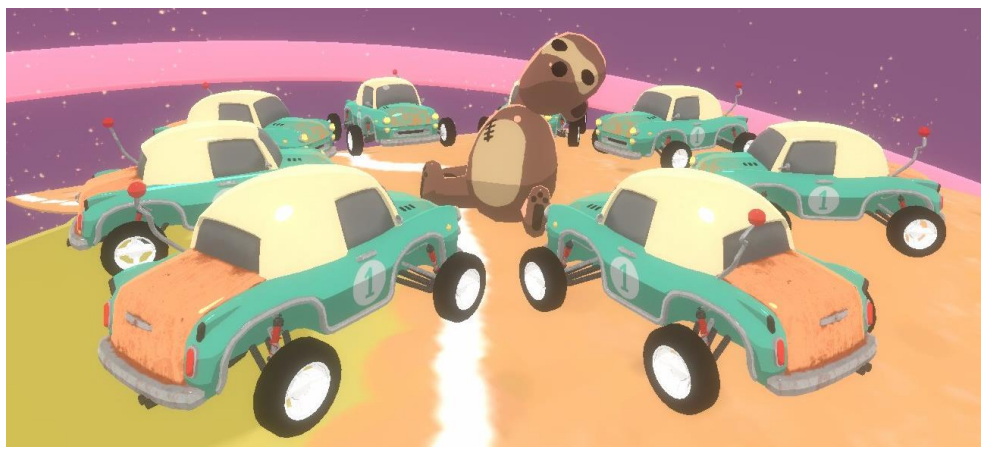
Developing a Utility-based AI for an Online Multiplayer Game

FALMOUTH UNIVERSITY

Luke Day

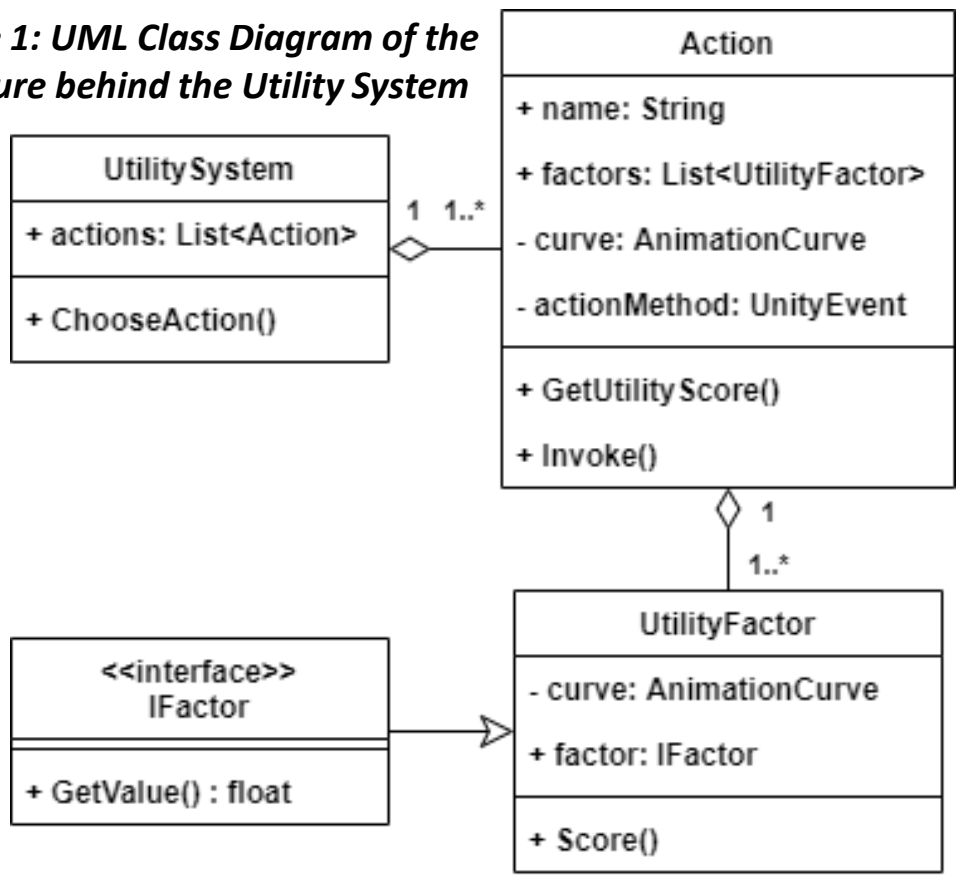
The Problem

Carma is a fast-paced, multiplayer, vehicle-combat game, in which players fight for the top of the leaderboard. However, a major issue with multiplayer games happens when matches aren't consistently filled with players, which means "the player experience is deemed unreliable [and] matchmaking lobbies are left empty" (Cook, 2014), thus making the game feel hollow.



