International Journal of Computer Science and Information Technology
ISSN: 3005-9682 (Print), ISSN: 3005-7140 (Online) | Volume 2, Number 1, Year 2024
DOI: https://doi.org/10.62051/ijcsit.v2n1.44
Journal homepage: https://wepub.org/index.php/IJCSIT/index

Gait Learning for Hexapod Robot Facing Rough Terrain Based on Dueling-DQN Algorithm

Liuhongxu Chen, Ping Du, Pengfei Zhan and Bo Xie*

School of Computer Science and Engineering, Sichuan University of Science and Engineering, Zigong 643000, China

ABSTRACT

In the handling of dangerous goods in explosive environments, robots are increasingly being used instead of human operators. Robots designed for operation in explosive environments are mostly equipped with tracked structures, which, due to their limited terrain adaptability, struggle to movement rugged landscapes. Hexapod robots, with their excellent maneuverability and adaptability, possess advantages in completing hazardous material handling tasks in such rugged terrains. One current challenge lies in enabling hexapod robots to autonomously adjust their gaits to cope with rugged terrain. This paper proposes a gait learning method based on the Dueling Deep Q-Network (Dueling-DQN) algorithm to address the gait adjustment problem of hexapod robots in sloped, terraced, and rugged terrain. The method combines lidar data and reinforcement learning to extract features from the lidar data to determine terrain types and foot coordinates. Finally, the Dueling-DQN algorithm and redundant phase strategy are employed to facilitate the motion of hexapod robots in these three types of rugged terrain. Simulation and prototype experiments are conducted to evaluate the Dueling-DQN algorithm's performance in terms of rewards and stability margins for the three types of terrain. During algorithm training on sloped, terraced, and rugged terrain, stable rewards and positive stability margins are achieved after approximately 490 iterations. The effectiveness and feasibility of the proposed method are further validated through Gazebo simulation and prototype experiments. In the context of movement in rugged terrain within explosive environments, the gait learning method based on the Dueling-DQN algorithm offers valuable insights into the control of hexapod robots.

KEYWORDS

Gait Planning; Reinforcement Learning; Hexapod Robot

1. INTRODUCTION

Explosive environments are commonly found in petrochemical plants, coal mines, and oil and gas extraction sites [1], where conditions are often complex and hazardous. On one hand, damaged structures may lead to unstable and confined spaces, making it difficult for rescue personnel to access the site. On the other hand, the presence of combustible and explosive materials, such as deactivated explosives, poses a permanent threat of injury to rescue personnel. Utilizing robots instead of human workers for these tasks can mitigate these risks [2]. Therefore, selecting suitable robots and devising methods for their movement in rugged terrain are essential for handling dangerous goods in explosive environments. Hexapod robots offer higher degrees of freedom as they require only discrete footholds during locomotion. Moreover, they demonstrate greater adaptability to rugged terrain compared to wheeled and tracked robots [3], making them more suitable for handling dangerous goods in explosive environments. Given the prevalence of rugged terrain in explosive environments, such as sloped, terraced, and rugged terrain, it is necessary to develop gaits for hexapod robots based on the

environment. The autonomous formation of gaits by hexapod robots to navigate sloped, terraced, and rugged terrain is currently a pressing issue. In recent years, various gait control methods have been proposed, categorized into three types: imitation learning, trend function, and reinforcement learning. Imitation learning involves improving hexapod robot gaits based on given action sequences. For instance, Hou [4] combined gait recognition with existing sequences and used convolutional neural networks to develop required gait profiles. Zhu Xiaoqing [5] proposed a reward-guided robot imitation gait learning algorithm. Such methods require manual gait template development based on terrain types, necessitating re-establishment for different terrains. Trend functions involve controlling hexapod robot footholds and gait types through designated functions. Cizek P [6] used position feedback from intelligent servo motors to sense ground reaction forces, allowing small hexapod robots to walk stably in rugged terrain. Gao Y [7] proposed an omnidirectional tracking strategy based on model predictive control and real-time replanning. Such methods rely on mathematical modeling of the environment and can only select fixed gait movements. Reinforcement learning, represented by algorithms such as Q-Learning and DQN, is based on environmental feedback. Liu [8] adopted an improved DQN algorithm-based free gait planning method. Youcef [9] proposed a distributed Q-Learning method. Ouyang Y [10] found that Dueling Deep Q-Network (Dueling-DQN) performs better in decision-making problems. To overcome the difficulty of wheeled and tracked robots in autonomously moving in rugged terrain in explosive environments, this paper proposes a method using the Dueling-DQN algorithm combined with hexapod robots to autonomously learn gaits in rugged terrain. Environmental data is extracted using laser radar to identify terrain and obtain foot coordinate ranges. The Dueling-DQN algorithm is used to learn the optimal foothold for each leg based on the terrain. Redundant strategies are employed to select the phases of hexapod robots to ensure stable robot body. The effectiveness and feasibility of the proposed method are validated through simulation and hexapod robot prototype experiments in sloped, terraced, and rugged terrain.

2. GAIT LEARNING BASED ON THE DUELING-DQN ALGORITHM

This section outlines how to employ the Dueling-DQN algorithm for gait learning in hexapod robots. Section 2.1 introduces the data processing methods for sloped, terraced, and rugged terrain using laser radar. Section 2.2 elaborates on the construction of the Dueling-DQN algorithm. Section 2.3 explains the establishment of the gait planner.

2.1. Terrain Feature Extraction Based on Lidar

Each type of rugged terrain possesses distinct characteristics, necessitating the design of corresponding terrain data processing methods for each terrain type.

For terraced terrain, as illustrated in Figure 1, edge points are sought in the planar data. Initially, the point with the maximum angle, denoted as p_u, is identified among the planar feature points after K scans. Subsequently, the nearest two points, p_j and p_k, are located within the point cloud set. These two points must belong to the same scan as the planar feature point. Their C value is computed according to equation (1) to determine if they lie on the plane. If so, using p_j as the reference, the search continues within the point cloud set until p_q does not fall on the plane. At this point, the Z-axis value of p_j is calculated, and this value, when increased by the distance between the laser radar and the ground, yields the height of the terrain during the K scans.

$$H_{ob}^{k} = Z_k + h_{liadar} \tag{1}$$

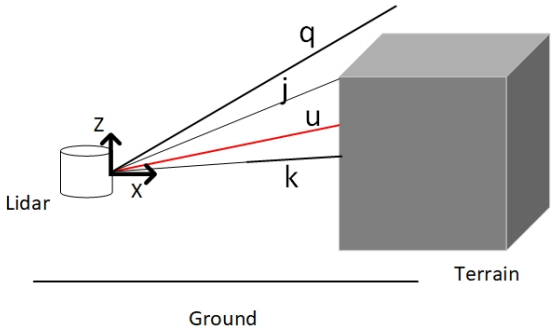

Fig. 1 Upper ladder-shaped obstacle data processing

The radar is mounted on the hexapod robot, and it measures the terrain height during movement to avoid height measurement errors. When the distance between the radar and the terrain reaches a preset distance threshold, the calculation stops. Finally, the average height of the terrain is computed.

