




A Soft Actor-Critic Approach for a Blind Walking Hexapod Robot with Obstacle Avoidance

Lei Wang¹, Ruiwen Li² , Ziwei Huangfu³ , Yishan Feng² and Yiyang Chen^{4,*} ¹ School of Automation, Wuxi University, Wuxi 214105, China; leiwang@cwuxu.edu.cn² School of Automation, Nanjing University of Information Science and Technology, Nanjing 210044, China; 202212490510@nuist.edu.cn (R.L.); 202312490404@nuist.edu.cn (Y.F.)³ Key Laboratory of Advanced Perception and Intelligent Control for High-End Equipment, Ministry of Education, Anhui Polytechnic University, Wuhu 241000, China; 2220320184@stu.ahpu.edu.cn⁴ School of Mechanical and Electrical Engineering, Soochow University, Suzhou 215137, China

* Correspondence: yychen90@suda.edu.cn

Abstract: This paper investigates a path planning approach for the walking and obstacle avoidance of a blind hexapod robot in various field conditions. Hexapod robots often perform field tasks in unstructured environments, and their external sensors are affected by weather and light. This paper proposes the use of internal sensors to sense the terrain and a slightly modified soft actor-critic algorithm to train the motion strategy. A hexapod robot is capable of walking smoothly on rough ground only using internal sensors that are not affected by weather factors, and the soft actor-critic approach is superior for overcoming high-dimensional issues for multi-degree-freedom robot motion in unstructured environments. The experiments showed that the hexapod robot not only traversed rugged terrain at a fixed speed but also possessed obstacle avoidance capabilities.

Keywords: hexapod robot; soft actor-critic; unstructured environment; adaptive motion; obstacle avoidance



Citation: Wang, L.; Li, R.; Huangfu, Z.; Feng, Y.; Chen, Y. A Soft Actor-Critic Approach for a Blind Walking Hexapod Robot with Obstacle Avoidance. *Actuators* **2023**, *12*, 393. <https://doi.org/10.3390/act12100393>

Received: 10 September 2023

Revised: 11 October 2023

Accepted: 19 October 2023

Published: 21 October 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Today, a growing number of robots are becoming present in people's lives. Various mobile robots are applied to replace humans, to perform tasks in challenging environments, such as desert exploration [1], wall climbing [2], and exploration in confined spaces [3]. Currently, in practical applications, mobile robots are mostly wheeled robots, whose ability to traverse terrain is limited by the size of their wheels relative to the terrain [4]. Other mobile robots such as tracked robots have certain advantages on rough terrain [5] but still face challenges in more complex environments.

In recent years, research on legged robots has made great progress, and legged robots exhibit amazing traversability compared to traditional wheeled and tracked robots [6–9]. Among legged robots, hexapod robots have better stability and fault tolerance than bipedal and quadrupedal robots [10,11]. Hexapod robots have a simpler structure and consume less energy than octopod and higher legged robots.

In the field of the mechanical structural design of hexapod robots, the dimensional design of a robot affects its ability to traverse rugged terrain [12]. A hexapod robot with a planar Schatz linkage structure can be stacked flat and is also capable of traversing rugged terrain, while ensuring a small size, easy storage, and easy transportation [13]. A radially symmetric hexapod robot can achieve the maximum workspace within the allowable range of a given size and certain physical constraints [14]. A hexapod robot with a leg-wheel structure can also perform well in heterogeneous scenarios and on rugged terrain [15–17].

Motion control of hexapod robots is a combination of various sensor information for joint control and coordination the between legs. The main current control methods are bio-inspired control [18], engineering-based control [19], machine learning-based control [20,21], and combinations of two or more. Bio-inspired control is typified by CPG

control, which is derived from bionics inspired by insects and allows hexapod robots to execute gaits without any loss of equilibrium or increased delay [22,23]. This method has already been put into applications, such as an underwater hexapod robot that can climb large-angle slopes [24], and a hexapod robot that generates gaits adapted to different environments [25,26]. In engineering, a hexapod robot foot trajectory tracking control method based on Udwadia–Kalaba theory was proposed, which had a faster error convergence and response speed compared with the traditional sliding mode space [27]. An omnidirectional tracking strategy based on model predictive control and real-time replanning was proposed, to address the difficulty of accurately tracking the reference trajectory of a hexapod robot leg [28,29].

Decentralized control strategies can be embedded in a RL approach, to adapt to different types of articulated robots [30]. The combination of CPG and RL provides a breakthrough in end-to-end learning for mobile robots [31]. Mimicking the brain of an insect, a hexapod robot can also have complex advanced cognitive control. It can recognize information about the environment and the transitions between various motions appropriate for each environment [32]. It can also handle a wide variety of perturbations and walk normally despite losing a leg [33].

For the current hexapod robots, plenty of problems still exist [34]. Small legs have a great impact on their dynamics and kinematic performance when they are in contact with the ground. A team investigated a capacitive tactile sensor that can effectively improve body speed and efficiency [35]. Hexapod robots performing tasks in the field often encounter the problem of visual sensor failure, due to weather and lighting, and also encounter puddles and other problems caused by the failure of the depth sensor. A blind hexapod robot was proposed that uses only on-board IMU data, joint angle data, and the leg–ground contact to sense the environment. This approach does not require expensive sensors and overcomes the effects of weather and light [36]. Robots are often faced with avoiding non-geometric obstacles. An unsupervised learning framework enables hexapod robots to autonomously generate physical segmentation models of the terrain in changing environments [37]. However, this framework requires visual sensors that are susceptible to weather and light, and this is not feasible for blind hexapod robots. Autonomous adaptation of robots to unknown and complex terrain is difficult [38], and the existing methods for robot foothold planning remain computationally expensive [39].

Since hexapod robots are vulnerable to weather and light while performing tasks in the field, this paper employs on-board IMU data and joint angle data to perceive the surrounding environmental information. This does not utilize the pressure information between each foot and the ground, which reduces the development cost, to give a simpler structure and meet the minimum walking conditions of a hexapod robot. In addition, this solution is unlikely to exhibit overfitting in a simulation and is easy to deploy to real robots.

The multiple redundant degrees of freedom in hexapod robots lead to a high-dimensional continuous state space, and a blind hexapod robot acquires limited surrounding information, making it difficult to adapt to rugged and complex terrain. This paper uses SAC in DRL to train the gait, which adds entropy terms to the objective function and gives the desired exploration ability for high-dimensional complex tasks [40]. This paper optimizes the strategy network in SAC, to adapt it to the structural–spatial relationship of a hexapod robot. It also optimizes the reward function in reinforcement learning, to generate better mobility strategies for walking and obstacle avoidance tasks. This paper trained hexapod robots with various typical types of reinforcement learning in a built training platform, and compared the training strategies, which proved the superiority of the algorithm used in this paper.

The contribution of this paper is to explore obstacle avoidance methods for blind hexapod robots, including the terrain avoidance of high slopes and pits. As well as exploring the application of a slightly modified SAC to the problem of walking and obstacle avoidance for a blind hexapod robot, and comparing it to PPO and TD3 in the same context.

2. Hexapod Robot System

In this section, the experimental hexapod robot system is introduced. The structure and sensor system of the hexapod robot are first described, followed by the control framework for the hexapod robot and the adaptive gait for legged robots. Furthermore, walking and obstacle avoidance for blind hexapod robots in rugged terrain is discussed.

2.1. Structure and Sensors

“JetHexa” is a hexapod robot produced by a company named “Hiwonder”. JetHexa is an open-source hexapod robot based on the ROS operating system. It is equipped with Jetson Nano, a steering gear, Lidar, camera, and other hardware configurations. The Jetson Nano is a GPU-based embedded development board produced by NVIDIA, and its ARM is loaded with the Ubuntu 18.04LTS system. The steering gear was independently developed by Hiwonder, its model is HX-35H, with a maximum torque of 35 kg.cm. The lidar has a range frequency of 4 k–9 k/s and a range radius of 16 m. The JetHexa is equipped with a LiDAR model EAI G4. It can realize robot motion control, mapping navigation, somatosensory interaction, and other applications.

The structure of the experimental hexapod robot in this paper is shown in Figure 1, in which the overall length of the hexapod robot is 397 mm, the width is 426 mm, and the height is 238 mm. The forward direction of the robot is the X-axis positive direction, the X-axis positive direction rotated counterclockwise by 90° is Y-axis positive direction, the vertical upward is the Z-axis positive direction, and the unit of measurement is mm. The coordinates of each joint are shown in Table 1.

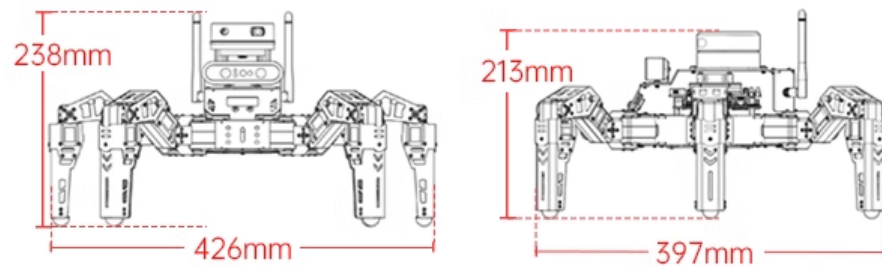


Figure 1. Dimensions of the hexapod robot.

