Ik heb de Unity tutorial (met gebruik van de basis scene) compleet kunnen maken, en werkend kunnen krijgen

Het leuke aan deze tool is dat inprincipe bijna “alles” wat je zou willen programmeren/visualiseren mogelijk is. Het probleem hierbij is dan ook gelijk dat omdat er zo enorm veel mogelijk is dat het ook erg ingewikkeld is om zelf een oplossing / uitwerking te bedenken met weinig kennis van de engine. Hierdoor is het dus ook gelijk nodig om ERG veel kennis op te doen voordat je iets nuttigs of moois zou kunnen maken. Bijvoorbeeld bij python plugin matplotlib kan je vrijsnel een mooi en nuttig plotje maken, maar als je daar 3d visualisaties wil maken kan je maar beter een andere tool pakken.

Ook doordat je eigelijk alles zou kunnen maken is dit programma niet specifiek agent based. Ik zou eerder dit project Object/scene based noemen waar je daarna van objecten, met wat eigen inzicht agents zou kunnen maken. Dit zorgt ervoor dat je niet strikt aan agent based simulaties gehouden word, en kan je ook andere dingen visualiseren / maken.

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

1. Een initiele staat $i_0 \in I$, waarbij ***I*** alle mogelijk interne staten zijn
2. 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:* $S \longrightarrow P$, waar ***S*** de staat of serie van staten van de omgeving is en ***P*** de perceptie van die omgeving
3. Een functie "Act" die een perceptie van de omgeving neemt en een toepasselijke actie kiest. Dus *Act:* $I \longrightarrow A$, waar ***I*** een interne staat is en ***A*** een actie.
4. Een functie "Update" die een staat ***I*** neemt (soms ***D*** genoemd) en perceptie ***P***, en een nieuwe staat ***I*** oplevert, dus *Update:* $I \times P \longrightarrow I$
   Dit is een korte samenvatting van wat er staat in: "An introduction to Multi-Agent Systems" chapter 1.1 up to 1.4. Die paragrafen kun je hier vinden.
   Natuurlijk worden deze functies stukken complexer wanneer je agenten complexer worden. Beschrijf nu in je eigen woorden wat deze functies zijn in de context van de simulatie uit je tutorial.

Met bovenstaande definitie en kijkende naar de opdracht geef ik nu met corresponderende nummers de context mee van de definitie:

1/2. De initiële staat van de tutorial is een wandering state van de AI. De enige verandering in “staat” is dat het continue enemy detected print als de tank inzicht is, en/of enemy touched print wanneer de tank de AI aanraakt. Dit zorgt er dus voor dat hij in merendeel van de tijd in de initiele staat zit, maar dan kan wisselen naar staat 2 die ik defineer als een “printing”state met meerdere mogelijke print opties. Ik zou niet deze 2 opties als een andere staat defineren aangezien hij deze print opties alle bij tegelijk zou kunnen hebben. Hij kan namelijk de tank zien, en aanraken.
3. Doordat de enige “Acting” actie die de Ai heeft is een print functie, hij past zijn route of wandering route niet aan met de nieuwe informatie als de tank in zicht is, hierdoor zou ik de AI

misschien niet eens een echte intelligente agent noemen kijkende naar de definitie van de paper (hoofdstuk 1.2.2) maar dus een “purely reactive agent.”
4. Ik zelf ben van mening dat deze functie niet eens echt bestaat, er is alleen een actie optie. Aangezien de agent niet echt zijn gedrag veranderd, maar alleen reageerd op zijn omgeving voorzich.

Hier onder geef ik aan wat deze agent is met betrekking tot de definities in de paper.

1. De agent zijn omgeving Accessible aangezien de simulatie in een gecontroleerde omgeving gebeurd.
2. De agent is Deterministic omdat er maar een gegeven aantal mogelijke reacties zijn die kunnen gebeuren, en mogelijke opties zijn om te doen. Hierdoor is de agent makkelijk te uitteleggen.
3. De agent is Episodic aangezien hij geen rekening houd met vorige acties of acties die mogelijk gaan komen. Hij leeft als het waren in het hier en nu.
4. De agent zijn omgeving is lichtelijk Dynamic aangezien een speler de tank zou kunnen verplaatsen buiten de macht om van de agent.
5. Ik zelf twijfel bij deze, maar denk dat bij deze simulatie relatief richting continue zou defineren. Meer omdat de tank op elke willekeurige locatie zou kunnen zijn, en zelfs uit de “simulatie doos” kan ontsnappen.Ook is de wander state van de agent willekeurig. Maar aande andere kant waarom ik twijfel is omdat er maar 3 mogelijke acties zijn, die kunnen gebeuren. Hierdoor zullen er niet enorm veel verassingen zijn.

Een situatie waar bijna alle dichotomies tegenovergesteld zijn van mijn simulatie (behalve dat ze allebei continue zijn.) is bijvoorbeeld een zelfrijdende taxi in de echte wereld. Deze moet met ongelofelijk veel verschillende dingen rekening houden, acties maken en een route plannen hierdoor moet hij ook de vorige episodes rekening houden en plannen waar hij hierna heen moet. Ook zijn de acties niet altijd

Als er drie dichotomies tegenovergesteld zijn dan is de simulatie wat interessanter om te visualiseren. Aangezien dit meer variatie geeft in de mogelijke acties die de agent zou maken en zijn moeilijker om zelf te bedenken. Maar aan de andere hand maakt dit de simulaties wel veel moeilijker om te maken, aangezien je mogelijk met veel tot bijna onmogelijke hoeveelheid situaties moet rekening houden tijdens het programmeren.