$$H_{ob} = \sum_{n}^{k} \frac{H_{ob}^{i}}{n} \tag{2}$$

When the H_{ob} value is less than the threshold of the maximum range of motion in the Z-axis direction of the hexapod robot's chassis, the terrain is labeled as a terraced terrain for traversal. Otherwise, it is labeled as terrain that cannot be traversed. For processing data related to rugged terrain, such as the scenario depicted in Figure 2, suppose during the Kth scan in the moving process, the Z-axis value of a point cloud data point P_a is h_a, and the Z-axis value of its nearest point P_b is h_b. The difference between the two points is computed as follows:

$$H_k = |h_a - h_b| \tag{3}$$

When the H_k value is greater than the value of H_k in the $K-4th$ scan, the point data is marked as a candidate value, and the X-axis value of P_b is calculated. This operation is repeated for each frame of data during motion. The coordinates of all candidate points are averaged in the X, Y, and Z directions to determine the coordinates of point A.

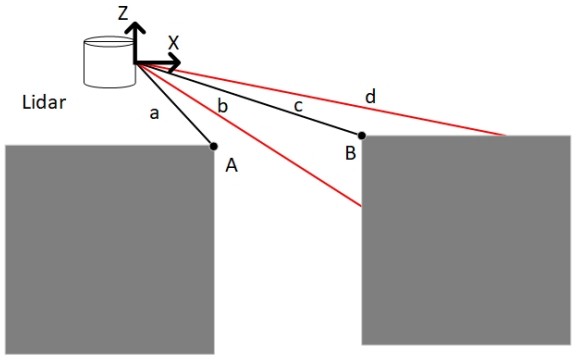

Fig. 2 Gully obstacle data processing

Similarly, to determine the coordinates of point B, we select point cloud data where there is no change in the Z-axis direction. Based on the difference in the X-axis values of P_c and P_d, compared to the difference in the X-axis values of P_c and P_d in the $K-4st$ scan, if the former is greater, point B is marked. By subtracting the X-axis and Z-axis coordinates of points A and B, respectively, we can obtain the distance and height difference of the ravine. When both feature points A and B exist in multiple scans and there is a negative spike in the Z-axis direction, the terrain at this location is marked as a ravine. If the length of the ravine is greater than the sum of the movement lengths of the

foot end and hind leg in the X direction of the robot, the ravine is marked as impassable; otherwise, it is marked as passable.

If only point A exists after multiple scans and there is a negative spike in the Z-axis direction, the terrain at this location is marked as a descending staircase, as shown in Figure 3. By selecting the neighboring points of point A and obtaining their values in the Z-axis direction, we ultimately determine the height of the descending staircase terrain.

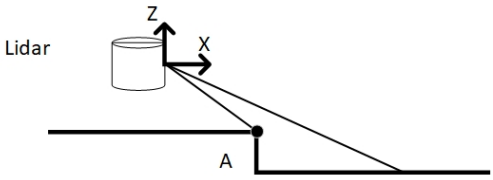

Fig. 3 Lower ladder-shaped obstacle data processing

For sloped terrain, the determination mainly relies on the variation in the Z-axis direction within the point cloud set, as depicted in Figure 4. Suppose, during the Kth scan, we search for three adjacent points: j, k, and l. The rate of change in Z-axis values between j and k, denoted as ΔZ_1, and between k and l, denoted as ΔZ_2, is computed to calculate the confidence value:

$$Confidence = \left|\Delta Z_1 - \Delta Z_2\right| \tag{4}$$

If $Confidence$ is less than 1, this operation is repeated in consecutive three scans. If the results of the three scans remain unchanged, all adjacent points j, k, and l in the three scans are selected, and the terrain is labeled as sloped terrain. Then, the slope angle of the terrain is calculated.

$$\bar{\theta} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{2}\left(arc\tan(\frac{z_k - z_l}{x_k - z_l}) + arc\tan(\frac{z_j - z_k}{x_j - z_k})\right)\right) \tag{5}$$

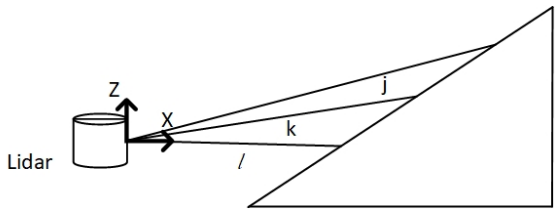

Fig. 4 Slope obstacle data processing

If the slope angle ($\bar{\theta}$) is greater than the maximum slope angle set for the hexapod robot, it is marked as a static obstacle; otherwise, it is marked as traversable sloped terrain.

Regarding foot-end data processing, the laser radar point cloud data is processed to construct a grid range centered on the initial posture, as illustrated in Figure 5. The entire grid is a square with sides equal to the maximum movement limit length of a single leg base axis, denoted as L_{sg}.

Time intervals are employed to collect data points that fall within the grid. Assuming the swing phase and support phase of a single leg of the hexapod robot are denoted as t_{sup}^{i} and t_{oup}^{i}, respectively, the entire cycle of a single leg is:

$$T_i = t_{\sup}^{i} + t_{oup}^{i} \tag{6}$$

The point cloud data collected within the selected range during the T_i cycle will be updated after the completion of the leg movement cycle. In cases where some point cloud data falls on the boundary lines of the grid divisions, measures are taken to discard them to avoid interference in subsequent gait learning.

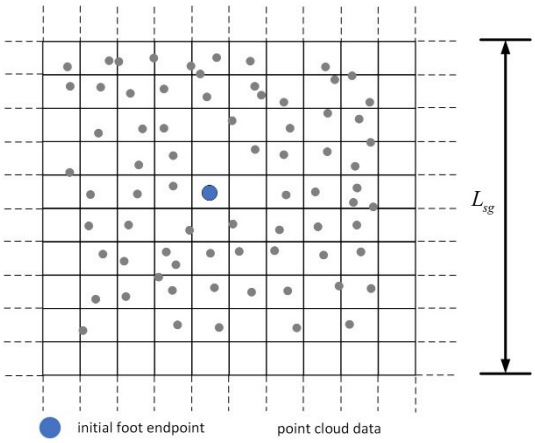

Fig. 5 Schematic diagram of the grid of foothold range

2.2. Dueling-DQN Gait Learning Algorithm

Dueling-DQN (Dueling Deep Q-Network) [14] is an algorithm that combines the Deep Q-Network (DQN) with the Dueling network architecture, aiming to enhance the performance of Q-learning in reinforcement learning tasks. The main idea behind Dueling-DQN is to decompose the Q-value function into a state value function and an advantage function. This allows the network to independently learn the value of states and the advantage of state-action pairs, thereby enabling more effective policy evaluation and optimization.