Table 1. Coordinates of the joints.

	Left Leg	Right Leg
front leg	(93.6, 50.805, 0)	(93.6, −50.805, 0)
middle leg	(0, 73.575, 0)	(0, −73.575, 0)
rear leg	(−93.6, 50.805, 0)	(−93.6, −50.805, 0)

Each leg joint of the hexapod robot consists of a high-voltage servo, an angle feedback device, and a temperature and voltage sensor. The angle feedback device can record the rotation angle of each joint, and the temperature and voltage sensor can understand the internal data of the servo in real time, to protect the servo. As shown in Figure 2, the hexapod robot has an inertial measurement unit (IMU) and an 18-angle feedback as proprioceptive sensors, which form the sensor network of the hexapod robot.



Figure 2. Structure of the hexapod robot.

2.2. Control Framework

As shown in Figure 3, the control framework can be mainly categorized into two stages: low-level control, and high-level control. First, the proprioceptive signals are obtained from the environment through the sensor network, which can determine its own attitude and the ruggedness of the surrounding environment. In the high-level control, the policy network receives the human-selected task types and proprioceptive signals, and then it generates robot trajectories and gaits corresponding to the tasks. In the low-level control, the motion trajectory of each joint of the hexapod robot is decomposed into angles and motion directions at certain time intervals, and finally the joints are precisely rotated by the positional PID control through feedback from the angle feedback device.

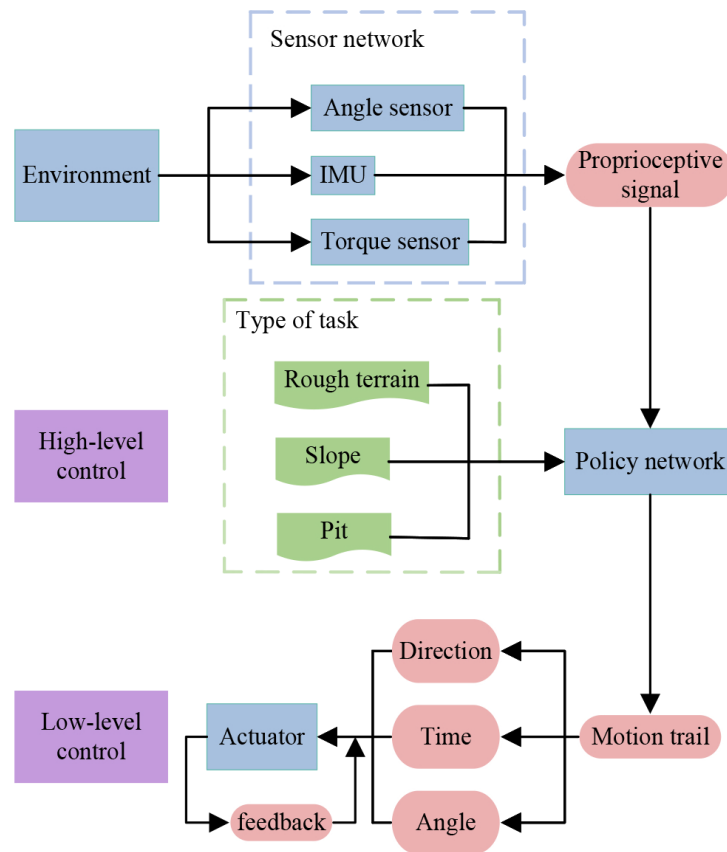


Figure 3. Motion Control Framework.

2.3. Adaptive Gaits

The motion of a hexapod robot is a high-dimensional combinatorial problem, and the best solution is to choose a suitable gait for a given terrain. A gait is a periodic motion that restricts the robot's movements to the appropriate terrain. A suitable gait ensures that the robot can traverse the terrain without damaging itself or the terrain, and is superior in terms of energy efficiency and smoothness. An appropriate gait is capable of crossing a given terrain with minimal movement, i.e., minimal wear and tear on the joints.

On structured terrain, a single gait can be used to pass through the terrain. The unstructured ground is complex and has many obstacles, and using adaptive gait allows the hexapod robot to autonomously fine-tune its gait according to its surroundings, to adapt to the unstructured ground. In this work, adaptivity is a locomotion strategy for a hexapod robot that relies only on proprioceptive signals to recognize information about its surroundings and thus change its entire gait.

2.4. Walking and Obstacle Avoidance

The blind hexapod robot exploits internal sensors to obtain information about its surroundings, which include an inertial measurement unit IMU on the torso, and angle feedback and torque feedback at each joint. The proprioceptive signals from the hexapod robot can be obtained after integrating the signal processing of the internal sensors. Proprioception is the perception of one's own space and position in three-dimensional space, and this also allows the body to perceive changes in limb movements [41]. For example, blind people are able to carry out normal behavior such as walking and eating through proprioception and touch even though they have lost their vision. Simple or complex sports such as walking, running, and playing basketball require proprioception [42]. The proprioceptive signals obtained from the sensor network are the trunk position, trunk linear velocity, trunk angular velocity, joint angle, and joint angular velocity.

The hexapod robot–environment interaction is discretized in the time dimension as a sequence of time steps. The state of the hexapod robot with the environment at each time step is represented by the vector s , which contains all proprioceptive signals acquired from the sensor network. The reward obtained by the hexapod robot interacting with the environment at each time step is denoted by r , which reflects how favorable the current environment is to the robot. The action performed by the hexapod robot at each time step is denoted by the vector a , which contains the motion trajectories of all the moving joints. As shown in Table 2, the interaction process between the hexapod robot and the environment is described using MDPs. When the hexapod robot walks on rugged ground, the state of each time step changes with the roughness of the terrain. As shown in Figure 4, the rougher the terrain, the larger its stance angle, and the drop of each leg is larger, thus adjusting the gait to adapt to the environment. In particular, when the hexapod robot is walking on high slopes or pitted terrain, its stance angle rises or falls sharply in succession. When it encounters an impassable high slope or pit, it can make an evasive maneuver at the next time step, according to the change of attitude angle.

Table 2. Interaction process between the hexapod robot and the environment.

t	$t + 1$...	$t + n$
a_t	a_{t+1}	...	a_{t+n}
s_{t+1}	s_{t+2}	...	s_{t+n+1}
r_t	r_{t+1}	...	r_{t+n}

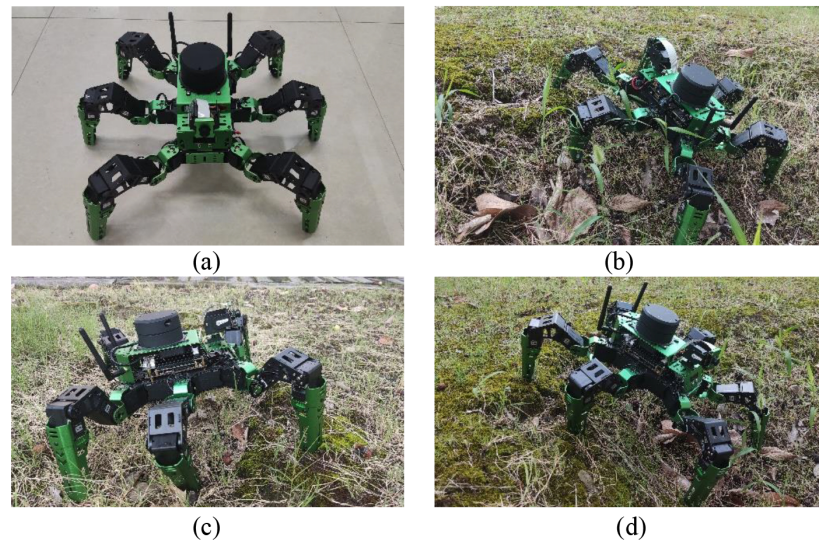


Figure 4. Postures of the hexapod robot on different terrains: (a) is a posture moving on flat ground, with a small Euler angle; (b) is a posture moving over a pit with a large variation in Euler angles and a large drop in each leg; (c) is a posture moving on a perlin with a small change in Euler angle and a small drop in each leg; (d) is a posture moving on a slope with a large change in Euler angle and a large drop in each leg.

3. A Soft Actor-Critic Algorithm

The interaction process between the hexapod robot and the environment uses MDPs, which satisfy the conditions for the use of RL. This paper uses SAC to train the motion strategy of the hexapod robot. SAC is a non-strategy DRL based on maximum entropy, where the introduction of maximum entropy gives a strong exploratory ability [43]. In addition, it has excellent robustness and high sampling efficiency, providing a stable learning framework for control tasks in continuous action space [44].

