USC Viterbi
School of Engineering

Video Game Programming

ITP 380 (4 Units)

Objective

This course provides students with an in-depth introduction to technologies and techniques used in the game industry today. At semester's end, students will have:

1. Implemented a variety of core game systems (incl. rendering, input, sound, and collision/physics)
2. Developed a strong understanding of essential mathematics for games
3. Written multiple-fully functional games with MonoGame/C#, both individually and as a team.
4. Learned the critical thinking required to continue further study in the field

Concepts

3D mathematics for games. Working with MonoGame. 3D graphics. Collision detection. Introduction to game physics. Events and scripting. Introduction to animation and A.I. Implementing game play. Getting a job in the game industry.

Prerequisites	CSCI 104 or ITP 365x
Instructor	Sanjay Madhav
Contact	**Email:** madhav@usc.edu
Office Hours	M/W 3:30–4:50PM and T 2-3PM, all in OHE 530H
Lab Assistant	TBA
Lecture	M 5-7PM in KAP 267
Lab	W 5-7PM in KAP 267

Course Structure

For the first half the semester, students will work on individual labs that reinforce the concepts covered during lecture. The first three individual labs complete "MonoStroids," a simple asteroids clone that's designed to teach the basics of game programming. The next four individual labs complete "MonoDefense," a tower defense game that includes a pathfinding AI.

During the second half of the semester, students will break into groups of 3-4 students and work on a larger game project. Further details regarding this project are provided on the next page.

The individual assignments must be completely *individually*. And of course, the group project will be completed with a group.

There is a midterm exam, but instead of a final exam, the final group project will be presented during the final exam time slot.

There will periodically be "pop" quizzes. These quizzes will generally be short 20-30 minute programming assignments, and can occur during any class session. There are a total of six quizzes, with the lowest quiz grade being dropped.

Textbooks

XNA Game Studio 4.0 Programming. Tom Miller and Dean Johnson. ISBN-10: 0672333457.

Game Engine Architecture. Jason Gregory. ISBN-10: 1568814135.

Grading

The course is graded with the following weights:

Individual Labs (5% each)	35%
Final Project	35%
Midterm Exam	20%
Quizzes (6 total, lowest grade dropped)	10%
TOTAL POSSIBLE	100%

Grading Scale

Letter grades will be assigned according to the following scale:

93%+	A
90-92%	A-
87-89%	B+
83-86%	B
80-82%	B-
77-79%	C+
73-76%	C
70-72%	C-
69	D+
67-68	D
66	D-
65 and below	F

Half percentage points will be rounded up to the next whole percentage. So for instance, 89.5% is an A-, but 89.4% is a B+.

There is no curving. Students will receive the grade they earn. Extra credit is generally not offered.

Final Project

At the midpoint of the semester, students will break into groups of 3-4 to work on a larger 3D game of their own design. Before commencing work on their project, students must have their proposal approved by the instructor. Working on a project which is not approved is not allowed.

Schedule

Week 8 – Form team and submit proposals for approval.
Weeks 9 to 11 – Work on core game feature sets.
Week 12 – “E3” Playable Demo Presentation (Graded)
Week 13 to 15 – Work on final polish features
Final Exam – Final Presentation (Graded)

Basic Requirements

The game must be implemented in C# using MonoGame. It must utilize 3D functionality, at a minimum, have 3D rendering, input, sound, collision detection, and some sort of physics.

The game must feel like a complete experience. A player should be able to load in the game, there should be some sort of instructions on how to play, and it should be able to restart and exit. If the game is level-based, it should end after the level(s) are complete. If it’s infinite gameplay, then it should play for as long as the player is successful.

Each team member is assigned responsibility for a specific technical challenge known as a “major feature.” Such features include a significant amount of gameplay work, A* pathfinding, vertex/pixel shaders, particle systems, skeletal animation, continuous collision detection, and so on. For advice on what constitutes a “major feature,” please consult with the instructor.

Project Management

Students are required to use Trello (http://www.trello.com) to manage the schedule for their final project. Further instruction will be given during class once work on final projects begins.

Proposal

Your team will be required to send out a proposal prior to commencing work on the project. The necessary components of the proposal will be covered in class.

Grading

Your final game project is **35%** of your overall semester grade. The final project breakdown is as such:

Individual Contribution	10%
“E3” Demo Grade	10%
Final Project Grade	15%
Total	**35%**

Individual Contribution – 10% of semester grade. This is based on how much code you contributed to the project, how well your assigned “major feature” was implemented, and the result of peer reviews within your group. If you did not contribute enough to the project, you will lose points in this section.