First, let's decompose the Q-function:

$$Q(s,a) = V(s) + A(s,a) - \frac{1}{|\mathcal{A}|}\sum_{a'} A(s,a') \tag{7}$$

Where $Q(s,a)$ represents the Q-value of action a taken in state s, $V(s)$ represents the value of state s, $A(s,a)$ represents the advantage of taking action a in state s, and $\mathcal{A}$ represents the set of all possible actions. The state value $V(s)$ is computed through the state value branch, while the advantage $A(s,a)$ is computed through the advantage branch. The formula is given by:

$$A(s,a) = Q(s,a) - V(s) \tag{8}$$

Using the Huber loss function, the error for the state value and advantage is calculated as follows:

$$L(\theta) = \frac{1}{N}\sum_{i}\left(r_i + \gamma \max_{a'} Q(s_{i+1}, a'; \theta^{-}) - Q(s_i, a_i; \theta)\right)^2 \tag{9}$$

Where $L(\theta)$ represents the loss function of the network, N represents the number of samples in the experience replay buffer, r_i denotes the reward at time step i, γ represents the discount factor, s_i represents the state at time step i, a_i denotes the action chosen at state s_i, θ represents the parameters of the neural network, and θ^{-} represents the parameters of the target network.

2.2.1. State space

Firstly, for the selection of point cloud data, the foot-end coordinate grid is evenly divided into a 10x10 grid, assuming a grid length of 10mm, as illustrated in Figure 6. When no terrain is identified, the densest points with the lowest elevation differences are selected for bounding. The point closest to the center of the circle is chosen as the coordinate data candidate point for the grid. If multiple points appear, one is randomly selected. The same process is applied to the small grids within the selected grid of the foot-end coordinates to complete the selection range of the single foot's coordinate points.

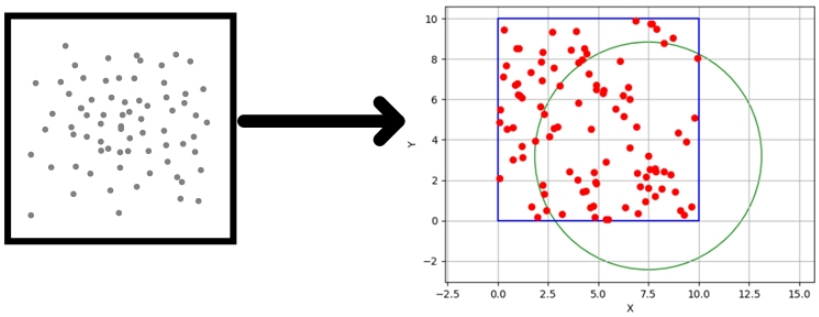

Fig. 6 Schematic diagram of grid screening candidate coordinates

It's essential to note that when rugged terrain is identified, the candidate points need to be adjusted flexibly based on the terrain. In the case of ascending and descending staircase terrains, priority is given to selecting point cloud data points that fall within the feature point range and the movement threshold range as candidate points. For ravine terrain, priority is given to selecting point cloud data points within the feature point range and within the movable threshold range as candidate points.

Regarding the foot-end state space, it mainly involves spatial mapping. The grid is divided into three matrices representing the range of foot-end coordinates, the X-coordinate, Y-coordinate, and Z-coordinate of the point cloud data. Each matrix has a size of 10x10, ultimately forming the space E of each foot-end.

$$E_i = \left(X_i, Y_i, Z_i\right) \tag{10}$$

The overall foot-end coordinate landing points are composed of the coordinates of the six legs combined.

$$M = \left(E_1, E_2, E_3, E_4, E_5, E_6\right) \tag{11}$$

The state space S mainly comprises external and internal information. When learning gaits on rugged terrain, it's necessary to consider the phase distribution of the six legs and the coordinates of the foot-end landing points. The state S is as follows:

$$S = \left(M, \tau, P_{target}\right) \tag{12}$$

Where τ represents the parameters of the hexapod robot's body, and P_{target} represents the target area of the body. τ includes:

$$\tau = \left(P_c, \kappa_\alpha^c, P, D\right) \tag{13}$$

P_c represents the centroid coordinates of the robot, κ_α^c represents the Euler angles of the robot itself, P represents the current foot-end coordinates of the robot, and D represents the current phase status of the robot.

2.2.2. Action space settings

The action space of the hexapod robot at time t can be represented as:

$$a_t = (Phase, P_k) \tag{14}$$

Where $Phase$ represents the phase value, P_k represents the coordinates of the next foot-end landing point. When the hexapod robot performs an action, it receives feedback from the environment and obtains information about the next state S_{t+1}.

2.2.3. Reward settings

The reward value is an indicator of the goodness or badness of a state. The composition of the reward value is as follows:

$$r(s,a) = r_m(s,a) + r_z(s,a) + r_s(s,a) \tag{15}$$

The reward value (r_m) is compared with the range of motion of the hexapod robot's foot-end movement based on feedback from the environment. If the selected foot-end coordinate point falls within the range of motion of the hexapod robot, a reward value is given; otherwise, a penalty value is given. r_z is a reward value based on centroid changes, determined by the forward distance traveled by the hexapod robot. r_s, on the other hand, is assessed based on stability margin. If the stability margin is less than 0, a penalty is given.

2.2.4. Algorithm process

The input to the algorithm consists of 10x10 matrices of the foot-end landing point selection range's X-axis, Y-axis, and Z-axis for each foot-end, while the output consists of the X, Y, and Z coordinates for each foot-end. The pseudo-code for the algorithm is as shown in Table 1.

Table 1 Dueling-DQN algorithm pseudo code

The pseudo-code for the Dueling-DQN algorithm
Initialize neural network Q and target network Q_target
Initialize the experience playback buffer ReplayBuffer
Set hyperparameters: learning rate lr, experience playback buffer size buffer_size, batch size batch_size, ε-greedy policy parameters, etc.
for episode in range(num_episodes):
Initialize environment state state
Initialize the total reward total_reward to 0
while not termination status:
Use ε-greedy strategy to select actions
Execute the action and observe the environment feedback reward and next state next_state
Store the transition (state, action, reward, next_state) into the experience replay buffer ReplayBuffer
Randomly sample batch data from ReplayBuffer (batch_size)
Calculate the target Q value target_q_values:
For each sample (state, action, reward, next_state):
Use the target network Q_target to predict the maximum action value next_max_action_value of the next state
Use the current network Q to predict the action values of the next state next_state_values
Calculate the state value function value_function and advantage function advantage_function
Calculate the final Q value
Calculate the loss function loss, usually the MSE loss function
Updating the parameters of neural network Q using gradient descent
Update the current state to the next state state = next_state
Update total reward total_reward
Update target network parameters

2.3. Gait planner

2.3.1. Phase selection

For the free change of phase in the gait learning of a hexapod robot, which guarantees the flexibility of the hexapod robot [15], the gait phase needs to be defined first:

$$Phase = [s_1, s_2, s_3, s_4, s_5, s_6] \tag{16}$$

The phase status s_i represents the phase of each foot-end. If s_i is 0, it indicates that the foot-end is in the support phase, while if s_i is 1, it indicates that the foot-end is in the swing phase. Since each leg has two states, there are a total of 64 phase configurations. To ensure the stability of the hexapod robot, it is required that there are no fewer than 3 support phases. Therefore, there are 40 possible phase configurations, as shown in Table 2.

