2025 IEEE/SICE International Symposium on System Integration (SII)
January 21-24, 2025, Munich, Germany

Curriculum Reinforcement Learning for Obstacle Avoidance Postures for a Hyper-redundant Manipulator

Keishi Hirade[1] and Ryuma Niiyama[1]

Abstract— **Redundant robots with more degrees of freedom than necessary for given tasks have attracted attention due to their flexibility, but they also increase the complexity of control. Especially for highly redundant robots, accurate motion planning and obstacle avoidance remain challenging. This research aims to develop a redundant robot arm that can perform reaching tasks while avoiding randomly appearing obstacles using reinforcement learning. We adopted the Proximal Policy Optimization (PPO) algorithm and conducted simulations in the Mujoco environment. The learning process consisted of a three-stage curriculum: reaching task, fixed obstacle avoidance, and random obstacle avoidance, gradually increasing difficulty to achieve efficient learning. Experimental results showed that the arm could adapt to complex environments and effectively reach target positions while avoiding obstacles. In particular, the system demonstrated high adaptability to randomly placed obstacles, successfully reaching within a maximum distance of approximately 0.07 m from the target position.**

I. INTRODUCTION

In the field of robotics, the control of hyper-redundant robot arms, such as ostrich-inspired manipulators, presents both great potential and complex challenges due to their high degrees of freedom and flexibility. In particular, obstacle avoidance in dynamic and unpredictable environments remains one of the crucial research topics in this field.

Conventional obstacle avoidance methods primarily targeted static or predictable environments. Khatib's potential field method [4] and LaValle's path planning method based on random exploration [6] enabled high computational efficiency and path generation in complex environments, respectively. However, these methods had limitations in adapting to dynamic environments. Aiming for adaptation to dynamic environments, new methods were proposed by Ge and Cui [2] and Wang et al. [14]. However, issues remained regarding the handling of multiple moving obstacles and computational costs.

Research on redundant robot systems has also progressed, with detailed analyses conducted by Siciliano [13] and Patel and Shadpey [11]. However, the complexity of control brought about by redundancy, especially in dynamic tasks, has been identified as a challenge. The studies by Burdick [1] and Mayne et al. [9] have also highlighted this complexity. As biologically inspired approaches, new attempts have been made, such as the snake-like robot research by Hirose and Yamada [3] and the adaptive neural network control method by Ma et al. [8]. However, challenges remain in applying these methods to three-dimensional spaces and large-scale systems. Or et al. proposed a curriculum-based reinforcement learning approach for controlling a tendon-driven high-DOF underactuated ostrich-inspired manipulator.[10].

Recent studies have demonstrated the potential of reinforcement learning in robotic manipulation tasks. Lin et al. proposed a snake-like path planning algorithm combining reinforcement learning with proprioception for redundant manipulators [7]. Their approach showed promising results in obstacle avoidance while maintaining smooth motion trajectories. Additionally, Shen et al. developed a reactive obstacle avoidance method based on reinforcement learning for redundant manipulators [12]. Their system demonstrated real-time adaptation capabilities in dynamic environments. These studies highlight the growing effectiveness of learning-based approaches in handling complex manipulation tasks.

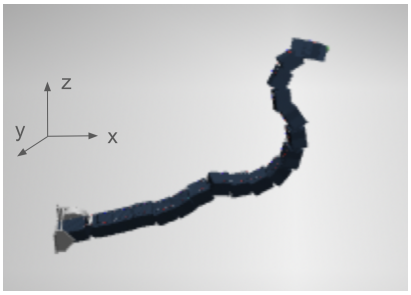
We chose an ostrich-inspired manipulator due to its characteristics that are well-suited for navigating complex environments. The unique joint configuration and wire-driven actuation present control challenges, making it an excellent test case for validating our learning approach.

