USC SCHOOL OF ENGINEERING

Programming Game Engines
ITP 485 (4 Units)

Objective This course provides students with an in-depth exploration of 3D game engine architecture. Students will learn state-of-the-art software architecture principles in the context of game engine design, investigate the subsystems typically found in a real production game engine, survey some engine architectures from actual shipping games, and explore how the differences between game genres can affect engine design. Students will participate in individual hands-on lab exercises, and also work together like a real game development team to design and build their own functional game engine by designing and implementing engine subsystems and integrating 3rd party components.

Concepts Engine subsystems including rendering, audio, collision, physics and game world models. Large-scale C++ software architecture in a games context. Tools pipelines for modern games.

Prerequisite CSCI-102, ITP-380

Lecture 2 hrs/week

Lab 2 hrs/week

Office Hours Mondays 1:00pm – 2:00pm, ITP Think Tank

Course Structure Lectures given on Monday. Project assignments and some supplemental lectures given on Wednesday; instructor and/or TA available for questions and help during Wednesday lab sessions, over email, and by appointment (at least 24 hours notice please).

Textbooks

Required:

1. *ITP-485 Course Notes.* Jason Gregory.
2. *Pro Ogre 3D Programming.* Gregory Junker. ISBN 1-59059-710-9.
3. *Mathematics for 3D Game Programming & Computer Graphics.* Eric Lengyel. ISBN 1-58450-277-0.

Recommended:

4. *3D Game Engine Architecture: Engineering Real-Time Applications with Wild Magic.* David H. Eberly. ISBN 0-122290-64-X.
5. *Large-Scale C++ Software Design.* John Lakos. ISBN 0-201633-62-0.

Grading Final grade is based upon the student's score on individual lab assignments, the midterm exam, and the grades earned on a multi-lab team-based project. Grades for each team lab are assigned based on an individual contribution component and an overall grade assigned to the lab as a whole. For the team project, it is especially important that students display their names on all documentation and source code you personally produce, so that the instructor will know which student contributed which material.

Individual Labs (Labs 1-6)	30%
Midterm Exam	30%
Team Project: Individual Contribution	20%
Team Project: Overall Project Grade	20%
TOTAL POSSIBLE	100%

Letter grades are assigned according to the following table.

94% - 100%	A
90% - 93%	A-
87% - 89%	B+
84% - 86%	B
81% - 83%	B-
77% - 80%	C+
74% - 76%	C
71% - 73%	C-
69% - 70%	D+
67% - 68%	D
65% - 66%	D-
0% - 64%	F

Policies It is the student's responsibility to complete both individual and team lab assignments on or before deadlines as set by the instructor.

Academic Integrity

- The use of unauthorized material, communication with fellow students during an examination, attempting to benefit from the work of another student, and similar behavior that defeats the intent of an examination or other class work is unacceptable to the University.

- Although working together is encouraged, all work claimed as yours must in fact be your own effort. Students who plagiarize the work of other students will receive zero points and possible referral to the Office for Student Conduct.

- All students should read, understand and abide by the University Student Conduct Code
  http://www.usc.edu/dept/publications/SCAMPUS/governance/gov03.html

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 your 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. The phone number for DSP is (213) 740-0776.

Programming Game Engines

ITP 485 (4 Units)

Course Outline

Week 1 – Introduction

- Course overview
- What is a game?
- What is a game engine?
- Engine differences between game genres
- Survey of game engine subsystems
- Review of C++ and best practices
- Game loops and real-time simulation
- Simple example: The PONG game loop
- Time in games
- Introduction to Ogre3D

Reading:

- Course notes: A.1 – A.7
- Junker: Chapters 1 – 4
- Ogre Manual (online): Sections 1, 2

Week 2 – No Lecture (Holiday)

Lab:

Pongre 1: Getting Started with Ogre3D

- OpenSVN set-up
- Install Ogre SDK
- Build and explore Ogre samples

Week 3 – 3D Math for Games

- Points, vectors and Cartesian coordinates
- Vector operations, dot and cross product
- 2D and 3D matrices, homogeneous coordinates
- Hierarchical coordinate frames, change of basis
- Other rotational representations
  - Axis+angle, Euler angles, quaternions
- Lines, line segments and rays
- Spheres
- Planes and frusta
- AABBs and OBBs

Reading:

- Course notes: A.8 (except A.8.12)
- Lengyel: Chapters 1 – 3
- OPTIONAL: Eberly: Section 2.2

Lab:

Pongre 2: Using the Ogre App Wizard

- Create project with Ogre app wizard
- Create simple bouncing ball (Ogre head) in 1D

Week 4 – 3D Graphics Fundamentals

- Coordinate frames used in rendering: object space, world space, view space, screen space
- Anatomy of 3D geometric data
- Transformation into screen space
- Rasterization: vertex attributes and interpolation, texturing
- Depth, alpha and stencil
- Culling and clipping
- Color and lighting

Reading:

- Course notes: C.1
- Ogre Manual (online): Sections 3, 4
- Lengyel: Chapters 0, 4, 6
- OPTIONAL: Eberly: Chapters 1, 3

