CS425 GAME PROGRAMMING 1

Jyh-Ming Lien

Fall 2019

Office hours: Wednesday 2:00 - 3:30 am in ENGR 4442

Description

- This course is a requirement of the ACS Game Design degree and can be used as a Senior CS Elective for the BS CS degree.

- The course will provide an introduction to technologies and techniques used in modern computer games, animations, and special effects.

- Students will gain knowledge and experience needed to build games, simulations, and animations.

- Therefore, the key elements of the course will be knowledge, experience, and (hopefully) fun!

In Short

- You will be **making a game engine** and use your game engine to **make games** in this class

Communications

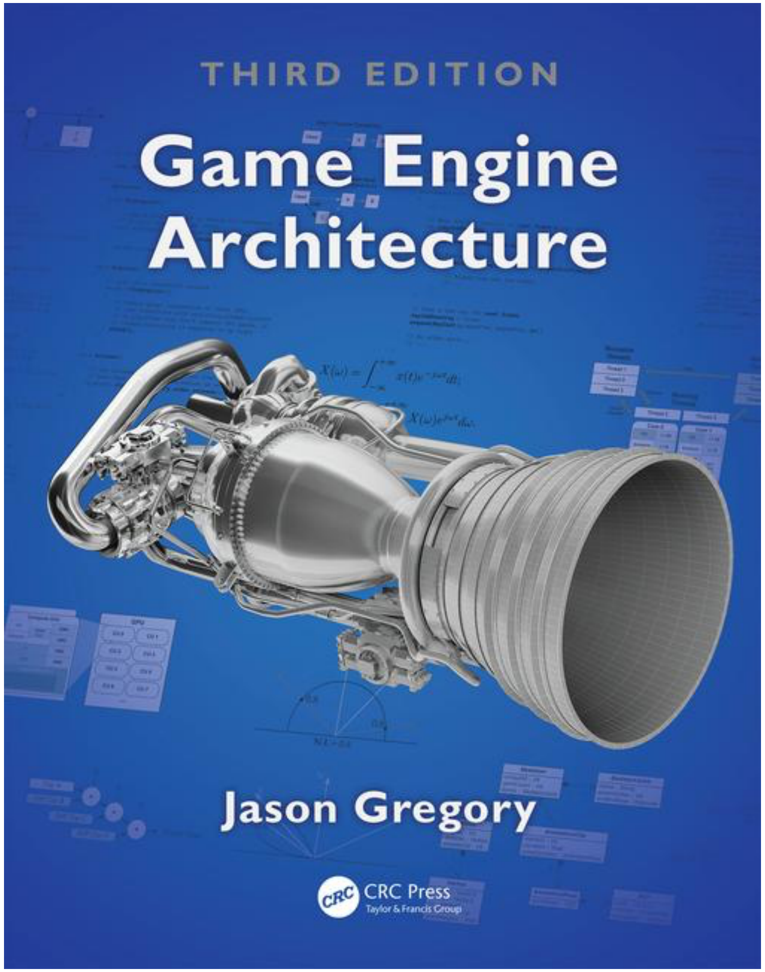
- **Blackboard**: http://mymason.gmu.edu/
  - All your grades appear here

- **Piazza**: is activated
  - All your announcements appear here
  - No private posts allowed
  - If you have private questions, come to my office hours or send me an email

Textbook

- Game Engine Architecture by Jason Gregory. ISBN# 9781315267845

- **Textbook is required**



Software and Tools

- Backbone: Simple DirectMedia Layer, a.k.a. SDL2 + OpenGL

- Language: C++

- Other tools: Maybe Unity

Course Outcomes

- a) ~~Become familiar with advanced techniques used in **object-oriented programming**~~
- b) Realize the manner with which the graphics pipeline may be utilized to create **animations, games, and simulations**
- c) Utilize **mathematical abilities** to create optimum algorithms
- d) Exhibit skills that demonstrate understanding of **physics simulations**
- e) Realize the appropriate techniques for maximizing **code reuse**
- f) Experience employing a variety of **data structures and algorithms**

Unofficial Objectives

- More C++ programming experience
- More experience with other people’s code/libraries
- Become more familiar with more components of game engines
- More 3D experience
- More math experience
- Gain experience with a commercial game engine?
- Add to demo reel
- Prepare for CS426

Grading and Late Policies

- Assignments (5) 50%
- Midterm Exam 10%
- Final Exam 10%
- Final Project: 25% (including 5% final presentation)
- Participation: 5%

- All required assignments should be completed by the stated due date and time. The total score of your final score will be 10 points less every extra day after the due date (i.e., the 100 total points will become zero after 10 days pass the due date).

- You will be given 0 point for missing your own presentation.

