Vlad Samonin

COMP 3190

COURSE PROJECT

"Proximal Policy Optimization in the Augmented Reality Environment"

1. Introduction.

1.1 Background

The goal of this paper is to expand on the topic of artificial intelligence applied in Augmented Reality. The augmented reality is the new way of interaction where the user can see digital 3d objects within the real environment. There are many advantages such as an ability to see the 3d world from all the angles, where the user can simply move, rotate the AR device and experience the full power of the 3d with six-degree of freedom. The six-degree of freedom allows not only to move through x,y, z-axis but also free of rotation in each of the axis.

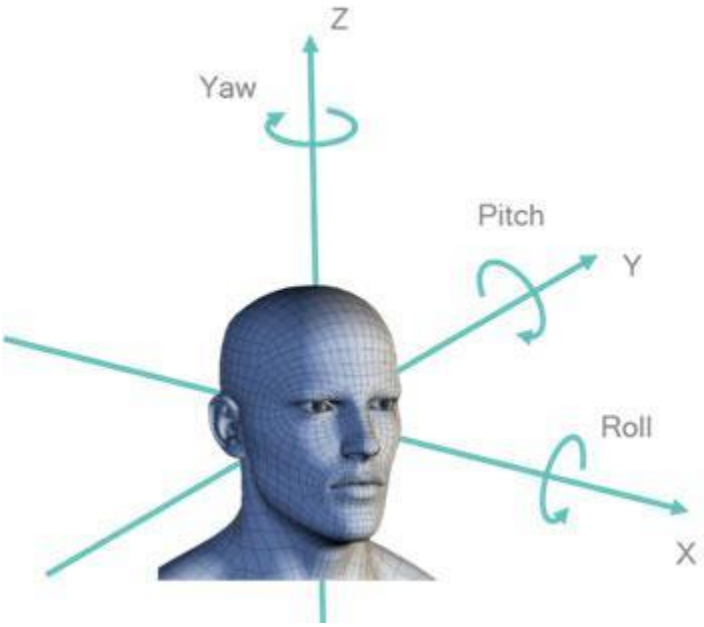

Figure 1.0 Six degree of freedom Illustration [1].

There are many challenges which a developer has to overcome when designing an AR application. One of them is how to apply Artificial Intelligence in the uncontrolled environment within different dimensions. Even simple physics applied for the AI agent in the AR game would be challenging and require more UX, Acceptance testing than the regular 3d game. Therefore different approach required for the developers while designing the AR application. The paper tends to describe some of the challenges, and development suggestions based on the AR project created for this. With using proximal policy optimization as a reinforcement learning model.

There are different platforms used for the reinforcement learning research and algorithms. Such as VizDoom, Mujoco and others [2]. They are created within the assumption of the control environment. For AR project the decision was made to go with the Unity game development engine. Since it allows not only to develop AR applications relatively easy and ability to integrate the existing assets from the Unity Store but also to target different platforms. Since the Unity is .NET based it allows to develop AR application for the Android, IOS and even AR glasses such as Microsoft HoloLens or the most recent one Magic Leap.

Unity team provided the open source tools Unity ML-Agents Toolkit for an ability to add PPO to the Unity project [3]. Where the machine learning techniques can be added to the project. Using this tool, the agents can be trained using reinforcement learning. Which is based on the Tensor Flow, whereby running python scripts within Unity the agents can be trained to perform certain tasks.

The reason why reinforcement learning is beneficial in certain scenarios such as AR game development. Is it the balance of the game, if the game is too easy such as playing against random AI the user will become bored and quit playing the game. On the other hand, if the user plays against extremely smart AI which uses A* algorithm, for instance, the game may become too challenging to play and the user will eventually quit playing as well. Therefore, if the AI is able to “think” and behave human-like making mistakes time to time and have the desire to win in the same time that’s an AI which user will want to face off.

1.2 Proximal Policy Optimization algorithm

PPO is based on the training the agent using the reinforcement by the neural network to make an optimization to the given function [4]. Which takes the observations from the user such as the distance to the goal, health and others. After that, it passes the trained data into the “Brain” which makes the decision which best action the agent should make in the given state. The PPO is run in the Python process by using TesnsorFlow and commutating with the Unity through the socket. There different useful parameters you can set such as Learning Rate, Batch Size, Time Horizon but for the simplicity of the project, the default parameters were chosen. The figure above illustrates the overall architecture of ML-Agents PPO algorithm. Where each Agent is connected to the Brain. Multiple Agents could be connected to the one brain object. Brain object is making the decision based on the observation results obtained from the Agent. And a single

Academy object exists which acts as a connector between Machine Learning Python API script and the Unity training environment.