In this study, we propose an approach applying reinforcement learning to the control of a hyper-redundant robot arm based on an ostrich-inspired manipulator design. As demonstrated by Kober et al. [5], reinforcement learning is suitable for controlling hyper-redundant robot arms due to its ability to handle high-dimensional state spaces and continuous action spaces. Our goal is to develop a redundant robot arm capable of performing reaching tasks while avoiding randomly appearing obstacles using the Proximal Policy Optimization (PPO) algorithm. To achieve this, we will utilize simulation environments to facilitate efficient learning. We employ a curriculum learning strategy, gradually increasing the complexity of tasks from basic reaching to fixed obstacle avoidance, and finally to random obstacle avoidance. This learning approach allows the system to build upon simpler skills before tackling more complex scenarios, leading to more robust and efficient learning compared to conventional methods that often attempt to learn complex behaviors all at once.

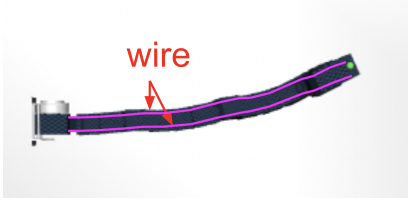
This research extends the application of reinforcement learning to highly complex, biologically-inspired robotic systems, demonstrating the scalability of these methods to high-dimensional control problems. Our work on random obstacle avoidance represents a advancement in adaptive control for hyper-redundant manipulators, addressing a key challenge in real-world robotic applications. Collectively, our research pushes the boundaries of control possibilities for highly flexible, multi-degree-of-freedom robotic systems, with potential impacts across fields ranging from industrial

[1]School of Science and Technology, Meiji University

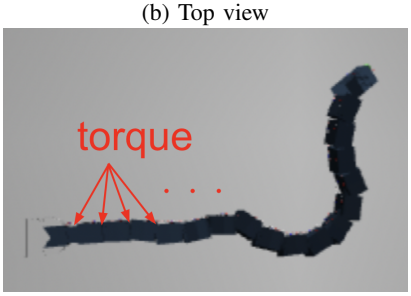
automation to search and rescue operations.



(a) 3D view



(b) Top view



(c) Side view

Fig. 1: Redundant robot manipulator from different perspectives. The manipulator consists of 18 joints connected in series, and its posture is primarily controlled by applying torque to the motors at each joint. Two tension wires (shown in red) running through the structure are attached to motors, and their lengths are adjusted to provide supplementary support for posture control.

II. Method

This research was conducted in a simulation environment. Specifically, we constructed an ultra-redundant robot arm as shown in Figure 1 using the Mujoco engine in simulation. This simulation model reproduces a flexible arm structure with 18 joints, equipped with realistic physical properties. By controlling the force exerted on these wires and adjusting the joint angles between each segment, as shown in Figure 1c, the arm can move in three-dimensional space. In this paper, the tip position is controlled by controlling the joint angles of 18 motors and by pulling two wires. In this research, we adopted a curriculum learning approach that gradually increases the difficulty of tasks. This method allowed the arm to build capabilities gradually, starting from basic movements to tackling more complex scenarios. The specific learning stages were divided into three stages. In the first stage, we conducted learning in an obstacle-free environment called the reaching task. The goal of this stage was to reach any point within the specified 3D space. In the simulation environment, the target point was visualized as a small sphere, and the arm's reaching accuracy was evaluated. In the second stage, we tackled the fixed obstacle avoidance task. Here, the goal was to reach the specified point while avoiding a fixed obstacle. By placing obstacles in the simulation and detecting collisions with the arm, we enabled learning of realistic avoidance movements. In the final stage, the random obstacle avoidance task, we aimed to reach the specified point in an environment where one obstacle is randomly placed. In this stage, the position of the obstacle changes with each episode, requiring the acquisition of a more versatile avoidance strategy. Taking advantage of the simulation environment, we could generate several obstacle placement patterns and use them for learning. Through this gradual approach, the arm was able to build skills from simple tasks to complex tasks, eventually realizing flexible movements with advanced obstacle avoidance capabilities. Implementation in the simulation environment enabled safe and efficient collection of large amounts of learning data while avoiding risks and time constraints associated with trial and error on actual machines.

