INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

Control and Simulation of a 6-DOF Biped Robot based on Twin Delayed Deep Deterministic Policy Gradient Algorithm

Phan Bui Khoi[1]*, **Nguyen Truong Giang**[1], **Hoang Van Tan**[1]

1 School of Mechanical Engineering, Hanoi University of Science and Technology, 01 Dai Co Viet, Hai Ba Trung, Hanoi, 100000, Vietnam

OPEN ACCESS

Received: 14.06.2021

Accepted: 24.07.2021

Published: 13.09.2021

Citation: Khoi PB, Giang NT, Tan HV (2021) Control and Simulation of a 6-DOF Biped Robot based on Twin Delayed Deep Deterministic Policy Gradient Algorithm. Indian Journal of Science and Technology 14(30): 2460-2471. https://doi.org/10.17485/IJST/v14i30.1030

* **Corresponding author**.

khoi.phanbui@hust.edu.vn

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: To study an algorithm to control a bipedal robot to walk so that it has a gait close to that of a human. It is known that the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm is a highly efficient algorithm with a few changes compared to the popular algorithm — the commonly used Deep Deterministic Policy Gradient (DDPG) in the continuous action space problem in Reinforcement Learning. **Methods:** Different from the usual sparse reward function model used, in this study, a reward model combined with a sparse reward function and dense reward function will be proposed. The application of the TD3 algorithm together with the proposed reward function model to control a bipedal robot model with 6 degrees of freedom will be presented. The training process is simulated in Gazebo/Robot Operating System (ROS) environment. **Finding:** The results show that, when choosing a reward model combined with a sparse reward function and a dense reward function suitable for the robot model, will help it learn faster and achieve better results. The biped robot can walk straight with an almost human-like gait. In the paper, the results from the TD3 algorithm combined with the proposed reward model are also compared with the results from other algorithms. **Novelty:** Applying the TD3 algorithm combined with the proposed reward model for the 6-DOF biped robot and simulating the robot's gait in Gazebo/ROS environment, ROS is a middleware that can be used to control a robot in a real environment in the future.

Keywords: TD3; biped robot; reinforcement learning; ROS; Gazebo

1 Introduction

In the development of digital technology, the digital age, the strong development in the field of artificial intelligence, robots are gradually being used in production activities and human life, robots can completely replace humans in difficult and dangerous jobs. The work efficiency of robots is also getting better and better, gradually helping people not have to perform works that are dangerous to health. Not only that, robots can also be sociable with humans, interact directly with people nowadays. With these benefits,

robotics research is attracting the interest of many researchers ever, many robot models are designed to resemble human shapes so that based on the human gait, the robot will be controlled in the most convenient way. One of the important factors in controlling robots is their ability to move, many ideas have been put forward for robots to be able to move as quickly and accurately as wheeled robots [1–4]. However, the disadvantage of robots that use wheels to move is that when facing an obstacle ahead, the robot must move in a different direction to avoid them, unlike humans with small obstacles, a human can completely step over them to continue. Therefore, learning, designing, and controlling a robot with a human shape, able to perform human-like movement will help the robot move more flexibly on more complex terrain.

There are many methods used to control biped robots, for traditional solution methods [5,6], we need to set up the Denavit-Hartenberg table for the robot, kinematic and dynamic analysis, design the motion trajectory for the joints. When robot model is changed, parameters are mostly calculated from beginning. One of the most commonly used methods in biped robot control is the zero-moment point (ZMP) [7,8] to set the motion trajectory of robot legs, however, this method suffers from many effects caused by noise, when there is noise from the environment, the robot can easily fall, so, it is difficult to apply in the real environment.

To prevent the robot from falling when there is noise during the robot's motion, several approaches are given to prevent the robot from falling [9–11], or the article [12] suggests a solution that is based on the signals from sensors, to detect the current state of the robot, and after that, it uses classification algorithm Support Vector Machine (SVM) which classifies whether the robot is about to fall and what position the robot is in to give the necessary control signals to prevent the robot from falling. However, after detecting the falling position of the robot, it is still necessary to take action based on robot dynamics and kinematics [13] to prevent the robot from falling.

In recent times, with the advancement of machines and fast computational processors, methods using neural networks are getting more and more attention for their efficiency and they help reduce the computational dynamics and kinematics of the robot. Reinforcement Learning (RL) is an algorithm in the field of artificial intelligence that is widely known because the way models are trained is as natural as how humans learn something new (trial and error) [14,15]. Therefore, RL has achieved impressive achievements [16] such as AlphaGo [17] of DeepMind won the professional player in Go or AlphaFold in the medical field [18]. RL is also applied to robotics [19] and has applications in manufacturing and life [20] such as applications for robotic arms [21–23] or Turtlebot [24]. For a biped robot, RL is applicable to help the robot have the ability to learn and perform a human gait when walking.

In the article [25], the RL algorithm is applied to find the suitable gait for the robot, their method uses model-based, they give appropriate actions based on the Poincare map model, with the algorithms using model-free, the learning process of robot does not need to be based on any other model, but it will learn to walk like a human by itself. One of the model-free RL algorithms commonly used for problems of the continuous action space is Deep Deterministic Policy Gradient [26], it is applied and it's the learning process is simulated the learning process in the article [27] for a bipedal robot which has only 4-DOF. And the training time for the robot is very long when using only the sparse reward. Although the way to set up the sparse reward is simple, it is a challenging problem [28] for the training process of the robot, the model is difficult to achieve the best optimization because the sparse reward function returns only one reward value during a trajectory. So, in order for the model to reach the optimal value faster, in this study, we will combine this sparse reward function with a dense reward function. When using the dense reward function, the model will reach convergence value faster because each step of the robot has a different reward value from the dense reward function to evaluate the output of the model. So the robot's data will be richer and each input has a suitable reward value.

Recently, TD3 [29] and TD3's variants have been applied to some types of robots such as the manipulator arm [30,31], 4-legged-Ant robot [32] which are simulated on the platform with the simple Pybullet environment. But it has yet to be applied and simulated for bipedal robots in ROS/Gazebo which can design complex environments and offer the possibility of practical activities with sensors [33]. Gazebo is a convenient simulation software to test these RL algorithms, and easily changing the parameters of the robot and the environment when using this software will help close the gap between the simulated environment and the real environment, which is convenient for robot control in future studies. Moreover, ROS is the middleware that can be used to apply the TD3 algorithm to the biped robot in the real environment. Therefore, it is necessary to evaluate reinforcement learning algorithms in ROS/Gazebo environment before putting the robot into the real environment.

