Autonomous System Control of

Multiple Robotic Arms Collaboration via

Machine Learning

by

Steve Lin

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved May 2023 by the
Graduate Supervisory Committee:

Hani Ben Amor, Chair
Sangram Redkar
Yu (Tony) Zhang

ARIZONA STATE UNIVERSITY

August 2023

ABSTRACT

Multiple robotic arms collaboration is to control multiple robotic arms to collaborate with each other to work on the same task. During the collaboration, the agent is required to avoid all possible collisions between each part of the robotic arms. Thus, incentivizing collaboration and preventing collisions are the two principles which are followed by the agent during the training process. Nowadays, more and more applications, both in industry and daily lives, require at least two arms, instead of requiring only a single arm. A dual-arm robot satisfies much more needs of different types of tasks, such as folding clothes at home, making a hamburger in a grill or picking and placing a product in a warehouse.

The applications done in this paper are all about object pushing. This thesis focuses on how to train the agent to learn pushing an object away as far as possible. Reinforcement Learning (RL), which is a type of Machine Learning (ML), is then utilized in this paper to train the agent to generate optimal actions. Deep Deterministic Policy Gradient (DDPG) and Hindsight Experience Replay (HER) are the two RL methods used in this thesis.

ACKNOWLEDGMENTS

Thanks for guidance from Prof. Hani Ben Amor about the whole thesis, MuJoCo and Reinforcement Learning.

Thanks for help from Michael Drolet in Interactive Robotics Lab about IRL Control and OpenAI Gym.

Thanks for help from Yifan Zhou in Interactive Robotics Lab about DDPG and OpenAI Gym.

TABLE OF CONTENTS

Page

CHAPTER

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

Chapter 1

INTRODUCTION

To this day, robot programming remains a challenge. Many tasks are still very difficult to implement in robots and fail when contact with the environment is made. Achieving high performance and accuracy still requires substantial time investment. One possible way to overcome these challenges is the use of modern Machine Learning (ML) algorithms and methodologies. In recent years, these approaches have shown remarkable achievements in a variety of tasks from autonomous driving to robotic table tennis.

This thesis focuses on applying Reinforcement Learning (RL), a type of Machine Learning (ML), to train an agent to work on dual-arm manipulation tasks. The two RL algorithms, Deep Deterministic Policy Gradient (DDPG) and Hindsight Experience Replay (HER), are used to train the agent to complete the grasping and manipulation tasks in an autonomous fashion. The reason why this thesis works on a dual-arm system (Liu *et al.*, 2021), instead of a single robotic arm, is because a dual-arm system can complete much more complicated tasks than those which a single arm can do. Also, there are many other tasks that can only be done by multiple robotic arms. Accordingly, both industry and academia is increasingly interested in using multiple arms. The specific objective of this paper is to introduce a complete framework for training such agents. Below, we summarize the components of our framework:

1.1 MuJoCo

MuJoCo (Tassa *et al.*, 2021) is a physics simulation engine which all important physics parameters can be easily tuned. For example, if now the dual-arm collaboration is going to be moved to outer space in order to assist an astronaut in daily lives or doing experiments, the parameters of the gravity - $(0, 0, -9.81)$ (the gravity in the three directions: x, y, z), can be changed to $(0, 0, 0)$ directly, which means that now the simulation environment becomes out of gravity. The experiments in this thesis are all simulated in MuJoCo.

1.2 OpenAI Gym

OpenAI Gym (Brockman *et al.*, 2016) provides a complete system with all the required settings to create a gym environment, such as HalfCheetah-v2, Humanoid-v2, Hopper-v2. This thesis requires a new gym environment with the Dual UR5 and a box. The new gym environment applies IRL Control to manipulate the Dual UR5.

1.3 IRL Control

IRL Control (Drolet *et al.*, 2022) is applied here to generate control signals for the Dual UR5 in the gym environment. IRL Control extends Operational Space Control (Nakanishi *et al.*, 2008), which is originally designed for a single robotic arm, to manipulate the Dual UR5. The code of OSC (Operational Space Control) is provided by ABR Control (DeWolf *et al.*, 2016)). Collision prevention and trajectory planning are the two main functions of IRL Control.

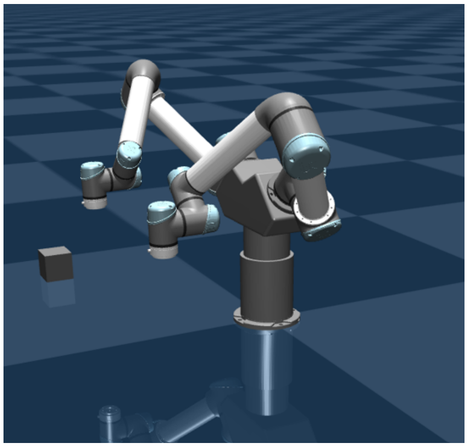

Figure 1. The Gym Environment in MuJoCo

1.4 UR5

UR5 is the robotic arm from Universal Robots to be used in this thesis. All experiments are based on the Dual UR5, which is composed of a base and two UR5s on each side of the base. No grippers is used in the experiments. Figure 1 shows the Dual UR5 and a box to be pushed away in MuJoCo.

1.5 Steps

There are three main steps in this thesis. The first step is to create the gym environment. The second step is to utilize IRL control to do the collision prevention and the trajectory planning for the Dual UR5. The third step is to connect the agent with the gym environment. Each step will be discussed in the following sections.

1.5.1 Gym Environment

The first step is to create a gym environment. The gym environment contains the Dual UR5. Since we do not use a gripper here, the end effector now becomes the wrist of the UR5. Also, there is a box in the environment for the Dual UR5 to push away.

1.5.2 IRL Control

The second step is to utilize IRL Control to manipulate the Dual UR5 in the gym environment. IRL Control is to generate a series of forces, and execute them on the Dual UR5 to make it move.

1.5.3 Connect the Agent with the Gym Environment

The third step is to create an interface to connect the agent with the gym environment, whose structure is shown in figure 2. The agent first generates an action and sends it to the gym environment. The gym environment then executes it, and return a new observation and the reward back to the agent. Thus, in order to connect the agent with the gym environment, it is required to design the observation space, the action space, and the reward function.

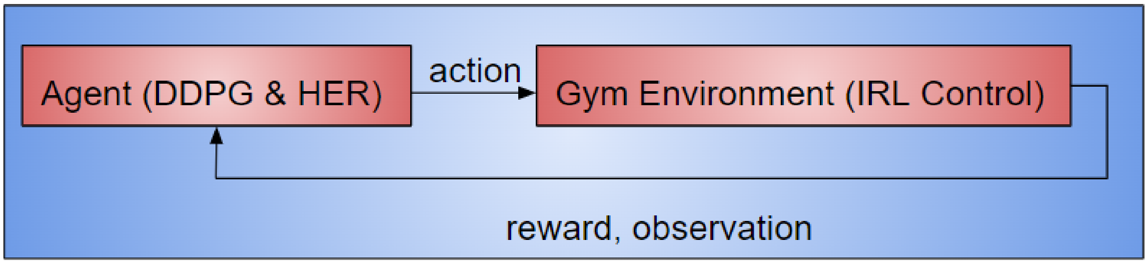

Figure 2. Flow Chart of Parameters Passing Between the Agent and the Environment

Chapter 2

METHOD

This paper utilizes DDPG and HER (Navale, 2021) to train the agent to learn how to push an object away (Amor *et al.*, 2019). IRL Control is applied here to convert the actions from DDPG and HER to control signals, and to send the control signals to the Dual UR5.

2.1 Deep Deterministic Policy Gradient (DDPG)

Deep Deterministic Policy Gradient (Lillicrap *et al.*, 2019) is mainly divided into two parts - "Deep" and "Deterministic Policy Gradient". The structure of DDPG is shown in figure 3 (Zhou, 2016). "Deep" represents Deep Q-Network (DQN), and "Deterministic Policy Gradient" represents the Actor-Critic method. Critic is based on Q-Learning and makes the agent to be able to update parameters after every step, instead of doing the update after every epoch. This results in a much faster learning process. As for the part of Actor, it is based on policy gradient and is able to generate actions in a continuous action space.

For more details about the part of Critic, it is trained by minimizing the loss function L:

$$L = \frac{1}{\mathbf{N}} \sum_{i=1}^{\mathbf{N}} [y_i - Q(s_i, a_i|\theta^Q)]^2 \tag{2.1}$$

N is the number of samples in a minibatch, $Q()$ is the Q-funcion of Critic that outputs a Q-Value, s_i is the state at step i, a_i is the action taken at step i, and θ^Q is the parameter set of Critic. The y_i is:

$$y_i = r_i + \gamma Q'[s_{i+1}, \mu'(s_{i+1}|\theta^\mu)|\theta^Q] \tag{2.2}$$

r_i is the reward obtained from step i, γ is the discount factor, Q' is the Q-function

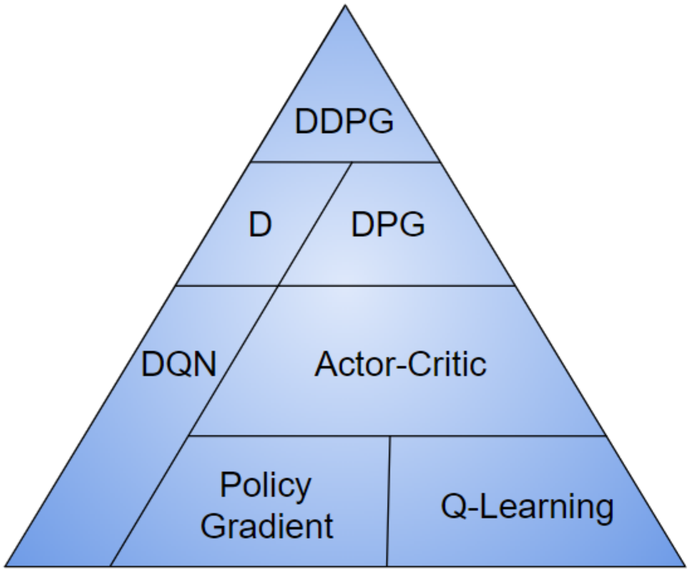

Figure 3. Structure of DDPG

of target Critic, μ' is target Actor, θ^{μ} is the parameter set of target Actor, and θ^{Q} is the parameter set of target Critic.