III. Simulation Environment

In this research, we constructed a robot arm simulation environment using the Mujoco engine. This simulation environment includes the following elements:

- Ultra-redundant robot arm with 18 joints, with a total length of 1.2m and weighing 2.57kgf
- Target point represented by a red sphere with a diameter of 1 cm
- Obstacles (cuboid shape) according to the task
- Simulation of physical phenomena such as gravity, friction, and collision
- Touch sensors for collision detection

The goal position was set as shown in Table I.

TABLE I: Goal Setting

Item	Value
Position	0.9m, 0m, 1.35m
Size	Sphere with radius 0.005m

The placement of obstacles was set according to the task, as shown in Table II and III.

TABLE II: Obstacle Placement Settings - Part 1

Task	Number of Obstacles	Placement Range
Reaching	0	-
Fixed Obstacle	1	Position: 0.7m, 0m, 1.28m Size: 0.036m, 0.36m, 0.36m

TABLE III: Obstacle Placement Settings - Part 2

Task	Placement Range
Random Obstacle	Randomly selected from two ranges:
	Limit1
	Position: x: 0.52m,
	y: -0.018m to 0.018m,
	z: 1.25m to 1.27m
	Size: x: 0.2m,
	y: 0.05m to 0.15m,
	z: 0.13m to 0.15m
	Limit2
	Position: x: 0.85m to 0.9m,
	y: -0.018m to 0.018m,
	z: 1.15m
	Size: x: 0.1m to 0.15m,
	y: 0.05m, z: 0.1m

In the random obstacle task, the position and size of the obstacle are set within the specified range at the start of each episode. This allows the arm to learn the ability to respond to various obstacle configurations. This simulation environment enables safer and more efficient verification of robot arm behavior under various conditions compared to experiments in the real world. Especially in the random obstacle task, changing the position and size of obstacles promotes learning of more versatile avoidance strategies.

IV. Reinforcement Learning Algorithm

In this research, we adopted the Proximal Policy Optimization (PPO) algorithm. PPO is a state-of-the-art policy gradient method with high stability and sample efficiency. Below, we detail the main components of the reinforcement learning setup for each task:

A. Common Settings

1) State Space: The state space consists of the elements shown in Table IV.

TABLE IV: Components of the State Space

Element	Representation
Angle of each joint	$\theta_1, \theta_2, ..., \theta_{18}$
3D position of arm tip	(x, y, z)
Target position	(x_g, y_g, z_g)
Obstacle information	3D coordinates of 8 vertices of the cuboid

Obstacle information is represented by the 3D coordinates of the 8 vertices of the cuboid representing the obstacle (total 24 dimensions). Also, in the reaching task, obstacle information is treated as an array of zeros to match the dimension of the state space. The state space is represented with a dimension of 48.

2) Action Space: The action space is the command value of each joint angle, which is an 18-dimensional continuous value:

$$a = (u_1, u_2, ..., u_{18}), \quad u_i \in [-1, 1]$$

Here, u_i is the normalized command value of each joint angle, which is scaled according to the joint angle limit specified in the simulation environment.

3) Learning Parameters: The hyperparameters for the PPO algorithm were carefully tuned through an iterative process of experimentation and evaluation. We explored a range of values for each parameter, assessing their impact on the learning performance and stability of the agent. The final set of hyperparameters, shown in Table V, represents the configuration that yielded the best results in terms of learning efficiency and task performance.

TABLE V: Hyperparameter Settings for PPO Algorithm

Parameter	Value
Learning rate	3e-5
Number of steps	4096
Batch size	128
Number of epochs	10
Discount rate γ	0.99
GAE parameter λ	0.95
Clipping range ϵ	0.3
Entropy coefficient	0.01
Timestep	10ms

B. Reward Function

For all tasks, the reward function takes the following general form:

$$R = R_d + w_p R_p + w_c R_c$$

Each component of the reward function plays a crucial role in guiding the agent's learning process:

Distance Reward (R_d): This component encourages the agent to minimize its distance to the goal. It is typically negative and proportional to the Euclidean distance between the arm's end effector and the target position.