This study will present a model of a 6 DOF — biped robot which is built to evaluate reinforcement learning algorithms, and how to simulate the robot's self-learning process in ROS/Gazebo so that the robot can walk like a human, the training method used is the TD3 algorithm [29], This study will show the effectiveness of the TD3 algorithm for a biped robot. With the TD3 algorithm as well as a combination of a dense reward [34] and sparse rewards for the robot's learning process, the training time for the robot is significantly reduced and the accumulative reward is also higher than when using other RL algorithms for the biped robot.

The next sections of the article will present the general theory of RL and TD3 algorithm (section 2), the shape and size of the robot model, and the training process used in the simulation (section 3) and section 4 will discuss results that we obtained in this research article.

Simulation results: *https://youtu.be/JVOLrORo0rA*

2 Reinforcement learning: TD3 algorithm

Reinforcement learning is an algorithm that thanks to the interaction of the agent with the environment to decide which action should be taken at each timestep. At a timestep, the agent receives state which is the agent's observation in the environment E, and receives a reward value corresponding to that state. The reward function is shown in (1).

$$R_t = \sum_{i=t}^{T} \gamma^{i-t} r(s_i, a_i) \tag{1}$$

(γ is the discount factor $0 < \gamma < 1$ and r_i is the reward when the agent at state s_i performs an action a_i), In (1), the reward function is calculated as the sum of the reward values r_i multiplied from' by the discount factor γ from the timestep t to T, T is the terminal of the agent's trajectory. γ^{i-t} means that further the state is (the bigger i is), the less effect on the current state's reward R_t. Through that reward function, the agent will decide which action $a_i \in A$ from the policy is most beneficial and an agent will perform that action so that the environment will resend the next state.

The goal of the algorithm is to find a set of parameters (ϕ) of the policy network π_ϕ to maximize the expected return or cumulative reward, the policy network is a network that outputs actions of the agent after receiving input as the state from the environment. The expected return is shown in (2)

$$J(\phi) = E_{s_i, r_i \sim E, a_i \sim \pi}[R_0] \tag{2}$$

For continuous actions, with a large action space, an actor-critic method needs to be used to find the optimal policy. The parameters ϕ of the policy network (in the actor-critic method known as the actor-network) are updated by calculating the gradient of expected return $\nabla_\phi J(\phi)$ mentioned in the deterministic policy gradient algorithm of D. Silver[35]. To evaluate the agent's action, a value function or a critic function $Q^\pi(s,a) = E_{s_i \sim p_\pi, a_i \sim \pi}[R_t \mid s, a]$ is used. $Q^\pi(s,a)$ evaluates the agent at state s and action a (an according to the policy π). The gradient of expected return is represented by (3).

$$\nabla_\phi J(\phi) = E_{S_i \sim p_\pi}\left[\nabla_a Q^\pi(s,a)|_{a=\pi(s)} \nabla_\phi \pi_\phi(s)\right] \tag{3}$$

There have been many articles, with many algorithms used in continuous action problems such as Proximal Policy Optimization (PPO)[36], Soft Actor-Critic (SAC)[37], Deep Deterministic Policy Gradient (DDPG)[26], Twin Delayed DDPG (TD3)[29], ..., with good results achieved with robotic arms[30,31], 4-legged-Ant robot[32], TD3 is an appropriate algorithm for our robot.

As mentioned in the article[29], TD3 is an actor-critic algorithm, it consists of one actor-network(or policy network) π_ϕ and two critic networks $Q_{\theta_i}\ (i = 1,2)$ and their target networks (one actor target network π_ϕ and two critic target networks $Q_{\theta_i'}\ (i = 1,2)$, θ_i is the set of parameters of the critic network that is used to approximate the critic function $Q^\pi(s,a)$).

The actor-network outputs the action values with input values that are state s or information obtained from the environment and added a noise[37]a $\sim \pi_\theta(s) + \varepsilon$ (with $\varepsilon \sim N(0,\sigma)$) and target action a' are also calculated: $a' \sim \pi_{\phi'}(s') + \varepsilon$ (with $\varepsilon \sim N(0,\sigma)$) , adding the clipped noise to the target action makes the target action closer to the original action. The noise used follows the Gaussian distribution $N(0,\sigma)$.

The critic network uses the Bellman equation[38] to update its parameters, these two critic network give two Q-values, and use the smaller $Q_{\theta_i'}(i = 1,2)$ value to calculate the target value y(r,s')(4)

$$y(r, s') = r + \gamma \min_{i=1,2} Q_{\theta_i'}(s', \pi_\phi(s') + \varepsilon) \tag{4}$$

After the target value y(r, s') is calculated, the parameters of critic networks were learned by minimizing the mean square error between $Q_{\theta_1}, Q_{\theta_2}$ and $y(r, s')$ (5), (6).

$$L_{\theta_1, D} = \underset{(s,a,r,s') \sim D}{E}\left[\left(Q_{\theta_1}(s,a) - y(r,s')\right)^2\right] \tag{5}$$

$$L_{\theta_2, D} = \underset{(s,a,r,s') \sim D}{E}\left[\left(Q_{\theta_2}(s,a) - y(r,s')\right)^2\right] \tag{6}$$

with D is the replay buffer consisting of transitions (s, a, r, s') received during agent's exploration process. When the number of transitions (s, a, r, s') stored in the replay buffer D is large enough, a batch size or a number of samples is randomly selected from D to calculate the loss value $L_{\theta_i,D}(i=1,2)$.

