Kernmodule 3 - Sergi van Ravenswaay - 3017869

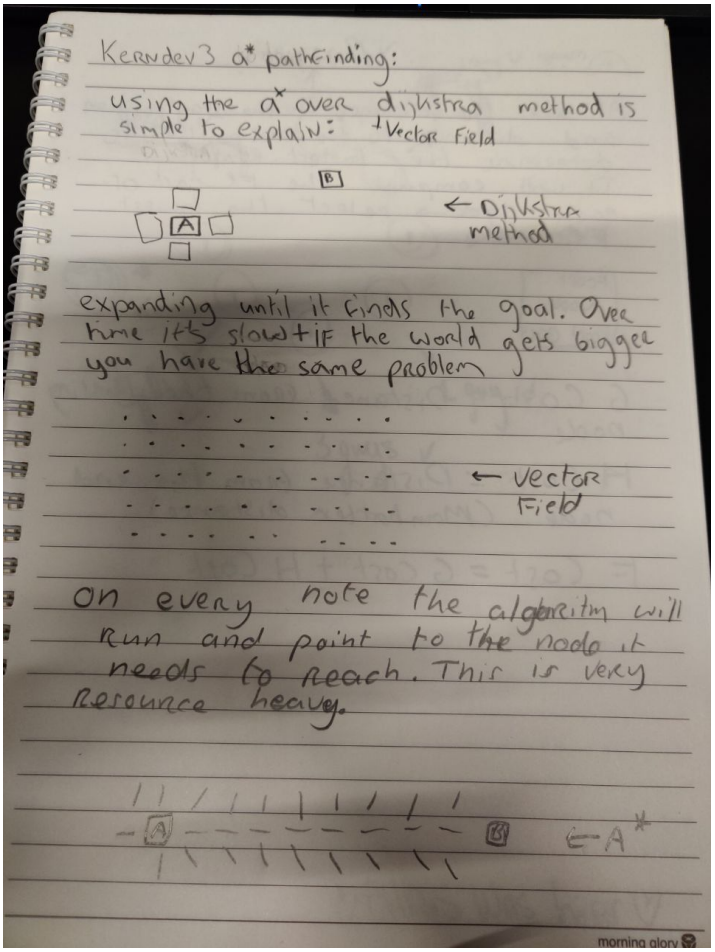
Mijn spel
Dit spel gaat over jij die als speler de schaapjes genoeg moet voeden en zorgen dat de wolven worden uitgeschakeld. Je kan op de map stro droppen die de schaapjes dan opeten. Zodra de schaapjes alles dat eten hebben gegeten krijgen ze weer energie om verder te roamen.

De wolven in het spel roamen en jagen op schaapjes die lagere energie krijgen en honger krijgen. Zodra deze schaapjes in het gebied van de wolf komen kan jij ze nog redden door ze te met water te beschieten. Op die manier gaan de wolven verder met het zoeken naar een andere prooi. Na verloop van tijd als je dit vaker doet terwijl ze een schaapje zoeken zullen ze naar een ander gebied gaan. Als de wolf jou een prooi