Progress Reward (R_p): This term rewards the agent for making progress towards the goal between consecutive steps. It is calculated as the decrease in distance to the goal from the previous step to the current step. This reward is crucial for providing a dense learning signal, especially in the early stages of training when the agent is far from the goal. Our experiments showed that without this progress reward, the agent struggled to learn effectively, often failing to make consistent progress towards the goal.

Collision Penalty (R_c): This negative reward discourages the agent from colliding with obstacles in the environment. It is typically a large negative value when a collision is detected, teaching the agent to avoid obstacles while navigating towards the goal.

Weights (w_p, w_c): Weights for each reward term. These weights were carefully tuned to balance the agent's focus on reaching the goal, making steady progress, and avoiding obstacles.

1) Reaching Task: In the reaching task, collision penalty is not considered as there are no obstacles. The reward function is set as follows

$$R_d = -\|p - p_g\|$$
$$R_p = 0.5 \times (\|p_{t-1} - p_g\| - \|p_t - p_g\|)$$
$$R_c = 0$$

Here, p represents the position of the arm tip, and p_g represents the target position. Also, p_t indicates the position of the arm tip at the current step t, and p_{t-1} at the previous step $t-1$.

2) Fixed Obstacle Task: In the fixed obstacle task, a collision penalty is introduced. The reward function is set as follows.

$$R_d = -\|p - p_g\|$$
$$R_p = 0.5 \times (\|p_{t-1} - p_g\| - \|p_t - p_g\|)$$
$$R_c = -1 \times (\|p - p_g\| + 1) \quad \text{(only upon collision)}$$

Here, the presence or absence of collision is determined based on the value of the touch sensor set in the simulator.

3) Random Obstacle Avoidance Task: In the random obstacle avoidance task, the same reward function as the fixed obstacle task is used, but the position of the obstacle changes with each episode. The reward function is set as follows:

$$R_d = -\|p - p_g\|$$
$$R_p = 0.5 \times (\|p_{t-1} - p_g\| - \|p_t - p_g\|)$$
$$R_c = -1 \times (\|p - p_g\| + 1) \quad \text{(only upon collision)}$$

C. Random Obstacle Avoidance Implementation

In implementing the training for the random obstacle avoidance task, we devised a device regarding the determination of randomly appearing obstacles. We defined two distinct sets of constraints (Limit1 and Limit2) for the size and position ranges of obstacles. At the beginning of each episode, we dynamically selected the constraint based on performance. We tracked the average reward for each constraint, increasing the probability of selecting the constraint with lower performance. This adaptive selection method promoted balanced learning across both constraints. Based on the selected constraint, we randomly generated obstacles, ensuring environmental diversity. This approach enabled the agent to adapt to environments of varying difficulty and learn more generalized avoidance strategies.

V. Results

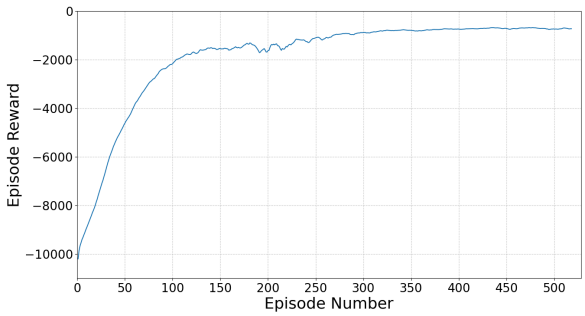
The experiments showed promising results across all three task types: reaching, fixed obstacle avoidance, and random obstacle avoidance. Figure 2 shows the results of the reaching task and fixed obstacle avoidance task, while Figure 3 shows the results of random obstacle avoidance.