The parameters ϕ of the actor-network are updated by maximizing Q_{θ_1} (7)

$$\max_{\phi} \underset{s\sim D}{E}\left[Q_{\theta_1}\left(s,\pi_{\phi}(s)\right)\right] \tag{7}$$

However, the TD3 algorithm updating parameters less frequently[29] will make the variance value lower so that the updated parameters (8) (9) are more accurate. To reduce the frequency of parameter updating, a delay factor d is used, and to avoid overestimation, a hyperparameter $\tau(0<\tau<1)$ is used which helps the parameters of target networks to be updated slowly.

$$\theta_i' \leftarrow \tau\theta_i + (1-\tau)\theta_i' \tag{8}$$

$$\phi' \leftarrow \tau\phi + (1-\tau)\phi' \tag{9}$$

The flow diagram of the TD3 algorithm is shown in Figure 1

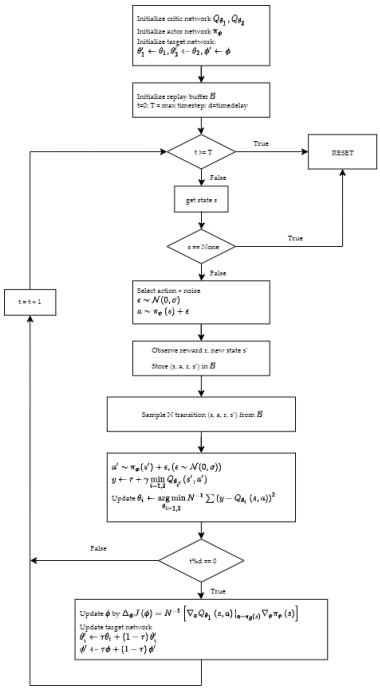

Fig 1. Flow diagram of TD3 algorithm. The two critic networks include 2 hidden layers, both layers have 256 neurons, the actor-network also has 2 hidden layers, and also contains 256 neurons, the discount factor γ is 0.99, the maximum time step is 1600, the batchsize N is 64, the delay factor d is 2, the learning rate of all networks is0.0001. ReLU is used for activation functions.

With these improvements compared with DDPG which is applied to the 4 DOF bipedal robot[27], therefore we will apply the TD3 algorithm to the 6-DOF biped robot with a more human appearance.

3 Model and training process

3.1 Biped robot model

The shape and size of the biped robot built is shown in Figure 2

The biped robot consists of seven links – one waist section, two thigh sections, two shank sections, and feet. The dimensions of links are all in meter. Symbols, mass, and materials of each link are shown in Table 1. Different types of joints to connect links are shown in Table 2. Therein, links "Ground", "Cyl_below", "Cyl_above" and "Horiz" only support simulation, not results.

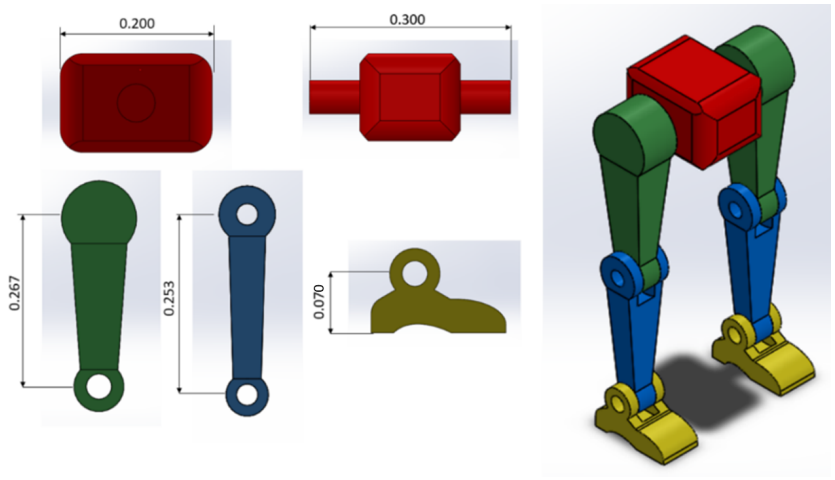

Fig 2. Shape and size of the biped robot built by 3Ddesign software

Table 1. Symbols, mass, and material of links

Link	Note	Material	Mass (Kg)
WAIST	Waist	Acrylonitrile butadiene styrene(ABS)	4.0200
TH_L	Left Thigh	Acrylonitrile butadiene styrene(ABS)	1.7324
TH_R	Right Thigh	Acrylonitrile butadiene styrene(ABS)	1.7324
SH_L	Left Shank	Acrylonitrile butadiene styrene(ABS)	0.7527
SH_R	Right Shank	Acrylonitrile butadiene styrene(ABS)	0.7527
F_L	Left Foot	Acrylonitrile butadiene styrene(ABS)	0.4111
F_R	Right Foot	Acrylonitrile butadiene styrene(ABS)	0.4111

Table 2. Relationship between links of robots

Parent link	Child link	Joint type
Ground	Cyl_below	Fixed
Cyl_below	Cyl_above	Prismatic
Cyl_above	Horiz	Fixed
Horiz	WAIST	Prismatic
WAIST	TH_L	Revolute
WAIST	TH_R	Revolute
TH_L	SH_L	Revolute
TH_R	SH_R	Revolute
SH_L	F_L	Revolute
SH_R	F_R	Revolute

The coordinate system of the joints is performed in Figure 3, the robot walks along the x-axis (in the absolute coordinate system). The rotation range of the six joints, i.e. the hip joints, knee joints, and ankle joints during the movement along the sagittal plane is tabulated in Table 3 and illustrated in Figure 4. Figure 4 shows the direction of rotation, flexion, and extension states of joints (hip, knee, and ankle). The desired angles (the output of the algorithms) will be normalized within these rotation ranges.

3.2 Training process

After the model is built by 3D design software, the robot model is converted to .stl format and .urdf file, .urdf file contains information about the mass, the moment of inertia, the angle limit of the joints, ... of the robot. With this format, the robot can

