Task 3: Design and Implement a State Machine for a NPC

Description of Real-World Environment (using raylib library)

The game world will be a rendition of a simple stealth game. The player is a bat that has to use its voice to keep the screen bright so you can visually see enemies. Enemies will patrol certain areas and will actively try to seek out the player when they are in range of them, the range will be quite short and will be visible when the screen is bright enough. They will patrol the area they hear the player from and will then go back to their usual patrol after a set amount of time. The goal of the game will be to reach the end of the maze of stealth sections. I'm also looking to provide a nice experience in terms of camera movement with the player and having it place the camera comfortably for each independent stealth section of the maze.

Functionality of the AI:

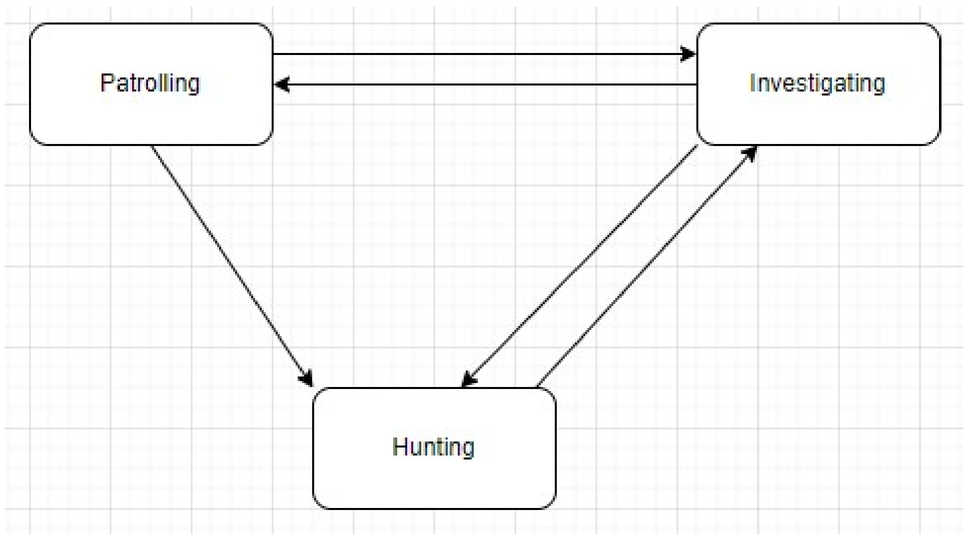
The enemy's actions will be controlled by a finite state machine filled with 3 behaviours. Those being Patrol, Hunt, and Investigate. The conditions for transitioning between these behaviours will be controlled by 3 transitions.

One of them being a response to a players input. When the player uses its ability to help brighten and see the window, this would mark that location and have any nearby enemies come and investigate the area surround that position.

The second transition would be a time-based counter. The countdown starting as soon as the enemies begin to investigate a location and after the countdown finishes, they would return to their original patrolling location and return to that behaviour.

A final transition of player proximity will be used for if the player gets too close to an enemy, they will begin to hunt them and try to catch them. Catching them would cause the player to lose the game and have to restart from the start.

Below is a diagram for how the state machine would operate:



Interaction with the Simulated Environment

The enemy AI would be constantly finding paths for themselves whether that be around their designated patrol section. Towards the player or around the players last known location. It's relatively simple but should be reliable in its ability to go towards the player whether they may be (as long as they are in a designated range) as well as stay within their allocated space. My current thought on how to implement this would be to add a distance limit on how far a new random node can be from the enemies set patrol node. So they would be centralised around a specific node and stay within a certain range of it. (Breaking this rule when having some form of interaction with the players location)

Difficulty Levels and Their Controls

There are a few different ways I can think of to control the difficulty levels in this game, most being quite simple and easy to implement / change. We could increase the number of enemies patrolling certain areas, increase their speed while hunting the player (this could be increased so much to the point where getting found once would be death) as well as increasing their "vision" range, meaning how close a player has to get to be found by an enemy.