A. Reaching Task

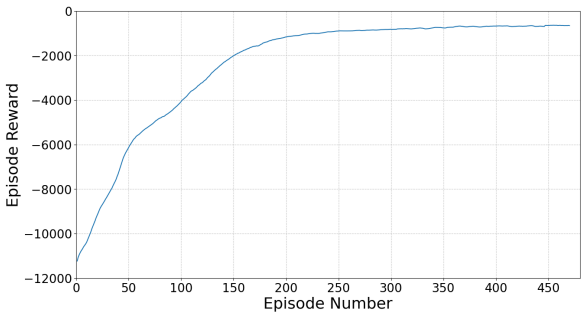
Figure 2a shows the learning curve for the reaching task. The learning process exhibits characteristics similar to those typically observed in reinforcement learning curves. First, a rapid gradient is observed in the initial 100 episodes, with a significant improvement in rewards. Subsequently, from 100 to 250 episodes, the reward continues to increase gradually, and a convergence trend is observed around 300 episodes. Beyond 350 episodes, no significant policy improvements are seen, and the value remains almost constant. Figure 2c shows the final configuration of the arm in a successful reaching task. From this image, it can be confirmed that the arm forms a gentle upward arc from the base to the tip as it moves towards the target position. The arm's posture is smooth, with no unnatural bending or twisting observed. The tip of the arm reached a position close to the target position. Looking at the specific data, the target position was (0.9, 0, 1.35), and the final position of the robot arm tip was (0.8537, 0.0028, 1.4031). Calculating from these coordinates, we can see that the arm tip reached a distance of about 0.0704 m from the target position. This result demonstrates that the arm was able to approach the target with high accuracy.

B. Fixed Obstacle Avoidance

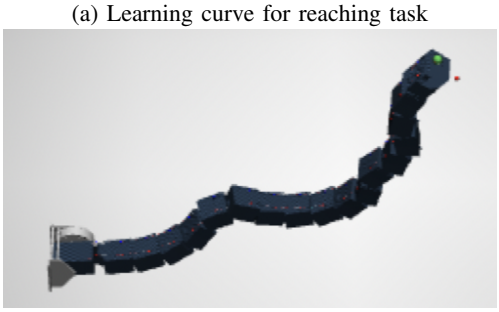
Figure 2b shows the learning curve for the fixed obstacle avoidance task. This curve exhibits characteristics similar to those of the reaching task. In the initial learning stage, especially in the first 150 episodes, rapid improvement is observed with a significant increase in rewards. Subsequently, from 150 to 250 episodes, gradual improvement continues, and a convergence trend begins to appear around 250 episodes. Beyond 300 episodes, although there are slight fluctuations in reward values, the overall performance remains in a stable state. Figure 2d shows the final configuration of the arm avoiding the fixed obstacle. From this image, it can be confirmed that the arm extends towards the target position while avoiding the obstacle represented by a red cuboid. In this task, the arm forms an "S-shaped" curve, bending upward from the base and then curving downward towards the tip. Despite this complex shape, the arm's posture remains smooth, with no unnatural bending or twisting observed. The tip of the arm reached a position very close to the target position. Looking at the specific data, the target position was (0.9, 0, 1.35), and the final position of the robot arm tip was (0.9335, 0.0264, 1.3677). Calculating from these coordinates, we can see that the arm tip reached a distance of about 0.0462 m from the target position. Our results is that the fixed obstacle avoidance task achieved a smaller distance between the arm tip and target position compared to the simpler reaching task. This result can be attributed to several factors. First, the presence of the fixed obstacle may have acted as a guide that constrained the solution space. Second, the additional collision penalty in the reward function for the obstacle avoidance task may have encouraged the agent to explore more carefully and develop finer control strategies.



