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 and ITP-380

Lecture 2 hrs/week

Lab 2 hrs/week

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 office hours, lab sessions, via email, and by appointment.

Textbooks

Required:

1. Jason Gregory. *Game Engine Architecture.* AK Peters. ISBN 978-1-56881-413-1.

Recommended:

2. Eric Lengyel. *Mathematics for 3D Game Programming & Computer Graphics.* Charles River Media. ISBN 978-1-58450-277-7.
3. Gregory Junker. *Pro Ogre 3D Programming.* Apress. ISBN 978-1-59059-710-1.
4. John Lakos. *Large-Scale C++ Software Design.* Addison-Wesley Professional. ISBN 978-0-20163-362-7.

Grading Final grade is based upon the student's score on individual labs and assigned problems, the midterm and final exams, and the grades earned on a multi-lab team-based project. 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 and Assigned Problems	20%
Midterm Exam	20%
Final Exam	20%
Team Project: Individual Contribution	20%
Team Project: Overall Project Grade	20%
TOTAL POSSIBLE	100%

Policies It is the student's responsibility to complete both individual and team-based assignments on or before deadlines as set by the instructor. Before logging off a computer, students must ensure that they have submitted to Subversion any work created during the class or lab session. Any work saved to the local computer's disk drive may be erased after restarting the computer. ITP is not responsible for any work lost. 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. Please contact your instructor for times and days for the current semester.

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. It is often difficult to distinguish between a culpable act and inadvertent behavior resulting from the nervous tension accompanying examinations. When the professor determines that a violation has occurred, appropriate action, as determined by the instructor, will be taken.
- 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 possibly be referred to Student Judicial Affairs and Community Standards (SJACS).
- All students should read, understand, and abide by the University Student Conduct Code listed in SCampus, and available at: http://www.usc.edu/student-affairs/SJACS/nonacademicreview.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 engine?
- Engine differences between game genres
- Survey of runtime engine subsystems
- Survey of tools and the asset pipeline

Reading:

- Course text: 1.2 – 1.7

Week 2 – Tools of the Trade

- Version control and Subversion
- Microsoft Visual Studio tips and tricks
- Profiling tools
- Memory leak / corruption detection
- Other tools

Reading:

- Course text: 2.1 – 2.5

Lab:

Pongre 1: One-Dimensional Bouncing Ball

- Create project by copying template
- Create simple bouncing ball (Ogre head) in 1D

Week 3 – 3D Math for Games

- Review: Vectors, Matrices, Quaternions
- Spheres
- Lines, line segments and rays
- Planes
- Splines

Reading:

- Course text: 4.1 – 4.5 (review), 4.6

Lab:

3D Math Problems

- Work problems, see typical applications, practice

Pongre 2: Bouncing, Paddles and Basic Gameplay

- Create simple box meshes for paddles
- Use Ogre's OIS human input device API to move paddles
- Detect collisions with paddles and bounce ball
- End/reset game if ball misses one of the paddles

Week 4 – More 3D Math for Games

- Integer and IEEE floating-point formats
- Hardware-accelerated vector math with SIMD

- Random number generation

Reading:

- Course text: 3.2.1, 4.7 – 4.8

Lab:

Pongre 3: Congratulations, It's a Game!

- Add HUD, scoring, and personalization of your choice

Week 5 – Software Engineering for Games

- C++ best practices
- Data, code and memory
- Errors, exceptions and assertions
- Memory allocation and management
- Container data structures
- Strings and hashed string ids

Reading:

- Course text: 3.1, 3.2.2 – 3.2.5, 3.3, 5.2 – 5.4
- RECOMMENDED: Lakos: Ch. 0; Ch. 5 sections 5.1 – 5.3, 5.7

Lab:

Pongre 4: Final Polish

- Polish your Pongre and make it kick some butt!

