CSC3232 Coursework

Welcome to CSC3232! This module is aimed at providing you with an understanding of video game technologies and simulation techniques. The areas investigated in this module can broadly be classified as being those that enhance the physical accuracy of your game simulation, those that enhance the appearance of intelligence and decision making in the simulation agents, and those that provide a solid structure around which the gameplay mechanics can be built. The lecture series will introduce and cover the theory behind these concepts, while the tutorial series will look at the practical implications of these concepts, and how they may be implemented in a commercial game engine. Together, these should provide you with experience with the module concepts, and the skills required to apply them successfully.

Module Assessment

There are 2 pieces of assessment for this module:

- Coursework project 1, submitted via NESS (50)
- Coursework project 2, submitted via NESS (50)

The coursework projects are designed to be complementary to each other, allowing you to build up a video game that includes a range of the common simulation techniques and technologies.

Coursework project 1 will focus on the physics aspects of gaming simulations (correct motion throughout an environment, and correct detection and resolution of collisions that occur between objects) and more simple artificial intelligence, while coursework project 2 covers design patterns, navigating through an environment, effects & audio, and evolving the gameplay simulation. Both pieces of coursework will be part of the same Unity project, and the exact nature of their implementation within that project is up to you. For both submissions you should just zip up the entire project folder, and submit it to NESS at the appropriate time. If your project folder is too large, you can instead submit a link to an appropriate code repository (such as the University GitHub) that can be accessed to obtain your project.

Coursework Theme

You are to use the Unity game engine to create a video game that incorporates the various elements covered in the module. While the exact graphical theme is up to you (it could be space, fantasy, or purely abstract), gameplay wise you should split the game into an ‘overworld’, that can be navigated to reach one or more ‘encounters’. The map screen and levels of *Super Mario Bros. 3* is one example of such a gameplay setup, while the starmap and battles of *FTL: Faster than Light* is another. You may use assets from the Unity asset store if you wish, but any such used should be things like meshes/audio/textures and so on – the code within the game should be primarily yours.

This mix of game conditions has been chosen to allow you the freedom to explore a game design that interests you while still allowing all of the technology elements of the module to be investigated and demonstrated. For instance, if your core game idea doesn’t blend well with pathfinding or adversarial AI, put it in the overworld section of the game, rather than hinder your main game idea. This means that it is to your benefit to make the encounters and overworld different ‘styles’ of gameplay – a turn-based adversarial overworld that leads to a number of physics-based challenges could be one approach, or a 2D platformer overworld that has a number of puzzle-based minigames within it.

Project 1 (Deadline: 16:00, Friday 12th November 2021)

Aim

The purpose of this coursework is to get your game idea underway, with a basic 'skeleton' of your game idea. You should have the basics of the overworld and encounters in place, such that you can go from the overworld, to an encounter, and back again. Neither the overworld or encounters are expected to be completed, just started sufficiently to demonstrate some core simulation elements.

You should also have begun to include implementations of the module elements that we have investigated so far – Newtonian dynamics to move some objects, some collision detection, resolution + feedback, some simple state-based AI, and the mathematical underpinnings of chance and state evolution in a game simulation.

Marking Scheme

- 10 marks available for Newtonian Physics
  - 5 marks for appropriate use of Newtonian physics
  - 5 marks for advanced usage (multiple gravity areas / changing mass / etc)
- 10 marks available for Collision Detection
  - 5 marks for making use of appropriate collision volumes
  - 5 marks for advanced usage (specific hit areas, dynamic changes over time etc)
- 10 marks available for Collision Response and Feedback
  - 5 marks for making use of appropriate collision response and feedback
  - 5 marks for advanced usage (physics material properties
- 8 marks available for state-based behaviours
  - 4 marks for basic state based behaviour (menu system, simple opponents)
  - 4 marks for advanced usage (hierarchical state machines etc)
- 12 marks available for probability and game design
  - 4 marks for making use of stochastic behaviour
  - 4 marks for evidence of gameplay modelling
  - 4 marks for demonstrating positive feedback loops

Deliverables

- Zipped Unity Project Folder OR zipped text file with link to repository (submitted via NESS).

Learning Outcomes

- Understand how to represent mathematical concepts in code
- Understand the implementation of Newtonian dynamics in a real-time simulation
- Deduce and apply the appropriate simulation technique given a particular simulation requirement.

Project 2 (Deadline: 16:00, Monday 13th December 2021)

Aim

The purpose of this coursework is to complete your game design, and add new features relating to the module material. Your game should have a fully working overworld and at least one type of encounter that can be played to completion. You should include gameplay elements relating to pathfinding, visual effects, audio, advanced AI, design patterns, and game logic. It is likely that different games will utilise these in different ways, and they will be judged by the quality of their implementation, and effectiveness at presenting a well structured game design.

Marking Scheme

- 10 marks available for pathfinding
    - 5 marks for making use of appropriate pathfinding
    - 5 marks for advanced pathfinding features (dynamic map changes etc)
- 10 marks available for advanced real-time AI techniques (GOAP / flocking / etc)
- 10 marks available for special effects and audio
- 10 marks for appropriate use of design patterns and game structure
- 10 marks available for advanced gameplay progression techniques (Min/Max, managing state etc)

Deliverables

- Zipped Unity Project Folder OR zipped text file with link to repository (submitted via NESS).

Learning Outcomes

- Develop familiarity with the nature of embedded map data, and its role in influencing path planning
- Understand the fundamental algorithms behind effective path finding in games
- Demonstrate an understanding of the interplay between game logic and physics computation
- Develop familiarity with code concepts common to game engineering, and their effective implementation.
- Understand the importance of managing game complexity and maintaining correctness of game logic.