4 hours)

  Standard lighting models and their implementation. Hidden-surface elimination using depth buffering, scanline coherence, and subdivision. Polygon filling.

- **Ray Tracing** (4 hours)

  Basic ray tracing techniques for generating shadows, mirror reflections, and refraction. Constructive solid geometry models.

- **Physically Based Rendering** (4 hours)

  Radiosity, bi-directional path tracing, global illumination.

- **Discretionary Topics** (5 hours)

  Chosen at the discretion of the instructor. Possibilities include: more depth on any of the foregoing, as well as human vision, colour theory, anti-aliasing, database amplification, animation, scientific visualization, graphics hardware support, higher-order curves and surfaces, and dynamic simulation.

1.7 Programming Environment

Programming assignments for this course will be implemented in C++ under the linux operating system, using OpenGL. The completion of a significant independent project is required (see Assignment 5). Some of the assignments make use of the Lua scripting language for parsing scene descriptions.

OpenGL is a standard graphics applications programming interface (API) that was developed as an extension and standardization of Silicon Graphics Incorporated's (SGI's) proprietary IRIS GL API. Lua is an interpreted scripting language intended to be embedded in other programs (in our case C++ programs). It is particularily widespread in games but used for many other tasks.

The *OpenGL Programming Guide* includes little information on how to program in OpenGL under X with all but the most trivial of interfaces. This was intentional, since OpenGL is supposed to be window system independent. However, it makes the kind of "small-scale" programming we do in assignments a little difficult.

1.8 Electronic Resources

The course URL can be used with to access the course web pages. These web pages have a large amount of information, including but not limited to this overview, the lecture notes and overheads, a glossary, sample exam questions, additional reference material, pointers to other Internet sources, and specifications for all the assignments. Much of this material is in hypertext.

Subscribe to the piazza. **It is your responsibility to read the piazza on a regular basis.** You should definitely read piazza before major events such as exams and assignment due dates. Changes to the material in the course web will be posted to the piazza, but the piazza is also intended as a public forum for discussion of the assignments. Feel free to post to the piazza. The TAs and instructor will monitor the piazza and answer any relevant technical or policy questions.

Material on reserve will be listed under "CS 488"; the instructor is permanently listed as "Stephen Mann" regardless of who is really teaching the course.

1.9 Assignment Evaluation

Each assignment will be accompanied by a list of objectives. The TA(s) will grant one mark for each objective met satisfactorily. The TA(s) may grant one bonus mark per assignment for such things as additional programming features, innovative approaches, etc. The TA(s) may also deduct marks for poor user interfaces, poor documentation, or other problems.

Graduate students will be expected to prepare a short technical report, with bibliography, for their course project, in addition to the items requested in the project statement.

1.10 Quizzes (if course moved online)

If the course moves online for all or part of the term, there will be a near weekly quiz (10 quizzes maximum) given in Learn. Each quiz will be worth 2% of your course mark; the quiz weight will be removed from the final exam weight (e.g., if there are two quizzes, the quizzes will be worth 4% of your course mark and the final exam will be worth 31% of your course mark).

If 4 or more quizzes are given, we will drop your lowest quiz score, which is intended to address internet access problems (if your internet connection is so poor as to be more disruptive during the term, contact the instructor on how to proceed; you may need to take an incomplete in the course).

The quizzes will be available in Learn for 1 day (technically starting at 11:59 PM the day before and ending at 11:59 PM the day of the quiz). ONCE YOU START THE QUIZ YOU HAVE ONE HOUR TO COMPLETE IT. Each quiz will have 6 questions, and will have a fixed set of answers (i.e., you will select your answer(s) by picking select boxes with the mouse; you will not enter any text for the quiz).

Your quiz will be marked immediately, and you will be shown any incorrect answers.

The intent of the quiz is to make sure you keep up with the readings and videos; most of the questions will be easily answered if you have done so. A few of the questions will be Harder, just because we care.

1.11 Rules for group work

No group work is allowed; all work on the assignments and project must be done by each individual student. However, students are allowed to discuss the ideas and concepts that are to be implemented.

