CMSI 371
Computer Graphics

http://dondi.lmu.build/spring2021/cmsi371

Spring 2021—online (see Brightspace for links)
TR 1:50–3:05pm, 3 semester hours
Office Hours: TR 3:30–6pm, W 4:30–6pm;
or by appointment *(don't hesitate to ask!)*

John David N. Dionisio, Ph.D.
email: *dondi@lmu.edu*
Virtual Doolan 102; (310) 338-5782

Objectives and Outcomes

This course explores the computer science subfield of *computer graphics*—the study and development of algorithms for synthesizing, manipulating, and displaying visual information. Long after the course concludes, my hope is that you will be able to:

1. **Represent, model, and create visual information digitally.**
2. **Manipulate and display visual information both computationally and mathematically.**
3. **Use and develop computer graphics APIs at different levels of abstraction.**

In addition to the course-specific content, you are also expected to:

4. **Follow disciplinary best practices throughout the course.**

Prerequisites/Prior Background

Mastery of a programming language such as JavaScript, Java, or C; expert knowledge of data structure and algorithm design; some familiarity with object-oriented programming, computer hardware, MVC, and event-driven programming.

Materials and Texts

No single book is required, but many are recommended, based on your learning style:

Theoretical/Mathematical Foundation

- Peter Shirley and Steve Marschner et al, *Fundamentals of Computer Graphics*, 3rd Edition, CRC Press, 2009

Practical/Programming

- Farhad Ghayour and Diego Cantor, *Real-Time 3D Graphics with WebGL 2*, 2nd Edition, Packt Publishing, 2018
- Jacobo Rodriguez, *GLSL Essentials*, Packt Publishing, 2013

Reference Works

- John Kessenich, Graham Sellers, and Dave Shreiner, *OpenGL Programming Guide*, 9th Edition, Addison Wesley, 2017—*a.k.a. "the Red Book"*
- Randi J. Rost, Bill Licea-Kane, et al., *OpenGL Shading Language*, 3rd Edition, Addison Wesley, 2009—*a.k.a. "the Orange Book"*

Assorted handouts, articles, and sample code will also be distributed throughout the semester. In addition, do not hesitate to look for further information regarding the concepts, techniques, tools, and paradigms that we will discuss.

Course Work and Grading

Your final grade will be based on the percentage of the points you get for the following deliverables against the total number of possible points:

Fun with *their* 3D library, part 1	100
Fun with *their* 3D library, part 2	100
Pixel-level primitives and filters	100
Static scene based on *our* polygon mesh library	100
Our own matrix library	100
Our lights, *our* camera…	100
…*Your* action!	200
Total	800 points

Percentages ≥ 90% get an A– or better; ≥ 80% get a B– or better; ≥ 70% get a C– or better. I may nudge grades upward based on qualitative considerations such as degree of difficulty, effort, class participation, time constraints, and overall attitude toward the course.

Term Portfolio

Your accumulated assignments for the semester comprise the *term portfolio*—the final, definitive artifact that demonstrates the course's outcomes. It is how you show whether you have, indeed, accomplished the objectives of this course.

An assignment's number is its due date in *mmdd* format, and it is always due by 11:59:59.999pm of that date. Point values are based on the state of your assignments at that moment.

Your portfolio for this course will consist of the following types of deliverables:

- An interactive 3D scene based on a high(er)-level graphics library (*1b*, *1c*, *2a*, *2b*, *3a*, *3c*)
- Simple image processing filters (*1a*, *2c*, *3b*)
- A 3D object library with polygon meshes, object composition, and instance transformations (*1a–1c*, *2a*, *2b*, *3a*, *3d*)
- Vertex and fragment shaders implementing commonly-used computer graphics algorithms and techniques (*2a–2d*, *3a*, *3b*, *3d*)
- An interactive 3D scene based on these libraries (*1a–1c*, *2a–2d*, *3a*, *3b*, *3d*)

The full range of *4a–4f* applies to all assignments.

Version Control

Version control is an indispensable part of today's computer science landscape in industry, the academe, and the open source community. We use version control heavily in this course: make sure that you get the hang of it.

None of the assignments can be completed (well) overnight; they should be the result of steady progress from the moment they are assigned to the date they are due. "One and done" submissions will negatively affect the final score.

Workload Expectations

In line with LMU's *Credit Hour Policy*, the workload expectation for this course is that for every one (1) hour of classroom instruction (50 scheduled minutes), you will complete at least two (2) hours of out-of-class work each week. This is a 3-unit course with 3 hours of instruction per week, so you are expected to complete 3 × 2 = 6 hours of weekly work outside of class.

Attendance

Attendance at all synchronous sessions is ideal, but not required. If you must miss class, it is your responsibility to notify me about this and keep up with the course asynchronously.

Due to the remote/online format of the class, it can be tempting to "spoof" yourself by logging in, deactivating audio and video, and ultimately tuning out. If you're planning to do this, don't bother—just watch the session recording later. Better for the class to know that you really aren't around than for us to *think* you're around, only to realize that you aren't when we try to interact with you.

The last day to add or drop a class without a grade of W is January 15. The withdrawal or credit/no-credit deadline is April 9.

Academic Honesty

Academic dishonesty will be treated as an extremely serious matter, with serious consequences that can range from receiving no credit to expulsion. It is *never* permissible to turn in work that has been copied from another student or copied from a source (including the Internet) without properly acknowledging/citing the source. It is your responsibility to make sure that your work meets the standard of academic honesty set forth in:

http://academics.lmu.edu/honesty

Americans with Disabilities Act

Students with special needs who require reasonable modifications, special assistance, or accommodations in this course should promptly direct their request to the Disability Support Services (DSS) Office. Any student who currently has a documented disability (ADHD, Autism Spectrum Disorder, Learning, Physical, or Psychiatric) needing academic accommodations should contact the DSS Office (Daum Hall 2nd floor, 310-338-4216) as early in the semester as possible. All discussions will remain confidential.

