State Machine Design

The environment simulated is a maze of nodes with a path flowing through valid nodes. It was made using an ascii map, with binary strings deciding the positions of nodes. 1 represents a valid node, and 0 represents an invalid node.

The AI swaps between two states: following and wandering. When the state machine agent is following, it targets the position of the gotopoint agent. The state machine agent has a speed that is half of the player agent.

When the program starts, the agent using the state machine starts by wandering around the maze. When it gets within 5 nodes of the player agent, the agent switches from wandering to following, where it chases the player agent until it reaches the player’s position. If the player gets further than 7 nodes away from the state machine agent, the state machine agent stops following and returns to wandering.

A screenshot of a video game

Description automatically generated

Wander behaviour works first by checking if the agent is currently already going to a node. If it isn’t, then a random node is picked, and the agent goes to it.

Follow behaviour works first by setting the target as the gotopoint agent. Next, it checks if the set target exits, and breaks if it does not. A floating point variable for distance is then set using the distance between the target’s current and last positions. If that distance is greater than the size of a node, the target’s last position is set and the follow agent goes to the gotopoint agent’s last position.

Gotopoint behaviour works by checking if the left mouse button is pressed, and going to the node at the position of the mouse cursor.

To enhance the difficulty, the speed of the state machine agent would be able to be changed. This would make it so the state machine agent either catches up to the gotopoint agent faster or slower. Another enhancement could be changing the state machine agent to have it “get tired” by slowing down after chasing the gotopoint agent for a set amount of time, with it being scalable by how long it takes for that to happen and how much slower it can go. One more example could be having the range that the state machine agent switches while within be wider or smaller. This would make it so the state machine agent either has more or less opportunities to start chasing the gotopoint agent.

Playtest Jasper 24/10/2024:

(version)

The purpose of this playtest is to evaluate whether the program runs as intended.

Did the controls feel intuitive?

Yes, click to move is a good choice. It was clear where you could move to.

Did the agents move how you would expect them to?

Pretty much, the change of colour indicated change of behaviour. Assumed the red one would chase you. The pink one wiggled.

How was the design of the nodemap?

It isn’t complicated enough to get a good grasp of the behaviour. It isn’t big enough for that either, hard to get away from follow agent. Also results in player wandering into chase range too frequently.

Playtester feedback:

Good overall, map should be bigger to show off how the AI work. Maybe some changes to colours would make it clearer what agent did what.

Observation notes:

The presence of a third agent didn’t make sense during gameplay.

Response to feedback:

Future changes would include making the state machine agent switch between red and green instead of blue and pink, and increasing the size and complexity of the map.