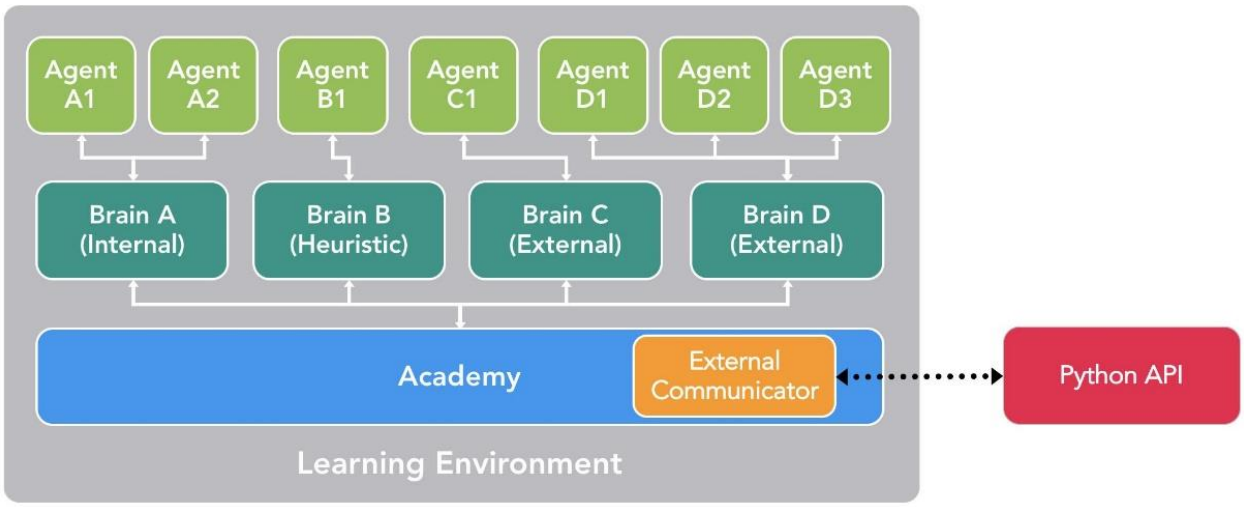

Figure 2. Overall Structure of the AI and Machine Learning Using PPO [2].

2.0 Unity AR Project with PPO training

To make the best of PPO reinforcement training topic of gaming chosen. The theme: “Pacman vs AngryBirds”.

Figure 3. The Unity Scene screenshot.

The overall idea is if the Agent “Angry Bird” touches PacMan 1 life is lost. And if PacMan shoots the agent than PacMan gets 1 score. The game continues until Pacman loses all lives.

2.1 PPO training

First trained the model using python script using Anaconda terminal [5].

```
mlagents-learn config/trainer_config.yaml --env=build/AI_Project --num-runs=10 –train
```

Where the script run Unity builds multiple times to train the agent based on the following reinforcement characteristics:

1. If Agent reaches the target it gets +1 reward point.
2. If Agent is closer to the target +0.1 point.
3. If Agent makes no progress (wasting time) -0.05 point.
4. If Agent is outside of the playground -1 point.

The logic can be found in the RollerAgent script. The script is written by following the “Making New Learning environment tutorial”[6]. However, it has been modified for the AR physics and the subgoals system added such as point 2 of reinforcement characteristics described above.

2.2 Vuforia API

For the Image Recognition, Vuforia API was chosen. Since it not only combines the ARcore and ARKit functionalities but also allows to target a bigger list of Android, IOS devices [7].

The image target was trained using Vuforia developer portal.