More details about Actor is shown at here. It is trained by doing the policy gradient:

$$\nabla_{\theta^{\mu}} J \approx \frac{1}{\mathbf{N}} \sum_{i=1}^{\mathbf{N}} \nabla_a Q(s_i, a_i|\theta^{Q}) \nabla_{\theta^{\mu}} \mu(s_i|\theta^{\mu}) \tag{2.3}$$

$\nabla_{\theta^{\mu}}$ is the gradient w.r.t. θ^{μ}, and ∇_a is the gradient w.r.t. a.

At last, we update all the parameters of both the target Actor network and the target Critic network via the two following equations:

$$\theta^{Q'} = \tau\theta^{Q} + (1-\tau)\theta^{Q'}(2.4)$$

$$\theta^{\mu'} = \tau\theta^{\mu} + (1-\tau)\theta^{\mu'}(2.5)$$

τ is the relative weighting between θ^{Q} and $\theta^{Q'}$, and between θ^{μ} and $\theta^{\mu'}$.

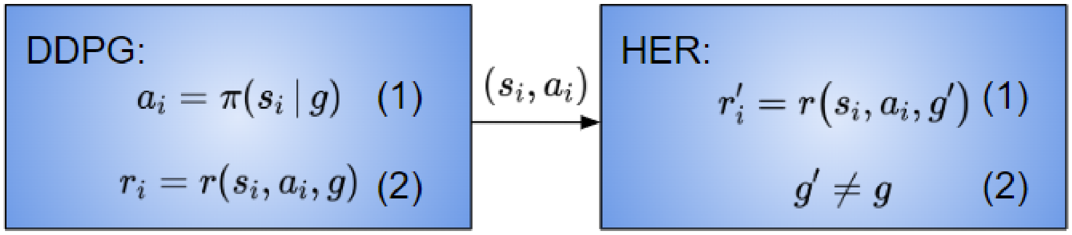
DDPG:
$a_i = \pi(s_i \mid g)$ (1)
$r_i = r(s_i, a_i, g)$ (2)
(s_i, a_i)
HER:
$r_i' = r(s_i, a_i, g')$ (1)
$g' \neq g$ (2)

Figure 4. Flow Chart About How HER Collaborates with DDPG

2.2 Hindsight Experience Replay (HER)

HER (Andrychowicz *et al.*, 2018) usually works with another Reinforcement Learning (RL) method, such as DDPG or Deep Q-Network (DQN). In this thesis, HER is applied to work with DDPG (figure 4). DDPG first generates an action a_i with a given goal g,

$$a_i = \pi(s_i|g) \tag{2.6}$$

and sends the pair of (s_i, a_i) to HER. HER then executes the (s_i, a_i) pair and calculate the reward again using the reward function $r()$, but with a different goal $g^{'}$:

$$r_i^{'} = r(s_i, a_i, g^{'})(2.7)$$

This is to learn more information or experience which has not yet been explored by the original goal g.

2.3 Design of the Observation Space, Action Space, Reward Function

It is required to first clarify the task and then define the observation space, action space, and the reward function. The reason is that all these three parts are affected by the content of the task. The task is to make the Dual UR5 to push the object away as far as possible.

2.3.1 Observation Space

The observation space contains five main elements: The vector in the form of $[x, y, z]$ from the left end effector to the object, the vector in the form of $[x, y, z]$ from the right end effector to the object, the distance between the left end effector and the object, the distance between the right end effector and the object, and the orientation of the object. Thus, the size of the observation space is 9.

2.3.2 Action Space

In the action space, each action contains one target position for both the left end effector and the right end effector to reach to. However, a target position in the action space only contains the x value and the y value. As for the z value of a target position, it will not be generated in an action. The z value of a target position is always 0.1 in each time of training and testing.

2.3.3 Reward Function

Each time of training and testing has two actions and their reward functions are different. Each action has its own specific meaning. The first action is to make the Dual UR5 to approach the object, and the second action is to make the Dual UR5 to push the object away.

2.3.3.1 Reward Type

Before defining the reward functions, it is required to decide the reward type first. There are two reward types - sparse reward and dense reward. Sparse reward is commonly used whenever a task is hard to define from the side of reward. Moreover, sparse reward only tells the agent whether the task is completed or not, usually 1 for successful and 0 or -1 for failed. As for dense reward, it is applied to guide the agent toward the goal by trying to tell the agent which action is "better" and which action is "worse" through different values of reward.

HER works well with sparse reward. However, this thesis applies HER to DDPG and uses dense reward. There are three main reasons about why this thesis utilizes dense reward. First of all, it is required to navigate the agent to start from the initial state to the goal during the whole process. Secondly, sparse reward is usually selected only when the task is hard to define in a reward function; otherwise, dense reward is considered first. The third reason is that the task in this thesis is to push the object away as far as possible, which doesn't work with sparse reward. This is because it is unable to define what a successful task is if the goal is "as far as possible" or "as low as possible". In other words, there will never be the best result, but only a better one will be.

2.3.3.2 The Reward Function for the First Action

Since the first action is to approach the object, the reward function contains three parts, which are shown in algorithm 1.

The first part of the reward is $reward_l$, which means that the closer between the

Algorithm 1: Reward function of the first action

```
Input  : $observation$, $object_{initial}$ (the initial object position), $object_{pos}$ (the
         real-time object position)
Output : reward of the first action
$distance_{box} = ||object_{initial} - object_{pos}||_2$
$vector_l, vector_r, distance_l, distance_r, orientation = observation$
if $distance_l <= 0.2$ then
    $reward_l = 2$
else
    if $distance_l <= 0.25$ then
        $reward_l = 1$
    else
        $reward_l = -distance_l$
    end if
end if
if $distance_r <= 0.2$ then
    $reward_r = 2$
else
    if $distance_r <= 0.25$ then
        $reward_r = 1$
    else
        $reward_r = -distance_r$
    end if
end if
if $distance_{box} >= 1.0$ then
    $ratio = 10$
else
    if $distance_{box} >= 0.5$ then
        $ratio = 5$
    else
        if $distance_{box} >= 0.1$ then
            $ratio = 2$
        else
            $ratio = 0.1$
        end if
    end if
end if
$\text{reward}_{box} = distance_{box} * ratio$
$\text{reward} = \text{reward}_l + reward_r + reward_{box}$
return $reward$
```

left end effector and the object, the higher the $reward_l$. The second part of the reward is $reward_r$, which implies that if the right end effector becomes closer to the object, then the $reward_r$ will be higher. All these two rewards, $reward_l$ and $reward_r$, are to make both end effectors to approach the object together by using the information from the observation. If the distance between an end effector and the object is bigger than 0.25, the corresponding reward will be negative, which is a type of penalty.

The third part of the reward is $reward_{box}$, and it depends on the object (box) moving distance. The longer the object moving distance, the higher the $reward_{box}$ is. Although the first action is expected to approach the object only, it may still touch and push the object away, so it is required to add the part of $reward_{box}$ to the reward of the first action. $reward_{box}$ is calculated by multiplying the object moving distance by a ratio. As for the ratio, it is obtained from the following principles: If the object (box) moving distance is bigger than or equal to 1.0, then the ratio of 10 is given; else if the object (box) moving distance is bigger than or equal to 0.5 and smaller then 1.0, then the ratio is 5; else if the object (box) moving distance is bigger than or equal to 0.1 and smaller than 0.5, then the ratio is 2; otherwise, the ratio will be set as 0.1, which is considered as a penalty for a short object moving distance. This design of ratio is to encourage the agent to make the Dual UR5 to push the object away as far as possible, since a longer object moving distance now can receive a much higher reward than usual.

2.3.3.3 The Reward Function for the Second Action

Since the second action is to push the object away, the reward is only composed of $reward_{box}$, which is shown in algorithm 2. The experiment results show that the target position in the second action may be very far away from the object. This is because the task for the second action is only to push the object away, and nothing else is considered. In other words, the second action does not require the left end effector or the right end effector to approach the object.

Algorithm 2: Reward of the second action

```
Input  : $object_{initial}$ (the initial object position), $object_{pos}$ (the real-time
         object position)
Output : reward of the second action
$distance_{box} = ||object_{initial} - object_{pos}||_2$
if $distance_{box} >= 1.0$ then
    $ratio = 10$
else
    if $distance_{box} >= 0.5$ then
        $ratio = 5$
    else
        if $distance_{box} >= 0.1$ then
            $ratio = 2$
        else
            $ratio = 0.1$
        end if
    end if
end if
$\text{reward}_{box} = distance_{box} * ratio$
$\text{reward} = \text{reward}_{box}$
return $reward$
```

Chapter 3

EXPERIMENTS

The experiment is to push the object away as far as possible and to be simulated in MuJoCo. The object is a box with the height, width, and length all equal to 0.1. There are two parts of the experiment, one simple and the other one complicated.

