CSC3232 Coursework Marking Scheme Report

This document lists the elements of CSC3232 coursework marking scheme that were met with this project, and how they were met.

Physics

Appropriate use of Newtonian physics

- Usage of Rigid Bodies
- Correct application of impulses to bodies
- Objects have appropriate mass quantities
- Game mechanics is physics-driven (e.g. immediate response to collisions)

Advanced Physics (multiple gravity areas/changing mass/etc)

- Physics properties are changed via scripts
- Mass/Physics is a gameplay mechanic
- Additional forces beyond simple motion-driven accelerations are provided (projectile trajectories, gravity interfering with the velocity)
- AI uses calculations to determine projectile forces

Basic Collision Volumes

- There is at least one collision volume
- There is more than one collision volume
- The collision volume is appropriate and matches the Game Object's mesh

Advanced collision volumes

- A single Game Object has multiple colliders
- Colliders are enabled/disabled via scripts
- Colliders can change their position programmatically
- Trigger volumes are used as part of player mechanics

Appropriate collision response and feedback

- Rigidbody responds to collisions realistically
- An object makes use of OnCollisionEnter/Exit
- Collision Layers are used to separate out collision types

Advanced collision response and feedback

- Trigger volumes are used to trigger gameplay events

Explanation

- Enemies have rigid bodies which react appropriately and realistically to knockback and projectiles, also showcasing that physics properties can be changed via scripts and that it is a deliberate gameplay mechanic. They also differ in mass due to size. Enemies can also push the player. The player and one of the enemies can fire a projectile which is affected by gravity. The "ShroudedEnemy" will do calculations to try and hit the player with the projectile accounting for gravity.
- There are multiple collision volumes throughout the scene, including on characters, pieces of the environment and on empty game objects. Colliders have been chosen appropriately to fit with the object it is on, especially on the enemies. The Cryo enemy has multiple colliders, where one is a trigger collider to cover the attack range. OnCollisionEnter/Exit is also used in certain places, in particular to account for when the player has left the Cryo enemy's attack range so that they stop taking damage. There is an empty trigger collider at the end of the level across the body of water (marked with a purple particle system) that is only enabled when the player has defeated all enemies. Similarly, there is an empty game object with a collider blocking access to the water area until all enemies are defeated, at which point it will be moved downwards, allowing the player through. Layers are used well, for example making sure the tempest

projectile only hits things on the player or enemy layers, or for the Cryo enemy's trigger collider so that it doesn't do damage to its AI enemy squad mate.
- A trigger volume is used to trigger a gameplay event where the floor becomes a bright fire/lave texture to allow for better visibility. Players can step into the volume to activate it but can only activate it once.

Graphics

Appropriate Use of Graphical Elements

- Multiple textures appearing in game
- Appropriate use of lighting
- GameObjects moving & rotating via script
- A navigable camera moving in 3D

Advanced Graphics

- A body of realistic looking water
- Scripted lighting/effects (e.g. weather, day/night cycle)
- Change object appearance via script

Explanation

- Multiple textures appear in the game, such as on prefabs, and on the ground/plane with a rocky texture. This texture changes appearance for a lava texture during a particular event triggered by a script through a trigger collider.
- Lighting is chosen to be baked or mixed and is used where emission of light makes sense such as with the particle systems.
- A rotating spear can be found in the level. Enemy characters/gameObjects move as well.
- The player can traverse the level in first person.
- A body of realistic looking water exists, where there are reflections, a moving wavy texture, and a ripple effect when the player moves across it.
- When the player defeats all the enemies, lights are scripted to come on the skull platform.

Pathfinding

- NavMeshAgents are used
- NavMeshObstacles are used
- AI makes decisions based on pathfinding

Explanation

- NavMeshAgents are used to allow the AI enemies to move within the level. NavMeshObstacles are used too, especially on the towers to make sure the AI can navigate around them. The radius of the NavMesh agent component was increased to ensure that AI don't get stuck on obstacles. Depending on the results of the CanSeePlayer() method then the AI will switch states, meaning that if a path to player is available then the AI make the decision to act on it (for example going to chase the player).

AI

State Machines [+10%]

- Usage of simple state-machines
- Usage of boolean or state-driven state machines
- Usage of object-encapsulation for modelling states
- Usage of hierarchical state machines or usage of external tools for generating state
- machines.

- State machines are triggered by external events or timeouts.
- Usage of probabilistic/stochastic state transitions.

Explanation

The AI makes use of all of these within the files: IEnemyState, EnemyStateMachineController, EnemyPatrolState, EnemyChaseState, EnemySearchState (which features probabilistic/stochastic state transitions between it and the patrol state). EnemyAttackState and ShroudedAttackState show more clearly the usage of hierarchical state machines. Object-encapsulation is also present as each state is contained within itself and inherits from the interface. Various Boolean checks are made to ensure that states are in the correct state.

Structuring NPCs

- NPCs are not orchestrated, and mainly react directly to the player or the environment.

Explanation

NPCs are not hardcoded to follow a set list of instructions to carry out. Instead, if they see the player they will react accordingly and either chase them or attack them. Their actions are a direct consequence of what the player chooses to do. If the NPC loses sight of the player, then they pick a random area to search. The use of NavMeshObstacles also means that the NPC is reacting to environment directly by being able to navigate it.

Advanced Features

- Appropriate usage of Prefabs
- Levels/Menus are separated into distinct scenes
- Evidence of code for limiting expensive computations (e.g., raycasting)
- Usage of Vector Fields
- Usage of Particle Systems

Explanation

- I have created prefabs of my own models and used them within the scene. This includes the enemies and projectile.
- Raycasting and coroutines are used
- Upon reaching the end of the level the scene is changed, and players are presented with a menu.
- Vector fields are used where the skull platform/spinning staff is. If the tempest projectile is fired from within the given area, then the projectile gets manipulated to be attracted to the centre.
- Third Party and self-made particle systems are present in the project