**Steering Behaviours – Leyla Asan Mane**

**What?** I have implemented the *seeking* and *fleeing* steering behaviours in my car racing game to show the possibilities of steering behaviours in game-making environments.

The seeking behaviour will follow a desired target. The fleeing behaviour will try to avoid a certain target.

The fleeing and seeking behaviours could easily be implemented into many AI-based games such as those that require the player to verse the environment, practise levels, tutorials and many more. Famous examples are zombie shooter games (seeking) and Destiny (aliens flee when hurt).

**Why?** I chose the steering behaviours task out of the five available because it seemed the most enjoyable for me and something I’m familiar with from the previous games I’ve made. I have some understanding of C sharp - my knowledge of C++ was easily translatable into C sharp.

**How?** The steering behaviours are an artificial intelligence system that is usually used in games to implement enemies, such behaviours that I have chosen are fleeing and seeking behaviours because they may be directly used in association with a car racing game. For example, the *seeking* steering behaviour could be used with PvE (player vs environment) if in easy mode because it will allow the player to win but may give an effect of the possibility of a loss if the vehicles are close to one another, unknown as to who will win.

**Components used:**

* NavMesh Agent: The cars were using the navMesh agent, which allows them to only stay on the baked path of the terrain and not go off-course. This is attached in the components sections of the cars.
* Scripts:

1. Avoider – The avoider script was used for the *fleeing* behaviour car. This script is pretty identical to the follower script, however, during the translate it will go backwards from the car’s vector.
2. ControlWithKeys – This is a script I made that takes advantage of Unity’s built-in input of the WASD and up/down/left/right keys and transforms the player’s car accordingly – I also made it rotate on the Y axis once the left/right or A/D keys are pressed to imitate the player’s car turning.
3. Follower – I took advantage of C sharp’s built in transform tools such as the ‘LookAt’ which allowed the AI car to ‘look’ at the player’s car by rotating the front towards it’s coordinates. I used the Translate method to move the AI’s coordinates towards the player at a rate of 50 pixels per second, multiplying it by the system’s time ‘deltaTime’ which considers the framerate of the system to be able to determine the time.

* Baking the terrain: I chose to use a terrain instead of a plane because I wanted to create textures on the terrain and bake it, baking it meant that I could enable navMesh agent to determine the path that the cars may move onto. To bake the terrain meant that the AI’s wouldn’t run off course as opposed to without it.

**Problems encountered:**

One obvious error is the fact that sometimes the cars run into the walls/off-path and this doesn’t look realistic or appealing, this is caused by problems I had regulating the navMesh agent’s radius, when I would adjust the values very slightly it caused significant differences that made it look even more unappealing to play in regard to game mechanics.

At first, I tried to use the plane instead of a terrain and it didn’t bode well because I was unable to apply textures/brushes to my map as imagined – I prefer a more 3D look.

The ‘avoider’ (flee behaviour) didn’t emerge as I initially planned out. I would have liked to have it randomly move to different locations when the player isn’t nearby, but I had enormous difficulties coding this so in the end I left it out. The avoider only reverses in motion when it detects the player and will stay in that place until it is in view of the player again, whereby it will try to reverse but may not be able to if a wall is behind it – blocking any movement. I still believe that this AI shows fleeing behaviour to a decent degree.

**Citations:**

1. <https://www.youtube.com/watch?v=sXQI_0ILEW4&index=2&list=LLlBeKUr6_hkc6wQw5Lp0Mmw>
2. <https://docs.unity3d.com/ScriptReference/Transform.LookAt.html>
3. <http://u3d.as/9Sf> - Metal floor texture.
4. <http://u3d.as/8Er> - Car textures.