3.1 Attributes of the Object

Since generalization is the key element of the experiments in this thesis, the position and the orientation of the object is required to be random during every time of training and testing. This is to test if the agent is able to make the Dual UR5 to push objects which are at different positions and with different orientations away.

3.2 The First part of the Experiment

In the first part of the experiment, we randomize the position of the object on the three purple line segmentations, which are respectively on $y = x$ (the blue line), $y = -x$ (the red line), and the y axis. This is shown in figure 5. The reason why we do not randomize the position of the object on the whole three lines is because only the three line segmentations are the effective working space for the Dual UR5. If the object is out of these ranges, which may be too far away from or too close to the Dual UR5, then the Dual UR5 is unable to touch the object, not to mention to push it away. We do not consider the orientation of the object here. The first part of the experiment is just to simplify the situation and to ensure that the model can be trained well.

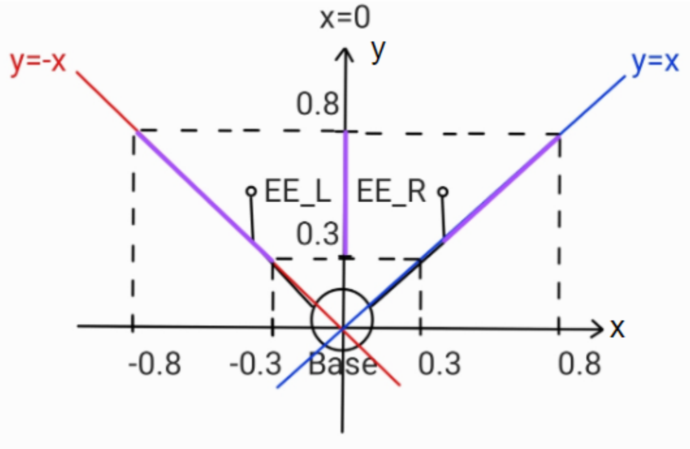

Figure 5. Way to Randomize the Position of the Object in the First Part of the Experiment

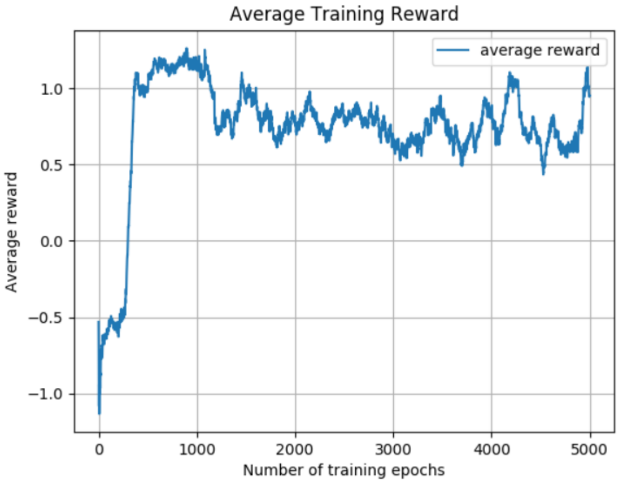

Figure 6. Average Training Reward for Training 5000 Times in the First Part of the Experiment

3.2.1 5000 Times of Training

The model is first trained for 5000 times. The line chart in figure 6 shows the average training rewards, which are always calculated by the last 100 rewards. Whenever the latest average reward becomes bigger than the current highest average reward, the model will then be updated.

As for the testing result, the bar chart in figure 7 shows the number of each range

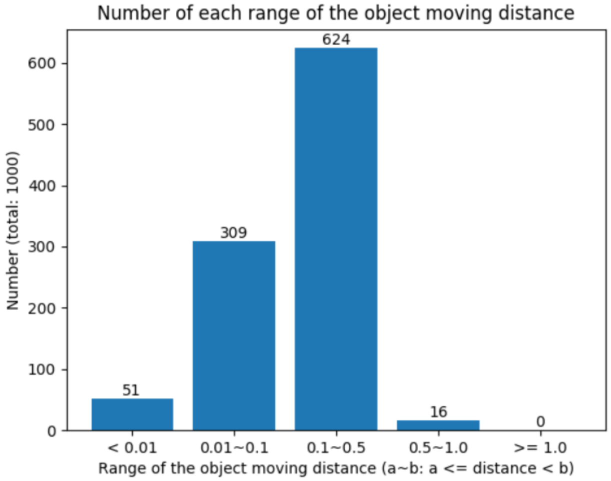

Figure 7. Bar Chart of the Testing Result for Training 5000 Times in the First Part of the Experiment

of the object moving distance for testing 1000 times. To evaluate the performance, all object moving distances are categorized into five ranges, which are respectively < 0.01, $0.01 \sim 0.1$, $0.1 \sim 0.5$, $0.5 \sim 1.0$, and $>= 1.0$. For example, the third bar with the number of 624 means that there are 624 out of 1000 times of tests that have an object moving distance in the range of $0.1 \sim 0.5$. By the way, although the range of < 0.01 is usually still bigger than 0, it is defined as not moving. The reason is that this tiny object moving distance is not caused by being pushed by the Dual UR5; instead, it is caused by while the object is loaded into the environment, the object just pops out and then drops onto the plane, which makes it to move a little bit forward or backward with a distance smaller than 0.01.

The percentage of each range of the object moving distance is shown in figure 8. This pie chart uses the same data as the previous bar chart, while the pie chart emphasizes the percentage of the numbers, and the bar chart emphasizes the real values of all the numbers.

Each following figure captures three frames from the demo of each testing, which are respectively the beginning of the testing, the end of the first action and the end of the second action. Thus, this contains three of the most critical steps in each testing. The first frame in figure 9 shows that the object in this test is at the right hand side of the Dual UR5 and close to it. Both end effectors approach the object in the second frame. The end effectors push the object away in the last frame.

The first frame in figure 10 shows that the object in this test is at the right hand

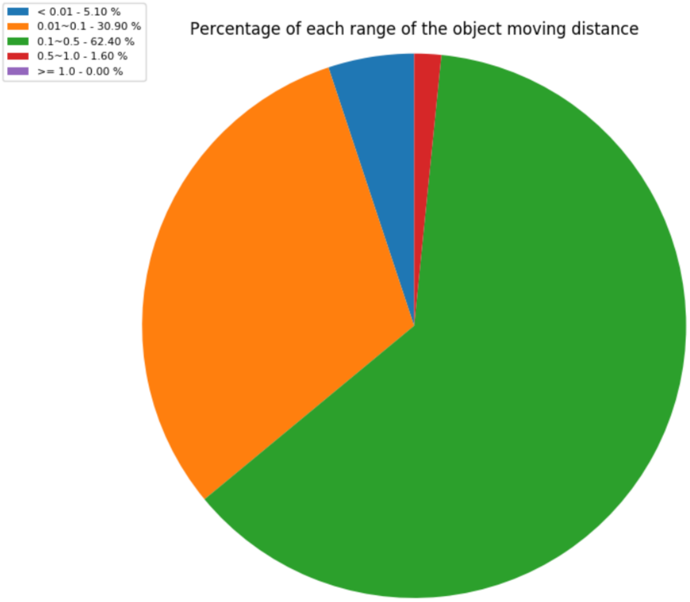

Figure 8. Pie Chart of the Testing Result for Training 5000 times in the First Part of the Experiment

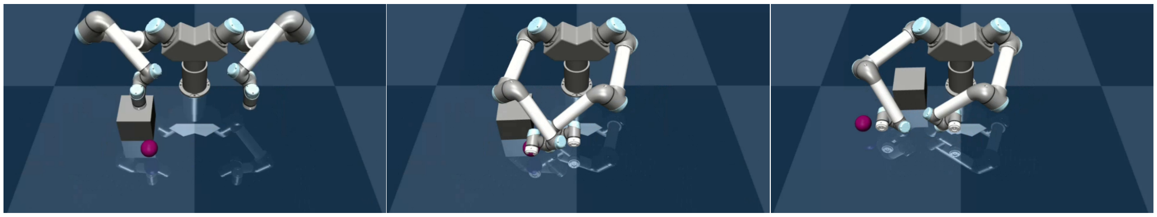

Figure 9. The Box is at the Right side, Close to the Dual UR5 (Test of Training 5000 Times in the First Part of the Experiment)

side of the Dual UR5 and a bit far away from it. Both end effectors approach the object and the right end effector scrolls the object away in the second frame. The right end effector pushes the object away in the last frame.

The first frame in figure 11 shows that the object in this test is at the right hand side of the Dual UR5 and very far away from it. Both end effectors try to approach the object but only the right end effector touches it in the second frame. The last frame shows that the right end effector pushes the object away.

The first frame in figure 12 shows that the object in this test is at the left hand side of the Dual UR5 and close to it. Both end effectors approach the object and scroll it a little bit in the second frame. The last frame shows that the object is scrolled away by both end effectors.