Table 2 Phase selection table

Num/ Leg	phase state (Support Phase: 0 Swing Term: 1)						Num /Leg	phase state (Support Phase: 0 Swing Term: 1)					
	1	2	3	4	5	6		1	2	3	4	5	6
1	1	1	0	1	0	0	21	0	1	0	1	0	1
2	1	1	0	0	1	0	22	0	1	0	1	0	0
3	1	1	0	0	0	1	23	0	1	0	0	1	1
4	1	1	0	0	0	0	24	0	1	0	0	1	0
5	1	0	1	1	0	0	25	0	1	0	0	0	1
6	1	0	1	0	1	0	26	0	1	0	0	0	0
7	1	0	1	0	0	1	27	0	0	1	1	1	0
8	1	0	1	0	0	0	28	0	0	1	1	0	1
9	1	0	0	1	1	0	29	0	0	1	1	0	0
10	1	0	0	1	0	1	30	0	0	1	0	0	0
11	1	0	0	1	0	0	31	0	0	1	0	1	0
12	1	0	0	0	1	1	32	0	0	1	0	0	1
13	1	0	0	0	1	0	33	0	0	1	0	0	0
14	1	0	0	0	0	1	34	0	0	0	1	1	0
15	1	0	0	0	0	0	35	0	0	0	1	1	1
16	0	1	1	1	0	0	36	0	0	0	1	0	0
17	0	1	1	0	1	0	37	0	0	0	0	1	1
18	0	1	1	0	0	1	38	0	0	0	0	1	0
19	0	1	1	0	0	0	39	0	0	0	0	0	1
20	0	1	0	1	1	0	40	0	0	0	0	0	0

According to the selection of foot-end landing points, the phase is chosen. Since the foot-end landing points mainly pertain to the swing phase, it's necessary to first confirm the leg numbers corresponding to the swing phase. Then, it needs to be determined whether there is redundant space in the leg transitioning from the swing phase to the support phase to support the movement of the next swing phase item. Assuming in the phase selection, support phase $p_{support}(s_1, s_3, s_5)$, swing phase $p_{swing}(s_2, s_4, s_6)$, the redundancy of each foot-end is calculated as follows:

$$redundancy = \phi_{swing}\left(E > p_{swing}^{j} \mid M_j\right) \tag{17}$$

The ϕ_{swing} represents the range of foot-end landing point coordinates in the support phase. It is divided into redundant space in the direction of motion towards the centroid, with the foot-end coordinates when the current swing phase item is landed as the reference. If any of these spaces have $redundancy$ lower than a threshold, priority is given to the tripod gait.

2.3.2. Foot placement selection

For the three types of rugged terrain, it's necessary to correspondingly match the three foot landing point models. When encountering the corresponding terrain, flexibly switch to the corresponding algorithm model. If none of the three terrains are encountered, the tripod gait is used for movement,

combined with the use of inverse kinematics to calculate the rotational joint angle values for each leg based on the landing point coordinates.

2.3.3. Feasibility analysis of foot landing point

The main focus is to assess whether there is collision between the lower leg and the terrain after the foot has landed. By employing the robot's forward kinematics [16], the coordinates of the robot's thigh $(pl_x[i], pl_y[i], pl_z[i])$ can be obtained, as illustrated in Figure 7. The foot coordinates and the thigh endpoint coordinates can be calculated using a two-point method.

$$\frac{x[i]-p_x[i]}{pl_x[i]-p_x[i]}=\frac{y[i]-p_y[i]}{pl_y[i]-p_y[i]}=\frac{z[i]-p_z[i]}{pl_z[i]-p_z[i]} \tag{18}$$

Further, a horizontal square with a side length of 10 is established around the two points, defining a region (x_1, x_2), (y_1, y_2) on the ground based on the projection of the two squares. By searching for points within this area in the point cloud, we determine whether their heights exceed the linear function. If so, a collision will occur.

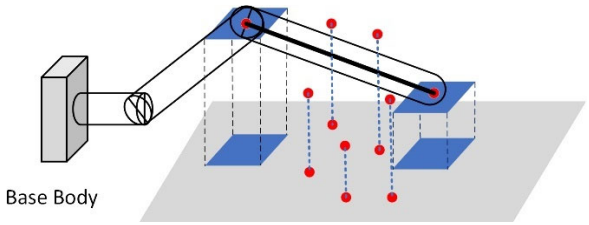

Fig. 7 Leg collision detection

3. EXPERIMENT

This section is primarily divided into two stages: simulation validation using Gazebo and physical validation using the prototype of the hexapod robot.

3.1. Simulation verification

For ease of control, the model's outline is simplified by adding collision properties to it, as shown in Figure 8.

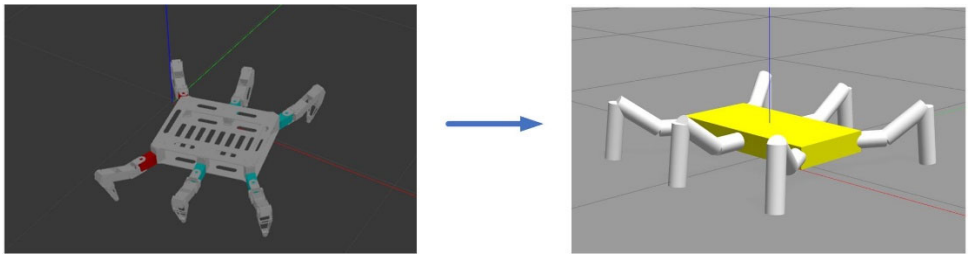

Fig. 8 Model conversion

As illustrated in Figure 9, constructing the gait learning environment primarily involves three types of terrain: slopes, trenches, and steps. During simulation, if there is a significant change in the body position, it is considered a motion error, and the task is restarted from the initial position. The slope has an incline angle of 33°, the trench has a width of 100mm, and the step has a height of 80mm.