Week 6 – The Graphics Pipeline

- Review: Triangle meshes, materials, texturing, transformation, lighting basics
- Pipelining concepts
- The rendering pipeline
  - Tools stage
  - Asset conditioning stage
  - Application stage: Visibility determination and primitive submission
  - Geometry processing and rasterization stages: GPU architecture
- Introduction to global illumination and programmable shaders

Reading:

- Course text: 10.1 (review), 10.2.1 – 10.2.5

Lab:

Team Project

- Teams decide on game design, write up minimal design docs

Week 7 – Rendering Engine Architecture

- Optimization: the driver of rendering engine architecture
- Primitive submission and render state management
- Sorting, alpha blending and Z pre-pass
- Visibility determination and scene graphs
- Rendering engine architecture
- Visual effects: Particles, overlays, decals, post processing
- Graphical tools for debugging and development

Reading:

- Course text: 10.2.6 – 10.2.7, 10.3 – 10.5, 9.1 – 9.8

Lab:

Team Project

- Teams work on engine and game play systems in parallel

Week 8 – MIDTERM EXAM

- Midterm review and preparation

Reading:

- Review prior weeks' readings

Lab:

MIDTERM EXAM during **lab hours**

Week 9 – Animation System Architecture

- Review of character animation fundamentals
- Blending: LERP and additive
- Procedural animation, IK and other forms of post-processing
- Compression techniques
- Animation system architecture and pipeline
- Interfaces between game characters and animation
- Animation state machines and layering

Reading:

- Course text: 11.6 – 11.12
- Ogre Manual (online): Section 8
- RECOMMENDED: Junker: Chapter 9

Lab:

Team Project

- Teams work on engine and game play systems in parallel

Week 10 – Advanced Collision Detection

- Review: Collision detection basics
- Fast-moving bodies and the bullet-through-paper problem
- The Gilbert-Johnson-Keerthi (GJK) algorithm
- The AABB prune and sweep algorithm
- Ray and sphere casting

Reading:

- Course text: 12.3.5.5 – 12.3.5.7, 12.3.6, 12.3.7

Lab:

Team Project

- Teams work on engine and game play systems in parallel

Week 11 – Advanced Rigid Body Dynamics

- Review of point mass linear dynamics, numerical integration
- Angular dynamics, moment of inertia
- Collision response
- Constraints and ragdolls
- Typical physics/collision system architectures
- API case studies in one or more of: Havok, PhysX, ODE
- Integrating physics into your game

Reading:

- Course text: 12.4.5 – 12.4.9, 12.5

Lab:

Team Project

- Teams work on engine and game play systems in parallel

Week 12 – Gameplay Foundation Systems

- Components of the gameplay foundation layer
- Runtime object model architectures
- Memory management and file I/O for level loading
- Streaming game worlds
- Memory management for dynamic objects

Reading:

- Course text: 14.2.1.1 – 14.2.1.2 (review), 14.2.1.3 – 14.2.1.6, 14.2.2, 14.4, 6.2.2.7

Lab:

Team Project

- Teams work on engine and game play systems in parallel

Week 13 – Engine Subsystem Integration

- Review: The game loop, time in games
- Updating a multi-object simulation in real time
- Integrating rendering, physics and animation into the game loop
- Multiprocessor game loops

Reading:

- Course text: 7.1 – 7.5 (review), 7.6, 14.6.1 – 14.6.3 (review), 14.6.4

Lab:

Team Project

- Teams work on engine and game play systems in parallel

Week 14 – Elective Topics

- Multiplayer networking
- Advanced lighting and rendering effects
- Terrain systems
- Advanced audio
- Intro to player mechanics and game cameras
- AI foundations: path finding; traversal; perception

Reading:

- TBD

Lab:

Team Project

- Teams work on engine and game play systems in parallel

Week 15 – Wrap-Up

- Overflow topics as necessary
- Final exam review

Lab:

Team Labs

- Teams perform final integration and add finishing touches
- Code freeze one day prior to Gold Master

Exam Week – Final Exam and Team Game Project Due