SAC has an additional entropy term $H(P)$ compared to the objective function of conventional DRL: $H(P)$ is

$$H(P) = \mathbb{E}_{x \sim p} [-\log P(x)], \quad (1)$$

where $P(x)$ is the distribution of x . The maximum entropy term makes the action output more random, thus improving the exploration and learning ability of the algorithm. Then, the optimal policy π^* in SAC is

$$\pi^* = \arg \max_{\pi} \sum \mathbb{E}_{(s_t, a_t) \sim \rho_{\pi}} [r + \alpha \cdot H(\pi(\cdot | s_t))], \quad (2)$$

where π is the robot's motion strategy, a denotes the action, s denotes the state, and r denotes the reward. Here, α denotes the temperature coefficient, a hyperparameter that determines the importance of entropy with respect to the reward, thus controlling the degree of randomness of the strategy.

Set the state s_t and the action a_t at time t , and the value function $Q_{soft}(s_t, a_t)$ is

$$Q_{soft}(s_t, a_t) = r + \gamma \mathbb{E}_{s_{t+1}, a_{t+1}} [Q_{soft}(s_{t+1}, a_{t+1}) - \alpha \log(\pi(a_{t+1} | s_{t+1}))], \quad (3)$$

where γ is the discount coefficient.

As for actor-critic, SAC has value networks and policy networks. To avoid using an overestimated Q network to compute the objective when maximizing, SAC introduces two online networks and two objective networks. The parameters of the two online networks are set as θ_1 and θ_2 , and the parameters of the two target networks are set as θ'_1 and θ'_2 . Then, the minimum value of the output of the two target networks are obtained as the

target value in the calculation. The parameters of the value network can be updated by minimizing the loss function as

$$J_Q(\theta_i) = \mathbb{E}\left[\frac{1}{2}(Q_{\theta_i}(s_t, a_t) - \hat{Q}_{\theta_i}(s_t, a_t))^2\right], \quad (4)$$

$$\hat{Q}_{\theta_i}(s_t, a_t) = r + \gamma[Q_{\theta_i}(s_{t+1}, a_{t+1}) - \alpha \log(\pi_{\varphi}(a_{t+1}|s_{t+1}))]. \quad (5)$$

The policy network parameters are denoted by φ , with the state s_t as input. A Gaussian distribution with mean π_{φ}^{μ} and standard deviation π_{φ}^{σ} is used as the output. Then, the action a_t can be obtained using a reparameterization with a_t as

$$a_t = f_{\varphi}(\tau_t; s_t) = \pi_{\varphi}^{\mu}(s_t) + \pi_{\varphi}^{\sigma}(s_t). \quad (6)$$

In the above formula, the f_{φ} function outputs the mean and standard deviation, τ_t is the standard normally distributed noise signal, and π_{φ}^{μ} and π_{φ}^{σ} are the mean and standard deviation of the Gaussian distribution, respectively.

SAC represents the policy with an energy-based model, which is an appropriate solution for difficult control tasks. To establish a link between the energy model and the soft value function, it is necessary to set a suitable energy function, and the energy function ϵ is set as

$$\epsilon(s_t, a_t) = -\frac{1}{\alpha} Q_{soft}(s_t, a_t). \quad (7)$$

Then, the policy π can be expressed as

$$\pi(s_t|a_t) \propto \exp\left(\frac{1}{\alpha} Q_{soft}(s_t, a_t)\right). \quad (8)$$

The policy network parameters are updated using a method that minimizes the Kullback–Leibler scatter. When the Kullback–Leibler scatter is smaller, the difference between the corresponding output behavioral rewards is smaller, and the policy converges better. The update function of the strategy network is

$$\begin{aligned} J_{\pi}(\varphi) &= D_{KL}[\pi_{\varphi}(\cdot|s_t) || \exp(\frac{1}{\alpha} Q_{\theta}(s_t, \cdot) - \log Z(s_t))] \\ &= \mathbb{E}_{s_t \sim D, a_t \sim \pi_{\varphi}}[\log \pi_{\varphi}(a_t|s_t) - \frac{1}{\alpha} Q_{\theta}(s_t, a_t) + \log Z(s_t)], \end{aligned} \quad (9)$$

where $Z(s_t)$ is used to normalize the distribution.

Finally, the parameters of the policy network are updated using the gradient descent method, denoted as

$$\nabla_{\varphi} J_{\pi}(\varphi) = \nabla_{\varphi} \alpha \log(\pi_{\varphi}(a_t|s_t)) + \nabla_{\varphi} f_{\varphi}(\tau_t; s_t) (\nabla_{a_t} \alpha \log(\pi_{\varphi}(a_t|s_t)) - \nabla_{a_t} Q(s_t, a_t)). \quad (10)$$

The larger the temperature coefficient α , the stronger its exploration ability. The adjustment of the hyperparameter α has a great impact on the training effect of the SAC, and α needs to be adjusted for different tasks and different training cycles. A temperature auto-adjustment mechanism [45], which obtains the optimal temperature coefficients for each step by minimizing the objective function, is denoted as

$$J(\alpha) = \mathbb{E}_{a_t \sim \pi_t}[-\alpha \log \pi_t(a_t|s_t) - \alpha H_0], \quad (11)$$

where H_0 is a predefined threshold for the minimum policy entropy. Algorithm 1 gives the pseudo-code for the SAC.

Algorithm 1 The training method of SAC

Input: initial a empty experience memory M and iteration number N .

Output: policy network $\pi(\theta)$

- 1: initial parameters $\theta_1, \theta_2, \varphi$
- 2: initial target network parameters $\theta'_1 \leftarrow \theta_1, \theta'_2 \leftarrow \theta_2$
- 3: **for** N **do**
- 4: **for** each environment step **do**
- 5: Observe state s_t and select action $a_t \sim \pi(\cdot|s_t)$
- 6: Execute a_t , observe next state s_{t+1} , reward r_t
- 7: Store $\{s_t, a_t, r_t, s_{t+1}\}$ in the experience memory
- 8: **end for**
- 9: **for** each gradient step **do**
- 10: Randomly sample a batch of samples (s_t, a_t, r_t, s_{t+1}) from M
- 11: Update Q-functions parameters:
- 12: $\theta_i \leftarrow \theta_i - \lambda \hat{\nabla}_{\theta_i} L_{\theta}(\theta_i)$ for $i \in \{1, 2\}$
- 13: Update policy parameters:
- 14: $\varphi \leftarrow \varphi - \lambda \hat{\nabla}_{\varphi} J_{\pi}(\varphi)$
- 15: Update temperature coefficient:
- 16: $\alpha \leftarrow \alpha - \lambda \hat{\nabla}_{\alpha} J(\alpha)$
- 17: Update policy network parameters:
- 18: $\theta'_i \leftarrow \tau \theta_i - (1 - \tau) \theta'_i$ for $i \in \{1, 2\}$
- 19: **end for**
- 20: **end for**
- 21: **return** policy network $\pi(\theta)$

4. Training Procedure

This chapter first introduces Mujoco, a training platform for blind hexapod robots, then, describes its process for training policy networks.

4.1. Training Platform

RL tends to require more samples than supervised and unsupervised learning. Because the training process of RL requires a large number of interactions and trials, it must constantly interact with the environment. This leads to some limitations in the application of RL in real time, especially when the real environment is costly or risky [46].

Using virtual environments for training is the preferred solution. Robots may encounter unexpected situations in the real world, which can lead to the destruction of equipment, injury, or other unforeseen consequences. By training in a virtual environment, potential risks can be avoided, and this ensures that the trained strategies are more reliable and safe when used in a real environment. The physics engine also provides complete control over the environment and tasks. It is easy to modify the parameters of the virtual environment, introduce different scenarios and contexts, and perform fast playback and replay. This makes RL debugging and optimization easier and more efficient. More importantly, training in a virtual environment provides easy access to rich data, including states, actions, and reward signals. These data serve as datasets for algorithms to perform model learning and strategy optimization. In addition, the physics engine records and saves detailed information of each training step for subsequent analysis and evaluation.

This paper established a training platform for hexapod robots using Mujoco [47], a commonly used physics simulation engine that is widely used in robot control, reinforcement learning, and motion planning. It is able to simulate the dynamic behavior of rigid body systems with multiple joints and contact forces. It is based on Newtonian mechanics, to calculate the collision, contact force, and dynamic behavior between objects. The geometry, inertial properties, and joint types of a hexapod robot are described through XML files. The terrain is represented by a grayscale map with topographic information. At the same time, the geometry, surface friction, and collision characteristics of the terrain

are also represented by XML files and generated in MuJoCo. Figure 5 shows a simulation model of the hexapod robot and terrain.