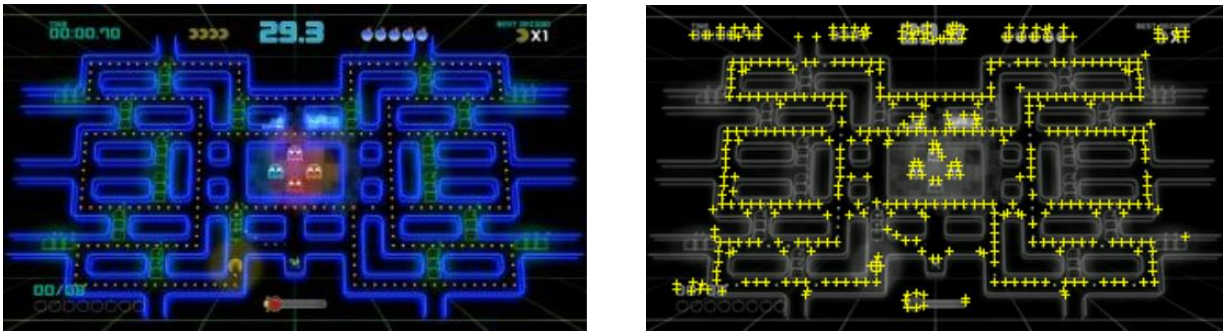

Figure 4. The trained image target using Vuforia API.

After the image trained data set is imported to the Unity project. The Vuforia API behaviour script upon recognizing the image target feature points activates the children bellow in our case

the 3dGame. The most important that it all happens in real time. Another great feature which Vuforia API has is extended tracking whereupon losing some of the tracking points it estimates the position of the 3d object attached based on the previous points [8]. This feature is very useful when the user needs to look away from the image target to concentrate on the gameplay.

The following game objects created for AR part of the development:

1. AR Camera
2. Vuforia API Image Target

2.3 Other Gameplay features

2.3.1 Canvas Menu

To get the user input the canvas UI object was created. Where the user can use push buttons to shoot into the enemies and use Joystick to control the PacMan. As well as adding another angry bird into the scene to make the game more challenging.

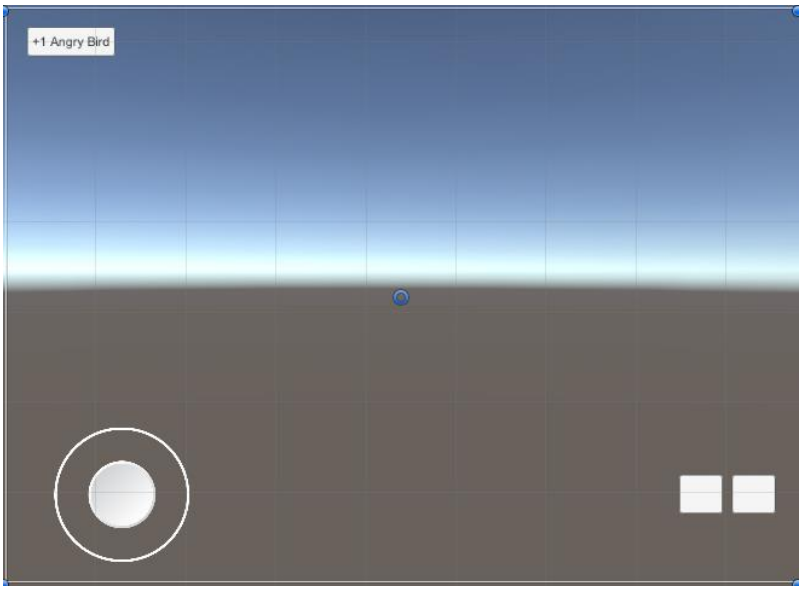

Figure 5. 2D Canvas menu in the Unity Scene to get the user input.

2.3.2 Scoring System

A used TextMeshPro asset for displaying the score of the player as well as # lives left [9]. In case of -1 life the text changes to “Game Over” And score doesn’t change if the user tends to continue the game.

```
UpdateScore.cs script manages the score system.
```

