IMGD 3000 - Technical Game Development I: Introduction

by

Robert W. Lindeman

gogo@wpi.edu

What to Expect

- This course is mainly about the nuts and bolts of creating game-engine code
  - Game architecture, algorithms, data structures, mathematics
  - Less about content
- Presupposed background:
  - IMGD-1001: The Game Development Process
  - CS-1101/2: Introduction to Program Design
  - CS-2102: OO design concepts
  - CS-2303: Systems programming
  - CS-3733: Software Engineering
  - In other words, you should be able to design and implement large systems
- Nice to have:
  - Computer graphics

What to Expect (cont.)

- Today, there are many game engines available
  - Provide a starting point for game creation
  - Usually provide
    - Tools for importing content (*e.g.*, models, textures, *etc.*)
    - Scripting language to handle high-level control
    - Cross-platform support
- We want you to learn what is *inside* these engines
  - We will use the **C4** Game Engine as one example of how things could be done
  - There are ***many*** ways to skin a cat!
  - Most games require you to extend the engine
    - HINT: Those are the ***really interesting*** jobs!
  - For C4, you will write game code on top of the engine

What to Expect (cont.)

- This course is about *game development* not C4
  - But you will learn C4
  - Focus on underlying methods
- This course is heavy on
  - Coding
  - C/C++, Scripting
  - Efficiency
    - Speed
    - Quality
- If you are a sophomore, you might want to wait a year, and take more CS
  - The sophomores in previous classes told me they wished they had waited

Summary of Syllabus

- Lectures and in-class exercises
  - Exercises designed to drive home concepts, or to get you thinking about projects
- In-Class Work (20%)
- ~3 "Smaller" Projects (40%)
- 1 Final Project (40%)
- Smaller projects will use C/C++ and the C4 codebase
- Final project will use C/C++ and the C4 codebase
- First project will be individual, rest team-based
- Clearly defined team roles
- All material on class website (www.cs.wpi.edu/~gogo/courses/imgd3000/)

Texts for the Course

- The Beginner's Guide to the C4 Engine

  By James Brady, A. A. Cruz, James H., and David Vasquez

  http://www.terathon.com/store/
  - WPI has a “site-license” for the book
  - Buy it from the Web if you would like a hard copy
- Excerpts from:
  - Object-Oriented Game Development
    By Julian Gold (2004)
    Addison Wesley, ISBN: 0-321-17660-X
  - Ultimate 3D Game Engine Design & Architecture
    By Allen Sherrod (2007)
    Charles River Media, ISBN: 1-58450-473-0

C4 Game Engine

- We have a site license for the C4 Game Engine
  - A Non-Disclosure Agreement (NDA) must be signed by all students to gain access to the source code
- We will also be using the online materials on the C4 Web site
  - www.terathon.com/c4engine/
  - There are very good user forums, a Wiki, *etc.* on the C4 site that you have free access to
- Please post in the appropriate sections
  - Most people are very helpful on the site, and they know you are coming ;-)
  - Create a profile
  - Set "Profession" to "WPI Student"
  - Email Eric Lengyel that you are a WPI student

Synchronized Tech & Art

- Joint teams of 3000 & 3500 students
- Work together
  - You focus on the **tech**
  - They focus on the **art**
- About 25-30 students each in 3000 & 3500
  - Need to find each other!
  - We will have one joint meeting per week
  - 3500 meets Mon/Thu 1:00pm-2:50pm
  - More on this later...

Projects

- Many phases to projects:
  - Understand/design/code/debug/test/eat/test some more
  - Encouraged to discuss approaches with others/in a group
  - On individual projects, work alone!
- Academic dishonesty (a.k.a., cheating):
  - Many reasons *not* to do it!
  - Immediate NR in the course
- Advice for doing well:
  1. Do the assigned reading (they are actually good books!)
  2. Come to class
  3. Ask questions (class, office hours, WPI GDC discussions)
  4. Make sure you understand things before coding
  5. Don't share your code with others!

Final Project

- Four- or five-person teams
- All teams start with the same initial idea & code
- Define your own extensions/changes
- You will focus mainly on technical aspects
  - Your 3500 team mates will take care of the art
- Interim deadlines to show progress
- Presentations will be done the last week of this course, where you will show your stuff
  - Hope to have industry participation

Course Support

- TAs
  - Paulo de Barros (pgb at wpi.edu)
  - Jia Wang (wangjia at wpi.edu)
- Please come to office hours (or other times)
- There is a GDC Forum for this course
  - http://forums.gdc.wpi.edu/
  - All project discussions should be posted there
  - You are encouraged to post screen-shots of your progress
  - Be careful when posting code -- don't give anything away!
  - Post any book errata there too

But First…

- What is a ***game engine***?
- How does it work?

