# Ease of Implementation

Many of the components necessary for this project were able to be taken from my previous AI project, which has a game manager, flocking, agents, and formations. This made implementing these features very easy as I already has most of the framework, with only minor changes necessary. I also had a graph and path finding from another project which could be implemented without any changes necessary. The largest issue I came across was with testing, due to how dependant both AI entities are on each other. Swarmers need a leader to follow and trigger their transition, and leaders need swarmers to control. This resulted in having to largely complete both entities before I could test them, making debugging particularly problematic.

# Performance

Swarmers work perfectly as intended in numbers around 70, staying with the leader and not falling too far behind. They become more troublesome, however, in larger numbers. Despite the leader having the impact of 500 swarmers, a large enough swarm has enough cumulative impact on itself to not be impacted by the leader. This results in the swarm ignoring the leader and going off in their own direction, only being pulled towards the leader when they get too far away, where the alignment force kicks in. The effect is like a wild swarm being leached to the leader, rather than following it.

In computational performance, the game is able to maintain 60 FPS with a small swarm, however when a larger swarm is in play, there is significant slowdown, typically down to 30-20 FPS. This is the result of all the swarmers having to ‘talk’ to each other since their all in such close proximity.

The leader’s path finding works as intended, however the result is very ‘point to point’ rather than following a path. This is the result of using an arrival behaviour. While not ideal, it is the most effective method, as the leader would end up constantly hitting walls otherwise.

# Improvements

In relation to large swarms not following their leader, this issue can’t be fixed in any simple way. One possible way would be to scale the leader’s importance with the size of the swarm, but I expect this to provide its own set of issues. An alternative of reducing the swarmers’ importance as the swarm increases may be more effective, but this would be at the cost of performance, and large swarms already have a significant performance cost.

With the current method being used for flocking, the performance of large swarms can’t be improved by much. However, if the leader was to determine the flocking forces for each swarmer, the performance would be drastically improved. A leader could go though its swarm and check if swarmers after the current swarmer are in range, then determine the forces for both swarmers, instead of the vector being fully iterated over by the full swarm. While more efficient, this method would also more tightly couple swarmers to leaders than what is necessary.

The best way to improve a leader’s path finding would likely be to use more nodes allowing a more precise path be found. This should allow a weaker arrival radius to be used, slowing the agent down less, and reducing the jagged movement between nodes.