Lab:

Pongre 3: Bouncing With Vector Math

- Use vector math to find new position of ball each frame, given velocity vector
- Calculate reflection of ball off a plane, use to bounce ball off all four walls of playfield
- Use Ogre's prefab plane models to visualize location of walls
- Extra credit: Allow walls to be at any angle, not just axis-aligned

Week 5 – Renderer Architecture

- Survey of typical game renderer layers
  - Drivers and hardware abstraction layer
  - Hardware-accelerated pipeline SDK
  - Low-level renderer
  - Material system and render state management
  - Scene graph
  - Interface to the game
- Hardware-accelerated 3D pipelines
  - Overview of OpenGL
  - Comparison to Direct3D
- Low-level renderer
- Material system and render state management
- Scene graphs

Reading:

- Course notes: C.2
- Junker: Chapters 5, 6
- Lengyel: Chapter 7
- OPTIONAL: Eberly: Sections 4.1, 4.2

Lab:

Pongre 4: Paddles & Basic Game Play

- Create simple box meshes for paddles
- Use Ogre's OIS human input device system to read joypad/keyboard and move paddles
- Detect collisions with paddles and bounce ball
- End/reset game if ball misses paddle and hits left/right edge of playfield

Week 6 – No Lecture (Holiday)

Lab:

Pongre 5: Congratulations, It's a Game!

- Add HUD, scoring, and personalization of your choice

Week 7 – Software Architecture for Games, Low-Level Engine Services

- Intro to software architecture for games
- Large-scale C++ systems
- Logical versus physical design
- Standard data types, strings and hashed string ids
- Subsystem start-up and shut-down
- Memory management: allocation, fragmentation, alignment
- File system, engine configuration and resource management
- Human input devices (HID)
- Debugging/development tools, debug drawing

Reading:

- Course notes: A.9, B.1 – B.10
- RECOMMNEDED: Lakos: Ch. 0; Ch. 5 sections 5.1 – 5.3, 5.7
- RECOMMENDED: Junker: Chapter 7
- OPTIONAL: Eberly: Section 2.1

Week 8 – MIDTERM EXAM

- Midterm review and preparation

Reading:

- Review course notes: Sections A and C

Lab:

MIDTERM EXAM during lab hours

Week 9 – Game Object Model 1

- What is a game object model?
- Survey of typical requirements and features
- Common object model architectures
- Attributes, RTTI and serialization
- World Editors

Reading:

- Course notes: D.1 – D.3
- OPTIONAL: Eberly: Section 8.1

Lab:

Team Labs

- Teams work on various engine and game play systems in parallel

Spring Break (No Lecture, No Lab)

Week 10 – Character Animation

- Skinned hierarchical animation
- Ogre's animation system
- Interface between player/non-player characters and animation

Reading:

- Course notes: F.1 – F.5
- Ogre Manual (online): Section 8
- RECOMMENDED: Junker: Chapter 9
- OPTIONAL: Eberly: Section 4.5

Lab:

Team Labs

- Teams work on various engine and game play systems in parallel

Week 11 – Player Mechanics

- Player mechanics in various game genres
- Basic 3D player mechanics
- Platformer and FPS player mechanics in depth:
  - Movement mechanics
  - Game cameras
  - Weapon systems

Reading:

- Course notes: H.1 – H.3

Lab:

Team Labs

- Teams work on various engine and game play systems in parallel

Week 12 – Game Object Model 2

- Event/messaging system
- Finite state machine support
- Scripting languages

Reading:

- Course notes: D.4 – D.6

Lab:

Team Labs

- Teams work on various engine and game play systems in parallel

Week 13 – Collision and Physics

- Collision
  - Collision detection basics
  - The Gilbert-Johnson-Keerthi (GJK) algorithm
  - Typical collision system architectures
  - Spatial subdivision trees
  - Broad phase, narrow phase, collision islands

- Game Physics
  - Basic rigid body dynamics
  - Force, acceleration, velocity, mass, momentum
  - Angular dynamics, moment of inertia
  - Collision response
  - Numerical integration
- Typical Coll/Phys APIs: ODE, PhysX, Havok

Reading:

- Course notes: E.1 – E.5
- Lengyel: Chapters 8, 11, 12, 14
- OPTIONAL: Eberly: Chapters 6, 7

Lab:

Team Labs

- Teams work on various engine and game play systems in parallel

Week 14 – Some Other Common Engine Components

- Game Audio
- High-Level Game Flow Systems
- HUD/GUI
- Particle Systems
- Movie Player
- Multiplayer Networking

Reading:

- Course notes: G.1 – G.6
- Junker: Chapters 10, 12
- Lengyel: Chapter 9

Lab:

Team Labs

- Teams work on various engine and game play systems in parallel

Week 15 – Wrap-Up

- Overflow topics as necessary
- Getting into the game industry – resumes, interviews, demos
- Life in the game industry

Lab:

Team Labs

- Teams perform final integration and put finishing touches on game engine project
- Code freeze one day prior to Gold Master

Week 16 – Final Game Engine Project Due