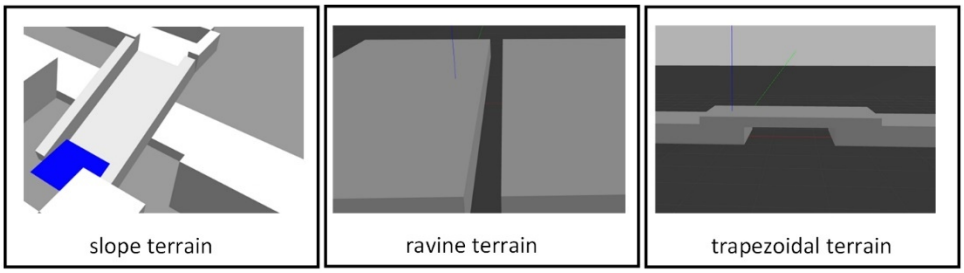

Fig. 9 trapezoidal terrain

Firstly, the reward values obtained during gait learning with the Dueling-DQN algorithm on sloping terrain, along with the variation in stability margin of the hexapod robot throughout the process, are depicted in Figure 10.

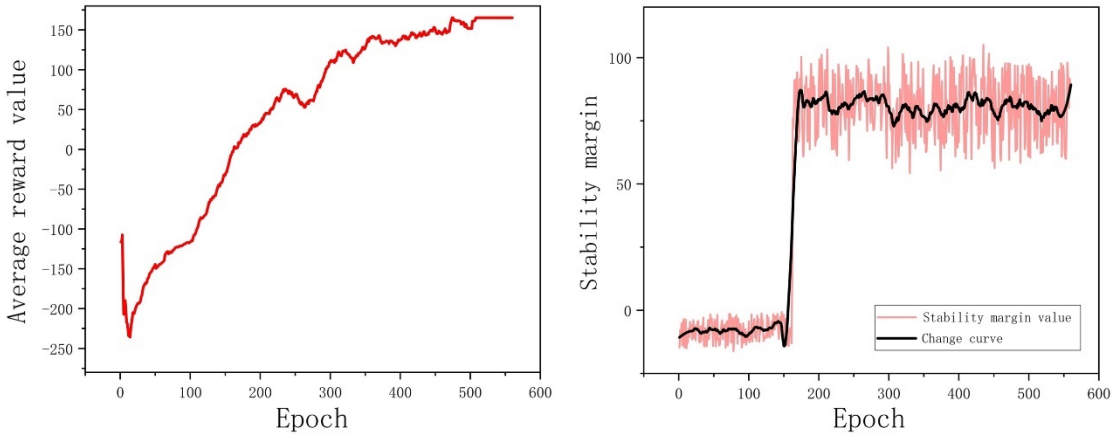

Fig. 10 Training results of Dueling-DQN algorithm in slope terrain

The simulation process of the hexapod robot on the slope is illustrated in Figure 11. After recognizing the sloping terrain, the hexapod robot begins selecting the coordinates of the footholds, gradually adjusting its center of mass parallel to the slope, and ultimately accomplishing the traversal of the sloping terrain.

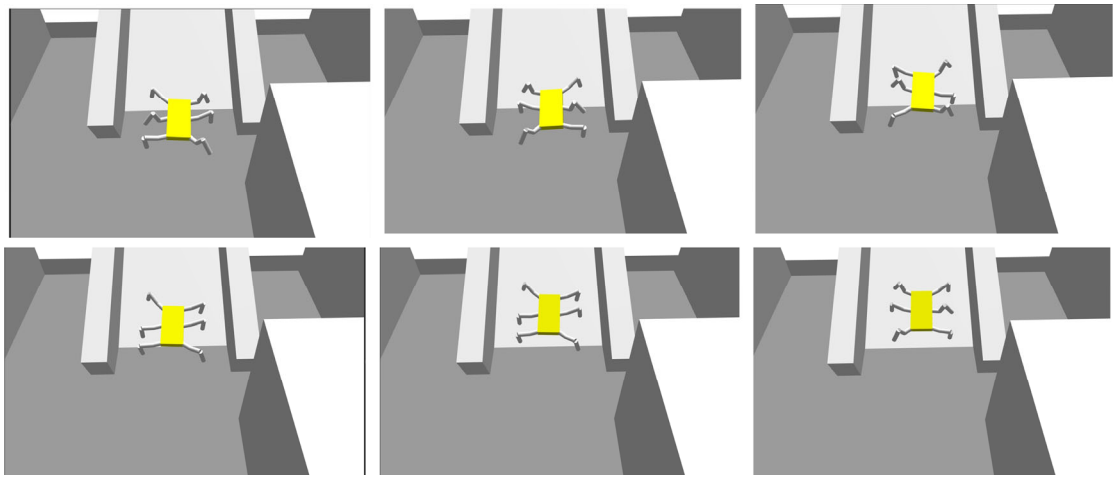

Fig. 11 Movement process of hexapod robot in slope-type terrain

The reward values obtained during gait learning using the Dueling-DQN algorithm in the ravine terrain, along with the stability margin changes of the hexapod robot throughout the process, are depicted in Figure 12. From the stability margin, it can be observed that after approximately 150 iterations, the robot's movement does not exhibit any imbalance. Moreover, over 500 iterations, the average reward value stabilizes and remains positive.

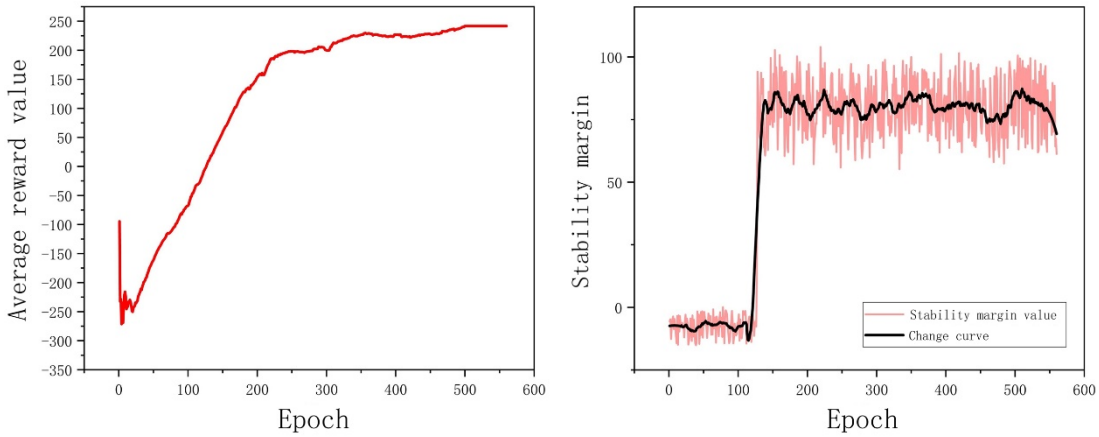

Fig. 12 Training results of Dueling-DQN algorithm in ravine-type terrain

The simulation of the hexapod robot crossing a ravine is illustrated in Figure 13. It can be observed that upon detecting the ravine, the hexapod robot autonomously adjusts its body position. It uses a quadrupedal gait to have the front legs cross the ravine, followed by positional adjustment using a quadrupedal gait. Subsequently, it employs a quadrupedal gait to have the middle legs cross the ravine. Then, it uses a tripod gait to advance while adjusting the position of the hind legs. Finally, it utilizes a quadrupedal gait to have the hind legs cross the ravine, thus completing the traversal of the ravine terrain.