1.12 Late and missed assignments

No late assignments will be accepted. Any assignment missed or submitted late will be assigned a mark of 0.

1.13 Submitting assignments and getting them back

Assignments are due at the time specified on the web page. More precisely, your assignment must be uploaded to LEARN by the specified time and day. Assignments will be marked by the end of the following lecture. **CREDIT WILL NOT BE GIVEN FOR LATE ASSIGNMENTS.**

If, on receiving a marked assignment, you have complaints or requests for clarification, these must be made to the TA(s) within one week of the return of the marked assignments to you. Beyond this time, the mark will not be altered.

1.14 Exams

The final assessment will be scheduled by the Registrar. The final will cover material from the entire course. If we need to shift to an online exam, the final exam will be administered in Crowdmark.

Questions from past midterm tests and final exams will be made available on the course web page. These questions serve the faculty as suggestive material for making up new exam questions, and it is strongly suggested that you work through the pertinent ones to prepare for exams. All exams will be closed book.

You MUST do all quizzes and the final assessment on your own, without discussing them with anyone else until AFTER the submission deadline for the quiz/assessment (i.e., not just your deadline, but the last time that any student in the class may submit it). However, the quizzes Open Book, and if the final exam is given online then the final exam will be Open Book (but an in person final exam will be closed book); you may use the course text and rewatch course videos during the time you are taking the quiz/online-assessment.

While not prohibited except as noted below, caution is urged in using other internet resources during an exam since the instructor will happily ignore alternate facts discovered on the internet (i.e., in the event of conflicting answers to questions, the course text and videos will be the definitive source for awarding points; in case of ambiguities, the instructor will have final say in what answers will be awarded points).

It is forbidden to access websites or any other resources where exam questions (and possibly solutions) have been made available, nor is it permitted to put CS 488/688

material in one of these sites. Doing so is grounds for academic discipline. When taking a quiz or an exam, you are required to certify that you have not accessed one of these sites.

1.15 FIPPA

The Freedom of Information and Privacy Protection Act says that a student's name can not be connected with anything else related to that student without their consent. In the course, you will complete assignments (A3,A4,project) in which you may have pictures that we want to put in the course web page gallery. We will implicitly assume consent to put these material on the course web page with your name associated with them. This will also identify the term in which you took the course, and the image will likely have a file name based on your uw email address.

If you do NOT want your name to appear in the web page, you should let us know and we will not put images from your assignments in the gallery. If at any time in the future, you want us to remove an image that has your name on it (as well as your name etc.), just let us know and we will remove it.

1.16 Questions

All questions about assignments, due dates, or anything else related to the course should be directed to the instructor during class, to the instructor and TA(s) during their office hours, or to piazza. Office hours will be posted to `piazza` at the start of term and will be recorded in the course web pages.

The TA(s) will be available during posted office hours to handle questions. Please *do not* bother the TA(s) at other times, in person, on the phone, or by direct e-mail, except in dire emergencies—TA(s) are students, too, and have other commitments in addition to CS488/688!

Note: confidential questions (about your marks, etc.), should be sent from your University of Waterloo account. Regardless, we will send confidential information such as grades **only** to your University of Waterloo account. You should make a habit of reading your mail on that account or setting up a forward file from that account to where ever you read your email.

1.17 Intellectual Property

Students should be aware that this course contains the intellectual property of their instructor, TA, and/or the University of Waterloo. Intellectual property includes items such as:

- Lecture content, spoken and written (and any audio/video recording thereof);
- Lecture handouts, presentations, and other materials prepared for the course (e.g., PowerPoint slides);
- Questions or solution sets from various types of assessments (e.g., assignments, quizzes, tests, final exams); and

- Work protected by copyright (e.g., any work authored by the instructor or TA or used by the instructor or TA with permission of the copyright owner).

Course materials and the intellectual property contained therein, are used to enhance a student's educational experience. However, sharing this intellectual property without the intellectual property owner's permission is a violation of intellectual property rights. For this reason, it is necessary to ask the instructor, TA and/or the University of Waterloo for permission before uploading and sharing the intellectual property of others online (e.g., to an online repository).