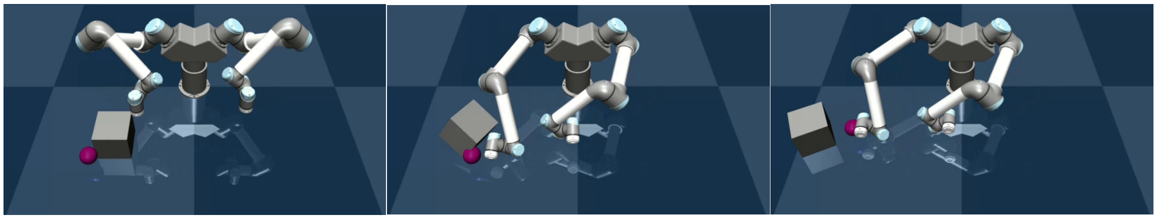

Figure 10. The Box is at the Right Side, a Bit Far Away From the Dual UR5 (Test of Training 5000 Times in the First Part of the Experiment)

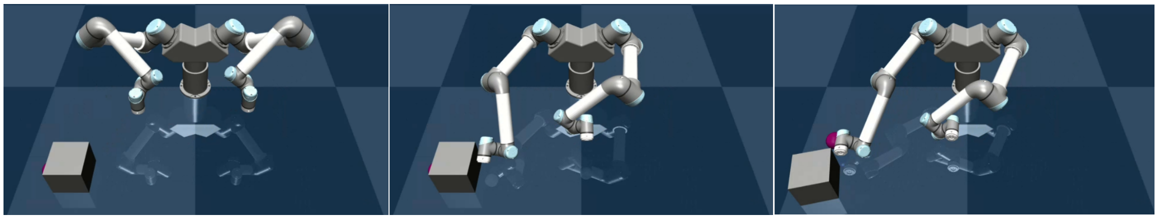

Figure 11. The Box is at the Right Side, Far Away From the Dual UR5 (Test of Training 5000 Times in the First Part of the Experiment)

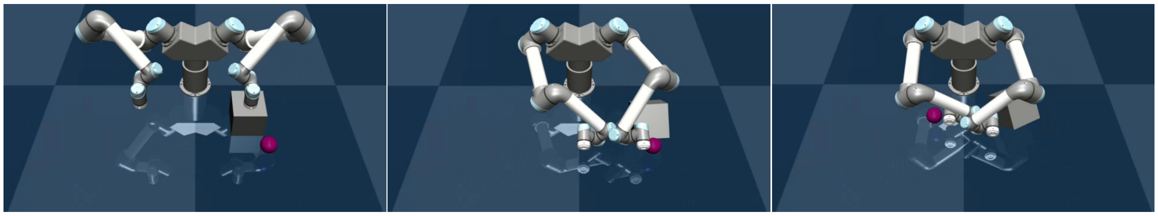

Figure 12. The Box is at the Left Side, Close to the Dual UR5 (Test of Training 5000 Times in the First Part of the Experiment)

The first frame in figure 13 shows that the object in this test is at the left hand side of the Dual UR5 and far away from it. Both end effectors try to approach the object but only the left end effector touches it in the second frame. The last frame shows that the object is pushed away by the left end effector.

The first frame in figure 14 shows that the object in this test is right in front of the Dual UR5 and close to it. Both end effectors touch the object and push it away a little bit in the second frame. The last frame shows that both end effectors keep pushing the object away.

The first frame in figure 15 shows that the object in this test is right in front of the Dual UR5 and far away from it. Both end effectors reach to the object in the second frame. The last frame shows that both end effectors push the object away.

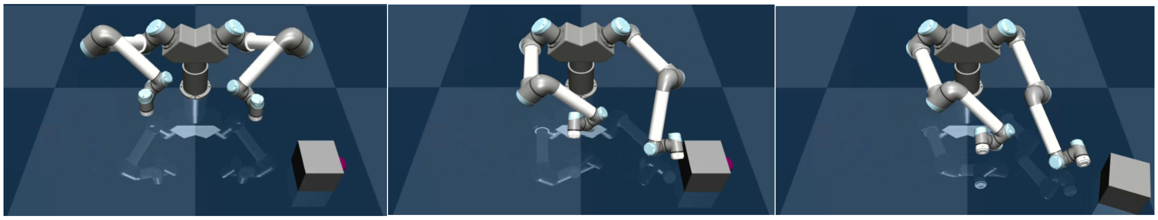

Figure 13. The Box is at the Left Side, Far Away From the Dual UR5 (Test of Training 5000 Times in the First Part of the Experiment)

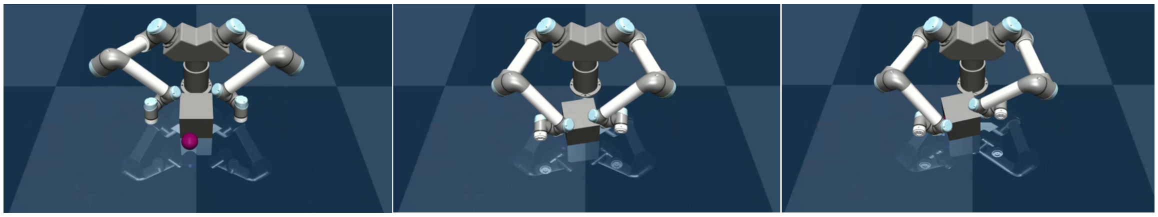

Figure 14. The Box is at the Middle, Close to the Dual UR5 (Test of Training 5000 Times in the First Part of the Experiment)

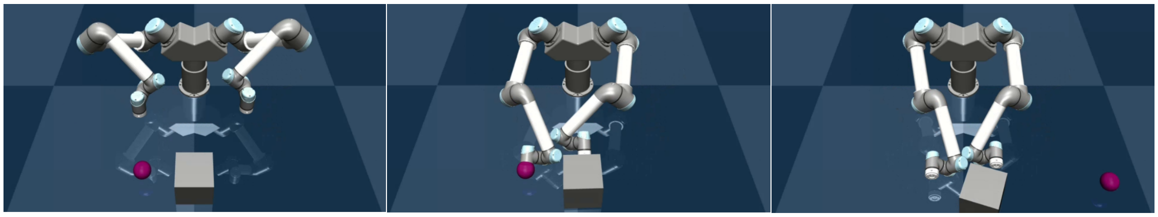

Figure 15. The Box is at the Middle, Far Away From the Dual UR5 (Test of Training 5000 Times in the First Part of the Experiment)

3.2.2 10000 Times of Training

The model is then trained for 10000 times and figure 16 is the relative average training reward line chart.

The testing result is shown in figure 17 and figure 18.

The first frame in figure 19 shows that the object in this test is at the right hand side of the Dual UR5 and a bit far away from it. Both end effectors try to approach the object but only the right end effector touches it and pushed it away in the second frame. The last frame shows that the object is scrolled far away by the right end effector.

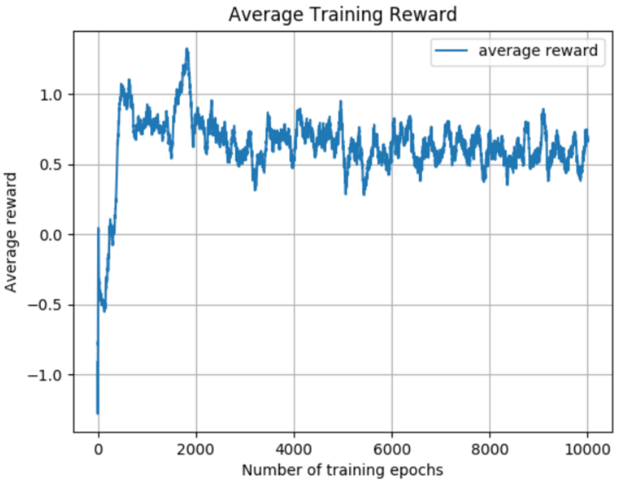

Figure 16. Average Training Reward for Training 10000 Times in the First Part of the Experiment

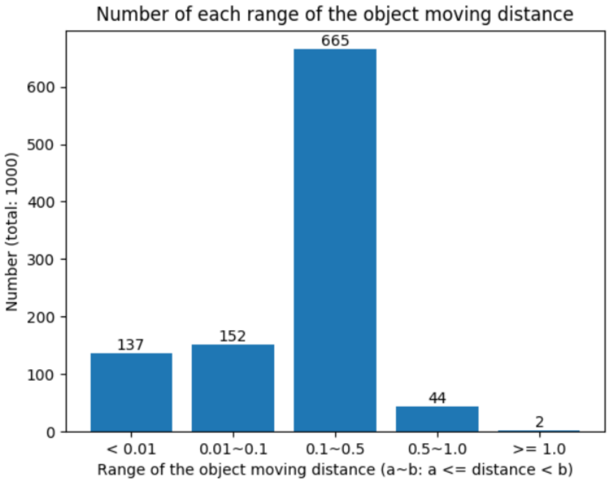

Figure 17. Bar Chart of the Testing Result for Training 10000 Times in the First Part of the Experiment

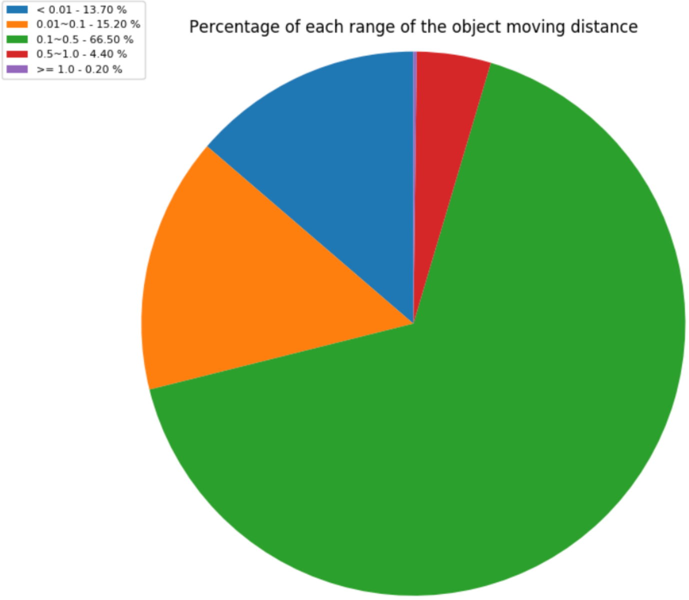

Figure 18. Pie Chart of the Testing Result for Training 10000 Times in the First Part of the Experiment

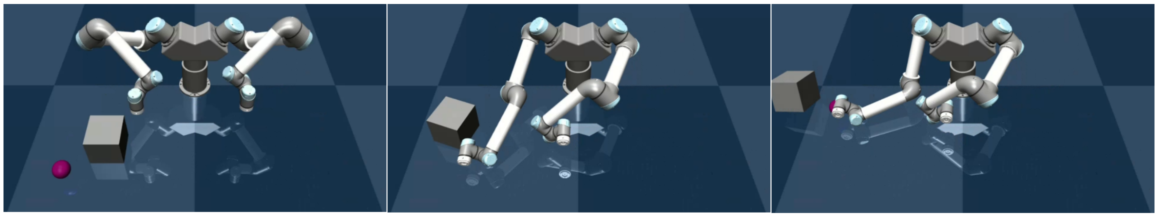

Figure 19. The Box is at the Right Side, a bit far Away From the Dual UR5 (Test of Training 10000 Times in the First Part of the Experiment)

The first frame in figure 20 shows that the object in this test is at the right hand side of the Dual UR5 and very far away from it. Only the right end effector touches it in the second frame. The last frame shows that the right end effector pushes the object away.