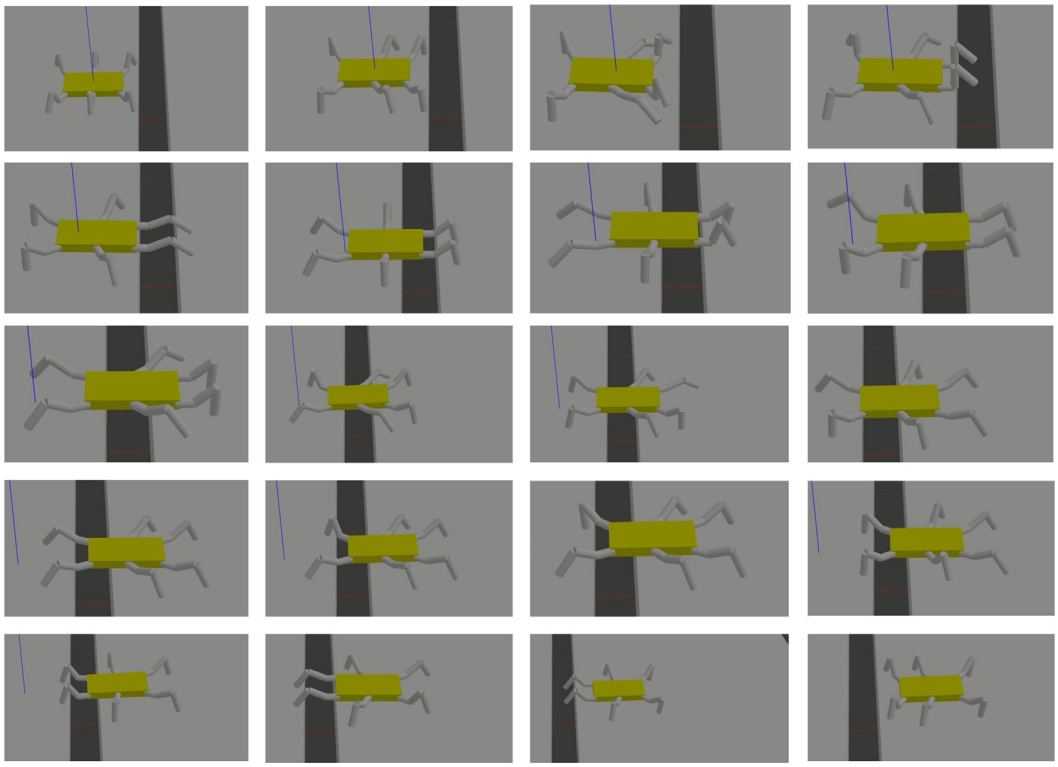

Fig. 13 Movement process of hexapod robot in ravine-type terrain

In dealing with terrains of trapezoidal shape, the process is primarily divided into two stages: ascending and descending. Dueling-DQN algorithm's rewards and the stability margin variation of the hexapod robot throughout the gait learning process in the ascending trapezoidal terrain are illustrated in Figure 14. It can be observed from the stability margin that the robot's motion stabilizes after approximately 420 iterations, and by the 500th iteration, the average reward stabilizes and becomes positive. Negotiating ascending trapezoidal terrains requires the robot to avoid collisions with the terrain surface, which could lead to body tilting. Consequently, multiple gait selections may be required, resulting in slower training compared to the first two terrain types.

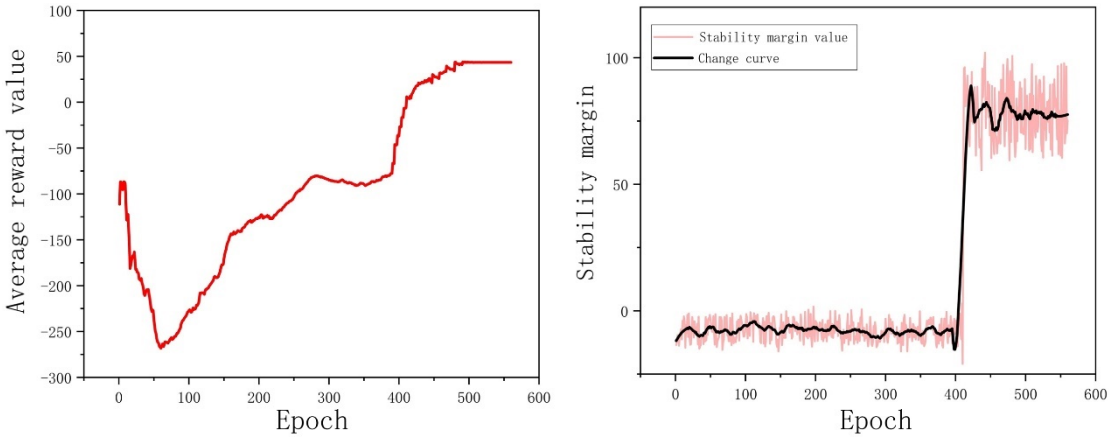

Fig. 14 Training results of Dueling-DQN algorithm in ladder-shaped terrain (upper)

The simulation process of the hexapod robot in the ascending trapezoidal terrain is depicted in Figure 15. It can be observed from the figure that upon terrain recognition, the robot adjusts its position and elevates its center of mass to prevent contact between the bottom of the body and the terrain surface. After placing the front legs on the terrain surface, it proceeds with a pentapedal gait. Then, once the middle legs are positioned on the terrain surface using a quadrupedal gait, it continues to move forward with a pentapedal gait. Finally, after placing the hind legs on the terrain surface using a quadrupedal gait, the hexapod robot completes the gait planning for the ascending trapezoidal terrain.