“E3” Demo Grade – 10% of semester grade, shared by all team members. Approximately four weeks prior to the final presentations, your group will be required to present an “E3” demo to the class. This demo must demonstrate the core mechanics of your game in a playable manner. It is not expected to be fully-featured, but the gist of your game should be apparent.

Final Project Grade – **15%** of semester grade, shared by all team members. It is broken down in the following manner:

30% – Basic project requirements met.
45% – Major features as outlined in proposal implemented.
25% – Overall quality of the game, relative to other projects both in this particular class section and historically.

Peer Reviews

The last week of class, students will be required to complete a peer review survey on Blackboard. The results of this evaluation will be taken into account when assigning the individual contribution grades.

Policies

Make-up policy for exams: To make up for a missed exam, the student must provide a satisfactory reason (as determined by the instructor) along with proper documentation. Make-ups are only allowed under extraordinary circumstances.

Late Assignments: Late individual assignments will be accepted with a 15% penalty per day late, up to a maximum of two days late (for -30%).

The final project will *not* be accepted late. It will be graded based on the submission at the required time of presentation.

Before logging off a computer, students must ensure that they have emailed or saved projects created during the class or lab session. Any work saved to the computer will be erased after restarting the computer.

ITP offers Open Lab use for all students enrolled in ITP classes. These open labs are held beginning the second week of classes through the last week of classes. Hours are listed at: http://itp.usc.edu/labs/.

Academic Integrity

USC seeks to maintain an optimal learning environment. General principles of academic honesty include the concept of respect for the intellectual property of others, the expectation that individual work will be submitted unless otherwise allowed by an instructor, and the obligations both to protect one's own academic work from misuse by others as well as to avoid using another's work as one's own. All students are expected to understand and abide by these principles. *SCampus*, the Student Guidebook, (www.usc.edu/scampus or http://scampus.usc.edu) contains the University Student Conduct Code (see University Governance, Section 11.00), while the recommended sanctions are located in Appendix A.

Students will be referred to the Office of Student Judicial Affairs and Community Standards for further review, should there be any suspicion of academic dishonesty. The Review process can be found at: http://www.usc.edu/student-affairs/SJACS/. Information on intellectual property at USC is available at: http://usc.edu/academe/acsen/issues/ipr/index.html.

In this class, all code submissions will be ran against current, previous, and future students using MOSS, which is a code plagiarism identification tool. If your code significantly matches another student's submission, you will be reported to SJACS.

Generally, the rule of thumb is that it is acceptable to discuss solutions to problems with other students, but once you are looking at someone else's code, it crosses over into the realm of cheating. It does not matter if this code is online or from a student you know, it is cheating in all situations. Do not share your code with anyone else in this or a future section of the course, as allowing someone else to copy off your code carries the same penalty as you copying the code yourself.

Note that there is one exception: for the final project, it is acceptable use external code (from the Internet) to solve certain problems, but please consult with the instructor before doing so to ensure you are not in violation.

Students with Disabilities

Any student requesting academic accommodations based on a disability is required to register with Disability Services and Programs (DSP) each semester. A letter of verification for approved accommodations can be obtained from DSP. Please be sure the letter is delivered to me (or to TA) as early in the semester as possible. DSP is located in STU 301 and is open 8:30 a.m.–5:00 p.m., Monday through Friday. Website and contact information for DSP: http://sait.usc.edu/academicsupport/centerprograms/dsp/home_index.html, (213) 740-0776 (Phone), (213) 740-6948 (TDD only), (213) 740-8216 (FAX) ability@usc.edu.

Emergency Preparedness

In case of a declared emergency if travel to campus is not feasible, USC executive leadership will announce an electronic way for instructors to teach students in their residence halls or homes using a combination of Blackboard, teleconferencing, and other technologies.

Please activate your course in Blackboard with access to the course syllabus. Whether or not you use Blackboard regularly, these preparations will be crucial in an emergency. USC's Blackboard learning management system and support information is available at blackboard.usc.edu.