The first frame in figure 21 shows that the object in this test is at the left hand side of the Dual UR5 and close to it. Both end effectors approach the object and push it away in the second frame. The last frame shows that the object is pushed away backward.

The first frame in figure 22 shows that the object in this test is at the left hand side of the Dual UR5 and far away from it. Both end effectors approach the object

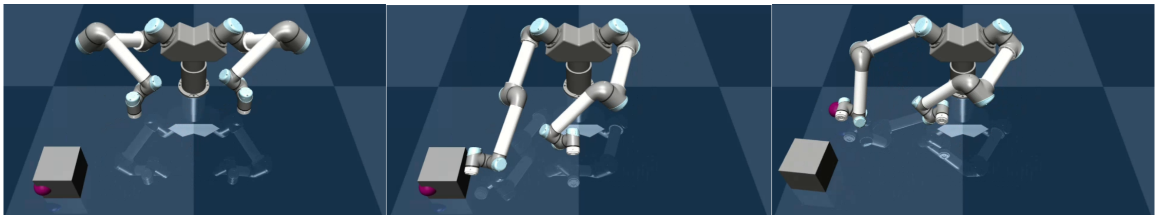

Figure 20. The Box is at the Right Side, Far Away From the Dual UR5 (Test of Training 10000 Times in the First Part of the Experiment)

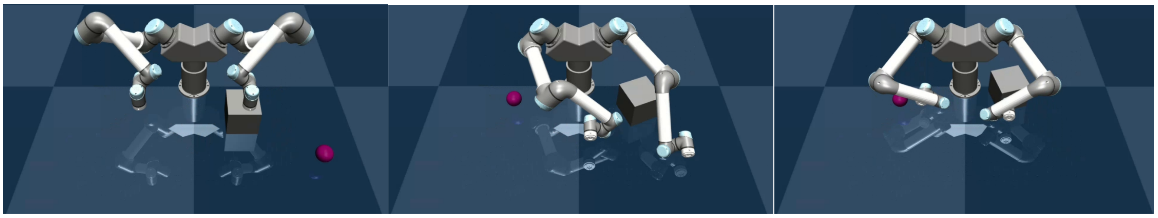

Figure 21. The Box is at the Left Side, Close to the Dual UR5 (Test of Training 10000 Times in the First Part of the Experiment)

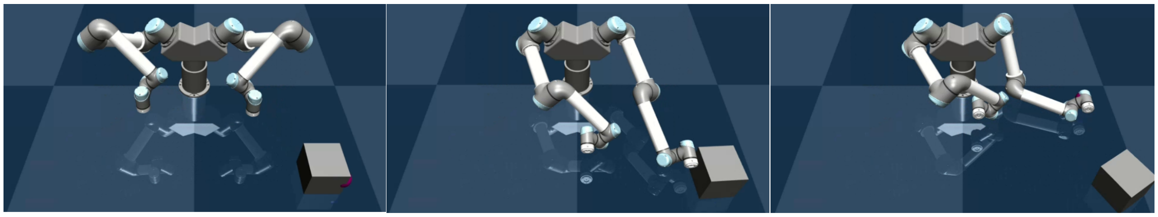

Figure 22. The Box is at the Left Side, Far Away From the Dual UR5 (Test of Training 10000 Times in the First Part of the Experiment)

and the left end effector pushes it away a little bit in the second frame. The last frame shows that the object is pushed farther away.

The first frame in figure 23 shows that the object in this test is right in front of the Dual UR5 and close to it. Both end effectors touch the object and push it away backward in the second frame. The last frame shows that the object is scrolled away backward with a long distance by both end effectors.

The first frame in figure 24 shows that the object in this test is right in front of the Dual UR5 and a bit far away from it. Both end effectors touch the object and push it away a little bit in the second frame. The last frame shows that both end effectors keep pushing the object away.

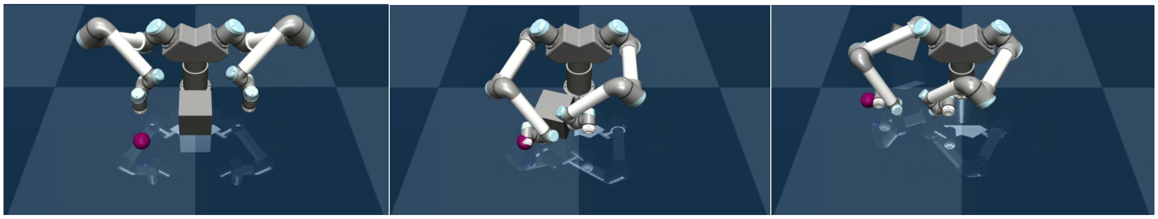

Figure 23. The Box is at the Middle, Close to the Dual UR5 (Test of Training 10000 Times in the First Part of the Experiment)

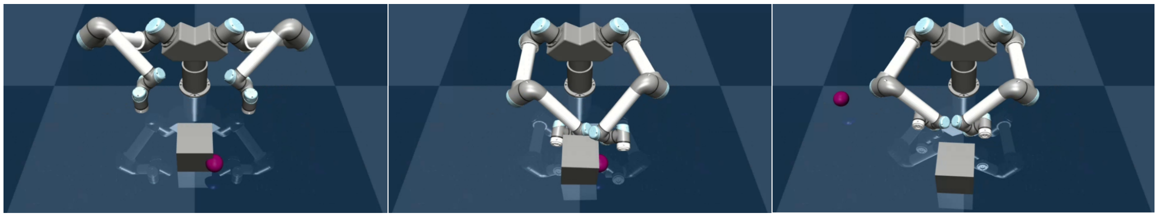

Figure 24. The Box is at the Middle, a Bit Far Away From the Dual UR5 (Test of Training 10000 Times in the First Part of the Experiment)

3.2.3 Comparison of Training 5000 and 10000 Times

The table 1 is to compare the testing result of training 5000 times and 10000 times. The numbers of the three ranges, 0.1 ∼ 0.5, 0.5 ∼ 1.0, and >= 1.0, all increase from the testing result of training 5000 times to the testing result of training 10000 times. Since the object moving distances in these three ranges are all bigger ones, it means that the performance of training 5000 time becomes better than that of training 10000 times.

Table 1. Comparison of Training 5000 and 10000 Times in the First Part of the Experiment

Range of object moving distance	< 0.01	0.01 ∼ 0.1	0.1 ∼ 0.5	0.5 ∼ 1.0	>= 1.0
Number of test for training 5000 times	51	309	624	16	0
Number of test for training 10000 times	137	152	665	44	2

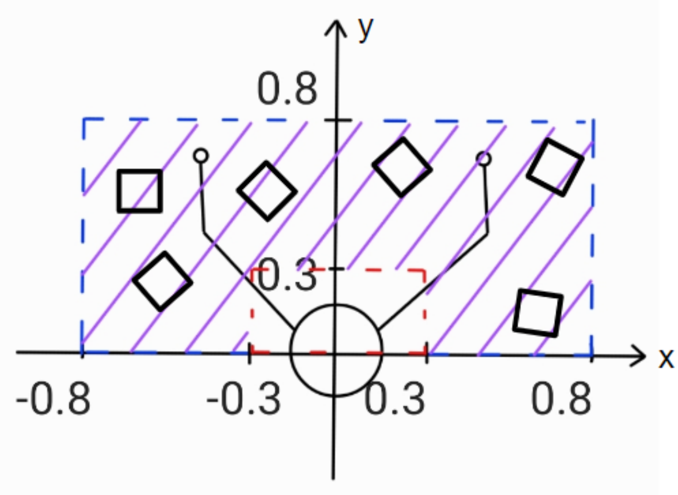

Figure 25. The Way to Randomize the Position and the Orientation of the Object in the Second Part of the Experiment

3.3 The Second Part of the Experiment

In the second part of the experiment, the task becomes more complicated as now the position of the object is randomized at the whole purple slash line area in front of the Dual UR5 (figure 25), which is much wider than the three purple line segmentations in the first part of the experiment. The second part of the experiment also randomizes the orientation of the object in the z direction as shown in the black boxes in figure 25, each with a different orientation. Thus, the object always stays on the $x - y$ plane steadily.

3.3.1 5000 Times of Training

The average training reward line chart for training 5000 times is shown in figure 26. It shows that the agent receives the highest average reward after nearly 2200 times of training.