Another, possibly more difficult to implement method, would be to give the enemies some vague idea of where the player is during their investigating behaviour. This would mean they'd be more likely to walk towards the player while close to them. We could put this on a weighted chance and change the weighting based on the difficulty, for example the hardest could be that they have a ~90% chance to walk straight towards you after hearing you.

I also think a large factor of difficulty will be based on level design and layouts of the stealth rooms. After having all the features up and running with minimal (hopefully 0) bugs this should be fun to play around with and try to make some challenging yet enjoyable levels.

Feedback

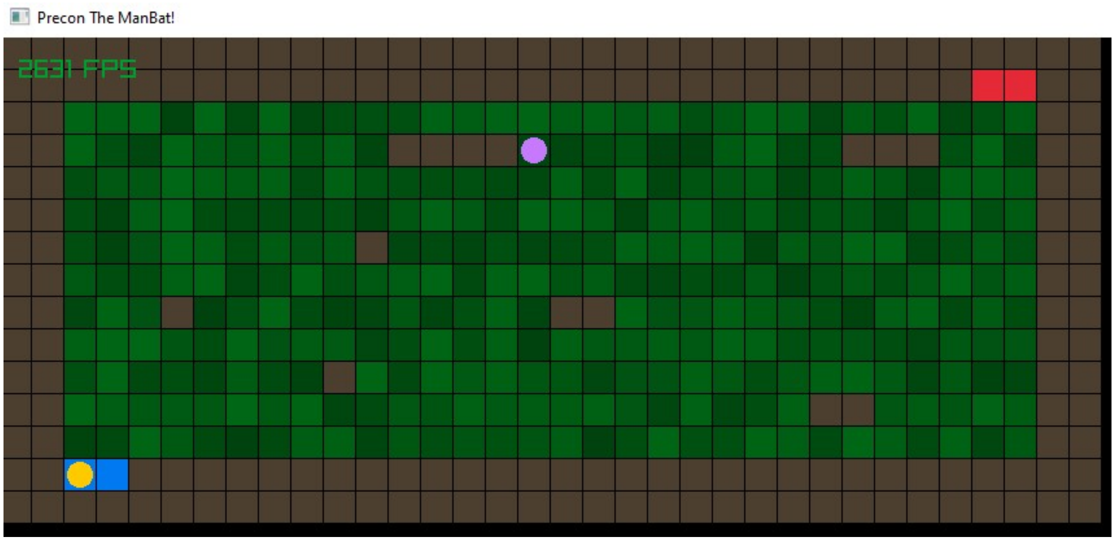
The first thing noticed was that it wasn't obvious to press the space bar to light the screen back up after it had turned dark, as there was no control tool tips on screen. After that was figured out it was super easy for the player to beat the stage and it was declared "too easy". Having the one level made make sense for a starting stage / tutorial level only.

One bug was found in playtesting where the enemy would increase its speed every time the space bar was pressed when in the investigating state. This would cause situations where the enemy would be so fast that it would be impossible to beat the stage at that point. Feedback revealed that this could be an interesting gameplay mechanic to add if it was more controlled.

A final note of feedback was that it could be interesting to experiment with which layers of gameplay disappear with the map turning dark. Having just the map turn dark but being able to see the enemies still could be an interesting mechanic. This led me to think about having enemies only be visible while they were moving and then slowly fade out when they stop, and if that could be an interesting gameplay mechanic. Or have the nearby area around them lit up.

Feedback Response

To get a better feel of how the game would actually play I changed the map layout to have a lot more obstacles in the middle. Which can be seen below. This definetly forced the player to use the mechanic to brighten the screen more often and made it feel more challenging and rewarding to play.



As for the enemies speed increasing bug, I decided to leave it in the game and turn it into a sort of feature. To make it function properly I added a 1 sec cooldown to the screen brightening mechanic. This way the player can't spam the ability and immediately make the enemy go supersonic, and turns the bug into a functioning mechanic, increasing the difficulty the more the player needs to memorise the map.