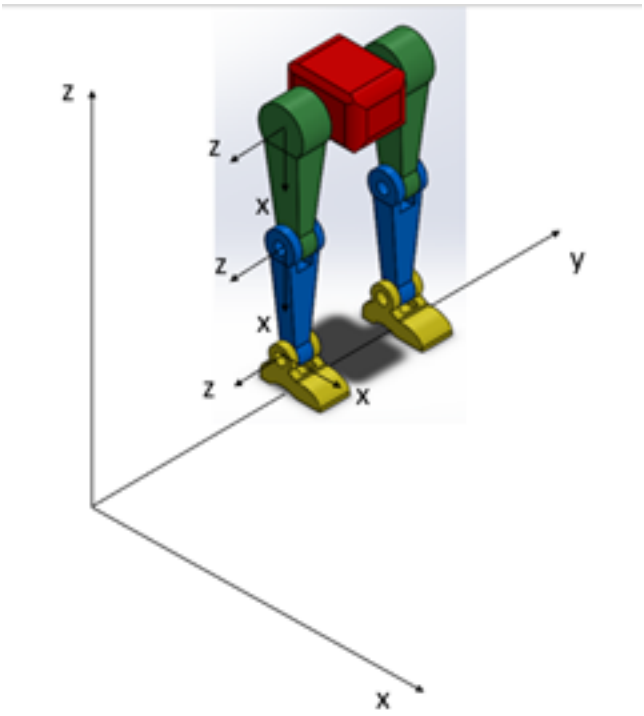

Fig 3. The coordinate system of the robot

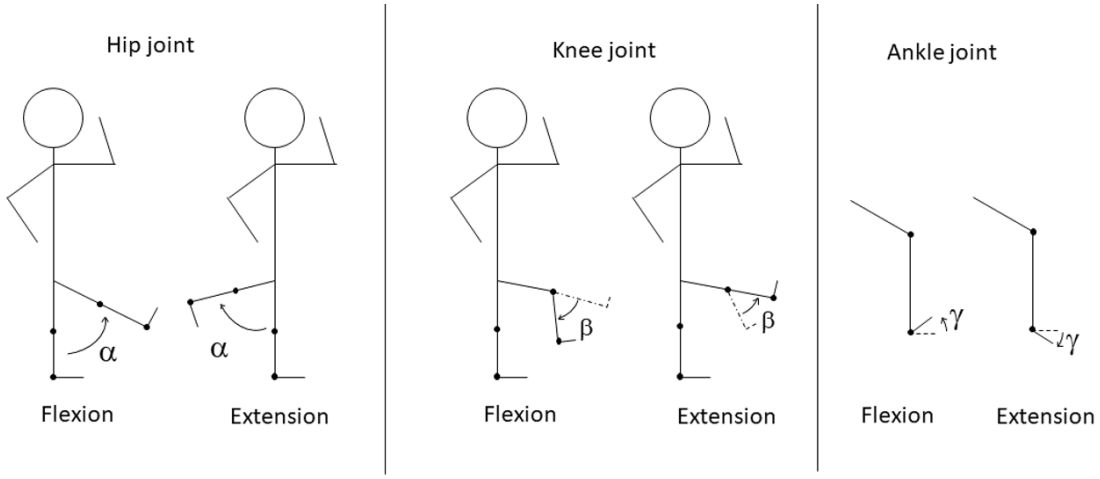

Fig 4. The direction of rotation, flexion, and extension states of hip, knee, and ankle joints.

Table 3. Limits of rotation of joints

Joint	Flexion(rad)	The extension (rad)	Annotation
Hip	0.7854	0.7854	α
Knee	1.3962	0.0012	β
Ankle	0.7854	0.7854	γ

be launched into Gazebo for simulating the robot training process, ROS is used for the operation of robots in this simulation environment with a frequency of 60Hz. During the training process, the robot will begin learning from random actions (a) (the actions are calculated from the actor-network and added a Gaussian noise), which is the exploration of the agent (biped robot). The sensors will send signals which are the next state (s') of the agent. At each timestep, when the robot performs the action calculated from the neural network, a reward (r) is also calculated to create a transition (s, a, r, s'). These transitions are stored in Replay Buffer, when the number of transitions in the Replay Buffer is large enough, the neural network will randomly select a number of transitions from the Replay Buffer as inputs to update its parameters according to the TD3 algorithm presented in section 2. This interaction between the neural network and the environment through action, state, and reward is shown in Figure 5. Information about the action, state, and reward used is shown below.

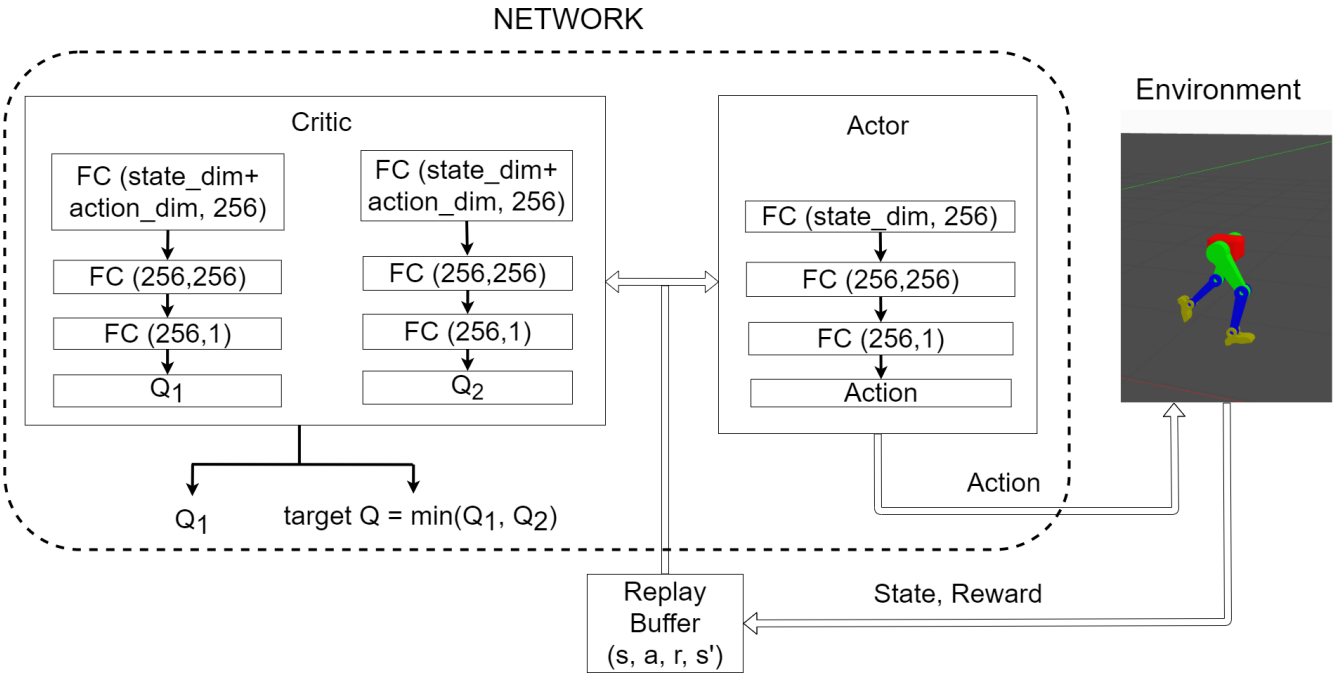

Fig 5. Robot bipedal training simulation system; FC (x,y) means that at this Fully Connected (FC) layer, x input neurons link to this layer's y neurons. The neural network structure obeys the TD3 algorithm.

