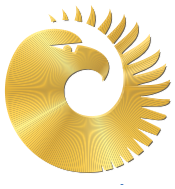

COM-391, Computer Graphics

American University of Central Asia
Software Engineering Department

1 Course Description

In this course, students will explore the core principles of modern 3-D computer graphics by building real-time 3-D engines through a series of hands-on laboratory tasks. They will learn how to leverage graphics accelerators using the OpenGL ES API to create visually rich computer-generated images, animations, and interactive applications. Lab exercises will guide students through each stage of engine development, covering geometry processing, shader programming, lighting, texture mapping, and performance optimization. By the end of the course, students will be able to research and analyze the functioning of complex real-time rendering systems, applying their enhanced programming and software design skills to develop fully functional and optimized 3-D graphics solutions.

2 Course Information

Course Materials
`https://github.com/auca/com.391`

Course Code
COM-391

Course ID
4954

Prerequisite
COM-119, Object-oriented Programming

Credits
6

Time and Place
Lecture: Monday 14:10–15:25, Room 410
Lab: Wednesday 12:45–14:00, Room 410

3 Contact Information

Professor
Dmitrii Toksaitov
toksaitov_d@auca.kg

TAs
David Koifman
koifman_d@auca.kg

Office
AUCA, Room 315

Office Hours
Available by appointment, either on-site or remotely, during work hours (Monday to Friday). Please contact your professor or TA to schedule a meeting.

4 Topics Covered

- Week 1–2: Introduction, Brief History, Dev. Environment (6 hours)
- Week 3–4: Vectors, Matrices, Scene Graph, Camera (6 hours)
- Week 5–6: Geometry, Buffer Objects, GPU Pipeline (6 hours)
- Week 7: Procedural Geometry (3 hours)
- Week 8–9: Materials, Shading, GLSL (6 hours)
- Week 10–11: Texturing and Mapping (6 hours)
- Week 12: Lambert, Phong, Blinn-Phong Shading (3 hours)
- Week 13–14: Instancing and other Rendering Optimizations (6 hours)
- Week 15–16: Real-time Graphics in Games, Building a Game with your Engine (6 hours)

5 Learning Outcomes

By the end of this course, students will:

1. **Acquire programming skills to write and test graphics applications**:
   - Implement and test programs using graphics programming concepts and APIs
   - Apply debugging techniques to identify and fix rendering issues and performance bottlenecks
2. **Develop proficiency in modeling procedural and object-oriented systems for graphics applications**:

- Use mathematical representations (vectors, matrices) to model transformations and geometry
- Represent 3-D objects and scene graphs using object-oriented principles

3. **Evaluate and use appropriate tools and techniques for graphics development**:
   - Use code editors, compilers, build systems, debuggers, and IDEs effectively
   - Work with graphics APIs (such as OpenGL ES) and GLSL shaders
   - Apply version control and collaborative development practices to graphics projects
4. **Design, plan, and critically evaluate graphics solutions to visual computing problems**:
   - Work with user requirements and technical constraints for real-time rendering
   - Assess possible solutions for visual fidelity, performance, and ethical implications
5. **Assess graphics systems in terms of quality attributes and trade-offs**:
   - Evaluate rendering techniques based on performance, visual quality, and resource usage
   - Analyze performance metrics and visual quality through empirical testing
   - Optimize graphics applications through techniques like instancing and efficient resource management
6. **Experiment with and analyze graphics techniques through practical applications**:
   - Complete projects demonstrating real-world applications of computer graphics
   - Communicate findings and defend technical decisions in presentations and documentation

6 Assignments and Exams

6.1 Moodle/e-Course Checkpoints

Students are required to maintain private GitHub repositories provided by the instructor for their assignments. They must periodically commit and push a specific number of lab solutions as directed by the faculty. Professors or teaching assistants will review the work either during the lab (on-site) or after the submission deadline (off-site) and assign points based on the completed assignments.

6.2 Labs and Projects

Throughout the course, students will be assigned several laboratory tasks. Additionally, they are required to develop one course project. Students are expected to defend their work to the instructor during both the Midterm and Final Examination periods.

7 Course Materials, Recordings and Screencasts

All course materials are available on GitHub at `https://github.com/auca/com.391`. By using GitHub, students will gain familiarity with the Git version control system and the widely-used GitHub service among developers.

Every class will be screencasted and uploaded to YouTube for student accessibility, though it's important to note that we do not guarantee every class will be recorded. Recordings will be done on a best-effort basis as time permits. Consider recording the class videos on your own computer if you need them to be available promptly. YouTube recordings can be located in the course repository at `https://github.com/auca/com.391`. While recordings provide flexibility, they should not be a substitute for attending classes. Active participation is crucial for success in this course. Each unexcused absence will result in a one-point deduction from your grade. Accumulating five or more unexcused absences may lead to an X grade. If overall attendance is poor, the instructor reserves the right to discontinue class recordings.