Course Outline

Week 1 (8/26 and 8/28) – Course Intro, Intro to C#, and Game Programming Basics

- Course Introduction
- The Game Loop
- "Time" in games
- Game Objects: What they are and different representations
- Data Structures Review

Reading: *Gregory*: §1.1 - §1.3, §7.4 - §7.5; *Miller*: Chapter 3

Lab: MonoStroids, Part 1: MonoGame/C# setup, Game Objects

Week 2 (9/4) – Vector Math and Games

- Vectors
- Applications
- Why vector math is valuable

Reading: *Gregory*: §4.1 - §4.2; *Miller*: Chapter 4 (pp. 41-53)

Lab: MonoStroids, Part 2: Loading/rendering models, spawning asteroids

No class 9/2 due to Labor Day.

Week 3 (9/9 and 9/11) – Rendering, Part 1

- Color systems/formats
- Rendering Pipeline
- Matrices, Coordinate Spaces, and Projections
- Painter's Algorithm and Z-Buffer

Reading: *Gregory*: §4.3, §10.1; *Miller*: Chapter 4 (pp. 53-83), Chapter 5

Lab: MonoStroids, Part 3: Basic input and collision

MonoStroids DUE Sunday, 9/15 @ 11:59PM

Week 4 (9/16 and 9/18) – Quaternions and Input/Sound

- Euler Rotations and Quaternions
- Human Input Devices and Event-Based Input Systems
- Sound basics

Reading: *Gregory*: Chapter 8 (pp. 339-350); *Miller*: Chapter 11 (up to 334), Chapter 13

Lab: MonoDefense, Part 1: Advanced game objects

Week 5 (9/23 and 9/25) – Artificial Intelligence

- AI state machines
- A* and basic path finding
- Navigation Meshes

Reading: AI reading on Blackboard

Lab: MonoDefense, Part 2: A* Pathfinding

Week 6 (9/30 and 10/2) – Collision Detection and Game Physics, Part 1

- Types of Collision Geometry
- Sphere/Sphere (swept) and Sphere/Plane Collision
- Rigid-Body Dynamics and Numeric Integration

Reading: *Gregory*: §12.1; §12.3.4; §12.3.5.1 - §12.3.5.3; §12.4.2 - §12.4.4

Lab: MonoDefense, Part 3: Core gameplay loop

Week 7 (10/7 and 10/9) – **Midterm Exam during lecture hours on 10/7**

Lab: MonoDefense, Part 4: Finalizing MonoDefense

MonoDefense DUE Sunday, 10/13 @ 11:59PM

Week 8 – (10/14 and 10/16) Camera Systems

- Field of view
- Basic camera styles
- Follow cameras

Reading: Camera reading on Blackboard
Lab: Form teams for group project.

Week 9 (10/21 and 10/23) – Rendering, Part 2

- Lighting, Shading, and Texture Mapping
- Advanced Mapping, Shadows, Particle Effects
- Vertex and Pixel Shaders overview

Reading: *Gregory*: §4.1 - §4.4.1, §10.2; *Miller*: Chapter 6 (pp. 105-121)
Lab: Continue work on group project

Week 10 (10/28 and 10/30) – Collision Detection and Game Physics, Part 2

- Raycasting (Ray/Plane and Ray/Sphere)
- Introduction to advanced (angular) dynamics

Reading: *Gregory*: §12.3.7; §12.2
Lab: Continue work on group project

Week 11 (11/4 and 11/6) – Event-Based Systems and Scripting Languages

- Closer look at event-based systems
- Scripting Languages
- Case Study: UI Mods in *World of Warcraft*

Reading: *Gregory*: Chapter 14 (pp. 773-817)
Lab: Continue work on group project

Week 12 (11/11 and 11/13) – Animation and Assorted Gameplay

- Evolution of Animation systems (2D to key framed to skinning)
- Skeletons and Poses
- Gameplay Systems for different genres

Reading: *Gregory*: Chapter 11 (pp 491-518); *Miller*: Chapter 6 (pp. 127-140)
Lab: **“E3” Group Project Presentations on 11/13 during lab hours**

Week 13 (11/18 and 11/20) – Networking in MonoGame

- Separating players
- Packet writer and Game services

Reading: *Miller*: Chapters 15 and 16
Lab: Continue work on group project

Week 14 (11/25) – User Interface Systems

- Event-based UI systems
- Implementing a HUD

Reading: UI reading on Blackboard
Lab: Continue work on group project
No class 11/27 due to Thanksgiving.

Week 15 (12/2 and 12/4) – Conclusion

- Where to go from here?
- Engines, mobile game, multiplayer games, etc.
- How to get into the game industry

Lab**:** Continue work on group project

Final Group Project Presentations on 12/11 @ 4:30PM