Action: At a timestep, through our robot-trained model, 6 action values which are desired angles of the joints are calculated so that the robot can move fast forward without falling. By the PID controller, the joints of the robot will be rotated to those desired positions. With ROS, PID coefficients are manually tuned so that the robot can rotate to those desired angles quickly, accurately, and consistently.

State: After performing these actions, the agent receives a state with 16 components from the environment including the robot's speed according to the robot's movement direction (x-axis), the robot's speed in a direction perpendicular to the ground (z-axis), the angle and the angular velocity of the hip joints, the angle and the angular velocity of the knee joints, the angle and the angular velocity of the ankles, the grounding state of the feet, we choose these components because these parameters can be measured by sensors located at the links, joints of the robot and it also characterizes the movement of the robot.

Reward: Based on the dense reward function in the article[34] for the bipedal robot, our reward function is designed by combining the sparse reward and dense reward function for our biped robot. With the sparse reward, we set a big bonus (+10) when the robot goes all the way and the biped robot will pay a heavy penalty (-10) if the robot falls, which contributes to the robot's balance problem. In addition, with dense reward, at each timestep, the robot with state received from the environment receives different reward values.

$$r = v_x - m\sum_{i=0}^{6} a_i^2 - n(z + h_0)^2 \tag{10}$$

where v_x (m/s) is the robot's speed in the robot's direction of movement (x-axis), a_i is the value of the robot's actions, z (m) is the waist position in the z-direction of the robot (origin at the robot's highest position in the upright state of the robot), a_i (rad)

are desired angles (action of agent), m and n are the constants for increasing the penalty value for the robot. According to our experience, we choose m = 3.0, n = 0.05.

Based on the human gait, the human waist will fluctuate up and down with a cycle during walking. So in order for the robot to have a similar human gait, h_0 is calculated in the reward function by averaging the robot's waist position between the robot's vertical state and its one-foot forward state. Call H and L are the positions when the robot is upright (H = 0) and when the robot steps one foot (L= -0.003m). Therefore, h_0 is calculated as (11).

$$h_0 = \frac{H + L}{2} \tag{11}$$

So, h_0 is equal to -0.0015m, which means that the robot's waist is around the z position = -0.0015 when the robot is in motion.

The dense reward function is intended to guide the robot to move as quickly as possible and the robot's waist coordinate in the vertical direction of the body does not change too much during the movement. Finally, we get a reward function after combing the above two functions for the robot when it is in the timestep i (12), l_e is the maximum distance used during training. 10 is a big bonus if the robot goes all the way, and -10 is the heavy penalty if the robot falls.

$$r_i = \begin{cases} 10 & \text{if } x_i > l_e \\ -10 & \text{if } z_i > 0.1 \\ v_{x_i} - m\sum_{j=0}^{6} a_j^2 - n\,(z_i + 0.0015)^2 & \text{otherwise} \end{cases} \tag{12}$$

4 Results

To see the effectiveness of the TD3 algorithm in the training simulation of a 6-DOF biped robot, we used two other RL algorithms, DDPG and SAC, to compare with TD3. The training process is carried out in about 10000 episodes, each with 1600 timesteps, each episode will end when the robot falls or walks 15 meters or the number of steps is more than 1600. TD3 with more advantages (section 2) helps the robot achieve a higher reward than the other two algorithms, and the reward fluctuation range is also smaller (Figure 5). In Figure 5, the solid line represents the reward value of the algorithms after every 100 episodes, the dashed line represents the average value of the rewards of those algorithms.

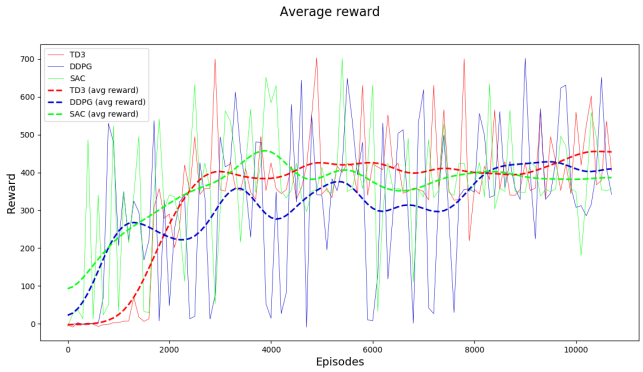

Fig 6. Average reward obtained when using 3 algorithms 1. TD3; 2. DDPG; and 3. SAC during training for biped robot in more than 10,000episodes

With the same reward function as above (section 3.2), the same number of episodes (10000 episodes), the same robot parameters, and the same hyperparameters of the algorithms such as learning rate or batch size… the average reward of the TD3 algorithm is higher but not much higher to the other two algorithms, because each algorithm makes the robot walk fast forward with different gaits. However, in Figure 7, we can see that when using the TD3 algorithm the robot has a more human-like gait than the other two algorithms, the two joints corresponding to the two legs of the robot have similar graphs on amplitude as well as the cycle which is the same as when humans walk forward. During human movement, when the left foot is the pivot foot, the right footsteps forward, the angle of the right hip joint tends to increase and the angle of the left hip joint tends to decrease and vice versa. So, in the graph of the hip joint in Figure 5, we can see that the TD3 algorithm helps the robot to reach that state, but the other two algorithms do not follow a certain rotation period. And the rotation period of the knee and ankle joints of the TD3 algorithm is also more stable than DDPG or SAC.