Access the course lectures remotely via Zoom at `http://com-391-zoom.auca.space`. When joining the Zoom session, students must identify themselves by providing their first and last names in Latin characters, properly capitalized.

8 Software

Students are advised to install the following software on their machines at the start of the course. Additional installations may be required later.

- Git: `https://git-scm.com`
- Python 3: `https://www.python.org`
- CMake: `https://cmake.org`
- Conan: `https://conan.io`

On macOS install Xcode Command Line Tools `https://developer.apple.com/xcode/resources`. On Windows install Visual Studio 2022 Community `https://visualstudio.microsoft.com/vs` or any other edition if you can acquire it legally

You can use any IDEs (like CLion) or code editors (like VS Code, Vim, Emacs) that you like. Please note that we can only provide support for CLion.

You can also work on Linux, but you will need to manage the installation process for drivers, tools, and editors by yourself.

9 Hardware

We have made every effort to choose a graphics API that will likely work on a wide range of machines, from high-end personal computers to low-powered notebooks. However, each year we encounter students whose computers have outdated GPUs or buggy drivers that are unable to run our code. Unfortunately, this means you will need to invest some effort into finding an environment that is compatible with our

course. Please be aware that we cannot provide extensions or preferential treatment if your machine is unable to handle our code. One option is to try a different native or virtualized environment (e.g., GNU/Linux) on your computer. Open-source drivers often perform better on older devices than those provided by the manufacturer. Additionally, you can reach out to the AUCA IT department for assistance. They may be able to provide a computer capable of running the programs, but please note that their resources are limited.

10 Reading

1. Computer Graphics: Principles and Practice by John Hughes (AUCA Library Call Number: T 385 C67 1996, ISBN: 978-0321399526)
2. 3D Math Primer for Graphics and Game Development, Second Edition by Fletcher Done and Ian Parberry (ISBN: 978-1568817231, available on the author's website)

10.1 Supplemental Reading

1. Mathematics for 3D Game Programming and Computer Graphics, Third Edition by Eric Lengyel (AUCA Library Call Number: QA76.76. C672 L46 2012, ISBN: 978-1435458864)
2. Foundations of Game Engine Development, Volume 1: Mathematics, by Eric Lengyel (AUCA Library Call Number: QA402.5. L46 2016, ISBN: 978-0985811747)
3. Foundations of Game Engine Development, Volume 2: Rendering, by Eric Lengyel (AUCA Library Call Number: QA76.76. C672 L462 2019, ISBN: 978-0985811754)

11 Grading

11.1 Moodle/e-Course Checkpoints

Your instructor will announce a periodic review of your work. You will be awarded up to the following number of points for such checks.

- Labs (25%)
- Project (10%)

11.2 Exams

- Midterm Exam (30%)
- Final Exam (35%)

11.3 Totals

- 100% is formed from the Moodle/e-Course Checkpoints (35%) and the two exams (65%).

11.4 Scale

- [94%–100]%: A
- [90%–94)%: A-
- [87%–90)%: B+
- [83%–87)%: B
- [80%–83)%: B-
- [77%–80)%: C+
- [73%–77)%: C
- [70%–73)%: C-
- [67%–70)%: D+
- [63%–67)%: D
- [60%–63)%: D-
- Less than 60%: F

Please note that requests for a higher grade due to points being marginally close will be ignored. For instance, 93.99 is an A-, NOT an A. Similarly, requests for extra assignments to boost points will also be disregarded.

12 Rules

First and foremost, in addition to all the rules listed in the syllabus document, students are required to follow the Code of Conduct of the American University of Central Asia.

12.1 Participation

Active work during the class may be awarded with extra points at the instructor's discretion. Poor student performance during a class can lead to points being deducted from the final grade.

Instructors may conduct pop-checks during classes at random without prior notice. Students MUST be ready for every class in order not to lose points. Students absent without a good reason from such classes with graded work will also lose points unless it is force-majeure circumstances. Instructors must be notified in advance about why a student is absent not to lose points.

12.2 Questions

We believe that a question from one student is most likely a question that other students are also interested in. That is why we encourage students to use the online

discussion board of the LMS (Learning Management System) that you use (e.g., AUCA e-Course System) to ask questions in public that other students can see and answer.

Do not post the complete source code for any task on the LMS discussion board. You will get zero for that work for any such public post. Do not ask generic questions about your code to know why it does not work. Please spend some time thinking about your code, debugging it.

12.3 Late Policy

Late submissions and late exams are not allowed. Exceptions may be made at the professor's discretion only in force-majeure circumstances. If you got ill, got severe personal issues, got problems with your computer or the Internet, you MUST notify instructors at least 24 hours in advance. Otherwise, we will not give you an extension. We will consider that you were procrastinating until the very last day. We will also not be giving more than one emergency extension throughout the course.

