Artificial Intelligence for Games

Plan:

I plan to use a finite state machine to create a game of tag. This will include a chase and flee function for a duck. The duck will chase and flee from the mouse depending on whether or not the duck is ‘it’. A possible alternative for decision making would be to use a ‘Behaviour Tree’.

Some pros to using this method is that FSM’s are great for small systems and this project fits into that category, however, some cons that come with it may include agents getting stuck in inescapable loops or even just setting up the transitions incorrectly. This is only a small project however and these problems shouldn’t pose too big an issue.

For the pathfinding in my game, I will use Dijkstra’s Algorithm to find the shortest path for the duck to follow or flee. I will create a weighted graph with some obstacles for the duck to avoid. As a possible alternative I could use ‘A Star’ for the duck’s pathfinding.

Pros to using Dijkstra’s Algorithm is that it calculates the shortest path, to and from every node on the graph, meaning for the small graph my game will have, the Duck’s Paths will be pre-calculated. The downside however, is that this method Is expensive and shouldn’t be used on larger graphs. It also means I can’t dynamically change the graph or weights without recalculating everything again.

Review:

The project ended up as I had planned it for the most part, however the game of tag wasn’t fully complete. The duck will only switch from chase to flee if I press the buttons to do so. The duck does however follow and flee from the mouses position.

As an improvement for next time, I would like to have the required skills to complete the task as this time I was very reliant on the help of the teacher. I would also like to achieve my goals through more efficient or more complex techniques as I tend to stick to simpler methods.

A different technique I could have used for pathfinding would be ‘A\*’ which is more efficient then Dijkstra’s as it doesn’t search the entire grid as Dijkstra’s does.