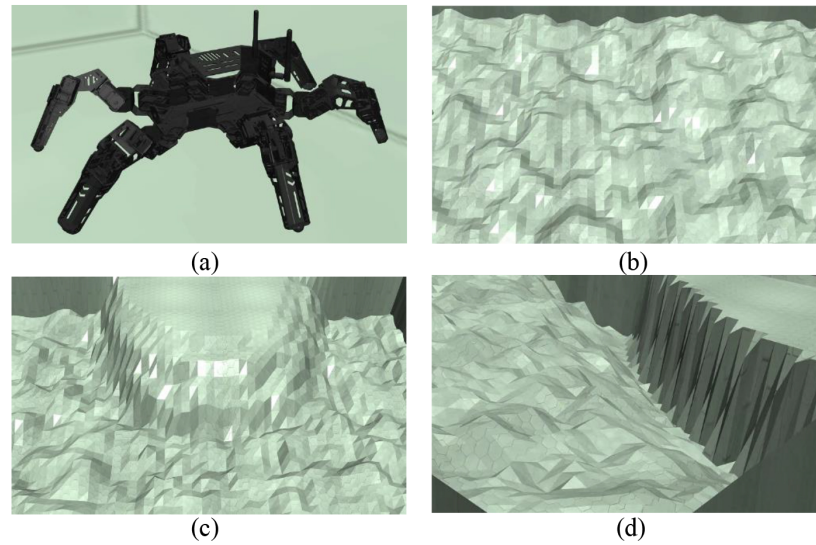


Figure 5. Hexapod robot and terrain models: (a) hexapod robot, (b) perlin, (c) slope, (d) pit.

4.2. Training Framework

Figure 6 shows the control framework of the hexapod robot based on SAC. SAC is implemented using pytorch and trained using Mujoco-py, to control the simulation environment. The state space contains all proprioceptive signals, including the torso position $Pose$, torso linear velocity V_{body} , torso angular velocity ω_{body} , joint angle θ_{joint} , and joint angular velocity ω_{joint} . Then, the state space vector S is

$$S = [Pose, V_{body}, \omega_{body}, \theta_{joint}, \omega_{joint}]. \quad (12)$$

The action space contains the motion trajectories of 18 joints used to control the robot and is represented by the vector A :

$$A = [T_1, T_2, \dots, T_{18}]. \quad (13)$$

The neural network structure used by SAC is shown in Figure 7. In this paper, the structure of the actor is modified. Two micro-neural networks are first used to extract features for highly relevant inputs, and then the features are put into a fully connected neural network. The actor uses a strategy network containing four layers, with the first input layer and the second hidden layer consisting of two micro-neural networks, as the input and output layers of the micro-neural network, respectively. The first micro-neural network input layer is $Pose$, V_{body} and ω_{body} , and the output layer has 128 torso-related features. The second micro-neural network input layer is θ_{joint} and ω_{joint} , and the output layer has 128 leg-related features. The third layer of the strategy network is a hidden layer containing 256 units that combines the features extracted by the two micro-neural networks. The fourth layer is the output layer, which outputs the mean π_{ϕ}^{μ} and the standard deviation π_{ϕ}^{σ} of action obeying a Gaussian distribution. The critic, on the other hand, uses two online networks and two target networks with the same structure, the first layer being an input layer containing 64 units with inputs A and S , the second and third layers having the same structure as the actor's hidden layer, and the fourth layer being an output layer that outputs the Q value, to evaluate the current action.

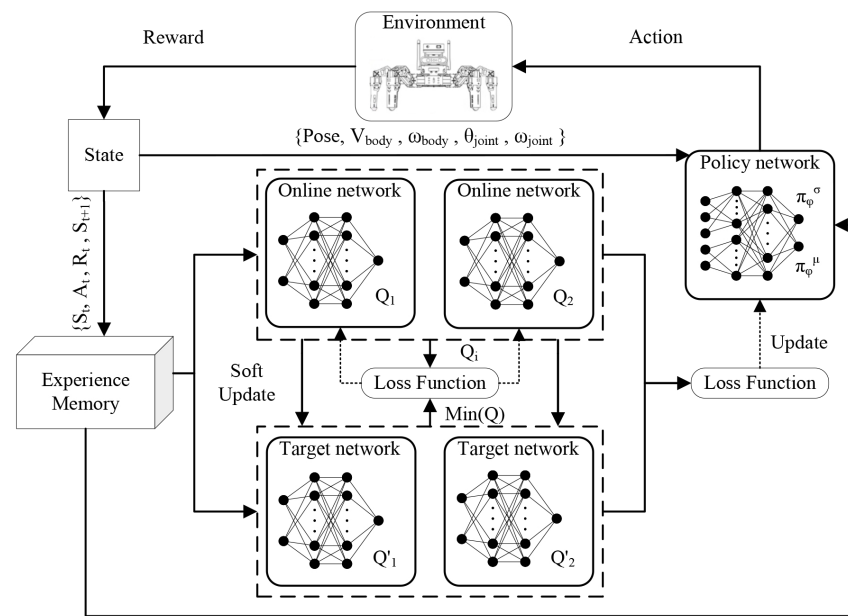


Figure 6. Control framework based on SAC.

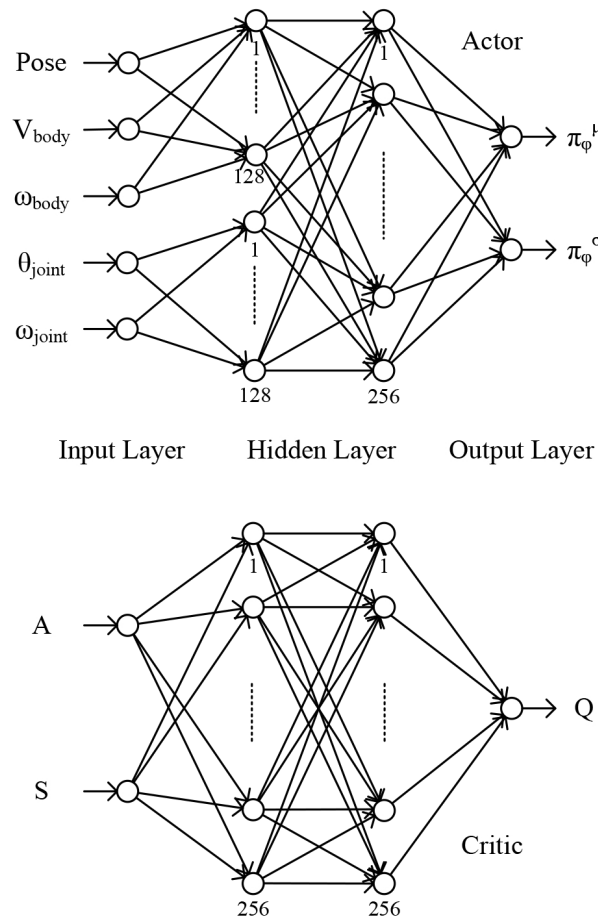


Figure 7. Neural network structure.

The reward function needs to be set according to the specific task and environment. When the hexapod robot is trained to walk on rugged ground, the total reward r_t consists of reward r_v and penalty r_θ , which is given by the expression

$$r_t = \lambda_1 r_v - \lambda_2 r_\theta. \quad (14)$$

The speed reward r_v encourages the robot to move forward, and the closer it moves to the set speed v_{tar} the greater reward it will receive. If the reward function is given as the travel distance to the goal, training results in an unstable or jumpy gait, and this makes the training time much longer. Let the current speed of the robot in a single step be v , and r_v be set to

$$r_v = \frac{1}{|v - v_{tar}| + 1} - \frac{1}{v_{tar} + 1}. \quad (15)$$

To ensure the correctness of the hexapod robot's heading, the penalty r_θ is set to correct the heading. Quaternions are in the complex form of Euler angles, which decompose the rotation of an object into a rotating vector and a rotating angle. If the rotating angle of the hexapod robot is denoted by q_w , then r_θ is set as

$$r_\theta = \sqrt{2 \arccos q_w}. \quad (16)$$

An additional penalty r_e is required for training a hexapod robot to avoid obstacles; utilizing the fact that the pitch and roll angle of robot changes drastically when it is going up a steep slope or going down a steep slope. This change can be sensed by the propriety sensors, which are utilized to allow the robot to avoid specific obstacles. Then, the penalty r_e and total reward r_t' are set as

$$r_e = (4\text{Pitch})^2 + (3\text{Roll})^2, \quad (17)$$

$$r_t' = r_t - r_e. \quad (18)$$

In this framework, the input to the policy network is the state space vector s_t and the output is the mean and standard deviation obeying a Gaussian distribution. The action space vector a_t is then generated using a reparameterization method. The reward r_t and the generated action a_t after evaluating the environment are then fed back to the hexapod robot to enter the next state s_{t+1} . The current state, action, reward, and next state are composed into an experience vector $\{s_t, a_t, r_t, s_{t+1}\}$, which is stored in an experience memory. For training, a small batch of experience samples are randomly selected from the experience memory, to iterate on the network parameters of the online network, and a soft update method is used to iterate on the parameters of the target network and policy network.