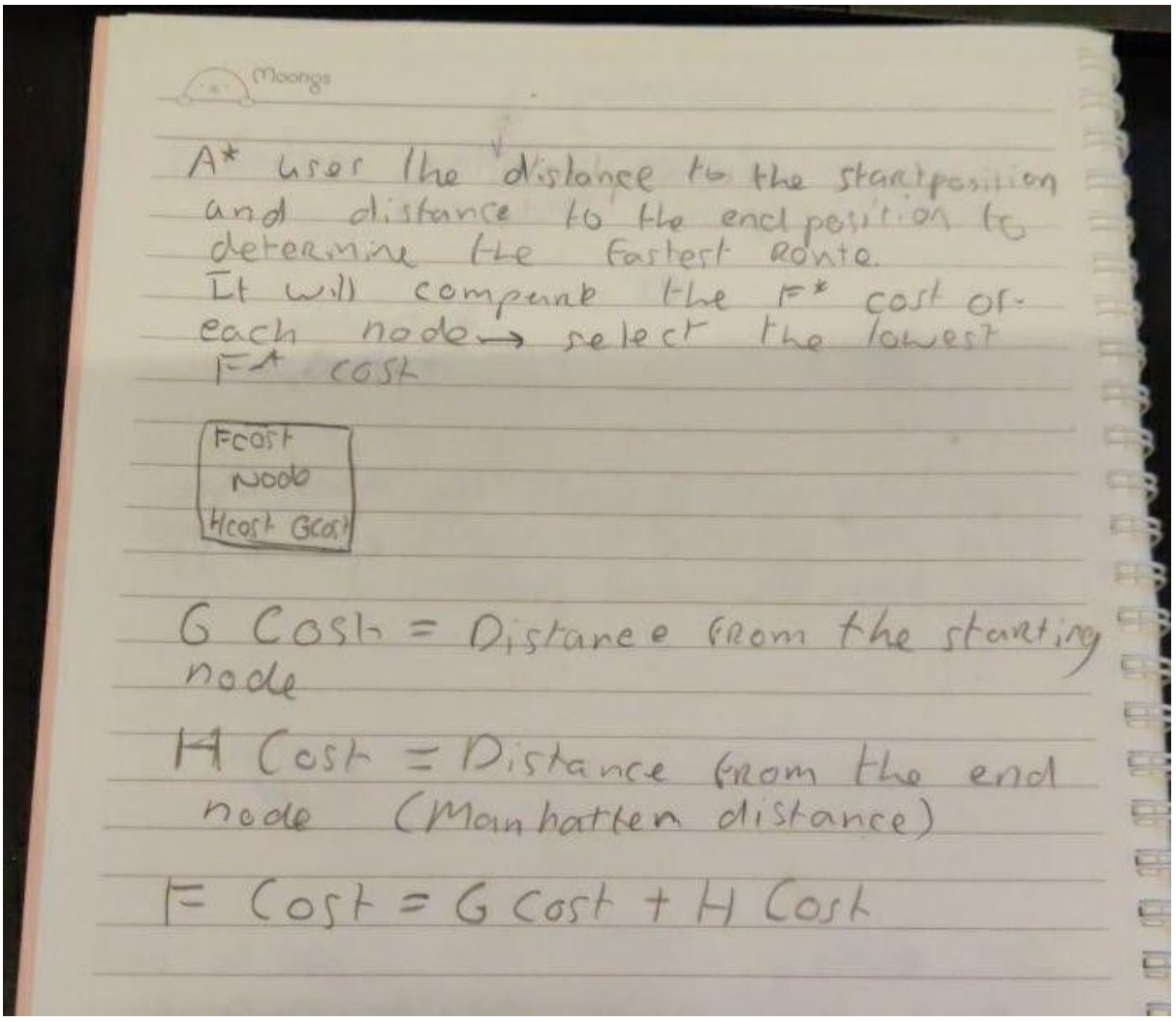
Als jij je lammetjes goed verzorgd zullen er steeds meer schaapjes bijkomen en ook meer wolven. Als jij de schaapjes slecht verzorgd dan zullen de schaapjes sterven.

Belangrijke mechanics
Om het spel balanced te houden heb ik verschillende features geïmplementeerd:

- Je kan maar vanaf een bepaalde range water schieten
- De schaap ziet het voedsel pas als het dicht bij hem wordt neergezet
- Je kan maar drie keer water schieten en dit zal weer regeneraten na een aantal seconden
- Wolven hebben een specifieke range waarin zij de schapen kunnen vangen



Implementatie Pathfinding
Research
Voor de pathfinding heb ik doorgelezen hoe verschillende pathfinding werken. Uiteindelijk heb ik voor A* pathfinding gekozen.



Toepassing

Uiteindelijk heb ik dit algoritme verwerkt in Unity en kan een object een begin en eindpositie aanvragen. Dit grid zal dan via horizontale en verticale kant een pad berekenen en deze returnen naar het object wat dit pad heeft aangevraagd. De volgende classes heb ik aangemaakt om deze pathfinding hierin te verwerken.