The testing result in figure 27 and figure 28 shows that the number and the percentage of the range of < 0.01 reach to 398 and 39.80%. This means that there are actually 398 out of 1000 times of test that the object was not even pushed away with any distance by the Dual UR5. Since the performance is not well (nearly 40% of the tests fail), it is required to train the agent for much more times.

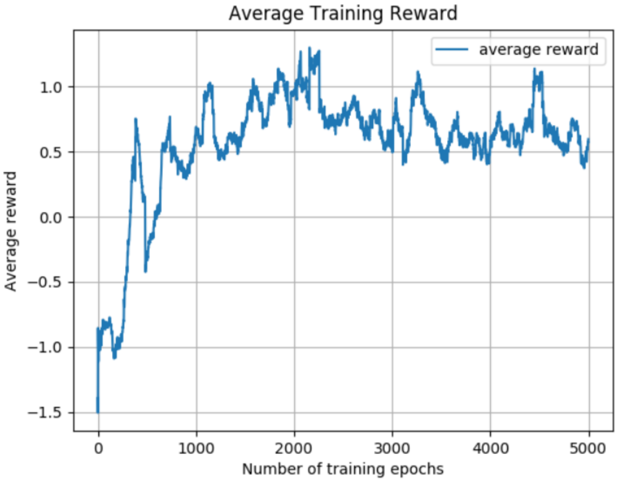

Figure 26. Average Training Reward for Training 5000 Times in the Second Part of the Experiment

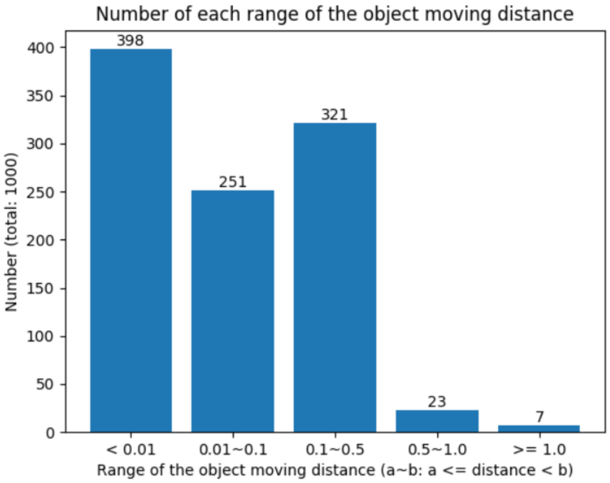

Figure 27. Bar Chart of the Testing Result for Training 5000 Times in the Second Part of the Experiment

The first frame in figure 29 shows that the object in this test is at the right hand side of the Dual UR5 and close to it. Both end effectors touch the object in the second frame. The last frame shows that the object is pushed away backward.

The first frame in figure 30 shows that the object in this test is at the right hand side of the Dual UR5 and a bit far away from it. Both end effectors scroll the object

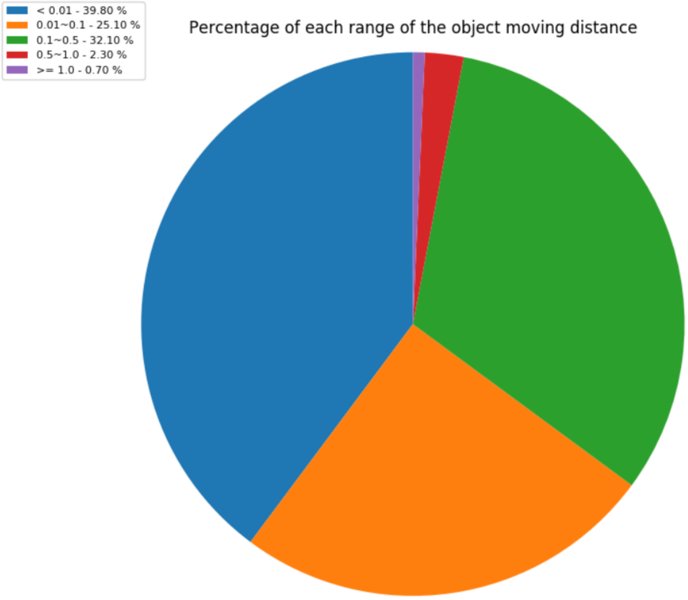

Figure 28. Pie Chart of the Testing Result for Training 5000 Times in the Second Part of the Experiment

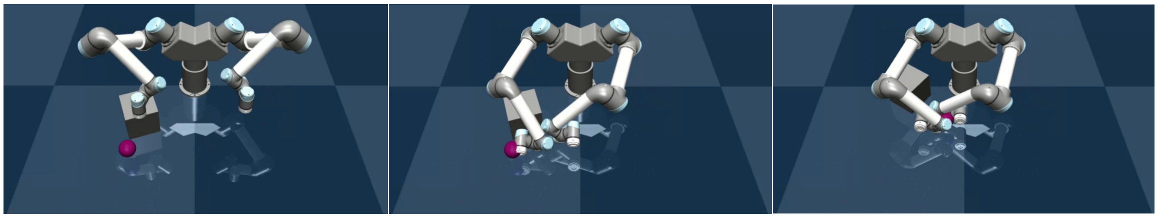

Figure 29. The Box is at the Right Side, Close to the Dual UR5 (Test of Training 5000 Times in the Second Part of the Experiment)

away in the second frame. The last frame shows that the object is kept being scrolled away.

The first frame in figure 31 shows that the object in this test is at the right hand side of the Dual UR5 and far away from it. The right end effector pushes the object away in the second frame. The last frame shows that the object is pushed away with a longer distance.

The first frame in figure 32 shows that the object in this test is at the left hand side of the Dual UR5 and a bit far away from it. The end effectors keep pushing the object away in the second and the last frame.

The first frame in figure 33 shows that the object in this test is at the left hand

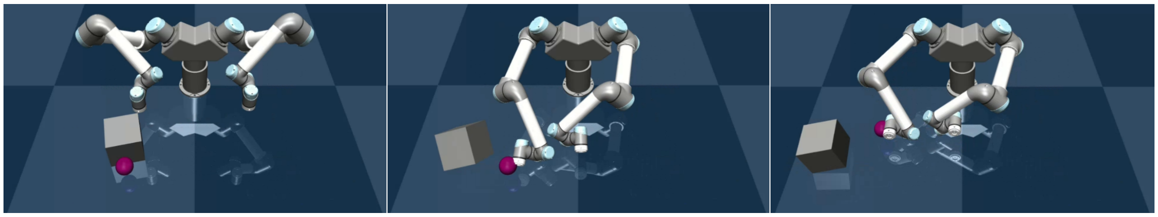

Figure 30. The Box is at the Right Side, a Bit Far Away From the Dual UR5 (Test of Training 5000 Times in the Second Part of the Experiment)

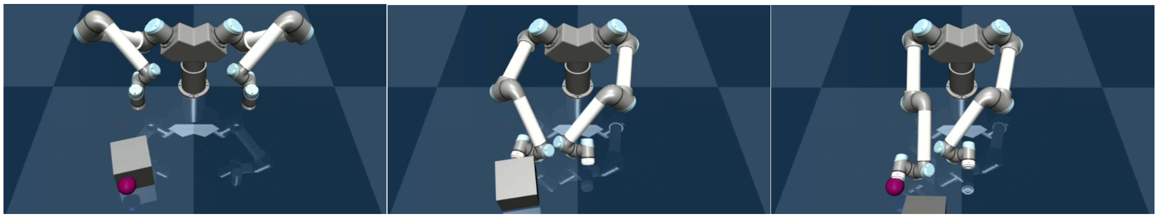

Figure 31. The Box is at the Right Side, Far Away From the Dual UR5 (Test of Training 5000 Times in the Second Part of the Experiment)

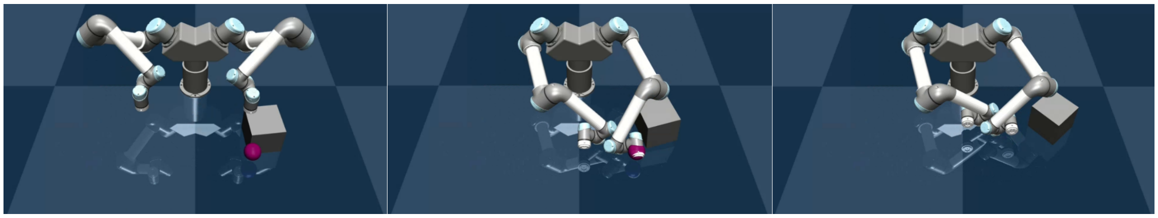

Figure 32. The Box is at the Left Side, a Bit Far Away From the Dual UR5 (Test of Training 5000 Times in the Second Part of the Experiment)

side of the Dual UR5 and far away from it. The left end effector pushes the object away in the second frame. The object is pushed farther away in the last frame.