Permission from an instructor, TA or the University is also necessary before sharing the intellectual property of others from completed courses with students taking the same/similar courses in subsequent terms/years. In many cases, instructors might be happy to allow distribution of certain materials. However, doing so without expressed permission is considered a violation of intellectual property rights.

Please alert the instructor if you become aware of intellectual property belonging to others (past or present) circulating, either through the student body or online. The intellectual property rights owner deserves to know (and may have already given their consent).

Chapter 2

CS488/688 F24 Assignment Overview

No food or drink is allowed in the lab.

There will be four programming assignments and a programming project for this course, which will be spread over a thirteen week term. These programs are worth a substantial portion of the final mark.

There will be a preliminary assignment that will be marked, but this mark will not be used in computing your final grade. This assignment is optional, but it gives a "no fault" trial to become familiar with the computing environment and the organisation of handing in and marking assignments. **We strongly suggest you do this assignment.** The TA(s) will assume that you have this familiarity for the regular assignments and the project, and they will make **no allowances** for lack of familiarity.

Note that you will be downloading a zip file that contains base code for each assignment as well as other files for the assignment. You are expected to start your assignments using the base code provided for Fall 2022. If you are retaking the course and wish to reuse code you wrote from a previous term, please contact the instructor.

2.1 Computing Environment

No food or drink is allowed in the lab.

There is a special lab for CS 488/688 in MC3007 . There are several Linux PCs in this room. Some of this equipment is extremely expensive. **Therefore, there is no food or drink allowed in the lab.** Anyone caught eating or drinking in the lab will be penalized by loss of marks on the next assignment (or final project!) from 1 mark to all marks at the instructor's discretion.

The computing environment will be under the linux operating system. The programming languages will be C++, and for some of the assignments, Lua.

You will be using C++ to build most of your user interface. You will also be extending the Lua language to interpret 3D scene description files by adding extra functionality in C++.

The Operating System will be Debian GNU/Linux on the PCs.

All assignments are submitted in Learn, and must run on the Linux PCs, and must run on one of these machines in MC 3007.

Graphics output will be through the PNG file format, and the OpenGL application programming interface (API).

You may obtain general UNIX and C++ help from the CSCF Consultants and specific help on the course from the TA(s), the instructor, and the newsgroup. There is also a great deal of information available through the course web, including a list of frequently asked questions.

Assignments will only be accepted if they execute on one of the Linux PCs.