Class Hierarchy

Figure 1: UML Class Diagram of the structure behind the Utility System



Pseudocode

Figure 3: Code snippet from the Utility System script

```
bestUtililty ← 0
bestAction ← null
FOR each action in actions
     avgUtility ← 0
     count ← 0
     FOR each factor in action.factors
          score ← action.GetUtilityScore(factor)
          avgUtility ← calculate new average utility
     NEXT
     IF avgUtility > bestUtility
          bestUtility ← avgUtility
          bestAction ← action
     END
NEXT
```

The Solution

I created intelligent AI bots using a utility-based system, which can effectively mimic player behaviour. I designed it to be easily configurable in the game engine (Figure 2).

The utility system uses 'actions' and 'factors', which "weigh into the calculation of the final utility score" (Graham, 2013). It then picks "the action with the highest expected utility" (Graham, 2013). This action is performed and the process is repeated.

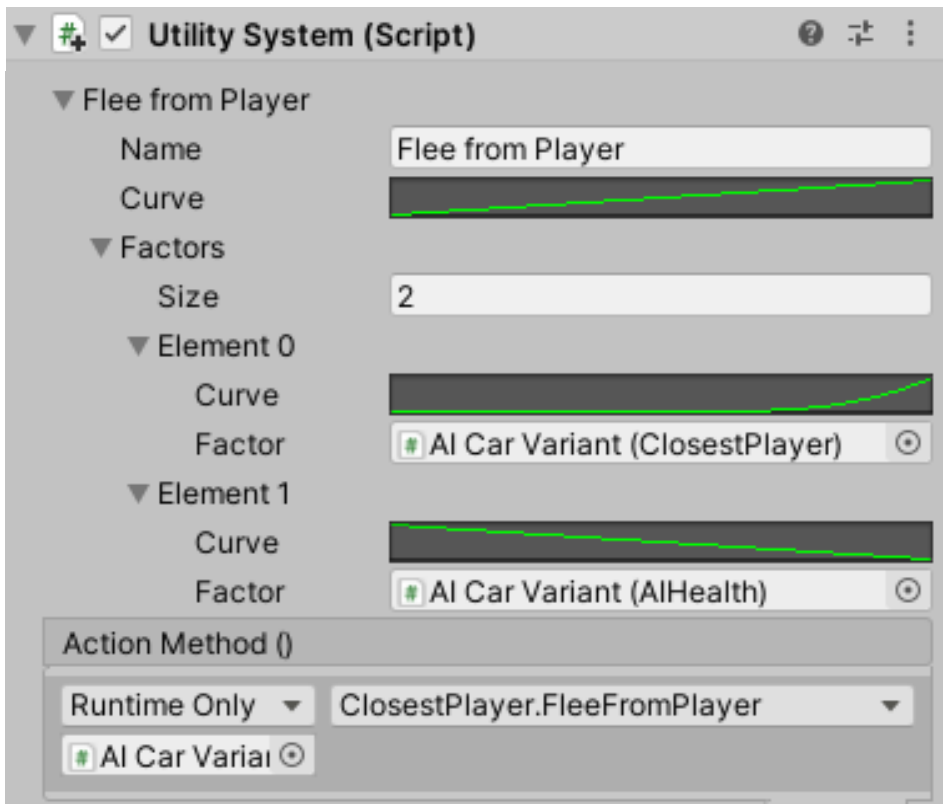


Figure 2: The Utility System, showcasing a 'Flee' action containing 2 factors

A custom navigation method was needed as Unity's 'NavMesh' doesn't work on a sphere. I generate a node structure at runtime and the AI navigate this mesh of nodes, using A* pathfinding.

The Results

By efficiently navigating the planets of Carma and executing the 'best' action, as a result of the Utility System, the AI agents behave intelligently and competitively. Not only has this had a positive impact for the team, by enabling them to easily test and balance features against the AI, but it has also greatly improved player experience and fulfilment.

A performance increase could be gained by removing the need of MonoBehaviour inheritance from factors, as this would greatly reduce the number of components on the AI cars. A further improvement for Carma's AI would be the introduction of a speed factor to the 'Attack' action, as presently it isn't aware of the threshold speed needed to deal damage. Finally, I can see this Utility System being used in variety of projects, outside of Carma, due to its modular and highly configurable design.

References

- Cook, D. (2014). *What I've learned about designing multiplayer games so far.* Retrieved from Gamasutra.
- Rabin, S. & Graham, D. (2013). *Game AI Pro*. pp.113-126.