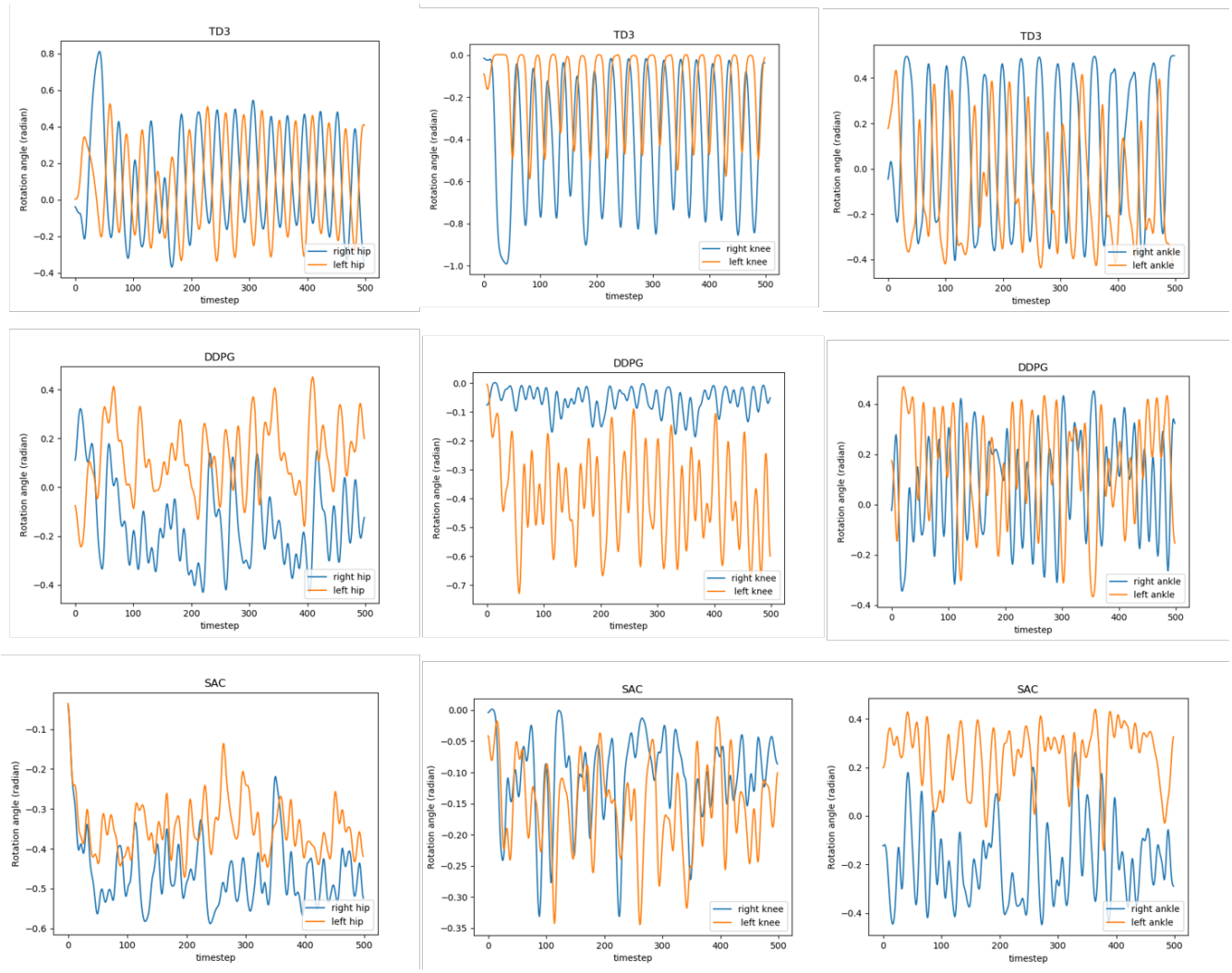

Fig 7. Rotation angle graphs of the joints from left to right (hip, knee, ankle) in 500 time steps of biped robot when using top-down algorithms (TD3, DDPG, SAC), the blue color represents the right joint rotation, orange color represents the left joint rotation.

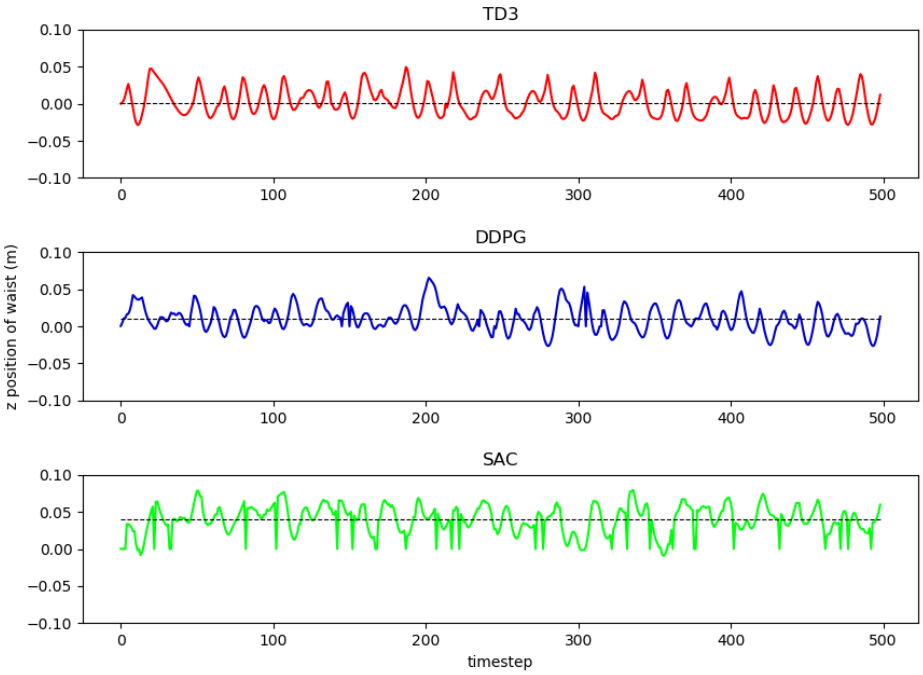

Fig 8. The waist position on the z-axis when the robot moves. The red, blue, and green lines are the z coordinate of the waist within the 500 time steps using TD3, DDPG, SAC algorithms. The dashed black line is the average z-value (in avg_z(TD3) = 0.0016m, avg_z(DDPG) = 0.010m, avg_z(SAC) =0.0393m)

The efficiency of the algorithm is also shown more clearly in Figure 8. In Figure 8, we see that when using the TD3 algorithm, the z coordinate of the waist changes around the position z = 0.0016m (close to the desired value according to our reward function (-0.0015m)), the other two algorithms have a larger difference than the desired value. The waist's up and down cycle are also more stable with TD3.

