# Automatic Flipper Control for Crawler Type Rescue Robot using Reinforcement Learning

Hitoshi Kono[1], Sadaharu Isayama[2], Fukuro Koshiji[3], Kaori Watanabe[4], Hidekazu Suzuki[5]
Department of Information and Communication Engineering, Tokyo Denki University, Tokyo, Japan[1]
Graduate School of Engineering, Tokyo Polytechnic University, Kanagawa, Japan[2]
Department of Engineering, Tokyo Polytechnic University, Kanagawa, Japan[3,5]
New Technology Foundation, Tokyo, Japan[4]

***Abstract*—In recent years, many natural disasters have occurred, and rescue robots have been used to gather information at disaster sites. Rescue robots, particularly crawler type rescue robots are operated through remote control by their operators via wireless communication or wired. However, certain robots have been known to not return owing to tipping over or disconnection of communication wires caused by missed operations. Therefore, studies have focused on automatic control of rescue robots. Adapting the rescue robot for uneven terrain or unexpected obstacle shape to travel in autonomous control situation is challenging. It requires complete autonomous control as well as partial control of the rescue robot, which necessitates assistance for teleoperation. This study proposed automatic flipper control of rescue robots using reinforcement learning for stepping over steps. The proposed method involved designing of the learning environment, reward setting, and system configuration for reinforcement learning. The input data for the training data were coarse-grained information using a distance sensor, gyro sensor, and GPS coordinates information. Reinforcement learning was performed through a physical simulation within an environment wherein the shape of a step changed once every 100 episodes. The robot's compensation was designed to reduce the impact on the robot's body by changing the robot's attitude angle. The learned knowledge, which is contained action-value function, was reused to verify that the flipper could be automatically controlled by the operator when the rescue robot is operated as moving along a direction remotely, and that the robot could step over steps.**

***Keywords*—Rescue robot; sub-crawler control; reinforcement learning; physics simulation**

## I. Introduction

In recent years, natural disasters as well as chemical, biological, radiological, nuclear, and explosive (CBRNE) disasters have become more frequent. Consequently, rescue robots have been used to gather information at disaster sites [1], [2], [3]. When an operator remotely operates a rescue robot, predicting the environmental conditions in which the robot will be placed is challenging. Thus, the operation is often performed in a complex environment. This places a heavy burden on the operator, and even a well-trained operator may mishandle the robot owing to tension and stress, which may result robots not returning to their original location. Therefore, research on automatic control or an appropriate control support system of rescue robots is currently underway to reduce the burden on operators and to ensure mission success [4], [5].

The operator controls the rescue robot remotely from a remote location via a control screen, relying on information obtained from the camera and sensors mounted on the robot. However, when running on stairs, there is considerable environmental information that even an experienced operator must pay attention to and check, which renders teleoperation difficult. In addition, training is required until the operator becomes familiar with the operation, which can result in a shortage of personnel in an emergency. The approach is not aimed at making the rescue robot operation completely autonomous, but to provide supplementary motion assistance to reduce the operator's workload and prevent mishandling, such as tipping over. In particular, automatic control of flippers has been studied for more than a decade [6], [7], [8], [9], [10], [11]. Conventional research has proposed active control methods, such as use of sensors to measure obstacles and road surface geometry and mechanically calculate flipper control, or the use of motor torque or contact sensors mounted on crawler belts. However, these approaches do not discuss the generalization performance of the method for various environments and its evaluation.

To address these issues, this study proposed an automatic control flipper using reinforcement learning (RL). RL is an algorithm wherein a robot learns the optimal decision making by updating its action-value function through trial-and-error behavior. By acquiring and automating flipper control through RL, the operator can remotely control the rescue robot by simply specifying the direction of movement. Thus, the robot can be operated by a non-expert, reducing the workload on the operator. Furthermore, by coarse-graining the data input during RL, the training data can be reduced, thereby enabling learning in a realistic amount of time. In this study, an automatic flipper controlled by RL was developed, and experiments involving physical simulations were conducted to confirm whether the robot could step over steps and the stability of the robot.

