Research on robot arm control based on Unity3D machine learning

Mu Lin¹,³, Lijun Shan¹ and Yongchun Zhang¹,²
¹Dalian Jiaotong University, Dalian, Liaoning, China
²Dalian Ocean University, Dalian, Liaoning, China
³E-mail: linmu_7@163.com

Abstract. Based on the Unity3D engine, the article uses deep reinforcement learning strategies to train the robotic arm through the reward function, and realizes machine learning and intelligent control of the robotic arm. After training and learning, the robotic arm can quickly and accurately find movement point in the environment and has high environmental adaptability. The application of powerful deep reinforcement learning strategies and virtual reality technology in engineering technology teaching has improved the teaching effect and teaching efficiency of mechanical structure cognition, curriculum design and other teaching links.

1. Introduction
In 2015, China proposed and implemented the strategy of manufacturing a strong country. Robotics cannot be achieved without automation in manufacturing. Traditional robot arm control is mainly based on single-chip microcomputers, sensors or embedded. The traditional robot arm is modified, and sensors are added to the traditional robot arm to make the robot arm have an external sensing function, similar to some external function of humans. Its flexibility is effectively improved, but the information obtained by the sensor is often very different from the environment [1].

Deep reinforcement learning combines the perceptual ability of deep learning with the decision-making ability of reinforcement learning, which can be controlled directly based on the input image. It is an artificial intelligence method closer to the way of human thinking [2].

In recent years, due to the emergence of some easy-to-use and scalable simulation platforms, deep reinforcement learning research and algorithm design have made significant progress. These simulation platforms can not only promote the improvement of algorithms, but also train models and deploy them in the real world through the virtual simulation platform. For example, we can deploy robots trained in the virtual simulation platform to work in the real world. The simulation platform provides a safe, controlled, and efficient training environment.

2. Background knowledge

2.1. Deep reinforcement learning
The principle of reinforcement learning is: if certain actions of the agent generate positive rewards, the probability of the agent performing these actions will increase in the future, otherwise, the probability of the agent performing these actions during the learning process will weaken. The agent obtains the current state s from the environment at each time point t, and then selects and executes an action at
from the action set \( A \), and it will get a reward \( r_t \) given by the environment, and after executing the action \( a_t \), it will cause the state to transition to \( s_{t+1} \).

Deep learning is a method in machine learning, specifically, a method for performing feature learning on input data. In deep learning, the main content is to learn the features of the input data and obtain the feature information through a hierarchical multilayer network, so that the machine "understands" the learning data and obtains the feature information.

Deep learning turns the extraction of high-level features from the original input data into reality. Although it performs very well in perception, it is not satisfactory in decision-making. In contrast, reinforcement learning excels in decision-making but does not stand out in terms of perception. Therefore, combining deep learning and reinforcement learning to form a deep reinforcement learning algorithm, which has complementary advantages, can provide an effective method for solving the perceptual decision-making problems of complex systems.

### 2.2. Virtual reality simulation platform

Most of the current virtual simulation platforms are based on some popular game engines. For example, the research work of early artificial intelligence (AI) in chess in 1950 and checkers in 1959, or the application of reinforcement learning to backgammon games in the 1990s, and related AI research in recent years, game engines have gradually become an important AI research platform. These simulation platforms can not only achieve algorithmic improvements, but also serve as a starting point for training models, which can then be deployed in the real world. A good example is training autonomous robots in a simulator and transferring them to real-world robots (Open AI 2018; Sadeghi, 2016).

But not all game engines are suitable as research platforms. In order to achieve more complex learning tasks for the model and to explore more advanced algorithms and methods, the complexity of the simulation environment itself should increase. In order to achieve integrated research on deep reinforcement learning, for a sustainable AI system, a virtual simulation platform is essential in the following key aspects: sensory complexity, physical complexity, and cognitive complexity [3].

1. Sensory complexity: Recent advances in deep learning have largely benefited from the ability of neural networks to process a large number of data processing capabilities based on vision, hearing, and text. For example, self-driving cars, home robots, and self-driving cars have benefited from the development of image recognition algorithms, especially convolutional neural networks.

2. Physical complexity: In the process of handling a large number of research tasks, researchers need rich sensory information and control schemes, so that the model can interact with the dynamic physical environment in a complex way, and finally realize what will be learned on the virtual simulation platform. The strategy is deployed in the real world.

