Decision Tree 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 decision tree agent is following, it targets the position of the gotopoint agent. The decision tree agent has a speed that is half of the player agent.

When the program starts, the agent using the decision tree 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 5 nodes away from the decision tree agent, the decision tree agent goes to the player’s last seen location and returns to wandering.

A diagram of a player

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 decision tree agent would be able to be changed. This would make it so the decision tree agent either catches up to the gotopoint agent faster or slower. Another enhancement could be changing the decision tree 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 decision tree agent switches while within be wider or smaller. This would make it so the decision tree agent either has more or less opportunities to start chasing the gotopoint agent.

Playtest 25/10/2024:

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

Did the controls feel intuitive?

Yes, simple, clear where you can and can’t go.

Did the agents move how you would expect them to?

Agents were clear, purple guy wanders wherever, light blue changes states.

How was the design of the nodemap?

It was big, multiple paths. Still felt a little tight, could still escape.

Playtester feedback:

Very straightforward, AI moved somewhat predictably, although the pink one wiggled a bit. Changing colour made it clear how the chaser would act. Very cool.

Observation notes:

No outstanding notes.

Response to feedback:

Will fix wiggling agents.