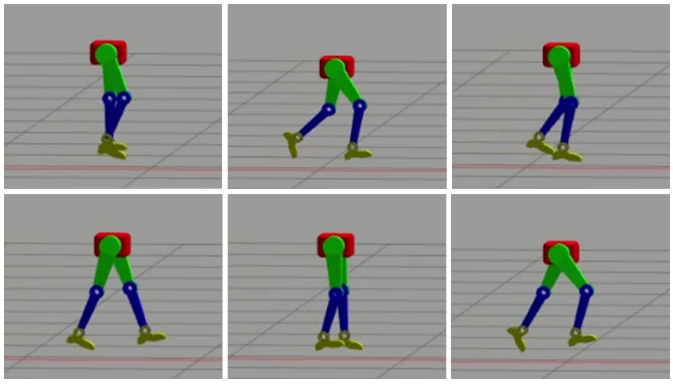

Fig 9. Simulation of the robot's walking gait after training with TD3

The results of simulating the gait of the biped robot in one step cycle from right foot forward to straight state to left foot forward and so on and so on is visualized in Figure 9.

During the research, we realized that the training time for the robot when using DDPG, SAC algorithm is almost double that of TD3 to reach 10000 episodes. When using only the sparse reward function (= 10 when reaching the destination; = -10 when falling) for the TD3 algorithm, in the same 10000 episodes, the robot still cannot go all the way.

5 Conclusion

In this study, a model of a biped robot is built to evaluate the algorithm with the shape of human legs, each leg of the robot has 3 degrees of freedom that can rotate in joint limits. The robot was trained by TD3 algorithm and simulated the training process by Gazebo/ROS, this algorithm will produce the desired rotation angle of each joint at each timestep when receiving information (state) from the environment. By PID controller, the joints will rotate to those desired angles. In this research, a reward function was given that helps robots learn faster and achieve higher efficiency as well. Unlike using only a sparse reward function, robot training time is significantly shortened by combining dense reward function and sparse reward function. This reward function helps the robot have a gait trajectory closer to humans. We have also compared TD3 with other RL algorithms (DDPG, SAC) when applying them with ROS for biped robots in the simulation environment. With the same number of timesteps during training, when the robot moves, the robot when using the TD3 algorithm has a more human-like gait than the DDPG and SAC algorithm, the two joints corresponding to the two legs of the robot have similar graphs on amplitude as well as the cycle (Figure 6) and the robot's waist position oscillates closer to the desired value in our reward function (the desired position of waist = -0.0015m and the average position of the waist when using TD3, DDPG and SAC: avg_z(TD3) = 0.0016m; avg_z(DDPG) = 0.01m; avg_z(SAC) =0.0393m). The training time of the TD3 algorithm is only half that of other algorithms.

With these achieved results, in the future, we will try to apply RL algorithms and ROS for a real robot so that it can walk in the real environment and develop more in order that the biped robot can walk in more complex terrain and we will try to improve the robot so that it can move in 3D space.

References

1) Mohamed AK, Youmin Z. Developments and Challenges in Wheeled Mobile Robot Control. In: Proceeding of the 10th International Conference on Intelligent Unmanned Systems (ICIUS). 2014. Available from: https://dx.doi.org/10.13140/RG.2.1.3595.3363.
2) Solea R, Filipescu A, Nunes U. Sliding-mode control for trajectory tracking of a Wheeled Mobile Robot in presence of uncertainties. *Proceedings of Asian Control Conference*. 2009;p. 1701–1706. Available from: https://ieeexplore.ieee.org/document/5276210.
3) Li W, Yang C, Jiang Y, Liu X, Su CY. Motion Planning for Omnidirectional Wheeled Mobile Robot by Potential Field Method. *Journal of Advanced Transportation*. 2017;2017:1–11. Available from: https://doi.org/10.1155/2017/4961383.
4) Morin P, Samson C. Motion Control of Wheeled Mobile Robots. 2008;p. 799–826. Available from: http://dx.doi.org/10.1007/978-3-540-30301-5_35.
5) Hernández SC, Rodriguez LE, R S, Gordillo JL. Kinematics and Dynamics of a New 16 DOF Humanoid Biped Robot with Active Toe Joint. *International Journal of Advanced Robotic System*. 2012;9:1–12. Available from: https://doi.org/10.5772/52452.
6) Xh B, Dermaku A, Likaj SA. Kinematics and and Dynamics Modeling of the Biped Robot. *IFAC Proceedings*. 2013;46:69–73. Available from: https://doi.org/10.3182/20130606-3-XK-4037.00032.
7) Vadakkepat P, Goswami D. Biped Locomotion: Stability, Analysis and Control. *International Journal of smart sensing intelligent system*. 2008;1:187–207. Available from: https://doi.org/10.21307/ijssis-2017-286.
8) Vukobratovic M, Borovac B. Zero-Moment Point-Thirty five years of its life. *International Journal of Humanoid Robots*. 2004;1:157–173. Available from: https://doi.org/10.1142/S0219843604000083.
9) Morisawa M, Harada K, Kajita S, Kaneko K, Sola J, E Y. Reactive stepping to prevent falling for humanoids. *Proceeding of IEEE-RAS International conference on Humanoid Robots*. 2009;p. 528–534. Available from: https://doi.org/10.1109/ICHR.2009.5379522.
10) Kajita S, Sakaguchi T, Nakaoka S, Morisawa M, Kaneko K, Kanehiro F. Quick squatting motion generation of a humanoid robot for falling damage reduction. *Proceeding of IEEE International Conference on Cyborg and Bionic Systems (CBS)*. 2017;p. 45–49. Available from: https://doi.org/10.1109/CBS.2017.8266127.
11) Morisawa M, Kaneko K, Kanehiro F, Kajita S, Fujiwara K, Harada K. Motion Planning of Emergency Stop for Humanoid Robot by State Space Approach. In: Proceeding of IEEE International Conferenceon Intelligent Robots and System. 2006;p. 2986–2992. Available from: http://dx.doi.org/10.1109/IROS.2006.282232.
12) Kim J, Choi T, Lee J. Falling avoidance of biped robot using state classification. *IEEE International Conference on Mechatronics and Automation*. 2008;p. 72–76. Available from: https://doi.org/10.1109/ICMA.2008.4798728.
13) Huang Q, Yokoi K, Kajita S, Kaneko K, Arai H, Koyachi N, et al. Planning walking patterns for a biped robot. *IEEE Trans Robotics Automation*. 2001;17:280–289. Available from: https://doi.org/10.1109/70.938385.
14) S R, Andrew GB. Reinforcement Learning: An Introduction. Cambridge, MA, USA. MIT Press. 2018. Available from: https://psycnet.apa.org/record/2019-19679-000.
15) Vicent FL, Peter H, Riashat I, B MG, P J. An Introduction to Deep Reinforcement Learning. *Found Trends Machine Learning*. 2018;11:219–354. Available from: http://dx.doi.org/10.1561/2200000071.
16) Hao-Nan W, Ning L, Yi-Yun Z, Da-Wei F, Feng H, Dong-Sheng L, et al. Deep reinforcement learning: a survey. *Frontiers Information Technology Electron Engineering*. 2020;21:1726–1744. Available from: https://doi.org/10.1631/FITEE.1900533.
17) Silver D, Huang A, Maddison CJ, Guez A, Sifre L, Drieleman GV, et al. Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature*. 2016;529:484–489. Available from: https://doi.org/10.1038/nature16961.
18) Andrew WS, Richard E, John J, James K, Laurent S, Tim G, et al. Improved Protein Structure Prediction using 2 Potentials from Deep Learning. *Nature*. 2020;577:706–710. Available from: https://doi.org/10.1038/s41586-019-1923-7.
19) Julian I, Jie T, Chelsea F, Mrinal K, Peter P, Sergey L. How to train your robot with deep reinforcement learning: lessons we have learned. *International Journal Robotics Research*. 2021;p. 1–24. Available from: https://doi.org/10.1177/0278364920987859.
20) Kober J, Bagnell J, Peters J. Reinforcement learning in robotics: A survey. *International Journal Of Robotics Research*. 2013;32:1238–1274. Available from: https://doi.org/10.1177/0278364913495721.

