Agent Coordination – Wolves

My idea was to build a wolf that forms a pack using local information. I made and improved my Wolf agent over three main versions, each trying to address issues that arose along the way.

In my first approach, the wolf agent had three states:

1) Search State - When a wolf does not see any other wolves, it follows a random search direction. The wolf continues moving in that direction, occasionally changing it with a 20% probability. This state is made to explore the world when the wolf is alone and look for other wolves and prey.
2) Pack Formation - When one or more wolves are visible, the wolf calculates the Manhattan distance to find its nearest mate. If no prey is visible, it moves toward the nearest wolf to form a pack.
3) Hunting State - If the prey is visible and the nearest wolf is close (distance < 3), the wolf switches to a hunting state. But instead of a direct chase, the wolf tries to surround the prey.

A problem in this version was that I used freeze: if the nearest wolf was very close (distance < 2), the wolf would freeze for 1–5 ticks to avoid jitter. This prevented the oscillation but also caused the pack to become stationary.

To address this problem, I tried to introduce a common pack movement direction. So, when no prey is visible and the pack is tightly grouped, instead of freezing, the wolves use a common movement direction that is randomly initialized and used until prey appears. The idea was to have the pack move together in one direction, rather than oscillating or stopping at one place.

This mostly worked but sometimes the code was constantly switching between gathering and using common movement which led to inconsistent movement, sometimes preventing the pack from moving consistently. Despite this, the wolves were able to capture the prey faster than in the previous version.

In the final version, I removed the freezing completely. Instead of stopping when wolves are close (distance < 2), I apply a small random offset. With this approach, the wolves no longer become stationary when tightly packed. And they continue moving, with small random movements to avoid jitter. Thanks to this the wolves move together in the pack, exploring the world and capturing prey much faster than before.

Homogeneous vs. Heterogeneous Team:

I experimented with three working versions of my wolf agent, all employing the same coordination logic (homogeneous team). I found this approach easier to implement as my idea from the beginning was to pack all wolves together and then move into a pack with all having equal roles. However, I experimented a bit with mixing different versions of my wolf agent in a single world and surprisingly they worked quite well.

Dealing with a RandomWolf in the Pack:

I tested different scenarios - 2 my wolves with 1 RandomWolf, and 1 my wolf with 2 RandomWolves. My wolves were able to either form a pack with another of my wolves or follow one of the random agents and form a pack with it.

Changing the Number of Wolves:

Increasing the number of wolves generally resulted in faster prey capture because more packs were formed and generally the probability of capturing the prey increases. With fewer wolves, the pack formation took longer, and prey was sometimes harder to capture.

Requiring More Wolves to Catch a Single Prey:

If the rules were modified to require more wolves (three instead of two), my coordination strategy still work since the wolves naturally form clusters (also bigger than 2). However, it might require more time to capture the prey.