The remainder of this paper is organized as follows. Section II discusses previous and related research. Section III proposes the method, which is the realization of adaptive behavior by controlling flippers of the rescue robot obtained through RL. Section IV presents computer simulation experiments using physical simulations employing the proposed method, and usefulness of the proposed method is indicated. Section V concludes the paper.

## II. Previous Research

In the rough terrain environment, rescue robots are equipped with a crawler belt to facilitate their own movement and a crawler belt for overcoming bumps and obstacles called a flipper or sub-crawler. Including the crawlers and

flippers required for movement, the rescue robot has 6 DOF, which is not intuitive for the operator. Therefore, research on teleoperation assistance in rescue robots, particularly partial automation of the crawler belt as a moving mechanism, has been conducted for more than a decade. Ohno *et al.* and Okada *et al.* are proposed active flippers of the rescue robot [6], [7]. These studies have tested multiple types of steps and have produced useful results. However, when assuming an unknown environment such as a disaster site, it is necessary to verify the effectiveness under various environmental shapes and conditions. Moreover, there are limits to the evaluation possible using only real robots.

As a successor to the above research, Rohmer *et al.* realized semi-autonomous control method over steps in a crawler type rescue robot [8], [9]. Similar to the above, an autonomous control system for the flipper was constructed and its ability to climb over steps was evaluated. However, this is simply a function of a part of the entire robot system and has not been discussed in depth.

Chen *et al.* and Kamezaki *et al.* developed arm mounted crawler type rescue robots [10], [11]. The robot had four arms and the flipper was highly maneuverable. In study [10], locomotion control method called compound motion pattern (CMP) for multi-limb robots was proposed. The actual robot could realize climbing of steps with semi-autonomous control. In study [11], instead of the two operators required to remotely control a robot, Kamezaki *et al.* developed a system that supported remote control with one operator and one autonomous system. These studies have achieved partial automation of functions for climbing over steps and have achieved various results. However, the verification of performance on stairs, uneven terrain, and random obstacle placement remains insufficient. Moreover, the actual environment is often unpredictable. Therefore, verification that takes randomness into account is necessary in the experimental environment, and there are limits to building a variety of environments in the laboratory, so a system that has been verified extensively through computer simulations needs to be applied in the real environment.

On the other hand, a flipper control method using machine learning has also been proposed, and the results learned using a physical calculation simulator have been applied to an actual robot, and very good results have been obtained [12], [13]. However, recent reinforcement learning using deep learning is high computational cost. Additionally, as sensors such as LiDAR have become cheaper in recent years, it is now possible to reflect higher-definition environmental shapes in physical calculation simulators.

## III. Proposed Method

### A. Proposed Method Overview

Previous research has primarily focused on evaluations such as operations on low stairs, and has not evaluated operations in environments with various shapes or those that simulate the actual environment. Furthermore, there is a lack of comprehensive discussion regarding environmental adaptation performance.

The overview of the proposed system is presented in Fig. 1. First, the robot model learns the flipper motion in advance through simulation. The learned results are transferred to the robot model in the practical phase and used when navigating through obstacles and stairs. Herein, the flipper is not operated by an operator; rather, it is controlled autonomously according to the environmental information observed from the sensor. Therefore, the rescue robot operator only needs to instruct the robot in the direction of movement, and the rescue robot can move over obstacles and stairs autonomously by operating its flippers. Thus, the operator no longer needs to control all 6DOF of the rescue robot, and only the controller is used to control at most the 2DOF motors that determine the direction of movement. Consequently, the burden on the operator in terms of remote operations is reduced.