21) Hai N, Hung ML. Review of Deep Reinforcement Learning for Manipulation. In: Proceeding of IEEE International Conference on Robotic Computing. 2019;p. 25–27. Available from: https://doi.org/10.1109/IRC.2019.00120.
22) Rongrong L, Florent N, Philippe Z, Michel D, Birgitta DL. Deep Reinforcement Learning for the Control of Robotic Manipulation: A Focussed Mini-Review. *Robotics*. 2021;10:1–13. Available from: https://doi.org/10.3390/robotics10010022.
23) Mellatshahi SN. Learning Control of Robotic Arm Using Deep Q-Neural Network. Electronic . 2021. Available from: https://scholar.uwindsor.ca/etd/8568.
24) Bedoya DR. Diseño de una estrategia para la planeación de rutas de navegación autónoma de un robot móvil en entornos interiores usando un algoritmo de aprendizaje automático. 2020. Available from: https://repositorio.unal.edu.co/handle/unal/78793.
25) Morimoto J, Cheng G, Atkeson CG, Zeglin G. A simple reinforcement learning algorithm for biped walking. *Proceeding of IEEE International Conference on Robotics and Automation*. 2004. Available from: https://doi.org/10.1109/ROBOT.2004.1307522.
26) Lillicrap TP, Hunt JJ, Pritzel A, Heess N, Erez T, Tassa Y, et al. Continuous control with deep reinforcement learning. *Proceeding of 4th International Conference on Learning Representations (ICLR)*. 2016. Available from: https://arxiv.org/abs/1509.02971.
27) Kumar A, Paul N, Omkar S. Bipedal Walking Robot using Deep Deterministic Policy Gradient. 2018. Available from: https://arxiv.org/pdf/1807.05924.pdf.
28) Guoyu Z, Qishen Z, Jiahao L, Jiangeng L. Efficient hindsight reinforcement learning using demonstrations for robotic tasks with sparse rewards. *International Jorurnal of Advanced Robotics System*. 2020;17(1). Available from: https://doi.org/10.1177/1729881419898342.
29) Fujimoto S, Hoof HV, Meger D. Addressing Function Approximation Error in Actor-Critic Methods. *Proceeding of International Conference on Machine Learning Conference (ICML*. 2018;p. 1587–1596. Available from: https://arxiv.org/abs/1802.09477.
30) Hou Y, Hong H, Sun Z, Xu D, Zeng Z. The Control Method of Twin Delayed Deep Deterministic Policy Gradient with Rebirth Mechanism to Multi-DOF Manipulator. *Electronics*. 2021;10:870. Available from: https://doi.org/10.3390/electronics10070870.
31) Myeongseop K, Dong-Ki H, Jae-Han P, Jung-Su K. Motion Planning of Robot Manipulators for a Smoother Path Using a Twin Delayed Deep Deterministic Policy Gradient with Hindsight Experience Replay. *Applied Science* . 2020;10:575. Available from: https://doi.org/10.3390/app10020575.
32) Stephen D, Wenfeng Z. Twin-Delayed DDPG: A Deep Reinforcement Learning Technique to Model a Continuous Movement of an Intelligent Robot Agent. *Proceeding of International Conference on Vision*. 2019;66. Available from: https://doi.org/10.1145/3387168.3387199.
33) José MC, Eduardo P, Lía GP, Jesús FC. ROS-Based Open Tool for Intelligent Robotics Education. *Applied Science*. 2020;10(21):7419. Available from: https://doi.org/10.3390/app10217419.
34) Duan Y, Chen X, Houthooft R, Schulman J, Abbeel P. Benchmarking Deep Reinforcement Learning for Continuous Control. *Proceedings of International Conference on Machine Learning*. 2016;p. 1329–1338. Available from: https://arxiv.org/abs/1604.06778.
35) Silver D, Lever G, Heess N, Degris T, Wierstra D, Ried-Miller M. Deterministic Policy Gradient Algorithms. In: Proceeding of International Conference on Machine Learning (ICML). 2014;p. 387–395. Available from: https://dl.acm.org/doi/10.5555/3044805.3044850.
36) Tuomas H, Aurick Z, Pieter A, Sergey L. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In: Proceeding of International Conference on Machine Learning. 2018;p. 1861–1870. Available from: https://arxiv.org/abs/1801.01290.
37) Rafael S. Noise, Overestimation and exploration in Deep Reinforcement Learning. 2020. Available from: https://arxiv.org/pdf/2006.14167v1.pdf.
38) Bellman R. Dynamic programing. *Science*. 1966;153:34–37. Available from: https://doi.org/10.1126/science.153.3731.34.