CSC8503 Coursework 2023/24

Due:14th December 2023

In this coursework you are to develop a game that includes physics, networking, and AI, to allow you to practice your coding skills, and reinforce your knowledge in these areas. The game takes place in a simple 3D environment, where the player controls an avatar using physics calculations, and must employ collision detection and resolution.

The codebase you have been provided with serves as a good starting point for the development of your game. The goal of the coursework is to then enhance this codebase with realistic physics and collisions, to allow the basics of the game to be played. From there, AI techniques can be used to add in some more complex movements of enemies that the player must interact with. You do not need to spend any time implementing graphical techniques, or finding new meshes and textures to apply to your game - the purpose of this coursework is just to investigate the new concepts introduced in the lectures and tutorials.

The coursework is divided into two parts. The first will primarily cover physics calculations and simple state machines, while the second covers networking and pathfinding. While the exact game implementation is up to you, the following video should provide some inspiration as to what could be replicated in your program: https://www.youtube.com/watch?v=Ov5GxFlHqUQ

Coursework Part A: Ripped Off!

It's heist time! You are to make a game in which the player must attempt to steal a guarded item from a building, with the game taking place in a 'voxelised' world. To help the player, there should be items that can be picked up and placed throughout the world to aid with movement, or used in some way. Points should also be awarded for performing actions in the world, to help indicate to the player that they are doing the 'right' thing. To win, the player must pick up the guarded item, and bring it back to their starting area. The player object should be controlled with the keyboard and mouse, via the application of forces or impulses. You may choose to apply torque to rotate the player object, or use functions to automatically change its orientation to the desired direction.

If the player successfully steals the item, or the time runs out, then the game will game end, and the player is shown their final score. The game should feature a simple menu, allowing the player to play again, or exit the game. During the game the player's current score, and number of items left to destroy should be displayed on screen.

The game you create should take place within a single virtual environment - the exact composition of this is up to you, but it should have obstacles placed around (built using a combination of any collision detection and resolution types that you can create) to make the world more interesting. The level should contain a number of simple challenges that the player must complete to open up areas containing items to help with the main mission. Pressing buttons to open doors, or having to find a key to open a door would be examples of these challenges. The level should also have a number of enemy AI characters. These characters should patrol the world – if the player comes near them, they should chase the player who will 'die' if the AI reaches them, causing them to reset back to their initial position. There should be some usage of raycasting within the world. This can be to help the player in some way (maybe the player can find a grappling hook powerup?), or it could be used as part of the AI (perhaps the enemy AI speeds up if it can see the player?).

Coursework Part B: The Usual Coursework

Everyone knows games are more fun with multiple players! The game should be expanded to include a networked multiplayer component. This should allow multiple players to connect to the same computer, and either compete or collaborate in getting the item back to the starting area.

The menu should be expanded to allow the creation of a server, or for connecting to an already running server by IP. The server should maintain a high-score table of best point totals achieved during gameplay – this should be updated and sent as appropriate to connecting players so that it can be viewed by pressing a key.

The game should also feature an expanded area containing the heist target item, and a new NPC character, the Undercover Agent. Somewhere in the game environment there should be a hedge maze, patrolled by an angry goose. The Undercover Agent should be controlled by the server, and should try to hunt down any player holding the heist item. The Undercover Agent should be 'smarter' than the other AI agents, making intelligent decisions to try and defeat the players – trying to navigate to an alarm button that will cause all of the doors to lock could be an example of this.

Deliverable Items – NESS Submission

1. Source code. Clean your solution in Visual Studio, and then zip your work folder.
2. A document containing at least four screenshots with descriptions of the features they represent, a list of any key/mouse presses that perform actions, and a link to a YouTube video of your coursework running.

Deliverable Items - Demonstration

Demonstrate the program running in the lab. The program should run full screen, and you should be able to demonstrate the various physics, networking, and AI features present. Multiple instances of the game can be launched from the same machine to demonstrate the ability to connect and play the game across a network connection, if necessary.

Marks Available (100)

Coursework Part A (50 marks)

Implementations of collision detection (AABB, Sphere, OBB, plane), collision resolution (impulse, projection, or penalty, with gameplay effects), application of forces, constraints, and simple state based artificial intelligence gain up to 50 marks.

A 'first class' submission would allow the player to move their character via the application forces, with the program successfully detecting and resolving collisions with the environment, and apply gameplay effects from collision interactions. The environment would contain at least one AI controlled character that uses state machine logic appropriately.

Coursework Part B (50 marks)

Implementations of networking (high score table + gameplay mechanics), advanced state machine usage (menu system, advanced character AI), and pathfinding (the goose) gain up to 50 marks.

A 'first class' submission would allow scores to be send across the network to connect players, for the players to be able to send and receive packets that allow their goats to move with consistency, and for the evil goose to be able to use appropriate pathfinding to move towards a player. The main menu and enemy AI should use appropriate state machines for their operation.