2.2 Accessing Multiple Machines

We recommend setting up an appropriate `ssh` public key to allow access between all the machines easily. If you have not yet made an ssh key, you can do the following (on any of the lab machines):

```
$ ssh-keygen -t rsa                                # (accept the defaults)
$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

This will allow you to `ssh` from any lab machine to any other lab machine without a password (unless you supplied one when generating the `ssh` key). This will also have the same effect for all of the other CSCF undergraduate hosts.

2.3 Lab Etiquette

> *"Can we make people take a shower before they come in the lab?"*
>
> – An olfactorily sensitive student.

Near due dates, the lab will be busy. **You should NEVER lock one of the lab machines to reserve it for yourself while you step out of the lab.** If the instructor finds a student in the class has locked one of the machines, the student may be given a mark of 0 on the next assignment that is due (or a 0 on the final project if that is the only thing left to handin).

If all the lab machines are in use, you may be able to run your program from a different machine using `ssh`. To do this, connect to the machine you wish to use (first asking the person sitting at that machine for permission using `ssh -X glXX`, where `glXX` is the machine you wish to use remotely. You should then be able to run your application on that machine. OpenGL may or may not work depending on the environment (and will probably be slow regardless). This is also possible from home.

At most two courses use the Graphics Laboratory and the machines in MC 3007 in any term. You will be made aware of which courses these are. You are encouraged to recognize legitimate students. If you see someone in the lab whom you do not know, please introduce yourself, state which course you are in, and ask them to do the same, particularly if they are doing something that is unfamiliar in the context of the course work you are doing. We have had trouble with "crashers" in the past, students who cracked the lab combination. They frequently came into the lab to play

with the machines and have been annoying and troublesome. Even worse, we have had machines stolen, so if we seem paranoid, it's because we have reason! If it becomes clear that someone has no business being in the lab, please request that he/she leave. If the request is refused or ignored, you should call UW Security to have him/her removed.

2.4 The OpenGL Graphics API

Some assignments will require *real-time, interactive* 3D graphical output. Programs may therefore make use of the OpenGL graphics library, an Application Programming Interface (API) that provides an interface to hardware accelerated graphics.

OpenGL (unlike the SGI's older GL), does not provide built-in window support. You will rarely need to write entire programs from scratch. In particular, we will be giving you a bunch of code that will take care of most of the grungy interface hacking. This is in the interests of time only, so we can give you more interesting graphics assignments.

The *OpenGL Programming Guide* has been specified as a reference text. We will be providing some instruction in class, but you are expected to learn the details of OpenGL on your own time. The lectures will only direct you to the sections you will need to know to complete the assignment. The course TA(s) are the primary consultants on OpenGL.

2.5 Assignment Outlines

In this section we will give brief descriptions of each of the assignments.

2.5.1 Assignment 0: An Introduction

This assignment is primarily designed to introduce you to the computing environment and make sure you understand how assignments are to be submitted. You will be modifying the user interface of a simple graphics program. This assignment will be critiqued but will not count in your final grade. **AGAIN, WE STRONGLY SUGGEST YOU DO THIS ASSIGNMENT.** In the past, students who skipped this assignment regretted it.

2.5.2 Assignment 1: OpenGL

This assignment will introduce you to 3D graphics programming under OpenGL in C++.

2.5.3 Assignment 2: Transformations and the Graphics Pipeline

You will build a transformation pipeline to visualize the difference between model, world, and view space. Your pipeline will only support wireframe rendering of a cube and a few gnomons, but will have to directly support perspective and 3D line clipping. You will build a graphical user interface to this program; your implementation effort will consist of the C++ code to implement a graphics pipeline using only line drawing routines that will be provided to you.

2.5.4 Assignment 3: Hierarchical Modelling and OpenGL Rendering

You will build a hierarchical modelling and rendering system on top of OpenGL. The program you will create will parse a description of a hierarchical model and render it, and then will allow 3D picking and interactive manipulation.

The hierarchical model will be described in a scene description "language" defined as an extension to Lua. This extension will be reused and extended in the next assignment.

2.5.5 Assignment 4: Raytracer

Using the hierarchical modeller you defined in the last assignment, you will build a raytracer. A raytracer is a rendering algorithm which can create extremely realistic images, but is not suited for real time implementation. Your raytracer will support spheres, cubes, meshes of planar convex polygons, and shadows. You will also be required to implement one additional feature of your own choice. This feature can be an acceleration technique, a new model type or construction technique, or an additional visual effect.

2.5.6 Assignment 5: The Project

This assignment will be a project of your own devising. A proposal is required, which we will critique and return to you. The project is worth a substantial part of your mark, and should have a significant 3D graphics component.

2.6 Ambiguities or Omissions in the Assignments

No assignment specifies completely what is to be done; some things are left to your discretion. Wherever something is ambiguous or omitted, you are to use your own judgment. The essential thing is to