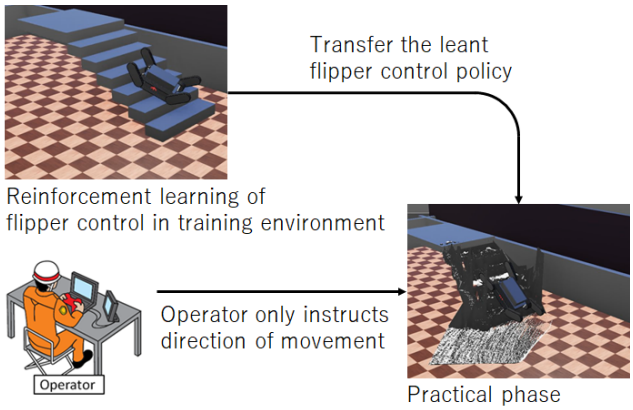



Fig. 1. Overview of proposed method.

The novelty of proposed method is that it provides randomness to the learning environment during the training stage. In addition, the actual environmental shape observed by LiDAR is used for evaluation. On the other hand, reinforcement learning is an algorithm that can discover solutions to behavioral rules through trial and error. By combining high-speed repetitive calculations by a computer, physical calculation simulations that take randomness into account, and reinforcement learning that can discover solutions, it is possible to give rescue robots unprecedented environmental adaptation performance. As shown in Fig. 1, wherein the training environment is a staircase and the height and depth of the steps of the stairs vary randomly within a certain range, the reinforcement learned action-value function exhibits good generalization performance. The robot model learns how to move the flipper in the training environment, and the acquired control is transferred to the robot model in the practical phase. Consequently, the flipper operates autonomously, and the rescue robot operator can overcome obstacles and run stairs simply by instructing the direction of movement. In this study, the robot model in the practical phase operates in a simulation to verify the method; however, in actual operation, the flipper control law learnt through RL is transferred to an actual rescue robot.

### B. Reinforcement Learning

RL is a machine learning method [14] that is modelled as the agent and the environment. The agent can interact with environment and the agent can perform an action $a$ ($\in A$) in the environment, which is described in state $s$ ($\in S$). By

executing an action, the state transition from current $s_t$ to $s_{t+1}$. The agent can observe the state $s$ and be rewarded $r$ from the environment. Then RL agent determines the optimal solution via trial-and-error and make its own policy to maximize the obtained rewards.

Many types of RL algorithms have been studied in decades. Q-learning, which is the most popular method, was adopted in this study [15]. Q-learning is defined as,

$$Q(s,a) \leftarrow Q(s,a) + \alpha\{r + \gamma \max_{a' \in A} Q(s',a') - Q(s,a)\}, \quad (1)$$

where, $s$ and $s' \in S$ are the states, $a \in A$ is the action, $\alpha$ is the learning rate ($0 < \alpha \leq 1$), $\gamma$ is the discount rate ($0 < \gamma \leq 1$), and $r$ denotes reward. Further, $Q(s,a)$ is the Q-function, and it is contains look-up tables called Q-table including all states and each action value pair.

When the agent selects an action based on $Q(s,a)$, the action selection function is used. In this research, the Boltzmann distribution type selection, a type of SoftMax method, was adopted. Action selection and selection probability calculation are described by

$$P(a|s) = \frac{\exp\{Q(s,a)/T\}}{\sum_{b \in A} \exp\{Q(s,b)/T\}}, \quad (2)$$

where, $T$ is a parameter that determines the randomness of the action selection.

### C. Environmental Setting

The learning process of RL requires several trials, and real-time RL with real robots is not practical owing to the learning time and number of trials. Therefore, in this study, as a learning environment, a simulation that repeated learning at high speed on a computer and was free from the possibility of a hardware failure of the robot was employed. As a computer simulation environment, a physical simulation system Webots was adopted [16]. This simulator can utilize the Open Dynamics Engine for physical calculations. Real time execution is also available and fast simulation mode can be selected based on computer performance. Python is used for programming of learning algorism, environmental definition, and robot model definition.

