1.

Het grootste voordeel van het gebruiken van de Unity game-engine is dat, omdat het een game-engine is, goed gebruikt kan worden om realistische simulaties te maken. Het bevat een hele grote tool set die je kan gebruiken om een ai ook echt 'te laten zien', zwaartekracht op uit te oefenen en fysiek objecten laten aanraken. Het presenteren van zo een simulatie is dan ook veel duidelijker voor de gemiddelde persoon. Al helemaal als je ook mooie modellen en physics erin stopt. Het nadeel is wel dat Unity heel groot is, en ook is bedoeld als game-engine. Dit zorgt ervoor dat je eigenlijk een gehele nieuwe logica en taal moet leren. Wat 'getComponent<>' doet, de logica achter 'vector3'. Unity is niet zo simpel om te gebruiken en bevat ook een beetje zijn eigen logica van hoe dingen werken.

Unity, vanwege zijn grootte, kan agent based zijn. Er is zelfs een custom library agent based modelling toolset speciaal voor unity gemaakt zoals ABMU. Je kan objecten een compleet autonoom script geven dat alleen voor zichtzelf werkt maar ook interactie kan hebben met zijn omgeving. Ze kunnen, in theorie, hun eigen beslissingen maken en ervan leren.

2.

1. De initiele staat i0 is in de context van de tutorial de veranderende variable die in start worden aangeroepen in de scripts Wander.cs, Sense.cs, Perspective.cs en Touch.cs. Vooral de 'Initialize()', elapsedTime = 0.0f, de eerste 'GetNextPosition()'
2. De functies See worden gedaan in de scripts Sense.cs, Perspective.cs en Touch.cs. In Sense.cs wordt vooral alleen van variable vastgezet en de script Perspective.cs mee aangeroepen. De functie UpdateSense() is wat de updated wat de ai aan het zien is 60 keer per seconde (60fps). De scripts Touch.cs wordt getriggered wanneer jij in aanraking komt met de ai zijn hitbox. Dus als als de state van de omgeving zo is dat de player binnen de raycast of hitbox zit van de ai, verandert te perseptie van de ai binnen 1/60 ste van een seconde.
3. In de context van de tutorial gebeurt er niet veel wanneer de ai een actie onderneemt. Als de P verandert naar wel een player zien/in hitbox, in de voor benoemde scripts, dan schrijft de AI een print() functie met dat er zicht is. Dit gebeurt 1x per hitbox touch en 60 keer per seconde bij het zien van de player.
4. In de tutorial wordt een update functie niet echt gebruikt. Als in er worden geen nieuwe staten geleverd. Als een player wordt gezien of niet gezien blijft de AI dezelfde acties ondernemen. Dit is waarom erbij wel de player zien het bericht 60x elke frame wordt gegeven. Zelfs de wander.cs file verandert niet door de P, waarmee de functie IxP->I niet wordt voldaan. Dus je zou kunnen zeggen dat de functie meer lijkt op $I^{n} \times P \rightarrow I^{n+1}$. Waar de I hetzelfde bijft.

3.

1. Inaccessible: De AI in onze tutorial heeft niet access tot alle informatie van de invironment. Het weet nooit waar de obstacelen
2. Deterministic: De gelimiteerde acties die de AI uitvoeren (een line printen en naar een locatie gaan) hebben geen enkel invloed op het environment. Waardoor elke action een gegarandeerd effect heeft van 'geen verandering'
3. Non-episodic: De Environment bevat geen episodes. De agent zit in een constante lijn aan updates dat zo snel gebeurt als dat Unity kan displayen (60fps)
4. Static: Zelfs met de acties van de AI blijft de environment onveranderd. Er is niks in de simulatie dat het environment verandert
5. Discrete: Het aantal acties die de AI kan verrichten aan de Environment is fixed omdat het 0 is. De agent kan niks veranderen aan het environment.

4.

Dynamic: Het zou goed gebruikt worden om de simulatie realistisch te maken met een player erin. Als de speler veranderingen kan aanmaken ipv de AI zal het dynamisch worden.

Accessible: Access hebben tot het gehele environment zou gebruikt kunnen worden om de AI perfect te laten navigeren door het veld heen. Waarmee de AI de goed kan calculeren naar welke positie het heen moet voor zijn acties.
Non-deterministic: Als het environment een RNG zou bevatten voor bepaalde acties van de agent zou de evironment non-determinstic worden, hiermee kan je bijvoorbeeld een simulatie maken waar de AI de speler probeert te zoeken maar kan falen.