IDENTIFY AND DESCRIBE WHAT YOU ARE DOING AND WHY.

Document your decisions and especially any added features you implement if you want them to be recognized.

If you think something is broken or brain-damaged, it may well be. Let us know **WELL BEFORE** the assignment is due and we may be able to fix it or modify the specification and benefit everyone in the class.

2.7 Code Credit

Code that is not efficient, well written, well structured, and/or well documented may result in the **deduction** of points. On the other hand, it is possible for non-functioning code to be given credit. Credit for code that does not work **must be requested in writing** in the manual that accompanies your handin; TA(s) will not grant code credit automatically.

What you are actually doing when requesting code credit is stating which objectives do not work properly and asking for the TA(s) to examine code to give you up to 1/2 credit for each non-functioning objective. For each missed objective, your request must state the following explicitly:

1. What code files do not work, and the relevant line numbers.
2. What execution features of the code in these files do not work (i.e. describe the current behaviour).
3. What you think the probable cause of the problem is.
4. What steps you have undertaken and will/would continue to undertake to track the problem down and correct it.

If you fail to provide this information, the TA may not give you credit for your code.

The code in question must be stylistically good, well organised, heavily commented, and clearly documented. You must make it easy for the TA to understand what you are doing in your code. If the TA cannot understand your code, you cannot expect them to give you any credit for it.

2.8 Cheating

The programming assignments are designed to give practice on the topics presented in lectures. It is important that you learn all you can by doing them.

One activity that promotes learning is discussing the assignments, in class, with the instructor, with the TA(s), and with others in the class. These discussions are to be **PRELIMINARY** in nature, exploring the possible tools, techniques, and issues involved in the solution. The discussions can take place in class, on the newsgroup, and in small groups of other class members. Feel free to participate.

The second activity that promotes learning is that each student, **individually**, works out a solution and composes **individual code** to execute the assignment. Each assignment is to be handed in with the following statement:

Declaration:

I have read the statements regarding cheating in the CS488/688 course handouts. I affirm with my signature that I have worked out my own solution to this assignment, and the code I am handing in is my own (with the exception of code/models mentioned in my documentation and whose use was approved by the instructor).

Signature:

Only assignments with the student's signature affixed to this declaration will be graded. If anyone is discovered not to have abided by their declaration, severe disciplinary action will be taken.

The minimum penalty for cheating of any kind on an assignment is a mark of -100% on that assignment, and a letter of notification to the Dean, who will enter a letter of reprimand into your file. Further action by the Mathematics Faculty Disciplinary Committee may also be undertaken; note that the penalties are severe, and can easily include expulsion.

If you have any questions about what may or may not constitute cheating, please ask the instructor so you don't end up cheating "by mistake".

2.8.1 Previously Written Code, Use of Other People's Code, ChatGPT

The code you submit should be code you wrote. For the project, it may make sense for you to use code written by someone else to assist you in your work. Further, you may wish to reuse code you wrote in previous terms. In both cases, you should check with the instructor before hand, and credit the code (even if it was code you wrote in previous terms) in your documentation for the assignment/project. However, you do not need to give credit for code that we provide for you. See the discussion of projects for more information on using code you previously wrote as part of your project.

You should also not use an AI like ChatGPT to assist you in writing your code. All the code you submit for credit must be your own. You may use an AI to help generate assets for your project that do not count as an objective (e.g., to generate a 3D model or to generate an image for texture

mapping). However, you should note in your project write-up that you have done so, being specific on what software you used and what it was used to create.

2.9 Feedback

The course was offered using SGI Indigo workstations for several years. The Indigo's were replaced with SGI Octanes, which are much faster machines. X workstations were added a some years ago. Eventually the whole lab became full of Linux PCs. Furthermore, the course has evolved from using C with Tcl/Tk, to C++ with Python/Tk, to C++ with GTK+ and Lua, to C++ with Qt and Lua, and most recently C++ with ImGui and Lua. The programming environment has thus been modified each term. It's still not perfect, and the constant modification gives entropy a chance to work. *Please* let us know as soon as possible if you spot any inconsistencies in the documentation or the provided code.

In addition, the instructor and TA(s) will strongly welcome any *constructive* criticism or suggestions you may have for making the programming environment better.

2.10 Additional information

- Make sure you don't change the specified default behaviour if your program has additional features. Let your extra features be optional and mention them in the README file.
- You can lose marks for the things that are not explicitly stated in the assignment objectives. Submission of poorly written code has already been mentioned, for example. Failure to put the executable(s) in the proper place is another example, and it will also result in deduction from the assignment mark. You can also lose a mark for submission of a program with a poor user interface.