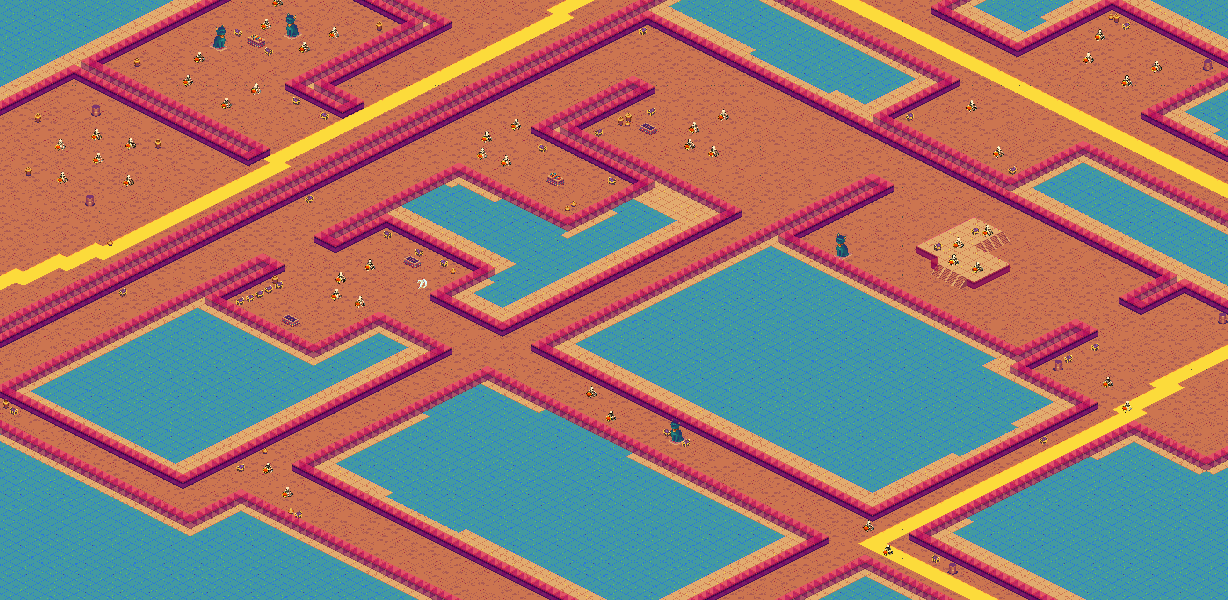
**Redeemer’s Crusade Project Report**

* **Jack Cleary**

*Redeemer’s Crusade* is a 2D isometric rogue like game consisting of various gameplay and AI elements.



An Example of a Level Within the Game

The main objective of the game is to progress through the current level and move on to the next, while earning enough points to pass through the level barrier. The player can earn points by both killing enemies (the skeletons that populate each level) and/or breaking chests.

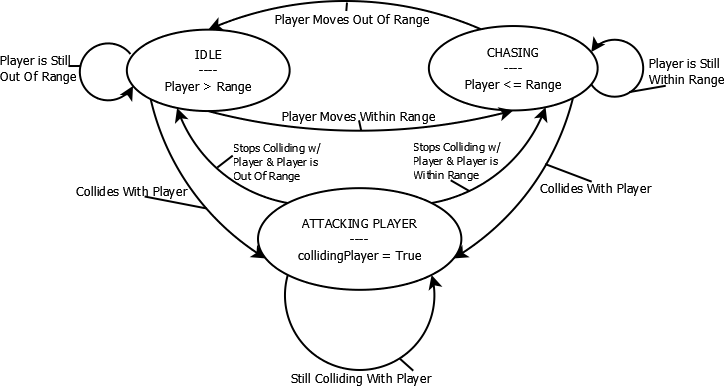
  

Skeleton Enemy Chest Player Character

The game currently consists of 3 levels, each increasing in difficulty and the points required to progress to the next sequentially.

AI was a large focus for Redeemer’s Crusade, with elements such as Finite State Machines, Line of Sight Chasing, Interception and Pattern Movement being incorporated into the game.

All NPCs in *Redeemer’s Crusade* contain an FSM which is curated to ensure they appear game aware and respond to the player’s actions in a fluid and natural manner. The FSMs were constructed using switch statements and states.



An Example of an FSM Diagram Detailing an FSM present in the Game

Line of Sight Chasing is an important aspect of the AI, it allows a level of realism to exist in *Redeemer’s Crusade* whereas the enemy NPC will stop chasing the player character if they are unable to see them in their field of vision, or if they are obstructed by an object. LOS chasing was primarily implemented using Ray-casting and Vectors. Enemy NPCs also have a variable range in which they can detect the player, so in order for the player to be chased by an enemy, they need to both be within their line of sight and close enough to be seen.



The Enemy is Unable to See the Player Through the Tree

If NPCs have been chasing the player character but he moves out of their sight range, the NPC will turn around in the opposite direction of the player and move to a new location, which is an implementation of Pattern Movement.

Vector Math is prevalent throughout the game in a large manner, along with LOS chasing it is used to calculate the direction of running and attack animations, the location of the player’s mouse, spawn positions for dead enemies, various rendering calculations, and others.

Interception is present within the game in the form of an NPC variant that is both smaller than the main skeleton type and faster and will also try to get ahead of the player character and predict their movements. This was also done with Vector math, where the player’s current position is multiplied by a float value to gain a rudimentary prediction and used to calculate the NPC’s direction.



Both NPC Variants