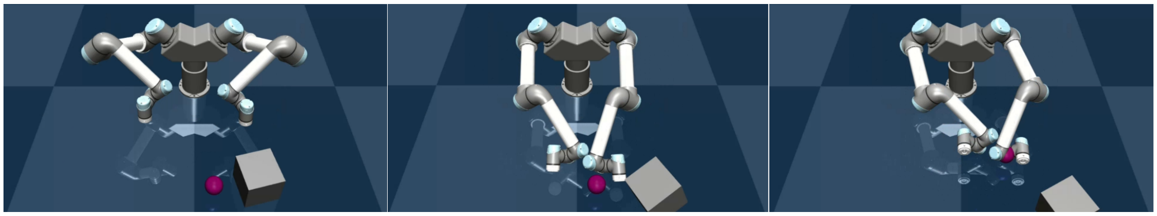

Figure 33. The Box is at the Left Side, Far Away From the Dual UR5 (Test of Training 5000 Times in the Second Part of the Experiment)

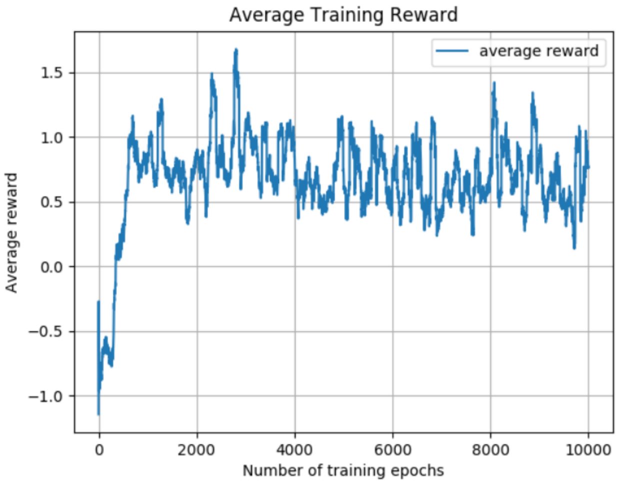

Figure 34. Average Training Reward for Training 10000 Times in the Second Part of the Experiment

3.3.2 10000 Times of Training

The model is then trained for 10000 times and the average training reward line chart is shown in figure 34.

The testing result in figure 35 and figure 36 shows that the number and the percentage of the range of < 0.01 decrease to only 358 and 35.80%.

The first frame in figure 37 shows that the object in this test is at the right hand side of the Dual UR5 and close to it. The end effectors scroll and push the object away in the second and the last frame.

The first frame in figure 38 shows that the object in this test is at the right hand side of the Dual UR5 and far away from it. The right end effector touches the object in the second frame. The last frame shows that the object is pushed away with a small distance by the right end effector.

The first frame in figure 39 shows that the object in this test is at the left hand side of the Dual UR5 and close to it. Both end effectors push the object away in the second frame. The last frame shows that the object is scrolled away backward by the end effectors.

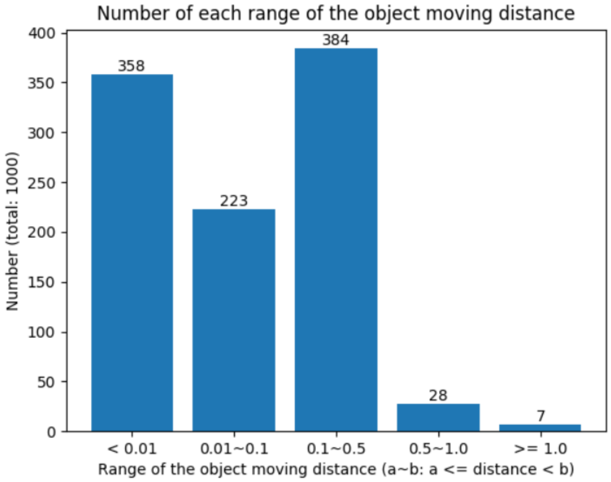

Figure 35. Bar Chart of the Testing Result for Training 10000 Times in the Second Part of the Experiment

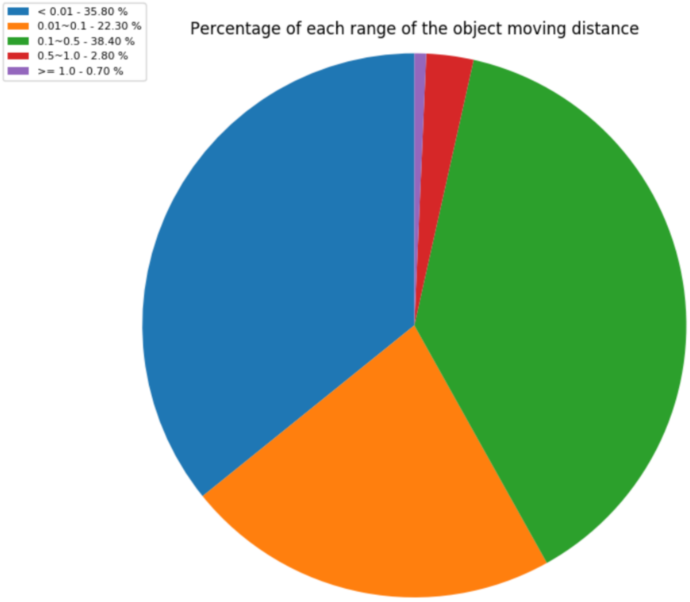

Figure 36. Pie Chart of the Testing Result for Training 10000 Times in the Second Part of the Experiment

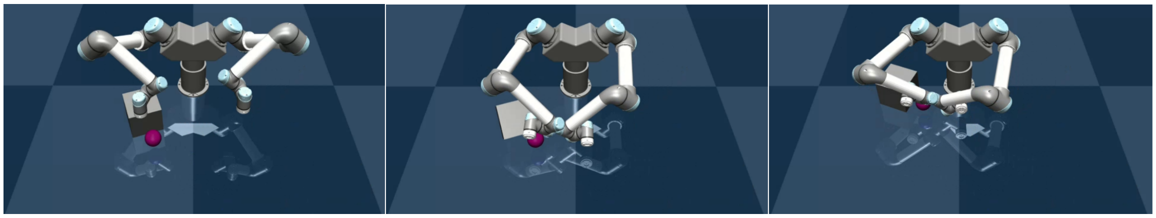

Figure 37. The Box is at the Right Side, Close to the Dual UR5 (Test of Training 10000 Times in the Second Part of the Experiment)

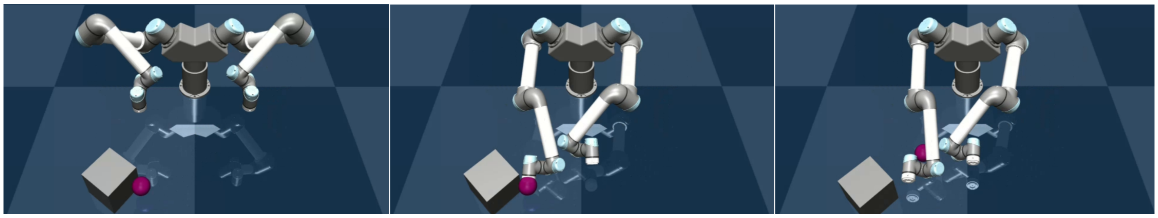

Figure 38. The Box is at the Right Side, Far Away From the Dual UR5 (Test of Training 10000 Times in the Second Part of the Experiment)

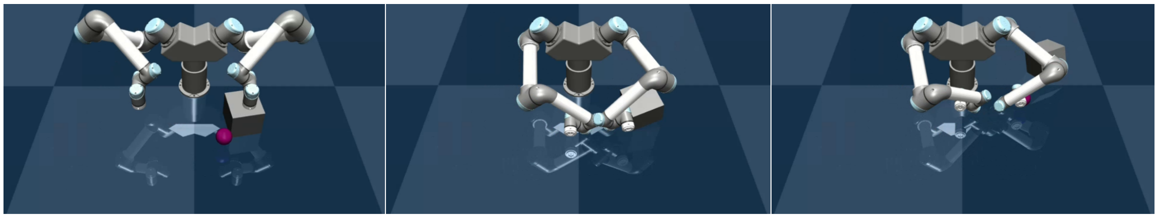

Figure 39. The Box is at the Left Side, Close to the Dual UR5 (Test of Training 10000 Times in the Second Part of the Experiment)

The first frame in figure 40 shows that the object in this test is at the left hand side of the Dual UR5 and far away from it. Both end effectors approach the object and the left end effector keeps pushing it away in the second and the last frame.

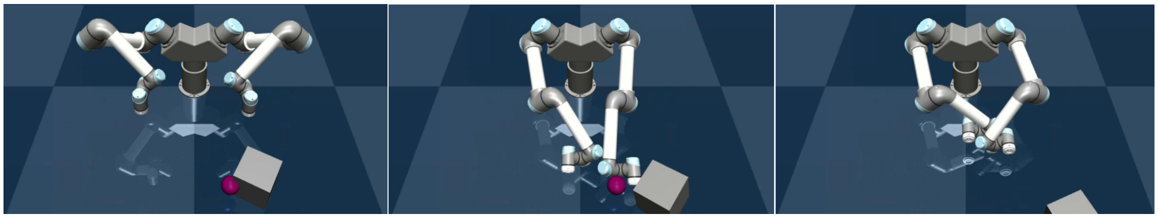

Figure 40. The Box is at the Left Side, Far Away From the Dual UR5 (Test of Training 10000 Times in the Second Part of the Experiment)

3.3.3 Comparison of Training 5000 and 10000 Times

In the table 2, the tests in the three ranges, $0.1 \sim 0.5$, $0.5 \sim 1.0$, and $>= 1.0$, are all with a longer object moving distance. The numbers of the tests of these three ranges from training 5000 times are bigger than or equal to those from training 10000 times. The results show that with much more time of training, the performance becomes better.

Table 2. Comparison of Training 5000 and 10000 Times in the Second Part of the Experiment

Range of object moving distance	< 0.01	0.01 ~ 0.1	0.1 ~ 0.5	0.5 ~ 1.0	>= 1.0
Number of test for training 5000 times	398	251	321	23	7
Number of test for training 10000 times	358	223	384	28	7

3.4 Comparison of the First Part and the Second Part of the Experiment

To compare the two parts of the experiment, the table 3 shows that the performance of the second part is worse than the performance of the first part as the summation of the numbers of the two ranges, < 0.01 and $0.01 \sim 0.1$, increases from the first part of the experiment to the second part of the experiment. This is because now the orientation of the object is taken into consideration, and the position of the object is randomized not only on the three line segmentations, but in the whole slash line area in front of the Dual UR5. Both reasons cause a much more complicated situation and a wider diversity of the environment that the agent will face with.