Fig. 2 presents an example simulation. Arbitrary objects can be generated for the environment in the simulation, and the objects can be configured based on sensor information acquired from the real environment. The robot's chassis, joints corresponding to motors for driving, crawler belts, and other components can also be configured in the simulation. Further, programmed and manual operations can be configured using a keyboard.

### D. Robot Configuration

This study was based on actual rescue robot for the system configuration, as shown in Fig. 3. Robot model (Fig. 3 (b)) was designed based on scale of actual rescue robot (Fig. 3 (a)). The robot in Fig. 3 (b) was equipped with various devices and sensors; however, the robot model used in the simulation was not equipped with these sensors; only the sensors necessary for RL were used.

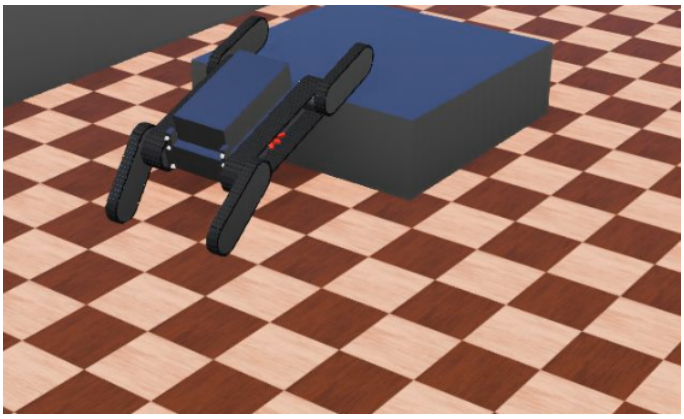
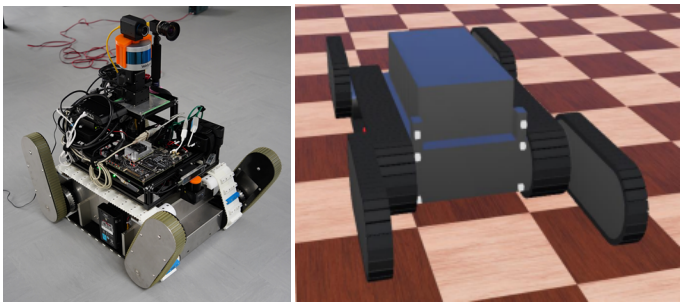


Fig. 2. Example of webots' simulation. The scene in this figure shows a rescue robot descending from a step while controlling flippers.



(a) Actual robot (b) Robot model

Fig. 3. Assumed actual crawler type rescue robot, and the robot model in the physics simulation. Body length, belt width, flipper's length, and width is set to actual size.

The robot model comprised the InertialUnit, GPS, and distance sensor to detect the object materials. Six distance sensors were installed at the front of the robot, and three on each side. The measurement direction of the distance sensor is shown in Fig. 4. InertialUnit was set such that the roll and pitch angles could be obtained in the range of $90°$ to $-90°$ with a resolution of nine. GPS was used to determine whether the robot moved forward or backward. The horizontal distance sensor acquired values $d$ [m] with four resolutions: $d < 0.3$, $0.3 \leq d < 0.6$, $0.6 \leq d < 1.0$, and $1.0 \leq d$. The upper and lower distance sensors acquired values with two resolutions with $d < 1$ or $1 \leq d$. The flipper could be controlled at 40, 20, 0, $-20$, $-40°$ with the horizontal direction as $0°$. The above settings reduced the state-action space $S \times A$ in the RL.