Six hours before the deadline for any work on the course, instructors will go into a silent mode. No questions will be answered about the work that has to be submitted, no requests to have office hours will be considered. However, at any other work time before the deadline, we will try our best to answer your questions and help you through Zoom or in our office.

12.4 Exam and Task Submission Ceremonies

Students MUST follow exam and task submission ceremonies. It means they MUST strictly follow all the rules specified by the instructors in written or verbal form. Failure to do so will result in lost points. Throughout your career, you will have to work with various supporting documents (contracts, timesheets, etc.). It is a good idea to start learning to work with such documents accurately early. We will remove points for not following these rules or even refuse to accept your exam defense or tasks submitted to us. We will also give zero for not following deadlines or the strict exam timing rules.

12.5 Administrative Drop

Instructors have a right to drop a student from the course for non-attendance. If you have five classes or more missed without an excuse, the faculty may consider dropping you by giving you the X grade.

12.6 Incomplete Grade

Similar to the policy for late exams, the grade I may be awarded only in highly exceptional circumstances. Students MUST initiate a discussion about receiving an I grade with the instructors well in advance and NOT during the last week before final exams.

12.7 Academic Honesty

Plagiarism is the act of copying or stealing someone else's words or ideas and presenting them as one's own. This definition encompasses various task elements, including but not limited to program code, comments, software documentation, abstracts, reports, diagrams, and statistical tables.

The following are examples of plagiarism in the context of a Software Engineering course:

- Presenting code written by others or AI as your own
- Purchasing code, software, or any project-related content from online platforms or other sources and submitting it as your own creation
- Using algorithms, patterns, or architectural designs without acknowledging the source
- Incorporating code snippets, sentences, design patterns, or any intellectual content from sources, published or unpublished, without proper citation
- Modifying someone else's code or design (e.g., changing variable names, changing the structure of the code) and claiming it as original
- Utilizing graphics, data sets, audio, video, or other elements from external works without proper acknowledgment.

Engaging in plagiarism is not only unethical but also undermines the educational process. The consequences for plagiarism in this course for all parties involved are as follows:

- First instance: The students will receive a grade of zero for the plagiarized work, and a report will be filed with the Registrar's Office.
- Second instance: The students will receive an F grade for the entire course.

It's important to note that both parties involved in plagiarism—the one who plagiarizes and the one whose work was copied—will face equal consequences. This underscores the imperative for honest students to exercise caution in ensuring the security of their work. It is the student's responsibility to guarantee that their assignments, code, or any related content can only be accessed by them and the course instructors. Sharing, unintentionally exposing, or not securely storing one's work can lead to unintended consequences and sanctions.

The use of artificial intelligence, including but not limited to generative AI tools, to complete any assignments, projects, or exams, either off-site or on-site, is strictly prohibited. If there is suspicion that a student has used AI assistance in their work, the student will be required to perform a similar task and answer relevant questions in a supervised setting with the course instructors. Failure to satisfactorily complete this task or to answer questions convincingly will lead to the conclusion that the work was AI-generated. In such cases, the Academic Honesty policies outlined previously will be enforced, including potential disciplinary actions. Note that for some tasks, the instructor may allow the use of AI tools, but you must obtain explicit written permission in the requirements to use them.

Students are advised against rote memorization of code for examinations. Relying solely on memorization is an ineffective learning strategy in programming. Examinations in this course may contain open-ended questions targeting the student's analytical and design skills, and memorization may lead to answers that are off-target and of subpar quality.

In addition to the rules outlined in this syllabus, we abide by all global university policies concerning plagiarism. Should the global university rules evolve to be more consequential or stringent than what is stipulated here, those university-wide regulations will take precedence over our course-specific rules.

12.8 Access and Support Services

In this course, we are committed to providing an inclusive learning environment that accommodates the diverse needs of all students. If you have a disability or require specific accommodations to participate fully in this course, please contact us as early as possible to discuss your needs. We will work together to ensure that appropriate adjustments are made to support your learning experience.

If you need guidance on improving your time management, presentation, writing skills, and study skills beyond the scope of Software Engineering parlance, we encourage you to connect with the Advising Office. The Advising Office offers a wide range of practical and creative workshops, and peer advisors can help you navigate the academic environment.

If you feel more comfortable working with other students, you can make an appointment with a student tutor from the WARC (Writing and Academic Resource Center). You can meet with them face-to-face or online. Visit the WARC webpage for more resources. Remember, team/group work with other students in this course is not allowed.

If you feel stressed, overwhelmed, find it difficult to manage your emotions, socialize, or have concerns related to your mental health, please consult the AUCA Counseling Service. There are excellent professionals working there; your visits are completely confidential and free.