What is a Computer Game? **User Perspective**

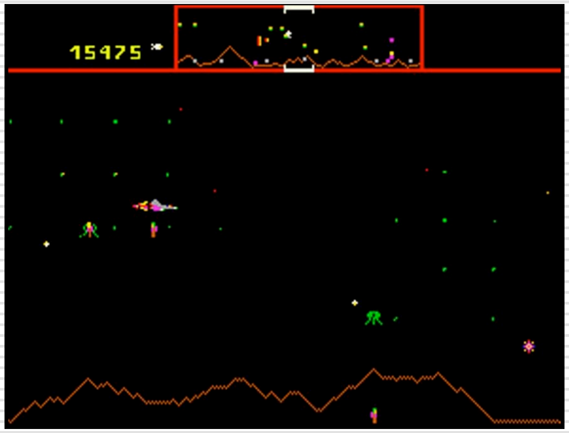
- A goal (or set of goals)
  - Save the Princess (solve these puzzles first)
  - Score points (get power ups)
  - Finish first (unlock features)
- A set of rules governing game play
  - Turn taking, like RPGs
  - Reaction to events, like Tetris' falling blocks
  - Legal actions
- Visual (audio, *etc.*) content
- Control techniques
  - Button mappings

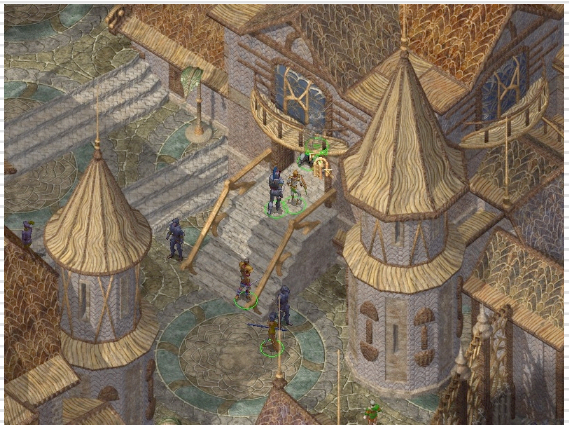
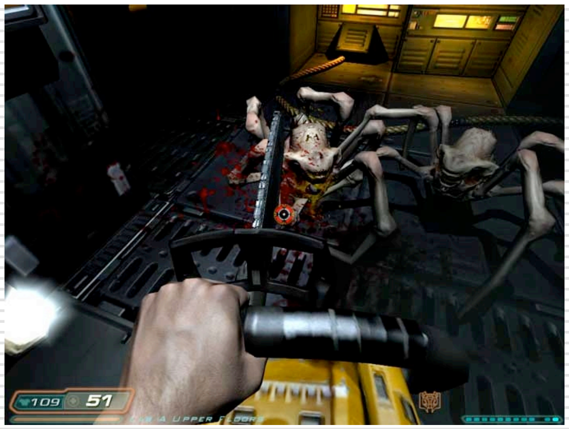
What is a Computer Game?
System Perspective

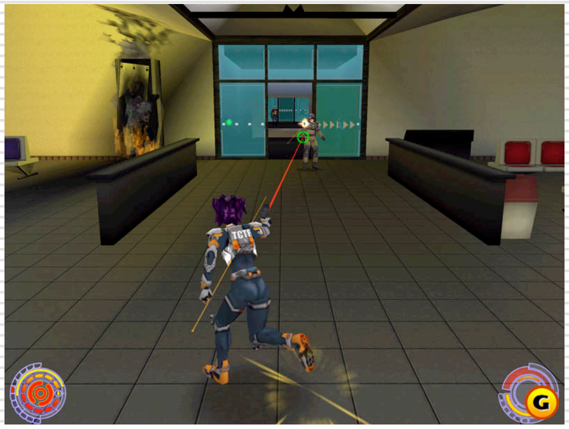
- A set of resources that are managed to support an entertainment (usually) application
- Graphical (audio, *etc.*) rendering
- A user interface
- Script handling
- Event processing
  - Time, collisions, *etc.*
- File I/O
- Asset-creation tools
  - Models, graphics, sound, *etc.*
- Optional
  - Networking
  - AI

Types of Games

- 2D (Tetris)
- Side-scroller
- 3D isometric
- 1st-person view
- 3rd-person view
- Others too