Please visit *http://www.lmu.edu/dss* for additional information. Ask for help as early in the semester as possible!

Also keep in mind that resources are available through the Library (*https://library.lmu.edu*) and Information Technology Services (*https://its.lmu.edu*). The DSS Office can help students connect with the appropriate person at the Library and ITS.

Topics and Important Dates

Correlated outcomes are shown for each topic. Specifics may change as the course progresses. University dates (italicized) are less likely to change.

January	3D graphics with JavaScript with a high-level library *(1b, 1c, 2a, 2b, 3a, 3c)*
January 15	*Last day to add or drop a class without a grade of W*
February	Graphics and memory *(1a, 2c, 3b)*; 2D graphics primitives *(1a, 3b)*; graphics with JavaScript and foundational libraries *(1b, 1c, 2a, 2b, 2c)*; introduction to programmable shaders *(3d)*
March 1–5	*Spring break; no class*
March	Object modeling *(1b, 1c)*; transforms *(2a, 2b, 3d)*; viewing and projection *(2b, 3a, 3d)*
March 31	*Cesar Chavez Day; no class (listed here for completeness)*
April 1–2	*Easter break; no class*
April	Lighting and shading *(2c, 3d)*; clipping and hidden surface removal *(2d)*; 3D scene workshop/lab sessions *(1a–3b, 3d)*
April 9	*Last day to withdraw from classes or apply for Credit/No Credit grading*
May 6	*Last set of term portfolio assignments due*

You can view my class calendar and office hour schedule in any iCalendar-savvy client. Its subscription link can be found on the course web site (it's too long to provide in writing).

Tentative Nature of the Syllabus

If necessary, this syllabus and its contents are subject to revision. Students are responsible for any changes or modifications announced or distributed in class, emailed to students' LMU Lion accounts, or posted on LMU's course management system, Brightspace. If you are absent from a synchronous class session, it is the student's responsibility to check Brightspace and with the professor to see if you missed any important class announcements. Students should not rely on word-of-mouth from classmates.

Course Evaluations

Student feedback provides valuable information for continued improvement. All students are expected to fairly and thoughtfully complete a course evaluation for this course. This semester, course evaluations will be administered online through the Blue™ evaluation system. You will receive an email notification at your Lion email address when the evaluation form is available. You may also access the evaluation form on Brightspace (*https://brightspace.lmu.edu*) dashboard during the evaluation period. Your responses will be anonymous and will not be linked to you in any way.

Course Outcomes

1 Represent, model, and create visual information digitally.

1a	*…in terms of pixels and geometric primitives.*	With a few exceptions, these outcomes will be demonstrated within a single, cumulative "scene" program to be developed throughout the semester. Assignments incrementally build on previous ones. It will thus be more important than usual that you keep up and keep current.
1b	*…in terms of polygon meshes: vertices, edges, and faces.*	
1c	*…as a composition of multiple discrete objects (scenes).*	

2 Manipulate and display visual information both computationally and mathematically.

2a	*Apply and implement transforms.*	In the same way that the study of general-purpose data structures starts with the structures themselves, then goes into algorithms and operations on those structures, so goes the study of computer graphics. Learning objective 1 looks at structure; learning objective 2 looks at algorithms and computations.
2b	*Project 3D objects onto a 2D viewport.*	
2c	*Perform color and light computations.*	
2d	*Be familiar with established algorithms such as clipping and hidden surface removal (HSR).*	

3 Use and develop computer graphics APIs at different levels of abstraction.

3a	*Develop a library of 3D objects.*	There is some overlap between these outcomes and the ones for learning objective 2. This is by design—the outcomes in objective 2 focus on understanding these computations conceptually; the outcomes in learning objective 3 look at your ability to implement them concretely. "Different levels of abstraction" pertain to *where* your implementation starts—high-level libraries allow you to write less code, but using someone else's assumptions and designs; low-level libraries expose full flexibility and power, but you need to write more code to fully tap that power.
3b	*Perform bit-level color manipulation.*	
3c	*Render a 3D scene using a high-level library.*	
3d	*Render a 3D scene using programmable shaders.*	

4 Follow disciplinary best practices throughout the course.

4a	*Write syntactically correct, functional code.*	Code has to compile. Code has to work. No errors, no bugs. Use unit tests as much as possible.
4b	*Use programming best practices, demonstrating principles such as DRY, proper separation of concerns, correct scoping of variables and functions, etc.*	This is the basis of good software design. It makes software easier to maintain, improve, and extend. Heed feedback well. *What you learn here will apply to future work in this field, in school and beyond.*
4c	*Write code that is easily understood by programmers other than yourself.*	This outcome involves all aspects of code readability and clarity for human beings, including but not limited to spacing & indentation, proper naming, presenting code in a manner that is consistent with its structure, documentation & comments when appropriate, and adherence to conventions or standards.
4d	*Use available resources and documentation to find required information.*	The need to look things up never goes away. Remember also that the course instructor counts as an "available resource," so this outcome includes asking questions and using office hours.
4e	*Use version control effectively.*	In addition to simply using version control correctly, effective use also involves appropriate time management, commit frequency, and descriptive commit messages.
4f	*Meet all designated deadlines.*	