Assignments

- Assignments (5+1) 50% + 1%
  - PA00: Setup your environments and try tutorials (1% participation)
  - PA01: Level loading, Scene Management and Scene Graph
  - PA02: Procedural Content Generation
  - PA03: Navigation and Path Finding
  - PA04: Game AI (Finite state machine and behavior tree)
  - PA05: Physics Engine

- Bonus points are possible
  - Example1: make a simple game out of the assignment
  - Example2: reproduce the same capability using Unity or Unreal Engine

Academic Integrity

- http://provost.gmu.edu/academic-integrity/
- http://www.cs.gmu.edu/wiki/pmwiki.php/HonorCode/CSHonorCodePolicies
- http://www.cs.gmu.edu/wiki/pmwiki.php/HonorCode/StatementOnAcademicIntegrity

- All code submitted should be yours (and not like your friends, classmates, or someone from online)
- Searching error messages is allowed (and encouraged)
- Discussing the high level methods and algorithms is ok
- Discussing implementation details is not
- When in doubt about where the line is, ask me

Tentative Schedule

- Today, Introduction
- See the rest of the schedule here
  - https://github.com/jmlien/CS425-2019/tree/master/lectures

Announcements

- First homework assignment will be posted today
  - Setting up SDL2 on your PC/laptop
  - Work on and understand the tutorials
  - Due September 4th before 11:59PM

What functionality should a game engine have? What goes into developing a game?

In your opinion what is the best game engine available?

SDL

- What does SDL provide?
  - Manages GUI, image, fonts, video, audio, input devices, threads, networking and timers

- What does NOT SDL provide?
  - Lighting
  - Cameras
  - Scene graph
  - Resource manager
  - …

- Is SDL a game engine?

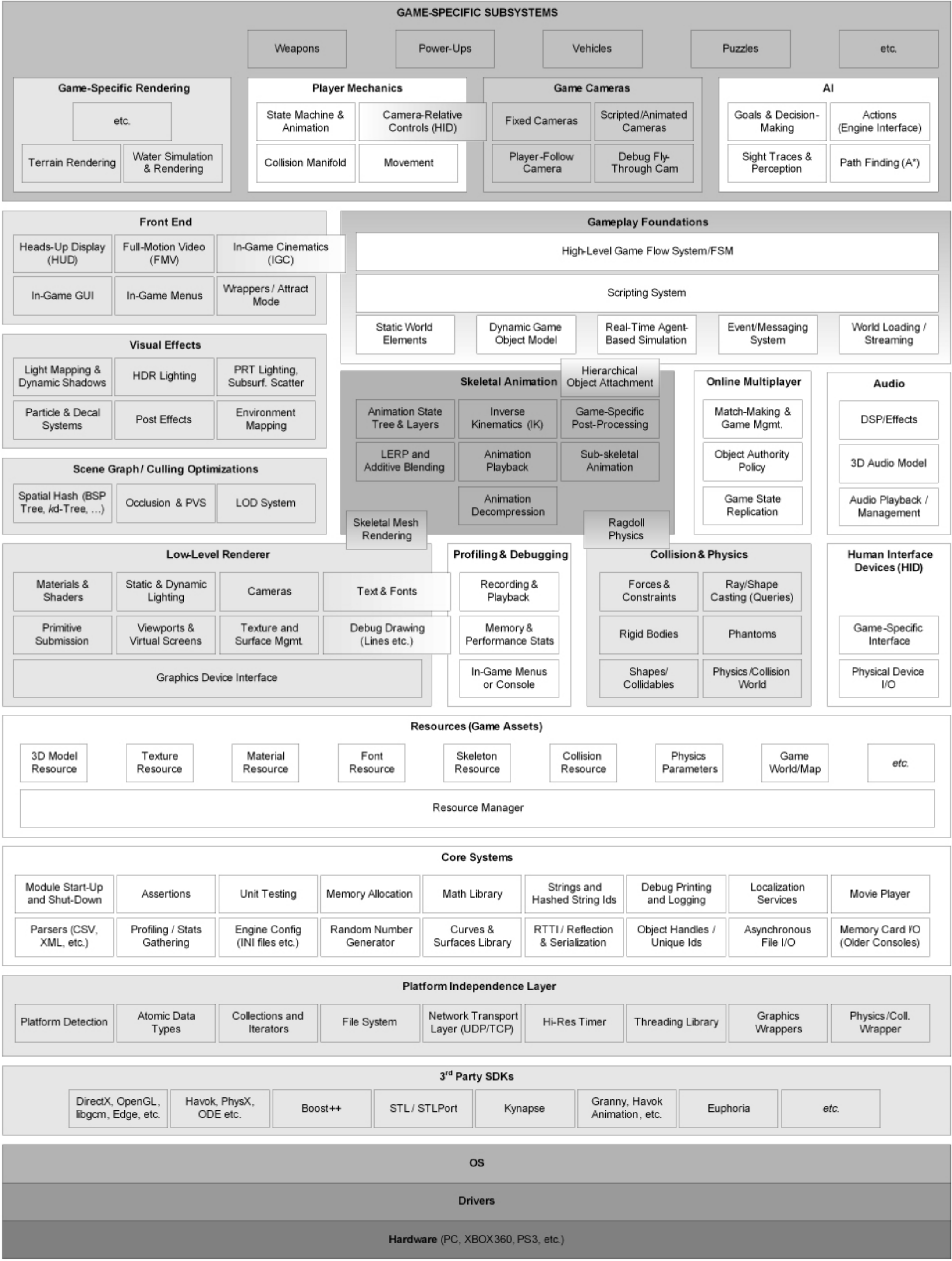
Video Games and Simulations

- Modeled mathematically
  - Approximation and simplification
- Agent-based
  - Vehicles, characters, fireballs, power dots, etc.
- Dynamic
- Interactive
- Real-time