Game Genres

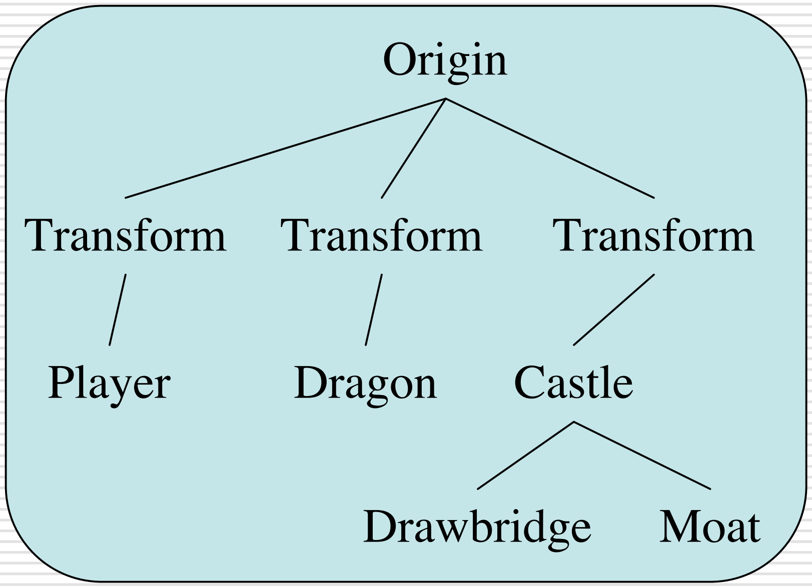
- Genre defined:
  - A category of artistic composition, characterized by similarities in form, style, or subject matter.
- First-person Shooter (FPS)
- Real-time Strategy (RTS)
- Action
- Sports
- Simulation
- Stealth
- Puzzler
- Party

Elements of a 3D Game

- Game engine
- Scripting
- Graphical user interface
- Models
- Textures
- Sound
- Music
- Support infrastructure
  - Web site
  - Support forums
  - Admin tools
  - Database

Game Engine

- Scene graph
  - Representation of the world
  - Includes characters
- Timing is very important
  - Events
    - Time-based
    - Multi-player
  - Synchronization
- Database of objects
- Networking
  - Between server and clients
  - Between servers
  - Between clients



Game Engine (cont.)

- Core utilities
- Rendering system
- Physics
- Artificial intelligence
- Input management

Core Utilities

- Data structures
- Game-state management
- Timers
- Memory management
- Journaling services
- File logging
- Performance profiling tools
- Encryption/decryption

Scripting

- Scripting languages provide easier path to building a game
  - Provides access to game-world objects (GWOs)
  - Allows most aspects of the game to be defined
  - Tie all parts of the game together
  - Leverage investment in engine development
  - Trade control for automation
- Sample scripting languages for games
  - Lua (www.lua.org)
  - Torque Script (www.garagegames.com)

Graphical User Interface

- Provides access to
  - Game menus (*e.g.*, save, load, boss)
  - Player status (*e.g.*, health, current speed)
  - Maps
  - Non-Player Character (NPC) dialog
  - Player-to-player chat

Models (Art Stuff)

- Objects are made from
  - Geometry (a.k.a., polygons)
  - Lighting
  - Textures
- Vertices and connectivity
  - Triangles
  - Triangle-strips
  - Meshes
  - Patches/surfaces

Texturing (Art Stuff)

- Created/manipulated using image processing software
  - Photoshop
  - Paint Shop Pro
- Mapped to geometry (models)
- Very powerful image enhancing techniques
  - Can be used for fake shadows, fake reflections, much more

Sound and Music

- One of the most-important elements of any experience is sound
- Sound effects
  - Make things more (hyper-) realistic
- Musical score
  - Sets the mood
  - Builds emotion
- Speech output
- Very important skill

Support Infrastructure

- Front-end for running games
  - Steam
- Web site
  - Promotion, log-in, etc.
- Support forums
  - Cheats, hints, discussion of new ideas
- Admin tools
  - User maintenance
  - Anti-cheating measures
- Database
  - Game-state maintenance

Our Focus

- We will focus mainly on *tech stuff*
  - How to program a game
  - How to control game flow
  - How to set the rules of play
  - How to support user interaction
- Less on content
  - Models
  - Textures

Expected Outcomes

- Understand the complexities of game development
- Be able to build individual parts of a game
- Build up your portfolio
- Work in Tech/Art teams
- Be ready for IMGD 4000!

Another Word About Projects

- Form teams of two tech and two art students
- Smaller projects
  - Tech people do tech parts, art people do art parts
- Final project
  - Do the design parts together
  - Choose roles for the rest
  - Meet intermediate milestones

Course Alignments

- We have aligned IMGD-3000/4000 and IMGD-3500/4500
- Teams can work on the same game for two terms
  - Advantages
    - "Better and Bigger" portfolio piece
    - Longer team experience
  - Challenges
    - Need careful coordination of time, people, and features
- You don't *HAVE* to do this!
  - In IMGD-4000, you can also use the Java Monkey Engine, instead of C4