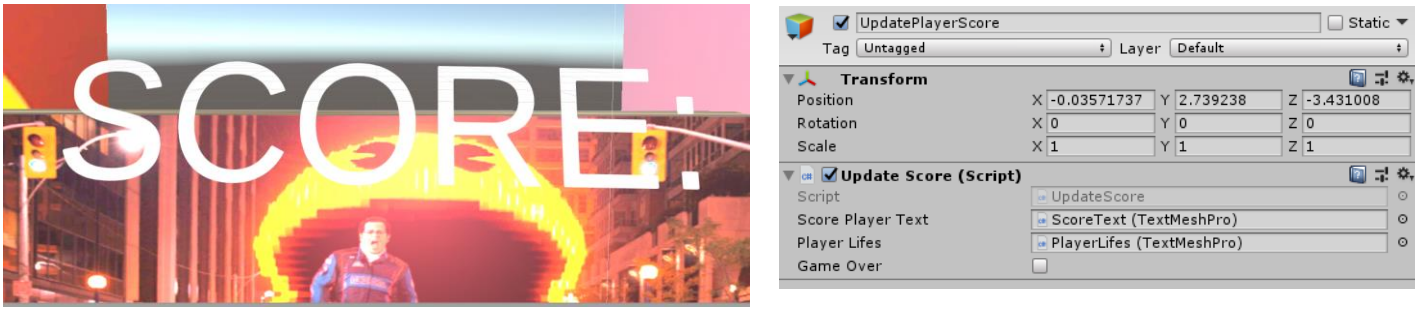

Figure 6. Update Player’s Score System for giving the feedback to the user.

2.3.3 Player and Angry Birds Agents

The Player is controlled by Joystick which is build by using Virtual Joystick Prefab [10]. And controlled by JoystickPlayer.cs in the custom scripts folder. The player can also use two guns available by pressing two square buttons which activate left or right gun accordingly.

There are four Angry Birds which are ml-agents. They are using the same Brain object with the internal trained model. The agent can be deployed by clicking the button “+1 Angry Bird”.

Even though the agents are using the same brain, they are sending different observations, therefore come to a different conclusion, and following different paths. They are appearing from different x-direction when they are being destroyed by the agent or fall from the plane.

2.3.4 Shooting system.

There are two scripts GunScripts.cs and Enemy.cs which are allowing the user to activate guns. The Agent has five lives. By using ray cast the GunScript can find out if the agent was hit or not. In case of hit the health is updated for the agent, and if it reached zero the agent is disappearing and score increment. The code is developed by following shooting with ray cast tutorial [11]. However, there are customization and modification made. Such as shooting the bullets in a random number from 1 to 10. Activating with the mobile touch of buttons and

connecting it with the custom scoring system and modifying the size of assets for AR use. There is also particle effect exist which is activated when the gun shoots and at the place where ray cast hits [12].

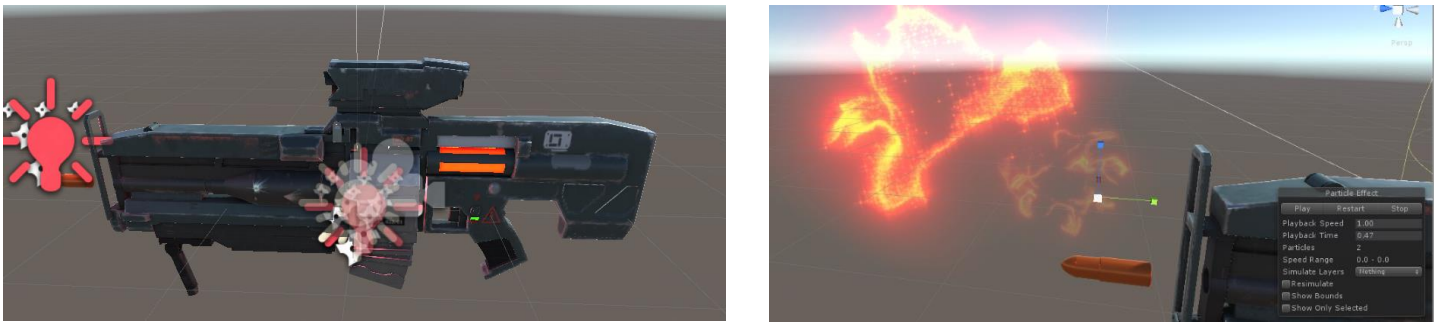

Figure 7. The Gun system with the particle effects.

2.4 Challenges

There are challenges existed in the development process. One of the biggest is to update the game for AR environment. Where not only the size of all game objects and particle effects needed to be scale down but also inside the code. For example, the distance for giving the reward to agent needed to be adjusted. By trial and error that was accomplished and tighten with image target.

