Game Engine Architecture
Semester 1

Objective This course provides students with an introduction to the theory and practice of video game programming. 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 using an existing game engine (e.g. XNA).

Concepts Real-time programming and the game loop, human interface devices, 3D rendering, collision detection, skeletal animation, rigid body dynamics, game object models, event-driven programming, game scripting languages.

Prerequisite Introductory Computer Science (Data Structures and Algorithms)
Introductory C++ Programming

Lecture 2 hrs/week
Lab 2 hrs/week

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

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.

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%

Game Engine Architecture

Semester 1

Course Outline

Week 1 – Introduction

- Course overview
- What is a game?
- Structure of a typical game development team
- Overview of the technologies that comprise a typical 3D game

Reading:

- Course text: 1.1 – 1.2

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

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
- Introduction to quaternions
- Comparison of rotational representations

Reading:

- Course text: 4.1 – 4.5

Assignment:

- 3D math problems

Week 4 – Time and the Game Loop

- The rendering loop
- The game loop
- Game loop architectural styles
- Abstract timelines
- Measuring and dealing with time

Reading:

- Course text: 7.1 – 7.5

Week 5 – Human Interface Devices

- Types of human interface devices
- Interfacing with a HID
- Handling various types of inputs, outputs

- Game engine HID systems
  - Dead zone
  - Detecting button-up and button-down
  - Chords, sequences and gestures
  - Control remapping
  - Context-sensitive controls

Reading:

- Course text: 8.1 – 8.6

Week 6 – Introduction to 3D Rendering

- Triangle meshes and tessellation
- Coordinate spaces and rendering transformations
- Lighting, color and texturing basics
- The virtual camera and projection

Reading:

- Course text: 10.1

Week 7 – Fundamentals of Character Animation

- Types of character animation
- Skeletons and poses
- Clips and the local time line
- Skinning

Reading:

- Course text: 11.1 – 11.5

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 – Collision Detection

- Collision detection basics
- Sphere vs sphere
- Axis-aligned bounding boxes
- Other collision primitives
- Optimization: Broad phase, narrow phase, spatial subdivision

Reading:

- Course text: 12.3.1 – 12.3.4, 12.3.5.1 – 12.3.5.4

Week 10 – Introduction to Rigid Body Dynamics

- Is physics fun?
- Point mass linear dynamics
- Numerical integration
- Survey of collision/physics middleware: ODE, PhysX, Havok

Reading:

- Course text: 12.1, 12.2, 12.4.1 – 12.4.4

Week 11 – Introduction to Game Object Models

- What is a game object model?
- World editors
- Distinction between offline and runtime object models
- Spawners
- Basics of game object updating and engine system integration

Reading:

- Course text: 13, 14.1, 14.2.1.1 – 14.2.1.2, 14.3, 14.6.1 – 14.6.3

Week 12 – The Resource Manager

- File system APIs for game engines
- Asynchronous file I/O
- Survey of tools and the asset pipeline
- Resource management basics

Reading:

- Course text: 6.1, 6.2.1, 6.2.2 (except 6.2.2.7)

Week 13 – Events and Scripting

- Events and message passing
- Scripting languages
- High-level game flow

Reading:

- Course text: 14.7, 14.8, 14.9

Week 14 – Game Audio

- Audio codecs and clip playback
- Audio rendering technologies: Dolby DTS, 5.1 surround, etc.
- 3D audio concepts
- Audio data management: banks, cues, etc.
- Overview of Microsoft XACT tool and runtime API

Reading:

- Microsoft XACT manuals (online)

Week 15 – Wrap-Up

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

Exam Week – Final Exam and Team Game Project Due

Game Engine Architecture
Semester 2

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 Game Engine Architecture: Semester 1 (or equivalent)

Lecture 2 hrs/week
Lab 2 hrs/week

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

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.

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%

Game Engine Architecture

Semester 2

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 – 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 1: One-Dimensional Bouncing Ball

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

Week 3 – 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 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 – Software Engineering for Games

- C++ best practices
- Data, code and memory
- Errors, exceptions and assertions

Reading:

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

Lab:

Pongre 3: Congratulations, It's a Game!

- Add HUD, scoring, and personalization of your choice

Week 5 – More Software Engineering for Games

- Memory allocation and management
- Container data structures
- Strings and hashed string ids

Reading:

- Course text: 5.2 – 5.4

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 Project

- 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