Table 3. Comparison of the Two Parts of the Experiment (5000 Times of Training)

Range of object moving distance	< 0.01	0.01 ~ 0.1	0.1 ~ 0.5	0.5 ~ 1.0	>= 1.0
Number of test in first part	51	309	624	16	0
Number of test in second part	398	251	321	23	7

3.5 Issues

This section is about all the main issues that occur in the experiments.

3.5.1 Dual UR5 Pushes the Object Away with Its Elbows

The Dual UR5 used to push the object away with its elbows. This is because if the object is pushed away with an elbow of the Dual UR5, then its moving distance will be extremely longer, and the agent will receive a much higher reward. However, the original goal is to make the Dual UR5 to push the object away with its end effectors (wrists), not its elbows. Thus, the solution is to adjust the ratio of each part of the reward to teach the agent to learn using the end effectors to push the object away.

3.5.2 The Two End Effectors Cannot Approach the Similar Points

Originally, the left end effector and the right end effector cannot approach the similar points. This means that they are very far away from each other, which causes that they are unable to touch and push the object away together. The reason is that there were originally two target positions generated in each action, one for the left end effector to reach to, and the other one for the right end effector to reach to. However, it is hard for the agent to train the model to make these two points to be close to each other. Thus, after modifying the action space, now each action contains only one target position for both the left end effector and the right end effector to approach.

3.5.3 End Effectors Only Reach to One of the Two Fixed Positions

At first, the Dual UR5 always reaches to one of the two fixed positions, which depends on whether the object is on the left side or on the right side of the Dual UR5. If the object is at the left hand side of the Dual UR5, then end effectors will reach to the fixed position which is on the left hand side, vice versa. The reason why the end effectors only reach to one of the two positions is because the training is still not enough. Thus, the agent can only generate two "simple" types of actions, one for when the object is on the left side, and the other one for when the object is on the right side. As for the objects at the left side, the agent is unable to generate different actions according to the actual positions of them. This is the same for objects at the right side. Thus, more times of training is required to avoid this problem.

3.5.4 End Effectors Cannot Reach to the Object

At first, neither the left end effector nor the right end effector can reach to the object. This is because the original target position in the action space was in the form of $[x, y, z]$. However, since the object is always on the plane and the plane is fixed at the height of 0.1 in each time of training and testing, the object will be at the same height of 0.1 forever. Thus, it is not necessary for the agent to generate the z value of the target position in the action space. The solution is to change the target position in each action from $[x, y, z]$ to $[x, y]$, while the z value is now always 0.1.

3.5.5 Average Rewards May Oscillate in a Range

It is very easy for average rewards to oscillate in a range. All experiments show that average rewards usually just keep going up and down in a range and are unable to breakthrough it. This is quite common in RL, which causes an agent to stop receiving a new highest reward and is unable to update the model anymore, since the model will only be updated once the average reward becomes higher than the highest average reward.

3.5.6 Forces Are Hard to Be Learned

The agent was originally designed to learn forces, which are exactly the control signals here. However, The absolute values of control signals which are generated by the agent are all just around 1, such as 0.9, 0.98, 0.959, etc. In contrast, the absolute values of real control signals from IRL Control range from 0.001 to 1000. In comparison, the range of control signals from the learned model is much smaller than the range of control signals from IRL Control. The result shows that with this learning result, the Dual UR5 will just lie down and keep twisting the last two joints once loaded into the environment, not to mention to approach the object and push it away. Since the agent fails to learn forces, learning target positions becomes an alternative solution.

Chapter 4

CONCLUSION

In conclusion, RL is able to solve the problem or improve the performance of robot programming. The reward function acts as an indicator that provides information about the goodness of state-action pairs (s, a). This thesis also shows that RL can train the agent to learn motor skills that can be applied to robotic control tasks in industry. Simple applications of robotic control are just like reach task, push task, pick and place task (Greg Brockman and other 370 authors, 2016) while complicated ones can be playing sports or robotic manufacturing.

This thesis proves that DDPG and HER work well together in learning robotic tasks. There are two main factors that can easily affect the training performance. The first factor is enabling the agent to keep receiving higher rewards. The second factor is the reward function. The experiments section shows that there may be a slight difference between the ideal task and the actual task and this depends on the reward function. An accurate reward function can make the actual task look almost the same as the ideal task and make the training process more efficient.

Chapter 5

FUTURE WORK

5.1 Triangle Method

The triangle method of calculating the reward is shown in figure 41. EE_L is the left end effector, EE_R is the right end effector, $obj\_current$ is the current object position, and $obj\_previous$ is the object position in the last step. $distance\_L$ is the distance between the left end effector and the object, $distance\_R$ is the distance between the right end effector and the object, $distance\_L\_R$ is the distance between the left end effector and the right end effector, and $distance\_obj$ is the object moving distance. It is helpful to first go through the definition of actions. The first action is to make both end effectors to approach the object, which means that the agent needs to make $distance\_L$ and $distance\_R$ decrease. Also, both end effectors is required to be as close to each other as possible in order to push the object away together, implying that it is better if $distance\_L\_R$ could be smaller. Surprisingly, there is a triangle which is exactly composed of these three distances - $distance\_L$, $distance\_R$, and $distance\_L\_R$. Pursuing all three distances to be smallest is equal to pursuing the area of the triangle to be smallest. Heron's formula is applied here to relate the three distances with the area of the triangle:

$$area = \sqrt{s(s - distance\_L)(s - distance\_R)(s - distance\_L\_R)} \tag{5.1}$$

$$s = \frac{distance\_L + distance\_R + distance\_L\_R}{2} \tag{5.2}$$

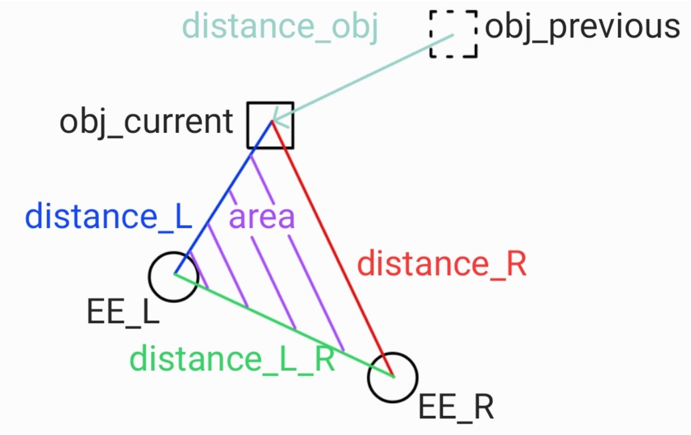

Figure 41. Triangle Method of Calculating the Reward

5.2 Average Rewards May Oscillate in a Range

For the problem of average rewards oscillating in a range and causing the model unable to be updated, a possible solution is to tune the number of rewards used to calculate the average reward. For example, the number of training is one million times. If we always use the latest ten rewards to calculate the average reward, then the value of the average reward will go up very easily, and the agent will be much more likely to receive a new highest reward to update the model. However, the average reward will also drop very easily, so the number of the rewards used to calculate the average reward cannot be too small. If now we always use the last 1000 rewards to calculate the average reward, then it will be very hard for the average reward to increase or decrease rapidly. Thus, if we can find the optimal number of the rewards used to calculate the average reward, then it can help improve many RL applications which have the same problem of being unable to keep receiving a new highest average reward.

REFERENCES

Amor, H. B., K. S. Luck, M. Vecerik, S. Stepputtis and J. Scholz, “Improved exploration through latent trajectory optimization in deep deterministic policy gradient”, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Macau, (IROS 2019) (2019).

Andrychowicz, M., F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel and O. Wojciech Zaremba, “Hindsight experience replay”, (2018).

Brockman, G., J. Terry, and other 370 authors, “Openai gym”, (2016).

DeWolf, T., P. Jaworski, D. Rasmussen, T. B. E. Hunsberger and F. A. S. Rocha, “Abr control”, (2016).

Drolet, M., H. B. Amor, Simon and R. Swaroop, “Irl control”, (2022).

Greg Brockman, J. T. and other 370 authors (2016).

Lillicrap, T. P., J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver and G. D. Daan Wierstra, “Continuous control with deep reinforcement learning”, (2019).

Liu, L., Q. Liu, Y. Song, B. Pang, X. Yuan and Q. Xu, “A collaborative control method of dual-arm robots based on deep reinforcement learning”, MDPI Applied Science (2021).

Nakanishi, J., R. Cory, M. Mistry, J. Peters and S. Schaal, “Operational space control: A theoretical and empirical comparison”, The International Journal of Robotics Research 27(6):737 (2008).

Navale, K., “Trajectory planning using her and reward engineering”, (2021).

Tassa, Y., S. Tunyasuvunakool, A. Quaglino, N. Gileadi, K. Zakka, K. Bayes, T. Erez, L. Burner, Balint-H, B. Nauck, K. Hartikainen, N. Nadeau, P. Mitrano, S. Traversaro, M. Lutter, R. P. Singh, gregor, L. Liu, J. Smith, limymy, jonas eschmann, P. Hawkins, G. Costamagna, R. Rachum, F. Romano, L. F. dos Santos, R. Vaxenburg, Z. Wang, stonfute, D. Butler and L. I. Viegas, “Mujoco200”, (2021).

Zhou, M., “What is ddpg”, (2016).