### E. Reward Design and Hyper Parameters

To realize flipper-based climbing of steps and stairs, the following reward functions were designed.

$$r(d_m) = \begin{cases} 0.05 & (d_m > 0.0) \\ -10 & (d_m \leq 0.0) \end{cases}, \quad (3)$$

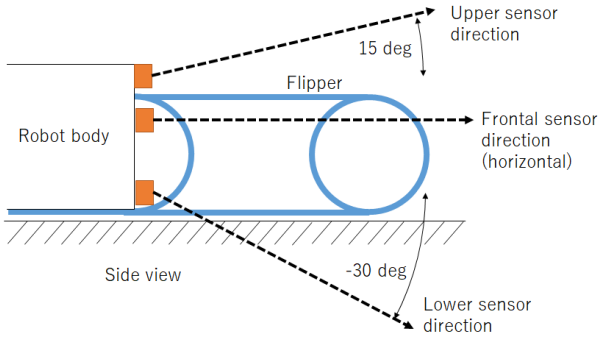



Fig. 4. Direction of distance sensor in side view. The robot is equipped with a distance sensor to measure the horizontal direction, sensor to measure the upper 15° to detect insurmountable obstacles, and sensor to measure the lower 30° to detect concavities.

$$r(\phi,\theta,\psi) = \begin{cases} -10 & (|\phi| > 45 \text{ deg}) \\ -10 & (|\theta| > 68 \text{ deg}) \\ -0.05 & (|\Delta\psi| > 0) \\ 0.05 & (\Delta\psi = 0) \\ -1 & (10 \text{ deg} \leq \Delta\theta^{\mathrm{f}} < 20 \text{ deg}) \\ -5 & (20 \text{ deg} \leq \Delta\theta^{\mathrm{f}} < 30 \text{ deg}) \\ -10 & (30 \text{ deg} \leq \Delta\theta^{\mathrm{f}}) \\ -1 & (10 \text{ deg} \leq \Delta\theta^{\mathrm{adv}} < 20 \text{ deg}) \\ -5 & (20 \text{ deg} \leq \Delta\theta^{\mathrm{adv}} < 30 \text{ deg}) \\ -10 & (30 \text{ deg} \leq \Delta\theta^{\mathrm{adv}}) \end{cases}, \quad (4)$$

where, $d_m$ is the distance the robot moves after selecting an action, $\phi, \theta, and \psi$ are the posture angles of the robot, and is a parameter generally expressed as a posture vector, which corresponds to roll, pitch, and yaw, respectively. Further, $\Delta\psi$ indicates the amount of change in the yaw angle over minute time. Both $\Delta\theta^{\mathrm{f}}$ and $\Delta\theta^{\mathrm{adv}}$ are the amount of change in the pitch angle, and they indicate the amount of change in pitch angle when the flipper is controlled and when the robot moves forward, respectively. Reward $r$ of (1) is calculated using $r(d_m)$ and $r(\phi,\theta,\psi)$ as follows:

$$r = r(d_m) + r(\phi,\theta,\psi), \quad (5)$$

This method is reward shaping like implementation [17]. Learning parameters for RL were set as: learning rate $\alpha$ of 0.1, discount rate $\gamma$ of 0.9, and temperature value for Boltzmann distribution $T$ of 0.2 to select action.

### F. Reusing of Action-value Function

When the rescue robot was used obtained action-value function in training environment to new environment, techniques were leveraged based on transfer learning in RL (hereinafter transfer RL) [18], [19]. The transfer learning proposed by Taylor *et al.* learning is a framework wherein an RL agent reuses the policies and action-value functions learned and acquired in a source task in another task called a target task. In the RL of this research, as an action-value function is acquired, a value function transfer type transfer reinforcement learning is used, and it is defined as the following equation.