5. Testing Results

In this paper, slightly modified SAC, PPO, and TD3 were used to train the hexapod robots during the training process, and the training results are shown in Figure 8. SAC and TD3 had a higher degree of exploration, so the rewards obtained in the pretraining period were much lower than those of PPO. The adaptive parameters introduced by SAC struck a balance between maximizing the expected cumulative reward and minimizing the entropy of the strategy. Adaptive temperature parameters can better adapt to different tasks and environments, and improve the performance of the algorithm. As a result, the rewards obtained after the convergence of SAC in the later stage were higher than for PPO. Figure 8 above shows that TD3 overfitted when trained for this task, and its reward maximum point was lower than with SAC. In summary, the modified SAC training was optimal for this task.

One objective of training a hexapod robot is a fixed walking speed, because the robot needs the right walking speed, no matter what terrain it is on. In this paper, the speed was set to 0.4 m/s for all three terrains of the training, or different speeds could be set separately. The walking speed of the trained hexapod robot is shown in Figure 9. In all three terrains, the robot started moving from rest, so the initial speed was 0 m/s. Then it accelerated to 0.4 m/s after an acceleration process of about 0.4 s, and stayed at about 0.4 m/s. Since all three terrains were rough terrains, the speed kept oscillating around the target speed. There was a strong deceleration process and acceleration process at

around 1.3 s during the robot's movement on the slope. The slowdown was due to the impassable slope, which hindered movement. After being blocked, the hexapod robot gradually changed its direction of movement, bypassed the slope, and accelerated to about 0.4 m/s, to continue walking.

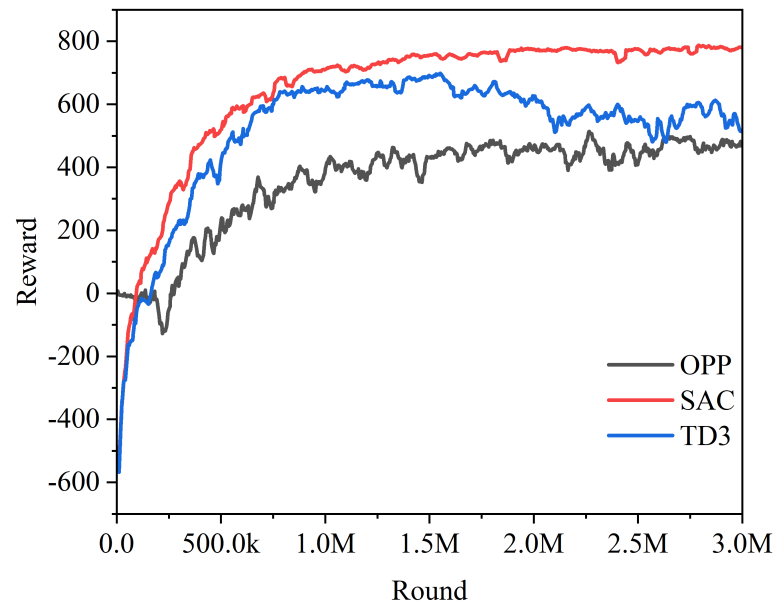


Figure 8. SAC , PPO, and TD3 training curves.

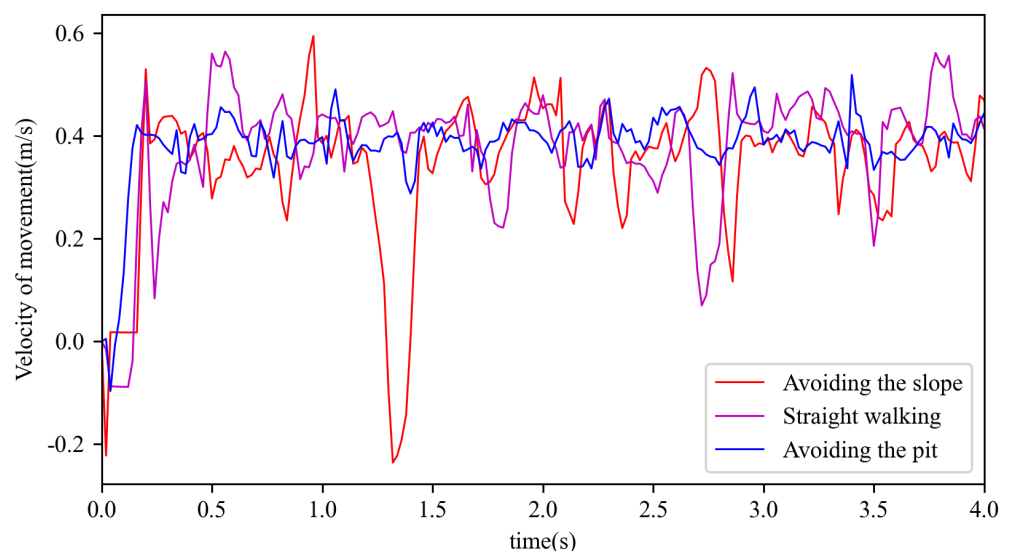


Figure 9. Movement speed of the hexapod robot.

In this paper, the stability of robot motion was judged on the basis of the vertical acceleration, oscillations around roll, and oscillations around pitch during the motion process. The vertical acceleration during the motion on the three terrains is shown in Figure 10, and the vertical acceleration of the robot kept oscillating from around 0 at the beginning of the stationary motion to the uniform motion, and its amplitude mostly kept in the range of 5 mm/s^2 to 10 mm/s^2 . The robot showed a large change at about 0.4 s during the pit movement, since the robot moved to the edge of the pit and changed its direction of movement. The standard deviation for avoiding the slope was 3.05 mm , the standard deviation for straight walking was 3.24 mm , and the standard deviation for avoiding the pit was 2.67 mm . The oscillations around roll during the motion on three terrains are shown in Figure 11. The standard deviation for avoiding the slope was 0.023 rad , the standard

deviation for straight walking was 0.033 rad, and the standard deviation for avoiding the pit was 0.025 rad. The oscillations around pitch during the motion on three terrains is shown in Figure 12. The standard deviation for avoiding the slope was 0.020 rad, the standard deviation for straight walking was 0.031 rad, and the standard deviation for avoiding the pit was 0.020 rad. From these figures, it can be seen that the standard deviations of vertical acceleration, oscillations in pitch, and oscillations in roll were small and the stability was acceptable.

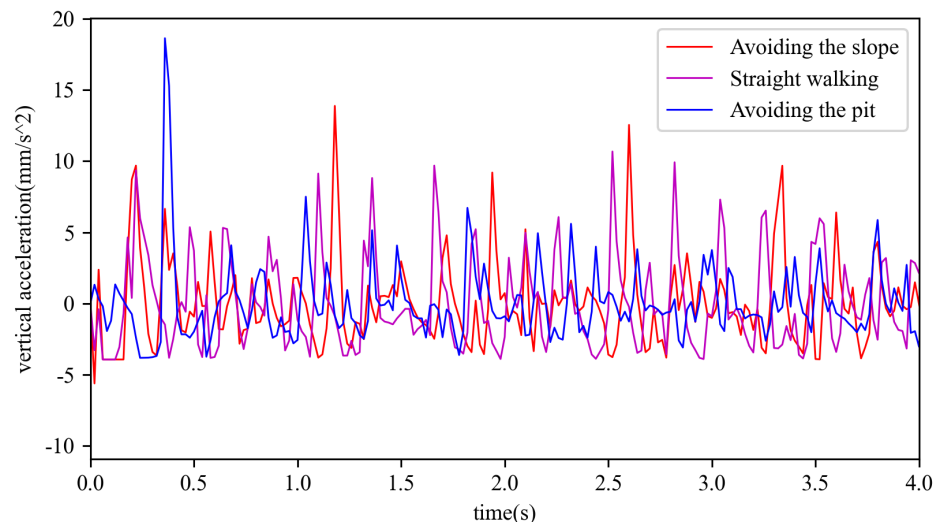


Figure 10. Vertical acceleration of the hexapod robot.

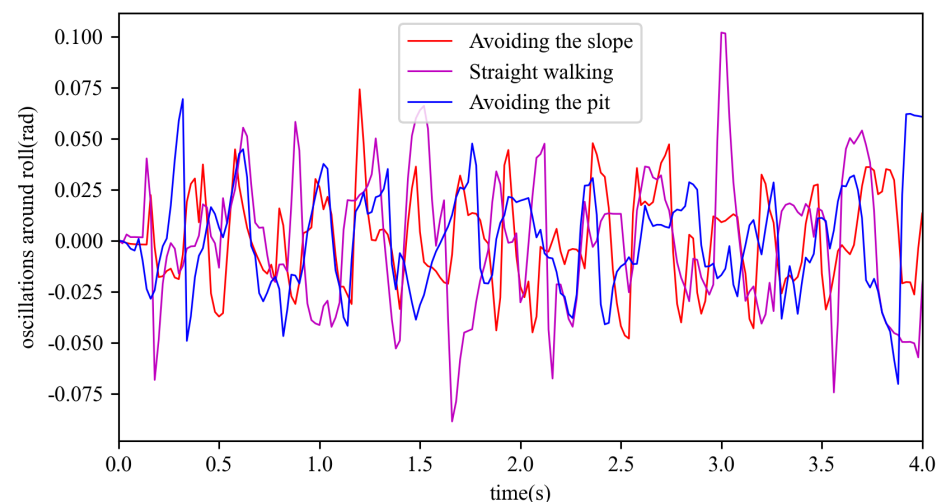


Figure 11. Oscillations around roll.