- Grid: In dit script worden er een grid met nodes aangemaakt. In de inspector kan je basisgegevens over het grid invullen (grootte, node radius, muurlayer etc.). Dit script doet naast maken ook het grid updaten om eventuele nieuwe muren, voedsel of entities te detecteren. Op die manier kan de data makkelijk worden doorgestuurd naar de verschillende objecten in de game.
- Node: Hier staat alle data in van elke node;
    - G-cost, F-cost & H-cost
    - Positie in de grid en Positie in de node
    - Of het een muur of voedsel is
    - Parent node; de node die de als volgende wordt aangewezen als deze in een pad wordt gebruikt.

- Pathfinding: In dit script staan alle pathfinding functies.
    - FindPath/GetFinalPath - Deze functie krijgt een begin en eindpositie mee en loopt daardoor het grid heen. Hier krijgt hij de nodes terug die van de begin naar het eindpositie lopen en voegt deze toe aan een lijst. Uiteindelijk wordt deze lijst gestuurd naar het object die deze posities heeft gegeven.
    - ReturnFood: Stuurt een node toe die hij krijgt van het grid waar voedsel kan liggen. Dit wordt aan de hand van een positie gedaan die hij binnenkrijgt.
    - GetManhattenDistance: Berekent de HCost tussen de twee nodes die hij binnenkrijgt.

  PathfindingManager: In dit script worden functies van de Pathfinding script aangeroepen. Om er voor te zorgen dat elk object de juiste data meekrijgt heb ik er een Queue ingezet. Deze zorgt er voor dat 1 voor 1 de spelers alle data binnenkrijgen. Ook heb ik in dit script een functie gemaakt die een link maakt naar het grid. Hier kan dan data van worden gevraagd voor voedsel.

Behaviour Tree

Voor de behaviour trees heb ik gebruikt gemaakt van Panda BT. Aan het begin had ik veel moeite om deze behaviour trees goed te begrijpen. Ik probeerde veel gedrag in een functie te doen en deze dan uit te voeren. Na onze meeting en ook zelf er wat beter na te kijken is het mij uiteindelijk gelukt de behaviour trees goed te implementeren. Ik heb de volgende AI's gemaakt die van hun eigen Behaviour gebruik maken.

Sheep:

Het schaap heeft het volgende behaviour:

- Roamen: Hij loopt rond en volgt een pad die hij van de A* heeft gekregen.
- Findfood: Als het schaap honger is vraagt hij op of er voedsel bij hem in de buurt is; zo ja dan vraagt hij een pad naar het voedsel op.
- Starving: Mocht hij geen voedsel vinden dan zal het schaap een beweging maken en opnieuw proberen naar voedsel te zoeken.

Wolf

De wolf heeft het volgende behaviour:

- Roamen: Hij loopt rond en volgt een pad die hij van de A* heeft gekregen.
- CheckSurroundings: Hij kijkt of er in de buurt een schaap is wat honger heeft. Deze zal langzamer lopen en kan hij stunnen. Mocht dat lukken dan kan hij hem opeten.
- Scared: Als hij wordt geraakt door water wat de speler schiet wordt hij bang. Deze zal hij het lammetje niet opeten. Ook verliest hij health.

Conclusie

Zelf ben ik na twee jaar uitstellen aan deze kernmodule begonnen. Zoals ik al heb aangegeven is dit absoluut niet mijn favoriete kernmodule, maar heb ik wel de afgelopen periode toch meer inzicht in A.I. gekregen. Zo heb ik na ons gesprek mijn hele BT's opnieuw gebouwd. Ik bouwde ik kleine functies en ik kreeg door hoe verschillende functies (fallback, sequence, race etc) werkte. Toen begon ik het toch interessanter en leuker te vinden dan eerst. Uiteindelijk ben ik toch blij dat ik een game met twee redelijk werkende A.I. kunnen maken.