CMSC 498M: Chapter 12
Final Review

Semester Review

What we covered:
- Introduction (history, game architecture) (2 lectures)
- Game Architecture (1 lecture)
- OpenGL Overview (3 lectures)
- Game and Graphics Engines (2 lectures)
- Modeling for Games (3 lectures)
- Animation (1 lecture)
- GPU Programming and Shaders (2 lectures)
- Artificial Intelligence for Games (4 lectures)
- Networking and Multiplayer Games (3 lectures)
- Sound for Games (1 lecture)
- Game Physics (3 lectures)
Introduction

History:
- Timeline
- Game genres
- Game demographics

Game Program Design:
- Basic elements of game design
- Game architecture - basic structural elements and relations
- Sample game program structure and timing
- Scene graphs

Interactive graphics and Geometry

OpenGL:
- Graphics Pipeline and General Structure
- OpenGL Drawing Primitives
- GLUT: Window Manipulation and User Interaction
- OpenGL Transformations and 3D viewing
- Lighting, shading, texture mapping in OpenGL

Geometry for graphics:
- Affine Geometry
- Coordinate systems
- 2-d and 3-d Affine transformations
- Transformations in OpenGL

Graphics architecture and shaders:
- Vertex and fragment shaders
- GLSL programming language
Graphics Engines and Ogre

Game and Graphics Engines:
- Engine core functionality
- Basic engine architecture

The Ogre 3D game engine:
- Ogre scene graph
- Ogre subsystem overview
- Examples

Game/graphics engine architecture:
- Core engine systems
- Scene graphs
- Renderers

Geometric Modeling

Geometric modeling:
- Modeling forms (meshes, blobs, parametric)
- Bezier and B-Spline curves and surfaces
- Modeling strategies for games
- Geometry instancing

Level of detail:
- Static and dynamic LOD
- LOD based on: Distance, size, eccentricity, velocity, depth of field

Animation:
- Kinematics, dynamics, interpolation
- Keyframing
- Skeletal animation
- Skin and bones
GPUs and Programmable Shaders

**GPU Architecture:**
- Rendering pipeline
- Vertex and fragment processing

**Shaders:**
- Vertex shaders
- Fragment shaders

**GLSL Shader Language:**
- Basic language structure
- Compiling/linking a GLSL program
- Vertex/Fragment shader examples
  - Phong shading
  - Toon shading
  - Normal mapping

Artificial Intelligence

**Basic AI for games:**
- AI Basics
- Agents
- Rule-based Approaches
- Finite-State Machines

**Path finding and Motion Planning:**
- Configuration spaces
- Waypoints and path planning algorithms
- Dynamic planning
- Potential-based path planning
- Flocking
- Particle systems
Networking, Online Games, and Sound

Networking:
- Networking: Protocols and packets
- OSI Network Layers: Physical, Data Link, Network, Transport, ...
- Socket programming and RakNet

Online Games:
- MMOG: latency and consistency
- Cheating and countermeasures

Sound:
- Basic definitions
- Sound engines and the aural pipeline
- Sound design

Physics

Basics:
- Basic physical quantities
- Kinematics (for particles and rigid bodies)

Rotation and force:
- Geometry and transformations
- Rotations (Euler angles and quaternions)
- Force: Springs, dampers, buoyancy, friction

Kinetics and integration:
- Force and torque
- Kinetics and the laws of force and motion
- Collision detection and response
- Numerical integration (Euler)
- Motion constraints
Student Presentations

<table>
<thead>
<tr>
<th>Name</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Greg Benjamin</td>
<td>BSP Trees</td>
</tr>
<tr>
<td>Steven Dobek</td>
<td>Pygame Engine</td>
</tr>
<tr>
<td>Trevor Hill</td>
<td>Cloth Simulation</td>
</tr>
<tr>
<td>Andrew Ferguson</td>
<td>Partially Visible Sets and Partially Audible Sets</td>
</tr>
<tr>
<td>Peter Enns</td>
<td>Automated Terrain Generation</td>
</tr>
<tr>
<td>Tiffany Chao</td>
<td>Unity 3D Engine</td>
</tr>
<tr>
<td>Todd Watson</td>
<td>Game Cheating Tools</td>
</tr>
<tr>
<td>Jess Gonzales</td>
<td>Sound for Games</td>
</tr>
<tr>
<td>William Goh</td>
<td>Designing Imperfect AI</td>
</tr>
<tr>
<td>Sina Iman</td>
<td>Network-Based Cheating</td>
</tr>
<tr>
<td>Richard Johnson</td>
<td>OpenTK</td>
</tr>
<tr>
<td>Nick Van Aersten</td>
<td>Smooth Animation and Camera Control</td>
</tr>
<tr>
<td>Chris Morton</td>
<td>Modeling in Blender</td>
</tr>
<tr>
<td>Patrick Bracken</td>
<td>XNA Game Studio</td>
</tr>
<tr>
<td>Mike Meyers</td>
<td>XNA Game Studio</td>
</tr>
<tr>
<td>Damon Earp</td>
<td>(TBA)</td>
</tr>
<tr>
<td>Douglas Hummel</td>
<td>(TBA)</td>
</tr>
</tbody>
</table>

What We Didn’t Cover

Topics we didn’t cover:
- 2-D game programming (sprites)
- DirectX
- Story telling and game design
- Business issues
- Software development tools and project management
- Artistic elements and tools (Maya and 3ds Max)
- User-interface issues
- Deeper aspects of game design (flow, balance, rewards)
- Ethical issues (and game ratings)
- Serious games