In this paper, contour maps were used to describe the paths of the hexapod robot moving over the three terrains. The purple curve represents the change in the center of gravity, and the color shading represents the altitude gradient. Figure 13 shows the motion path of the robot on perlin, for which the motion path of the robot was straight within a deviation of ± 25 cm. Figure 14 shows the motion path of the robot when avoiding pits; in the figure, the robot detects the pit at around $x = 22$ cm and then keeps moving towards the edge of the pit without stepping into it. Figure 15 shows the motion path of the robot when avoiding the slope, in which the robot detects the slope at about $x = 30$ cm, and then changes the direction of motion towards the top. In summary, the hexapod robot in this paper has the ability to walk and avoid obstacles in rugged terrain, and Figure 16 shows the performance of the hexapod robot in the simulator.

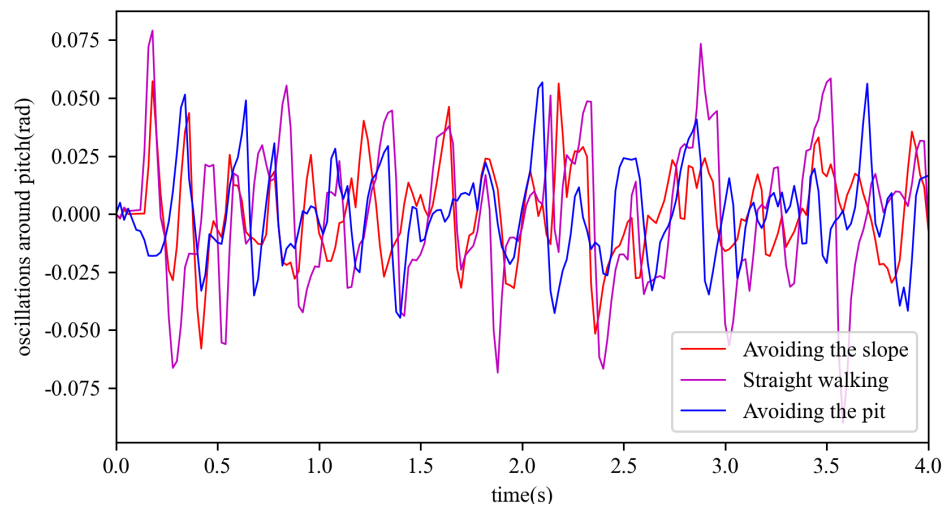


Figure 12. Oscillations around pitch.

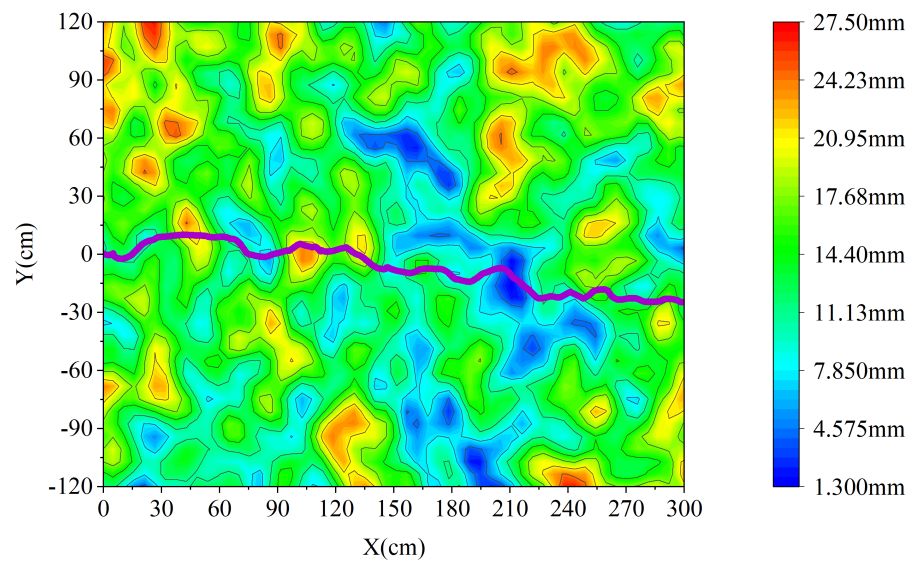


Figure 13. Motion paths of the hexapod robot for perlin.

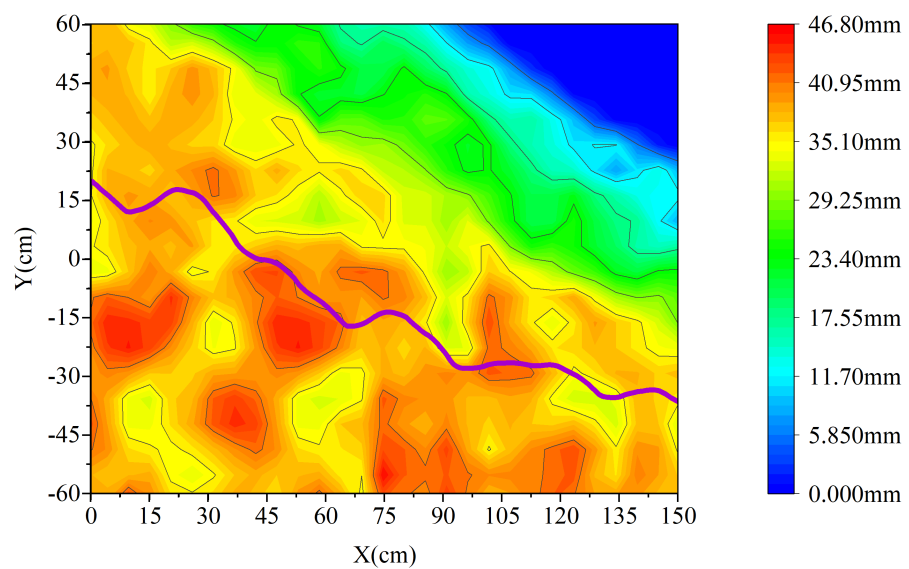


Figure 14. Motion paths of the hexapod robot for pits.

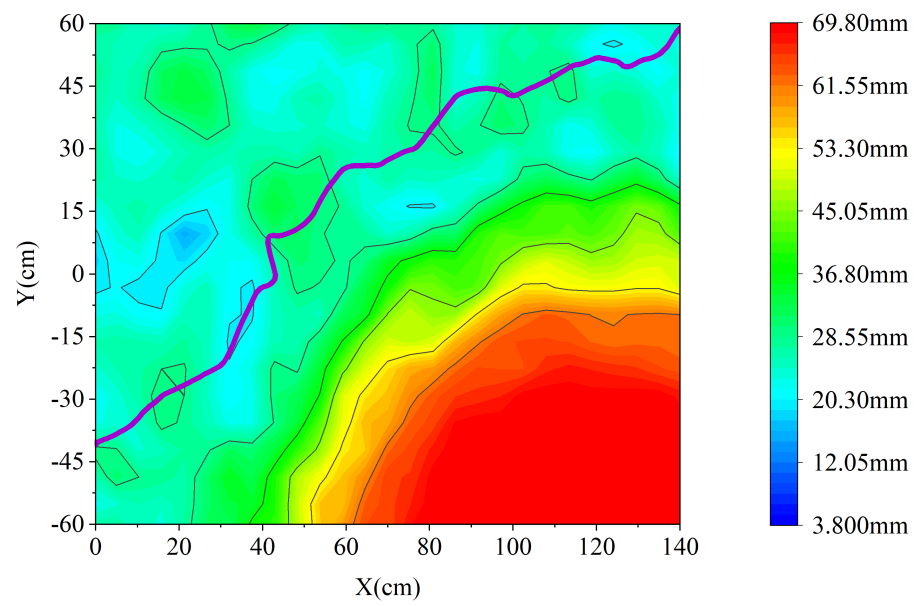


Figure 15. Motion paths of the hexapod robot for slopes.