3. Cognitive complexity: The third type of complexity is called cognitive or spatiotemporal complexity. For example, some AI research does not include complex visual effects or complex physical interactions. The complexity comes from the uncertainty of time and the openness of space during the execution of the model. An effective simulation environment should enable designers to create such a simulation environment for its learning model. These complex tasks can be decomposed into cognitive structures of different depths. Tasks can be presented in a sequential or generalized manner. Corresponding to the real world, this is often the case for task processing, and, over time, is it capable of Learning new tasks in a new space and new environment is considered the key to continuous learning, which can eventually make it a training system with artificial intelligence.

This article builds a robotic arm virtual laboratory based on deep reinforcement learning and virtual reality technology. The system uses the Unity3D engine and uses the robotic arm as a research object to develop a virtual simulation teaching system. AI training is performed on the robotic arm to realize machine learning of the robotic arm and intelligent control. The system was released on the PC side and used in the teaching and training of robotic arm control. Establish a virtual laboratory to realize first-person perspective roaming, on-site observation of the robotic arm structure, and intelligent control simulation to solve the cognitive teaching and intelligent control problems of complex
mechanical structures. The application of powerful deep reinforcement learning and virtual reality technology to engineering technology teaching has improved the teaching effect and teaching efficiency of mechanical structure recognition, motion control, curriculum design and other teaching links.

2.3. Unity ML-Agents tool
Unity is a game development platform consisting of a game engine and a graphical user interface called the Unity Editor. Unity was originally created in 2005 and has continued to grow. It is now used by developers for various interactive simulations, from mobile and browser-based games to high-budget console games or AR / VR experiences. Unity, as the world’s most popular creative engine, is at the intersection of deep reinforcement learning and games. In the past, agents needed to achieve the behaviors we wanted to achieve by artificial coding, but now they can continuously learn in the training environment. Ways to teach agents to improve development efficiency. Deep reinforcement learning is a disruptive technology, and deep reinforcement learning in Unity has opened a new chapter in the history of artificial intelligence [4].

Unity ML-Agents is an open source Unity plugin that allows games and simulations to be used as an environment for training intelligent agents. Through the easy-to-use Python API, Agents can be trained using reinforcement learning, imitation learning, neuron evolution, or other machine learning methods.

3. System construction

3.1. System construction plan
The process of developing a deep reinforcement learning virtual laboratory based on the Unity3D engine is essentially a process of developing virtual reality content in combination with deep reinforcement learning strategies. First, 3D modeling software is used to build a simulation model of the robotic arm and the laboratory environment. The 3D model is imported into the Unity3D engine. Through program scripts, interactive functions and machine training are added so that students can observe the robotic arm from a first-person perspective in a virtual laboratory to achieve immersive interaction and increase students' awareness of the robotic arm. The system frame structure is shown in the Figure 1.

![System frame structure](image)

**Figure 1.** System frame structure.

3.2. Robot arm deep reinforcement learning training environment
The deep reinforcement learning training of the robotic arm is implemented based on Unity ML-Agents. In the Unity deep reinforcement learning tools, the performer of the action is called an agent,
which is embedded in the environment; the strategy is the ultimate goal of the action execution. Brain is responsible for providing decision-making strategies for its related agents to guide the execution of the action; there will be a state before and after the action is executed, and the change of the two states will obtain a reward value according to the criteria of the strategy; the implementation of Academy and external training tools Information interaction and overall training. The relationship between "Agent", "Brain" and "Academy" is shown in the Figure 2:

![Figure 2. "Agent", "Brain" and "Academy" relationship.](image)

The official recommended environment for Unity ML-Agents is: Python2/3 64-bit; jupyter notebook; TensorFlow (1.0+) (Training); Visual Studio 2017; Unity3d 2017+. This article is based on a recent version of Unity ML-Agents, and uses ml-agents-0.8.0 and Anaconda3 to implement a robotic arm deep reinforcement learning environment as shown in Figures 3 and 4.