The second problem existed with Physics of the game since the Agents used built-in gravitational force there became of problem where they felt outside of the field and never came back to the original position. It's due to the different dimension, in the Unity physics tighten to x,y,z but in the Augmented Reality they are tightened according to x,y,z established by Image Target. Therefore, the solution was first to change the setting of the AR Camera so that world center is set to be around the target. The second adjustment to avoid the agents being deployed to the wrong location, the button created "+1 Angry Bird" so upon the image recognition and 3d model of the scene displayed the user can activate the Angry Bird. And finally, walls were added to the scene to avoid the agents fell off from the floor.

3. Conclusion

In conclusion, the project is successful. The video demo can be found here [13]. Also, the APK file along with the image target is attached to the project. The proximal policy optimization is implemented successfully in the Augmented Reality environment. Where the user is competing

with the Artificial Intelligence human-like. It has a brain, it’s competitive and at the same time it can make mistakes. The Augmented Reality is adding a new way for the AI to interact with human and therefore has great potential!

References:

[1]. “On-device motion tracking for immersive VR: Freedom from wires,” Qualcomm, 11-Aug-2016. [Online]. Available: https://www.qualcomm.com/news/onq/2016/08/11/device-motion-tracking-immersive-vr-freedom-wires. [Accessed: 27-Nov-2018].

[2]. Juliani, Arthur, Berges, Vincent-Pierre, Esh, Gao, Yuan, Henry, Hunter, Mattar, Marwan, Lange, and Danny, “Unity: A General Platform for Intelligent Agents,” [astro-ph/0005112] A Determination of the Hubble Constant from Cepheid Distances and a Model of the Local Peculiar Velocity Field, 07-Sep-2018. [Online]. Available: https://arxiv.org/abs/1809.02627. [Accessed: 27-Nov-2018].

[3]. Juliani, A., Berges, V., Vckay, E., Gao, Y., Henry, H., Mattar, M., Lange, D. (2018). Unity: A General Platform for Intelligent Agents. *arXiv preprint arXiv:1809.02627*. https://github.com/Unity-Technologies/ml-agents.

[4]. Unity-Technologies, “Unity-Technologies/ml-agents,” GitHub. [Online]. Available: https://github.com/Unity-Technologies/ml-agents/blob/master/docs/Training-PPO.md. [Accessed: 27-Nov-2018].

[5]. Unity-Technologies, “Unity-Technologies/ml-agents,” GitHub. [Online]. Available: https://github.com/Unity-Technologies/ml-agents/blob/master/docs/Training-ML-Agents.md. [Accessed: 27-Nov-2018].

[6]. Unity-Technologies, “Unity-Technologies/ml-agents,” GitHub. [Online]. Available: https://github.com/Unity-Technologies/ml-agents/blob/master/docs/Learning-Environment-Create-New.md. [Accessed: 27-Nov-2018].

[7]. Getting Started with Vuforia Engine in Unity. [Online]. Available: https://library.vuforia.com/articles/Solution/vuforia-fusion-supported-devices.html. [Accessed: 27-Nov-2018].

[8]. Getting Started with Vuforia Engine in Unity. [Online]. Available: https://library.vuforia.com/content/vuforia-library/en/articles/Training/Extended-Tracking.html. [Accessed: 27-Nov-2018].

[9]. Getting Started with Vuforia Engine in Unity. [Online]. Available: https://library.vuforia.com/articles/Solution/vuforia-fusion-supported-devices.html. [Accessed: 27-Nov-2018].

[10]. “Joystick Pack,” Unity Asset Store - The Best Assets for Game Making. [Online]. Available: https://assetstore.unity.com/packages/tools/input-management/joystick-pack-107631. [Accessed: 27-Nov-2018].

[11] Brackeys, “Shooting with Raycasts - Unity Tutorial,” YouTube, 19-Apr-2017. [Online]. Available: https://www.youtube.com/watch?v=THnivyG0Mvo. [Accessed: 27-Nov-2018].

[12] “Sci-Fi Weapons,” Sci Fi Weapons | Dev Assets. [Online]. Available: http://devassets.com/assets/sci-fi-weapons/. [Accessed: 27-Nov-2018].

[13] V. Saminin, "Pacman vs Angry Birds AR PPO", YouTube, 2018. [Online]. Available: https://youtu.be/gkq5ZYaK1RQ. [Accessed: 06- Dec- 2018].