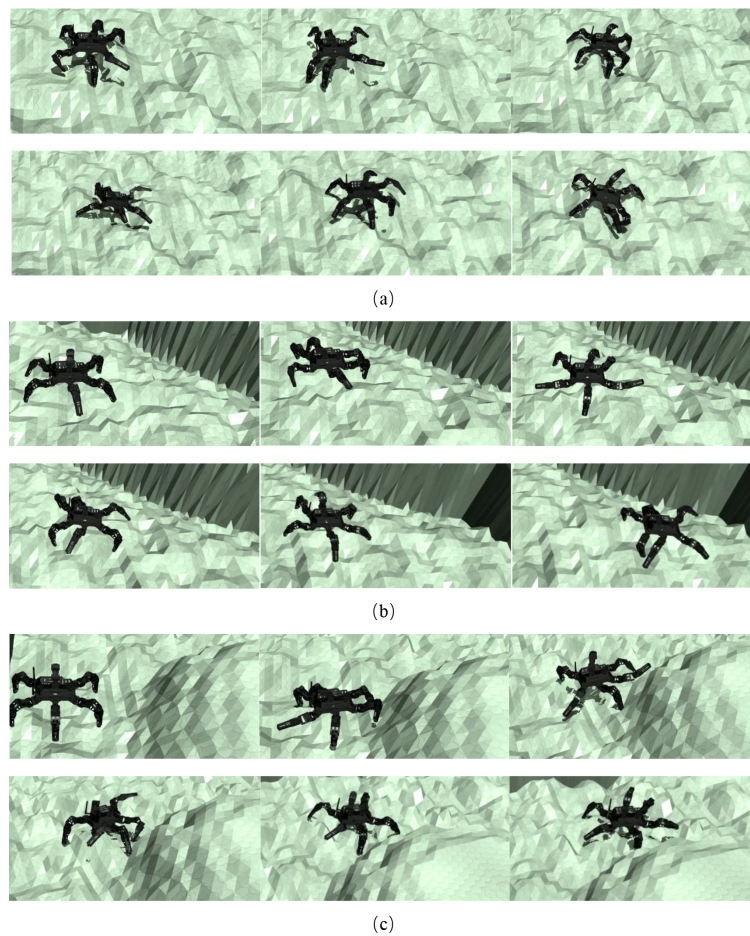


Figure 16. (a) The straight walking process of the hexapod robot. (b) The process of avoiding the valley for the hexapod robot. (c) The process of avoiding hills for the hexapod robot. Github: <https://github.com/rw12123/jethexa>.

6. Conclusions and Future Work

This paper presented a control method for a hexapod robot, which not only allows the hexapod robot to be protected from weather and light while performing tasks in the field, but also to avoid specific obstacles. A modified strategy network actor is used as the robot's motion strategy, which is obtained through SAC training. Due to the different tasks, the corresponding policy networks need to be trained separately, and corresponding physical environments are established, which should be consistent with the actual environment as much as possible. The trained strategy network can only rely on proprioceptive sensory signals to command the hexapod robot to perform adaptive movements, and realize stable walking at a specific speed and pass through the rugged ground, as well as a certain obstacle avoidance ability. This provides a solution for the movement control of hexapod robots in wild mountains. A scheme is provided for the motion control of a hexapod robot in wild mountainous areas.

The method given in this paper requires manual selection of a specific policy network when facing different terrains. In future work, we will train a network so that the robot learns to classify terrain and choose the best motion strategy during movement. The network enables the machine to adapt to a variety of terrain combinations [48,49]. ILC is a control method that optimizes the current system based on historical experience. Optimization of the robot's end-of-foot trajectory, in conjunction with ILC, could be used to improve the robot's locomotion performance [50,51]. In addition, shared control combines human intelligence and robot intelligence, and the framework could fully utilize the advantages of human–robot collaboration to better train the strategy network [52].

Author Contributions: Conceptualization, R.L.; methodology, R.L.; software, R.L.; validation, R.L.; formal analysis, R.L.; investigation, R.L.; resources, R.L., L.W., and Y.C.; data curation, R.L.; writing—original draft preparation, R.L.; writing—review and editing, Y.F., Z.H., L.W., and Y.C.; visualization, R.L.; supervision, L.W. and Y.C.; project administration, R.L., L.W., and Y.C.; funding acquisition, L.W. and Y.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Science Foundation of China (No. 62103293), the Natural Science Foundation of the Higher Education Institutions of Jiangsu Province (No. 22KJB120009), the Natural Science Foundation of Jiangsu Province in China (No. BK20210709), the Wuxi Innovation and Entrepreneurship Fund “Taihu Light” Science and Technology (Fundamental Research) Project under Grant (No. K20221045), the Start-up Fund for Introducing Talent of Wuxi University (No. 2021r045 No. 2023r027), and the Innovative Leading Talents in Universities of Xishan Talents Program (No. 2022xsyc001).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Public Involvement Statement: There was no public involvement in any aspect of this research.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CPG	central pattern generator
RL	reinforcement learning
SAC	soft actor-critic
DRL	deep reinforcement learning
PPO	proximal policy optimization
TD3	two-delay depth deterministic strategy
MDPs	Markov decision processes
PID	proportion integration differentiation

References

1. Dupeyroux, J.; Serres, J.R.; Viollet, S. Antbot: A six-legged walking robot able to home like desert ants in outdoor environments. *Sci. Robot.* **2019**, *4*, 13. [[CrossRef](#)] [[PubMed](#)]
2. Gao, Y.; Wei, W.; Wang, X.; Li, Y.; Wang, D.; Yu, Q. Feasibility, planning and control of ground-wall transition for a suctorial hexapod robot. *Appl. Intell.* **2021**, *51*, 5506–5524. [[CrossRef](#)]
3. Buchanan, R.; Wellhausen, L.; Bjelonic, M.; Bandyopadhyay, T.; Kottege, N.; Hutter, M. Perceptive whole-body planning for multilegged robots in confined spaces. *J. Field Robot.* **2021**, *38*, 68–84. [[CrossRef](#)]
4. Chen, Y.; Cheng, C.; Zhang, Y.; Li, X.; Sun, L. A neural network-based navigation approach for autonomous mobile robot systems. *Appl. Sci.* **2022**, *12*, 7796. [[CrossRef](#)]
5. Lee, G.; Ogata, K.; Li, C. Autonomous shape-variable crawler: One-dimensional displacement coordination for constant upper frame posture. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 14968–14977. [[CrossRef](#)]
6. Song, X.; Zhang, X.; Meng, X.; Chen, C.; Huang, D. Gait optimization of step climbing for a hexapod robot. *J. Field Robot.* **2022**, *39*, 55–68. [[CrossRef](#)]
7. Homchanthanakul, J.; Manoonpong, P. Continuous online adaptation of bioinspired adaptive neuroendocrine control for autonomous walking robots. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 1833–1845. [[CrossRef](#)] [[PubMed](#)]
8. Thor, M.; Manoonpong, P. Versatile modular neural locomotion control with fast learning. *Nat. Mach. Intell.* **2022**, *4*, 169–179. [[CrossRef](#)]
9. Manoonpong, P.; Patane, L.; Xiong, X.; Brodoline, I.; Dupeyroux, J.; Viollet, S.; Arena, P.; Serres, J.R. Insect-inspired robots: Bridging biological and artificial systems. *Sensors* **2021**, *21*, 7609. [[CrossRef](#)]
10. Li, H.; Qi, C.; Gao, F.; Chen, X.; Zhao, Y.; Chen, Z. Mechanism design and workspace analysis of a hexapod robot. *Mech. Mach. Theory* **2022**, *174*, 104917. [[CrossRef](#)]
11. Xu, K.; Lu, Y.; Shi, L.; Li, J.; Wang, S.; Lei, T. Whole-body stability control with high contact redundancy for wheel-legged hexapod robot driving over rough terrain. *Mech. Mach. Theory* **2023**, *181*, 105199. [[CrossRef](#)]
12. Li, H.; Qi, C.; Mao, L.; Zhao, Y.; Chen, X.; Gao, F. Staircase-climbing capability-based dimension design of a hexapod robot. *Mech. Mach. Theory* **2021**, *164*, 104400. [[CrossRef](#)]
13. Yao, S.; Yao, Y.-A.; Liu, R.; Liu, C. A portable off-road crawling hexapod robot. *J. Field Robot.* **2022**, *39*, 739–762. [[CrossRef](#)]
14. Rastgar, H.; Naeimi, H.R.; Agheli, M. Characterization, validation, and stability analysis of maximized reachable workspace of radially symmetric hexapod machines. *Mech. Mach. Theory* **2019**, *137*, 315–335. [[CrossRef](#)]
15. Zhang, G.; Ma, S.; Liu, J.; Zeng, X.; Kong, L.; Li, Y. Q-whex: A simple and highly mobile quasi-wheeled hexapod robot. *J. Field Robot.* **2023**, *40*, 1444–1459. [[CrossRef](#)]
16. Orozco-Magdaleno, E.C.; Gomez-Bravo, F.; Castillo-Castaneda, E.; Carbone, G. Evaluation of locomotion performances for a mecanum-wheeled hybrid hexapod robot. *IEEE-Asme Trans. Mechatron.* **2021**, *26*, 1657–1667. [[CrossRef](#)]
17. Chen, Z.; Wang, S.; Wang, J.; Xu, K.; Lei, T.; Zhang, H.; Wang, X.; Liu, D.; Si, J. Control strategy of stable walking for a hexapod wheel-legged robot. *ISA Trans.* **2021**, *108*, 367–380. [[CrossRef](#)]
18. Larsen, A.D.; Buescher, T.H.; Chuthong, T.; Pairam, T.; Bethge, H.; Gorb, S.N.; Manoonpong, P. Self-organized stick insect-like locomotion under decentralized adaptive neural control: From biological investigation to robot simulation. *Adv. Theory Simul.* **2023**, *2023*, 2300228. [[CrossRef](#)]
19. Dupeyroux, J.; Passault, G.; Ruffier, F.; Stéphane, V.; Serres, J. Hexabot: A small 3D-printed six-legged walking robot designed for desert ant-like navigation tasks. In Proceedings of the IFAC World Congress 2017, Toulouse, France, 9–14 July 2017.
20. Chen, Y.; Zhou, Y. Machine learning based decision making for time varying systems: Parameter estimation and performance optimization. *Knowl.-Based Syst.* **2020**, *190*, 105479. [[CrossRef](#)]
21. Wang, L.; Dong, L.; Huangfu, Z.; Chen, Y. A fuzzy neural network controller using compromise features for timeliness problem. *IEEE Access* **2023**, *11*, 17650–17657. [[CrossRef](#)]
22. Gao, Z.; Shi, Q.; Fukuda, T.; Li, C.; Huang, Q. An overview of biomimetic robots with animal behaviors. *Neurocomputing* **2019**, *332*, 339–350. [[CrossRef](#)]
23. Gutierrez-Galan, D.; Dominguez-Morales, J.P.; Perez-Pena, F.; Jimenez-Fernandez, A.; Linares-Barranco, A. Neuropod: A real-time neuromorphic spiking cpg applied to robotics. *Neurocomputing* **2020**, *381*, 10–19. [[CrossRef](#)]
24. Ma, F.; Yan, W.; Chen, L.; Cui, R. Cpg-based motion planning of hybrid underwater hexapod robot for wall climbing and transition. *IEEE Robot. Autom. Lett.* **2022**, *7*, 12299–12306. [[CrossRef](#)]
25. Bal, C. Neural coupled central pattern generator based smooth gait transition of a biomimetic hexapod robot. *Neurocomputing* **2021**, *420*, 210–226. [[CrossRef](#)]
26. Barrio, R.; Lozano, A.; Martinez, M.A.; Rodriguez, M.; Serrano, S. Routes to tripod gait movement in hexapods. *Neurocomputing* **2021**, *461*, 679–695. [[CrossRef](#)]
27. Wei, J.; Tao, G.; Zhang, J.; Yuan, L. Foot trajectory following control of hexapod robot based on udwadia-kalaba theory. *Nonlinear Dyn.* **2023**, *111*, 14055–14075. [[CrossRef](#)]
28. Chen, Y.; Chu, B.; Freeman, C.T. Iterative learning control for robotic path following with trial-varying motion profiles. *IEEE/ASME Trans. Mechatron.* **2022**, *27*, 4697–4706. [[CrossRef](#)]
29. Gao, Y.; Wang, D.; Wei, W.; Yu, Q.; Liu, X.; Wei, Y. Constrained predictive tracking control for unmanned hexapod robot with tripod gait. *Drones* **2022**, *6*, 246. [[CrossRef](#)]