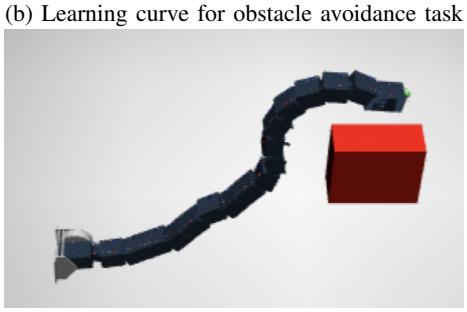
(a) Learning curve for reaching task



(b) Learning curve for obstacle avoidance task

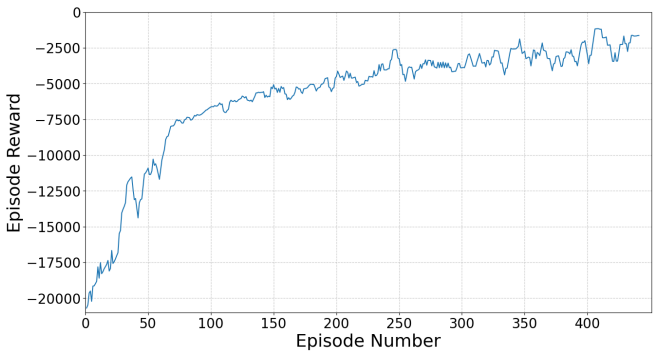


(c) Robot arm in reaching task

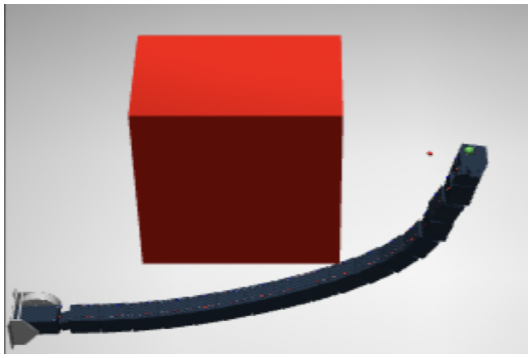


(d) Robot arm in obstacle avoidance task

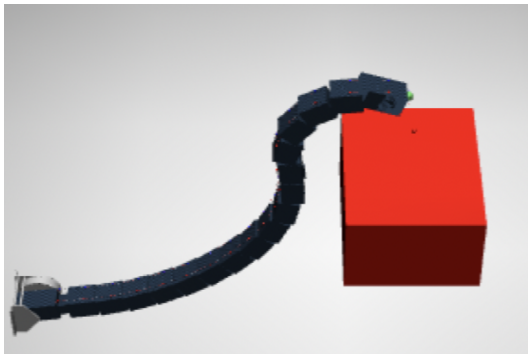
Fig. 2: Comparison of robot arm obstacle avoidance and reaching patterns



(a) Learning curve for random obstacle avoidance



(b) Robot arm in random obstacle avoidance in limit1



(c) Robot arm in random obstacle avoidance in limit2

Fig. 3: Comparison of robot arm movements in random obstacle avoidance

C. Random Obstacle Avoidance

Figure 3 shows the learning curve and arm configurations for the random obstacle avoidance task. Figure 3a represents the learning curve for the random obstacle avoidance task. Consistent improvement was observed as learning progressed, with performance enhancement continuing up to about 70 episodes. Subsequently, gradual improvement was observed until around 300 episodes. Figures 3b and 3c illustrate the arm configurations avoiding obstacles with randomly selected shapes and placements. These images demonstrate the arm's ability to flexibly adapt to different obstacle arrangements. In Figure 3b, the arm curves to pass beneath the obstacle, while in Figure 3c, the arm forms a shape that goes over and around the obstacle. This diversity in shapes indicates that the arm can generate optimal joint angles for random environments and effectively avoid obstacles. Specifically, in the situation shown in Figure 3b, the target position was (0.9, 0, 1.35), and the final position of the robot arm tip was (0.974, -0.0072, 1.37). Calculating from these coordinates, we can see that the arm tip reached a distance of about 0.073 m from the target position. On the other hand, in the situation depicted in Figure 3c, for the same target position (0.9, 0, 1.35), the final position of the robot arm tip was (0.894, -0.0055, 1.451). In this case, the arm tip reached a distance of about 0.98 m from the target position.

VI. Discussion

The results of this study demonstrate the effectiveness of reinforcement learning in controlling ultra-redundant robot arms modeled after an ostrich's neck. The arm learned tasks that gradually increased in complexity from reaching tasks to fixed obstacle avoidance and random obstacle avoidance, demonstrating the adaptability and effectiveness of the learning approach proposed in this study. The adoption of the curriculum learning approach appears to have been effective. The large number of episodes required for convergence in the learning curve indicated the difficulty of this task. Additionally, the final error between the total length of the robot arm and the distance to the target was about 6.6%. This indicates that despite the challenges posed by the high degree of redundancy of the arm, a high level of precision was achieved. A direct numerical comparison with other reinforcement learning methods such as SAC and DDPG remains as future work. One clear advantage of our method is that the curriculum learning strategy significantly reduced the complexity of the learning process. This reduction in complexity is particularly important for hyper-redundant systems where the high dimensionality of the state and action spaces makes direct learning challenging.