1. Create the ML-Agents-2020 environment in the Environments of Anaconda Navigator by using the Create command.

![Figure 3. Create the ML-Agents-2020 environment.](image)

2. Enter Open Terminal, and install "ml-agents-envs" and "ml-agents" in the download directory of ml-agents-0.8.0 through the "pip install" command.

```
C:\Windows\system32\cmd.exe

(ml-agents-tutorial) C:\Users\Seto>cd /d C:\Users\Seto\Downloads\Compressed\ml-agents-0.8.0
(ml-agents-tutorial) C:\Users\Seto\Downloads\Compressed\ml-agents-0.8.0>pip install -e ml-agents-envs
```

![Figure 4. Install "ml-agents-envs" and "ml-agents".](image)
3.3. Robotic arm modeling and virtual lab construction

In the process of creating a 3D model of the robotic arm (as shown in Figure 4), the information contained in the model (name, size, unit, coordinate, axis, material, etc.) must conform to the production specifications, which helps model programming and software between import and export [5]. The virtual laboratory modeling process is shown in the Figure 5.

![Virtual laboratory modeling process](image)

**Figure 5.** Virtual laboratory modeling process.

After modeling the robotic arm, you need to assign the texture map to the corresponding model, and finally package the robotic arm model and export it in FBX format. It should be noted that during the export of the model, the Y-axis of the model must be set upward so as to be consistent with the coordinate system in the Unity3D engine. At the same time, the "embedded media" option must be selected so that the texture file on the model is imported the Unity3D engine will only be saved in a separate folder to avoid loss of textures. Create a new project using the Unity3D engine and import the model exported from 3ds Max into the project. At the same time, the lab environment was built using similar modeling techniques and resource packs, such as the Skybox that comes with the Unity3D engine.

3.4. Deep reinforcement learning strategy training robot arm

First, Unity3D opens the Unity SDK project in ml-agents-0.8.0. Create Academy, create a new empty object, add a C # script, change the inheritance class to Academy, and add the using ML Agents namespace. A similar process creates an Agent (the inherited class is Agent).

Create "Brain" for "Academy" from "Resources" and add it to Academy and agent.

Set the robot arm to the parent-child relationship according to the hierarchy, and realize the degree of freedom by the key point "point". Then set the joints of the robot arm and the target object in the Agent script, and start the training of the robot arm to grab the target object. As shown in Figure 6.

![Training logic diagram](image)

**Figure 6.** Training logic diagram.

Through TensorBoard to observe the training process, it can be found that as the training progresses, its cumulative reward and estimated value are constantly rising, Various vectors and speed losses are decreasing. As shown in Figure 7.
Figure 7. Training process diagram.

The above data shows that the training is effectively performed. After several hours of training, the trained binary file is finally loaded in Unity3D, and the Play button above the editor is clicked to realize the observation of the training result as shown in Figure 8: When the target moves randomly, the robotic arm can accurately grasp it.

Figure 8. Training result graph.

4. Conclusions
With the continuous development of deep reinforcement learning and virtual reality technology, content development combining deep reinforcement learning strategies and virtual reality technology has become more and more simple. The Unity3D deep reinforcement learning agent tool can help research developers in complex learning scenarios. Effective training agents have greatly improved the development efficiency. Compared with existing solutions, the virtual laboratory system has been improved in terms of convenience and interactivity, and can be applied to the teaching of courses related to mechanical specialty. Students can better understand the theoretical explanation of teachers through virtual laboratory.
Acknowledgement
The research in this article is sponsored by the research project of Dalian Science and Technology Bureau in 2018, "3D Interactive Experience Platform Based on Virtual Reality Technology" (2018J13FZ133).

References
[1] Li Wei 2010 Research on Control of Multi-DOF Robot Arm Based on Virtual Instrument[D]. Southeast University
[2] Wang Xuesong, Zhu Meiqiang and Cheng Yuhu 2014 Principles and Applications of Reinforcement Learning[M]. Beijing: Science Press
[3] Zhang Minmin, Xu Heping, Wang Xiaojie, et al. 2017 Google TensorFlow machine learning framework and application[J]. Microcomputer & Application 26(10) 58-60
[4] Wang Yongheng, Geng Shaofeng and Gao Hui 2017 A proactive decision sup-port method based on deep reinforcement learning and state partition[J]. Knowledge-Based Systems
[5] Ding Yu and Lin Mu 2018 Virtual training of reducer assembly based on Unity3D[J]. China Equipment Engineering 99-100