30. Sartoretti, G.; Paivine, W.; Shi, Y.; Wu, Y.; Choset, H. Distributed learning of decentralized control policies for articulated mobile robots. *IEEE Trans. Robot.* **2019**, *35*, 1109–1122. [[CrossRef](#)]
31. Lele, A.S.; Fang, Y.; Ting, J.; Raychowdhury, A. Learning to walk: Bio-mimetic hexapod locomotion via reinforcement-based spiking central pattern generation. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **2020**, *10*, 536–545. [[CrossRef](#)]
32. Sun, C.; Yang, G.; Yao, S.; Liu, Q.; Wang, J.; Xiao, X. Rhex-t3: A transformable hexapod robot with ladder climbing function. *IEEE/ASME Trans. Mechatron.* **2023**, *28*, 1939–1947. [[CrossRef](#)]
33. Schilling, M.; Paskarbit, J.; Ritter, H.; Schneider, A.; Cruse, H. From adaptive locomotion to predictive action selection—Cognitive control for a six-legged walker. *IEEE Trans. Robot.* **2022**, *38*, 666–682. [[CrossRef](#)]
34. Cully, A.; Clune, J.; Tarapore, D.; Mouret, J.-B. Robots that can adapt like animals. *Nature* **2015**, *521*, 503–507. [[CrossRef](#)] [[PubMed](#)]
35. Wu, X.A.; Huh, T.M.; Sabin, A.; Suresh, S.A.; Cutkosky, M.R. Tactile sensing and terrain-based gait control for small legged robots. *IEEE Trans. Robot.* **2020**, *36*, 15–27. [[CrossRef](#)]
36. Azayev, T.; Zimmerman, K. Blind hexapod locomotion in complex terrain with gait adaptation using deep reinforcement learning and classification. *J. Intell. Robot. Syst.* **2020**, *99*, 659–671. [[CrossRef](#)]
37. Xu, P.; Ding, L.; Li, Z.; Yang, H.; Wang, Z.; Gao, H.; Zhou, R.; Su, Y.; Deng, Z.; Huang, Y. Learning physical characteristics like animals for legged robots. *Natl. Sci. Rev.* **2023**, *10*, nwad045. [[CrossRef](#)] [[PubMed](#)]
38. Ngamkajornwiwat, P.; Homchanthanakul, J.; Teerakittikul, P.; Manoonpong, P. Bio-inspired adaptive locomotion control system for online adaptation of a walking robot on complex terrains. *IEEE Access* **2020**, *8*, 91587–91602. [[CrossRef](#)]
39. Sun, G.; Sartoretti, G. Joint-space cpg for safe foothold planning and body pose control during locomotion and climbing. *IEEE Robot. Autom. Lett.* **2022**, *7*, 9889–9896. [[CrossRef](#)]
40. Chen, P.; Lu, W. Deep reinforcement learning based moving object grasping. *Inf. Sci.* **2021**, *565*, 62–76. [[CrossRef](#)]
41. Han, J.; Waddington, G.; Adams, R.; Anson, J.; Liu, Y. Assessing proprioception: A critical review of methods. *J. Sport Health Sci.* **2016**, *5*, 80–90. [[CrossRef](#)]
42. Qu, X.; Hu, X.; Zhao, J.; Zhao, Z. The roles of lower-limb joint proprioception in postural control during gait. *Appl. Ergon.* **2022**, *99*, 103635. [[CrossRef](#)] [[PubMed](#)]
43. Zang, W.; Song, D. Energy-saving profile optimization for underwater glider sampling: The soft actor critic method. *Measurement* **2023**, *217*, 113008. [[CrossRef](#)]
44. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. *arXiv* **2018**, arXiv:1801.01290.
45. Deng, L.; Li, S.; Tang, X.; Yang, K.; Lin, X. Battery thermal- and cabin comfort-aware collaborative energy management for plug-in fuel cell electric vehicles based on the soft actor-critic algorithm. *Energy Convers. Manag.* **2023**, *283*, 116889. [[CrossRef](#)]
46. Haarnoja, T.; Zhou, A.; Ha, S.; Tan, J.; Tucker, G.; Levine, S. Learning to walk via deep reinforcement learning. *arXiv* **2018**, arXiv:1812.11103.
47. Todorov, E.; Erez, T.; Tassa, Y. Mujoco: A physics engine for model-based control. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Algarve, Portugal, 7–12 October 2012; pp. 5026–5033.
48. Yang, C.; Weng, G.; Chen, Y. Active contour model based on local kullback-leibler divergence for fast image segmentation. *Eng. Artif. Intell.* **2023**, *123*, 106472. [[CrossRef](#)]
49. Li, M.; Wang, Z.; Zhang, D.; Jiao, X.; Wang, J.; Zhang, M. Accurate perception and representation of rough terrain for a hexapod robot by analysing foot locomotion. *Measurement* **2022**, *193*, 110904. [[CrossRef](#)]
50. Chen, Y.; Chu, B.; Freeman, C.T. Generalized iterative learning control using successive projection: Algorithm, convergence, and experimental verification. *IEEE Trans. Control Syst. Technol.* **2020**, *28*, 2079–2091. [[CrossRef](#)]
51. Zhuang, Z.; Tao, H.; Chen, Y.; Stojanovic, V.; Paszke, W. An optimal iterative learning control approach for linear systems with nonuniform trial lengths under input constraints. *IEEE Trans. Syst. Man Cybern. Syst.* **2023**, *53*, 3461–3473. [[CrossRef](#)]
52. Xu, P.; Wang, Z.; Ding, L.; Li, Z.; Shi, J.; Gao, H.; Liu, G.; Huang, Y. A closed-loop shared control framework for legged robots. *IEEE/ASME Trans. Mechatron.* **2023**, early access.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.