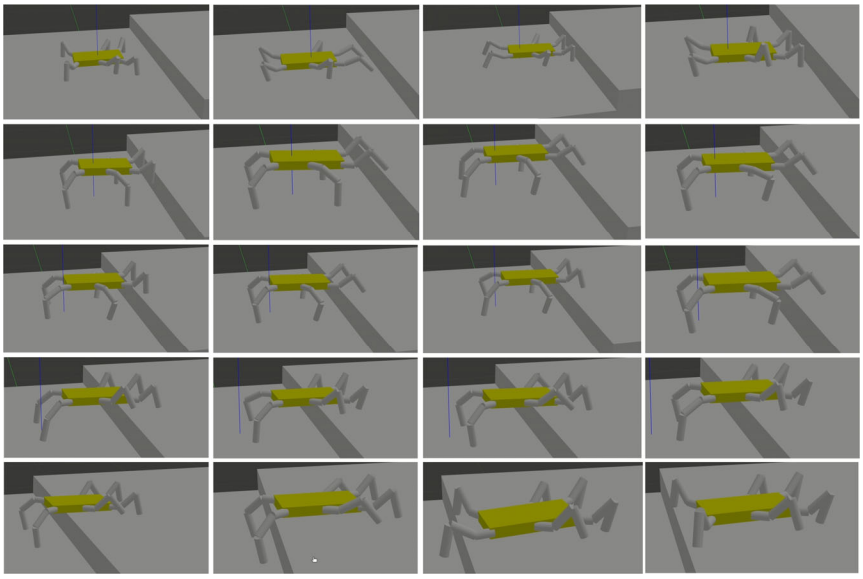

Fig. 15 Movement process of hexapod robot in the ladder-shaped terrain (upper)

Continuing with the training on the downward slope terrain, the rewards obtained during the gait learning process with the Dueling-DQN algorithm, along with the stability margin change of the hexapod robot throughout the entire process, are depicted in Figure 16. From the stability margin perspective, it is evident that after approximately 490 iterations, the robot's movement becomes balanced, and around 550 iterations, the average reward stabilizes and becomes positive.

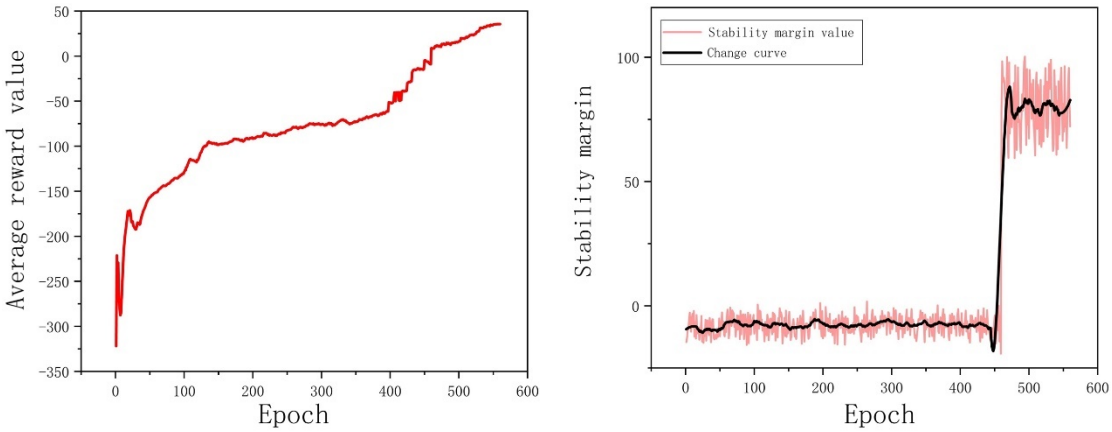

Fig. 16 Training results of Dueling-DQN algorithm in ladder-shaped terrain (down)

The simulation results of the hexapod robot on the downward slope terrain are illustrated in Figure 16. From the figure, it can be observed that upon recognizing the terrain, the hexapod robot utilizes a quadruped gait to advance, maintaining the front legs on the horizontal surface to keep the center of mass height unchanged. The robot continues to move forward using the quadruped gait, placing the middle legs on the horizontal surface. Subsequently, it switches to a quintuped gait to proceed further. Finally, the robot places the hind legs on the horizontal surface, adjusting the center of mass height, thus completing the gait planning.

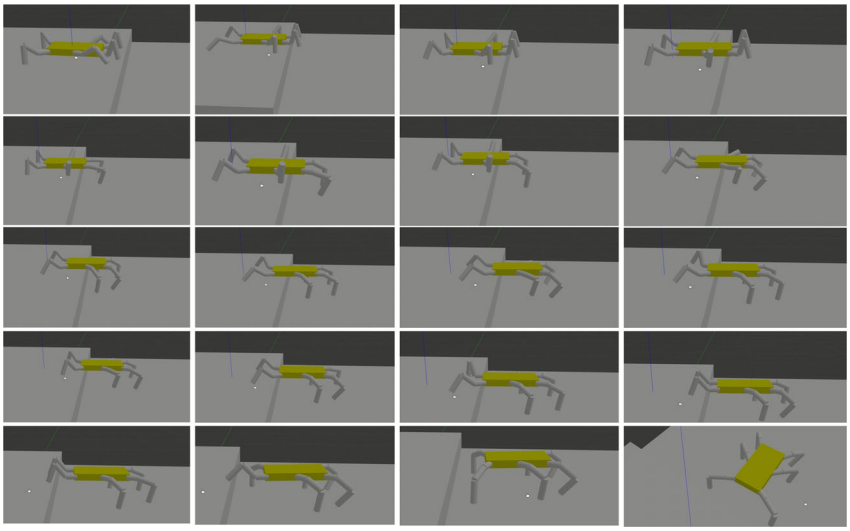

Fig. 17 Movement process of hexapod robot in the ladder-shaped terrain (down)

3.2. Experimental verification

The trained algorithm is deployed onto the hexapod robot prototype for physical validation. Laser range finder data is utilized to construct an elevation map in RVIZ, as depicted in Figure 18. This elevation map serves as a groundwork for the motion data of the hexapod robot.

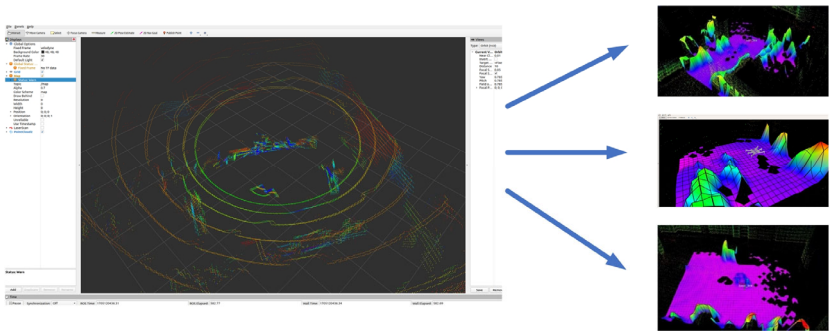

Fig. 18 Build a high-level diagram

To further validate the feasibility of the gait planner, the laser rangefinder is mounted on a servo hexapod robot. A serial communication program is developed to send data via serial ports, along with the establishment of a communication protocol. The angle values of the 18 joints are transmitted to the main control board of the hexapod robot. The angle control of the Feite servos follows a protocol where values between 0 and 2047 represent positive directions (0°-180°), while values between 2048 and 4095 represent negative directions (0°-180°). Therefore, the transmitted data is in the form of a 16-bit integer, which needs to be split into two 8-bit transmissions. The main control board receives the data and invokes servo control functions to manipulate the joints of the hexapod robot. Due to the varying initial angles of each servo, individual calibration is required for each servo to zero.