VII. Conclusion

This study presents an approach to controlling hyper-redundant robot arms using reinforcement learning, utilizing an ostrich-inspired manipulator. Our system demonstrates reaching and obstacle avoidance in complex environments, with important implications for robotics fields requiring flexibility and adaptability. The potential applications are wide-ranging, from search and rescue operations where a small camera attached to the arm's tip could be utilized, to space exploration where the arm could navigate around obstacles. Future research will focus on transferring the learned policy to physical robot systems and integrating sensors for enhanced environmental recognition. By combining biologically inspired principles, as exemplified by our ostrich-inspired manipulator, with advanced machine learning techniques, we are pushing the boundaries of robotic manipulation.

References

[1] J.W. Burdick. On the inverse kinematics of redundant manipulators: characterization of the self-motion manifolds. In *1989 International Conference on Robotics and Automation Proceedings*, pages 264–270 vol.1, May 1989.

[2] S.S. Ge and Y.J. Cui. Dynamic Motion Planning for Mobile Robots Using Potential Field Method. *Autonomous Robots*, 13(3):207–222, November 2002.

[3] Shigeo Hirose and Hiroya Yamada. Snake-like robots [Tutorial]. *IEEE Robotics & Automation Magazine*, 16(1):88–98, March 2009. Conference Name: IEEE Robotics & Automation Magazine.

[4] Oussama Khatib. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. In Ingemar J. Cox and Gordon T. Wilfong, editors, *Autonomous Robot Vehicles*, pages 396–404. Springer, New York, NY, 1990.

[5] Jens Kober, J. Bagnell, and Jan Peters. Reinforcement Learning in Robotics: A Survey. *The International Journal of Robotics Research*, 32:1238–1274, September 2013.

[6] S. LaValle. Rapidly-exploring random trees : a new tool for path planning. *The annual research report*, 1998.

[7] Yue Lin, Hongbin Wang, Jing Zhang, and Xiangyuan Wu. Snake-like path planning algorithm based on reinforcement learning and proprioception for redundant manipulator. *International Journal of Advanced Robotic Systems*, 19(4), 2022.

[8] Jianjun Ma, Shuzhi Sam Ge, Zhiqiang Zheng, and Dewen Hu. Adaptive NN Control of a Class of Nonlinear Systems With Asymmetric Saturation Actuators. *IEEE transactions on neural networks and learning systems*, 26(7):1532–1538, July 2015.

[9] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, June 2000.

[10] Keung Or, Kehua Wu, Kazashi Nakano, Masahiro Ikeda, Mitsuhito Ando, Yasuo Kuniyoshi, and Ryuma Niiyama. Curriculum-reinforcement learning on simulation platform of tendon-driven high-degree of freedom underactuated manipulator. *Frontiers in Robotics and AI*, 10:1066518, July 2023.

[11] Rajni Patel and F. Shadpey. Control of Redundant Robot Manipulators - Theory and Experiments. *Lecture Notes in Control and Information Sciences*, 316, January 2005.

[12] Yue Shen, Wei Wang, Chenguang Yang, and Dewei Gan. Reinforcement learning-based reactive obstacle avoidance approach for redundant manipulator. *Entropy*, 24(2):279, 2022.

[13] Bruno Siciliano. Kinematic control of redundant robot manipulators: A tutorial. *Journal of Intelligent and Robotic Systems*, 3(3):201–212, September 1990.

[14] Li Wang, Yushu Liu, Hongbin Deng, and Yuanqing Xu. Obstacle-avoidance Path Planning for Soccer Robots Using Particle Swarm Optimization. In *2006 IEEE International Conference on Robotics and Biomimetics*, pages 1233–1238, December 2006.