Programmeren opdracht 3: Intro to agents and agent-tools - individueel

Naam : Sam van Roosmalen

Studentnummer : 1782358

Groep : 7

Inleiding:

Ik heb de agent tool unity gekozen. Ik heb in unity zelf gewerkt met een 3D project. En geprogrammeerd in C# in Vscode. In de Tutorial, die je als project kan openen in unity. Zit in de folder assets een scriptje New behavior script. Als disclaimer: Het environment is niet af, er ontbreken veel dingen, maar in de uitleg beschrijf ik het volledige environment.

## Volg de tutorial en omschrijf daarna in één paragraaf wat deze tool anders maakt dan andere programmeertalen, wat zijn de voor- en nadelen?

*Is het om een of andere reden niet gelukt de tutorial af te maken: beschrijf waar het mis ging?*

Het is gelukt om een tutorial af te maken, maar niet gelukt om een agent te maken. In mijn omgeving is er een bal die vooruit beweegt, en op zijn weg zijn er kubussen toegevoegd. Als de ball de kubus aanraakt vernietigd de bal de kubus en beweegt verder. Maar het is geen omgeving die dingen mee geeft aan de agent. Om vervolgens echt te handelen in een oneindige loop van states acties en percepties.

Dus voor een uiteindelijk product had ik bedacht dat de wereld random kubussen spawnt. En dat deze bal vooruit beweegt en elke keer als hij een kubus waarneemt de kubus verwijderd. Als hij geen kubus waarneemt dan beweegt hij vooruit.

*Als je een bestaand voorbeeld gebruikt (Netlogo heeft er vele): beschrijf de kernfunctionaliteit van het voorbeeld en voeg minimaal een element toe aan de omgeving en licht toe. (bijvoorbeeld: het meest eenvoudige economiemodel in Netlogo is agenten die random geld uitdelen, je kunt dat aanpassen naar geld uitdelen aan agenten die dichtbij hen staan.*

Ik heb niet een bestaand voorbeeld gebruikt.

## We definieren een staat en drie functies waarmee we een stateful agent abstract omschrijven:

*Een initiele staat i0∈I, waarbij****I****alle mogelijk interne staten van de agent zijn*

Alle mogelijke interne staten zijn bewegen, waarnemen en verwijderen.

*Een functie "See" of "Perceive", die een mapping maakt van elke staat in de omgeving tot een staat die de perceptie van de agent van de omgeving aangeeft. Dus: See: S⟶P, waar****S****de staat of serie van staten van de omgeving is en****P****de perceptie van die omgeving*

Vanaf de initiele staat maakt de bal een waarneming S, als de S een kubus ziet binnen 1 grid vakje, dan is P de percieve dat er een kubus in de weg staat. Als de P geen kubus is handelen de twee volgende acties daar op, door gaan of verwijderen.

*Een functie "Act" die een kijkt naar de staat van de agent en een toepasselijke actie kiest. Dus Act: I⟶A, waar****I****een interne staat is en****A****een actie.*

De interne staat is bijvoorbeeld dat er een kubus in de weg staat (P is een kubus waargenomen recht voor hem). De kubus krijgt dan als actie om de kubus te verwijderen, als de kubus is verwijderd is de nieuwe perceptie dat er geen kubus is dus de bal vooruit kan bewegen.

*Een functie "Update" die een staat****I****neemt (soms****D****genoemd) en perceptie****P****, en een nieuwe staat****I****oplevert, dus Update: I×P⟶ I*

De update oftewel ‘next’ gebruikt de oude state en i0 en samen met de nieuwe perceptie een nieuwe i berekent namelijk i1. In mijn machine zou dat dan zijn. De oude state zijnde er is niks in de weg dus de bal beweegt vooruit. De nieuwe state van de omgeving is dat er een kubus recht voor hem in de weg komt te staan. De see functionaliteit van de agent zou dan die kubus zien en ‘percieven’ daarmee berekent die met de oude state dat hij nu niet verder kan en als actie de kubus zou moeten verwijderen.

## Beschrijf je omgeving op basis van de dichotomiën die [hier](https://canvas.hu.nl/courses/27428/files/2223184/download)op pagina 6 beschreven staan, en licht toe (dus niet alleen termen opsommen):

Accessible vs inaccessible

Mijn model is zeker accessible, omdat het model simpel is in de zin de agent heeft altijd een antwoord op de situatie en de situatie is absoluut, zonder dat je dingen toevoegd.

Deterministic vs non-Deterministic (Stochastic)

Het is niet-deterministisch omdat er een random tussen stap gedaan wordt, namelijk de kubussen worden random gespawned

Episodic vs non-episodic (Sequential)

Mijn environment is episodic omdat er geen correlatie is tussen de ene actie en de andere. Elke keer herhaalt die dezelfde overweging eigenlijk, namelijk lopen of verwijderen.

Static vs Dynamic

Mijn environment is dynamic omdat het random inspawnen van objecten een ander process is dan de agent zelf.

Discrete vs continuous

Mijn environment zou continuous zijn, omdat er geen einde is en geen finiet aantal stappen.

## Bedenk een voorbeeld waarbij minimaal 3 dichotomies precies tegenovergesteld zijn en beschrijf of het veranderen van je omgeving op deze manier wel of niet iets zou toevoegen aan je simulatie. Bijvoorbeeld: in een verkeerssimulatie maakt het soms wel echt uit voor de resultaten van je simulatie of de omgeving accessible is, maar bij het Nägelschreckenberg-model niet. Probeer te bedenken wat het effect van dit soort keuzes wat betreft je omgeving is op wat je kan leren/wat voor resultaten er uit je simulatie komen

Als ik het zou veranderen naar **discreet** **inplaats** **van** **continuous**, zou er dus een finiet aantal kubussen geplaatst moeten zijn en een eind streep moeten bestaan. Je zou bijvoorbeeld kunnen zeggen dat je dan een specifiek geval simuleert om dan te testen hoe het object (de bal) daarop zou reageren. Persoonlijk zie ik hier niet het nut van in, aangezien mijn enviroment nu niks specifieks onderzoekt en dan ook niet. Maar je zou het argument kunnen maken dat er een specifiek scenario gemaakt zou kunnen worden en dan simuleren hoe de bal (een representatie van een object) daar mee om gaat. Daarmee zou het ook meteen **Static** **zijn** **inplaats** **van** **dynamic**, omdat er een finiet aantal stappen bestaan.

Je zou het enviroment **non-episodic kunnen maken inplaats van episodic** door een correlatie maken tussen de geplaatste kubussen. Bijvoorbeeld als de ene kubus verwijderd wordt de andere uitwijken voor de bal die beweegt naar de eindstreep. Er vanuit gaan dat het hier boven geschreven uitgewerkt zou zijn. Hier geldt ook weer dat het enviroment nog niks representeert of onderzoekt. Je zou kunnen zeggen de bal is een vrachtwagen en de kubussen worden auto’s. Je rijdt in tegenovergestelde richting. Als je botst tegen de ene auto (kubus) wijken de andere uit. Om maar te simuleren dat bestuurders een ongeluk uit de weg gaan.

## Conclusie

Als conclusie concludeer ik dat het niet optimaal is om unity te gebruiken, de leer curve is te groot om in een week verder te gaan met een groepje. Al hoewel je mooie simulaties kan maken en veel mogelijkheden hebt. Bijvoorbeeld je kunt in 3D werken. Er zijn veel tools om dingen te doen zonder te programmeren, zwaarte kracht bijvoorbeeld. Maar het programmeren in C# is te lastig om snel te leren met zijn alle. Ik zou dus ook niet aanraden om met Unity te werken.