In real-world scenarios, to validate the effectiveness of the gait controller in controlling the hexapod robot, a physical slope environment is set up with a slope inclination angle of approximately 33°, as illustrated in Figure 19.

Fig. 19 Slope type obstacle experimental site

During the experimental validation using the hexapod robot prototype, the movement of the robot throughout the entire slope experiment is depicted in Figure 20 within the experimental site.

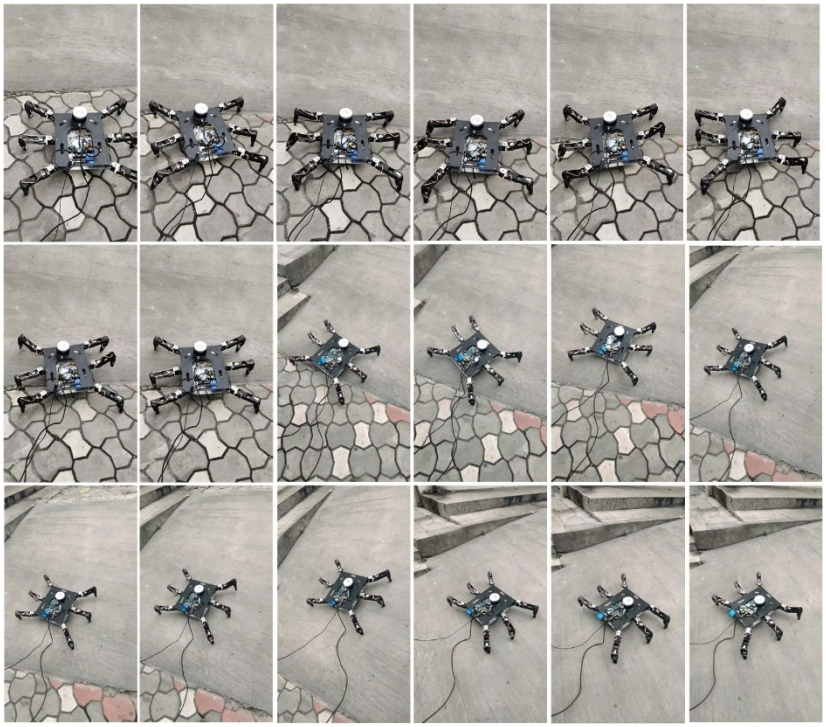

Fig. 20 Movement process of hexapod robot in slope experimental site

A experimental validation environment for the ladder-type and trench-type terrains was constructed using cardboard boxes, as shown in Figure 21 The trench width was set to 100mm, and the ladder height was set to 80mm.

Fig. 21 Joint Experimental Site

The entire movement process is illustrated in Figure 22

Fig. 22 Hexapod robot joint field movement process

4. CONCLUSION

Aiming at the challenge of gait planning on rugged terrain in explosive environments, a hexapod robot gait planning method based on Dueling-DQN algorithm is proposed. The proposed method has the following characteristics. First, LiDAR is used to obtain terrain feature data to enhance the perception of environmental data; at the same time, the Dueling-DQN algorithm is used to train foot footholds to improve the stability of the hexapod robot body; a gait planner is designed to use redundant strategic steps State flexibility.

The proposed method effectively solves the challenges faced by hexapod robots in handling dangerous objects in explosive environments, such as rugged terrain, difficulty in control, and inability to independently form terrain handling strategies. The feasibility of the method was verified through Gazebo simulation. This method has important practical value in gait planning for handling dangerous objects in explosive environments, and can be extended to future hexapod robots in field operations.

ACKNOWLEDGEMENTS

This work was funded in part by Sichuan Province Science and Technology Department Key Research and Development Project (Grant No. 2022YFG0347) the Natural Science Foundation of Sichuan (2023NSFSC1315).

REFERENCES

[1] Luccioni B M, Ambrosini R D, Danesi R F. Analysing explosive damage in an urban environment[J]. Proceedings of the Institution of Civil Engineers-Structures and Buildings, 2005, 158(1): 1-12.

[2] Li Y, Li M, Zhu H, et al. Development and applications of rescue robots for explosion accidents in coal mines[J]. Journal of Field Robotics, 2020, 37(3): 466-489.

[3] Liu K, Gao F, Chen Z, et al. Foothold Planning and Body Posture Adjustment Strategy of Hexapod Robot in Complex Terrain[C]//International Conference on Mechanism and Machine Science. Singapore: Springer Nature Singapore, 2022: 2023-2035.

[4] Hou S, Cao C, Liu X, et al. Gait lateral network: Learning discriminative and compact representations for gait recognition[C]//European conference on computer vision. Cham: Springer International Publishing, 2020: 382-398.

[5] Zu Xiaoqin, Chen Yaolu, Ran Dengyu. Autonomous gait learning for hexapod robot based on reward guidance [J]. Journal of Beijing University of Technology, 2021 (2).

[6] Cizek P , Zoula M , Faigl J . Design, Construction, and Rough-Terrain Locomotion Control of Novel Hexapod Walking Robot with Four Degrees of Freedom per Leg[J]. IEEE Access, 2021, PP(99):1-1

[7] Gao Y, Wang D, Wei W, et al. Constrained predictive tracking control for unmanned hexapod robot with tripod gait[J]. Drones, 2022, 6(9): 246.

[8] Liu Y, Xu Y. Free gait planning of hexapod robot based on improved DQN algorithm[C]//2020 IEEE 2nd International Conference on Civil Aviation Safety and Information Technology (ICCASIT. IEEE, 2020: 488-491.

[9] Youcef Z, Pierre C. Control of the trajectory of a hexapod robot based on distributed Q-learning[C]//2004 IEEE International Symposium on Industrial Electronics. IEEE, 2004, 1: 277-282.

[10] Ouyang Y. Task offloading algorithm of vehicle edge computing environment based on Dueling-DQN[C]//Journal of Physics: Conference Series. IOP Publishing, 2021, 1873(1): 012046.

[11] Gu Y, Zhu Z, Lv J, et al. DM-DQN: Dueling Munchausen deep Q network for robot path planning[J]. Complex & Intelligent Systems, 2023, 9(4): 4287-4300.

[12] Peng B, Sun Q, Li S E, et al. End-to-end autonomous driving through dueling double deep Q-network[J]. Automotive Innovation, 2021, 4: 328-337.

[13] Jiang W, Bao C, Xu G, et al. Research on autonomous obstacle avoidance and target tracking of UAV based on improved dueling DQN algorithm[C]//2021 China Automation Congress (CAC). IEEE, 2021: 5110-5115.

[14] Wang Z, Schaul T, Hessel M, et al. Dueling network architectures for deep reinforcement learning[C]//International conference on machine learning. PMLR, 2016: 1995-2003.

[15] Guo Jian, Liang Yongjie, Zhang Xiaojia, et al. Kinematics analysis and calculation of hexapod robot based on gait planning [J]. Machine Tools and Hydraulics, 2023, 51(05): 66-73.