Game Engines

- Reuseable core components
- Extensible to different games with minor modifications
- Often tied to a particular platform and game genre
  - Optimization

Runtime Architecture



GAME-SPECIFIC SUBSYSTEMS

Weapons
Power-Ups
Vehicles
Puzzles
etc.

Game-Specific Rendering
etc.
Terrain Rendering
Water Simulation & Rendering

Player Mechanics
State Machine & Animation
Camera-Relative Controls (HID)
Collision Manifold
Movement

Game Cameras
Fixed Cameras
Scripted/Animated Cameras
Player-Follow Camera
Debug Fly-Through Cam

AI
Goals & Decision-Making
Actions (Engine Interface)
Sight Traces & Perception
Path Finding (A*)

Front End
Heads-Up Display (HUD)
Full-Motion Video (FMV)
In-Game Cinematics (IGC)
In-Game GUI
In-Game Menus
Wrappers / Attract Mode

Gameplay Foundations
High-Level Game Flow System/FSM
Scripting System
Static World Elements
Dynamic Game Object Model
Real-Time Agent-Based Simulation
Event/Messaging System
World Loading / Streaming

Visual Effects
Light Mapping & Dynamic Shadows
HDR Lighting
PRT Lighting, Subsurf. Scatter
Particle & Decal Systems
Post Effects
Environment Mapping

Skeletal Animation
Hierarchical Object Attachment
Animation State Tree & Layers
Inverse Kinematics (IK)
Game-Specific Post-Processing
LERP and Additive Blending
Animation Playback
Sub-skeletal Animation
Animation Decompression
Skeletal Mesh Rendering
Ragdoll Physics

Online Multiplayer
Match-Making & Game Mgmt.
Object Authority Policy
Game State Replication

Audio
DSP/Effects
3D Audio Model
Audio Playback / Management

Scene Graph / Culling Optimizations
Spatial Hash (BSP Tree, kd-Tree, …)
Occlusion & PVS
LOD System

Low-Level Renderer
Materials & Shaders
Static & Dynamic Lighting
Cameras
Text & Fonts
Primitive Submission
Viewports & Virtual Screens
Texture and Surface Mgmt.
Debug Drawing (Lines etc.)
Graphics Device Interface

Profiling & Debugging
Recording & Playback
Memory & Performance Stats
In-Game Menus or Console

Collision & Physics
Forces & Constraints
Ray/Shape Casting (Queries)
Rigid Bodies
Phantoms
Shapes/ Collidables
Physics /Collision World

Human Interface Devices (HID)
Game-Specific Interface
Physical Device I/O

Resources (Game Assets)
3D Model Resource
Texture Resource
Material Resource
Font Resource
Skeleton Resource
Collision Resource
Physics Parameters
Game World/Map
etc.
Resource Manager

Core Systems
Module Start-Up and Shut-Down
Assertions
Unit Testing
Memory Allocation
Math Library
Strings and Hashed String Ids
Debug Printing and Logging
Localization Services
Movie Player
Parsers (CSV, XML, etc.)
Profiling / Stats Gathering
Engine Config (INI files etc.)
Random Number Generator
Curves & Surfaces Library
RTTI / Reflection & Serialization
Object Handles / Unique Ids
Asynchronous File I/O
Memory Card I/O (Older Consoles)

Platform Independence Layer
Platform Detection
Atomic Data Types
Collections and Iterators
File System
Network Transport Layer (UDP/TCP)
Hi-Res Timer
Threading Library
Graphics Wrappers
Physics /Coll. Wrapper

3rd Party SDKs
DirectX, OpenGL, libgcm, Edge, etc.
Havok, PhysX, ODE etc.
Boost++
STL / STLPort
Kynapse
Granny, Havok Animation, etc.
Euphoria
etc.

OS

Drivers

Hardware (PC, XBOX360, PS3, etc.)

Components

- Virtual world/game world
  - Terrain, objects, characters, etc.
- Mechanisms for movement
  - Objects, characters, cameras, etc.
- Interaction
  - Event-based programming
- Camera(s) into the world
- Graphics/Rendering
- Audio
- Networking
- …

Conclusion

- Went over the logistics, objectives, topics of CS 425
- Briefly introduced the concept of a game engine
- Assignment: Setup SDL2