$$Q_t(s_t,a_t) \leftarrow Q_t(s_t,a_t) + Q_s(\chi(s_t),\chi(a_t)). \quad (6)$$

where, $Q_t(\cdot)$ is action-value function in target task, and $Q_s(\cdot)$ is obtained action-value function in source task. Further, the function $\chi$ is called inter-task mapping, it has the function of mapping $a$ and $s$ of the source task to $a$ and $s$ of the target task. Inter-task mapping is considered transferring action-value function among heterogeneous agents, it is defined as $\chi : S_t \mapsto S_s|_{s_t \in S_t, s_s \in S_s}$ and $\chi : A_t \mapsto A_s|_{a_t \in A_t, a_s \in A_s}$. Further, $S_t$ and $S_s$ are the sets of the state $s$ in the target and source tasks, respectively. In addition, $A_t$ and $A_s$ are the sets of the action $a$ in the target and source tasks, respectively. When the agents are homogeneous between tasks, inter-task mapping is not required for the transfer. In this case (6) is modified as follows:

$$Q_t(s,a) \leftarrow Q_t(s,a) + \tau Q_s(s,a), \quad (7)$$

where, the parameter $\tau (0 \leq \tau \leq 1)$ is called the transfer rate, and is used to avoid negative transfer such as phenomenon like an over learning [20], [21]. Eq. (7) implies that the action-value function of the target task agent was initialized by the sum of the initial value of the action-value function of that agent and the action-value function of the source task to be reused. In this study, $\tau$ was set as 0.5.

## IV. Experiment using Physics Simulation

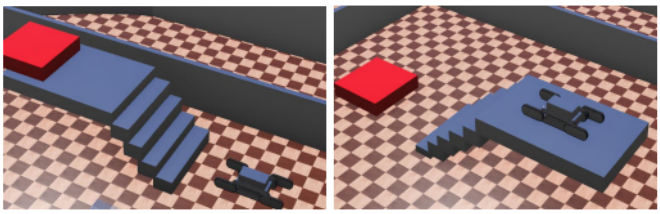
### A. Conditions

We conducted a comparative experiment to confirm the usefulness of automatic flipper control of a rescue robot using RL. The comparison target was a rescue robot whose flipper control angle was fixed at 45°. This is because when comparing the proposed method with human remote control, humans cannot always perform the same operation every time, thereby rendering it difficult to perform quantitative evaluation with reproducibility in mind.

Three environments were prepared to compare the running performance of the proposed method and the fixed flipper, and are described below along with the training environment for reinforcement learning.

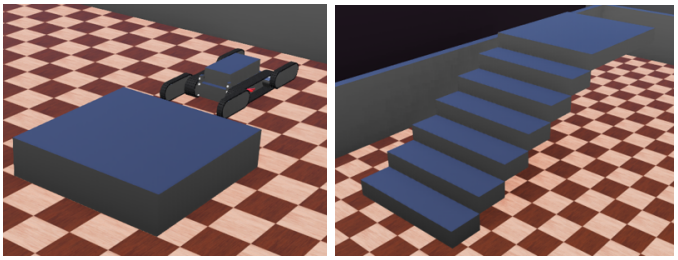
*1) Training environment:* In the training phase, stairs were used as the type of environment, and the robot performed RL on the behavior of climbing and descending the stairs (see Fig. 5). The robot's body crawler only moved forward and performed RL to control the flippers during the process of climbing and descending stairs. The climbing and descending environments of the stairs changed randomly at every 100 episodes. The height of one step of the stairs changed randomly with the height $H$ [m] being $0 \leq H \leq 0.3$ and the depth $D$ [m] being $0.15 \leq D \leq 0.2$ for each episode. The number of stairs was fixed at five.

The flipper control timing was selected and executed when the robot body moved 0.3 [m]. Simultaneously, the action-value function was updated via Q-learning.



(a) Climbing situation (b) Descending situation

Fig. 5. Learning environment when the training. The robot performs reinforcement learning on the shapes of the ascending and descending stairs, and the generated action-value function is common. In this figure, the red blocks indicate the goal areas of each environment, and are deliberately visualized. It is not displayed in the actual simulation.



(a) Single-step stair (b) Multi-step stairs

Fig. 6. Artificially created single-step and multi-step stairs. In contrast to the training environment, single- and multi-step stairs have fixed step heights and depths.

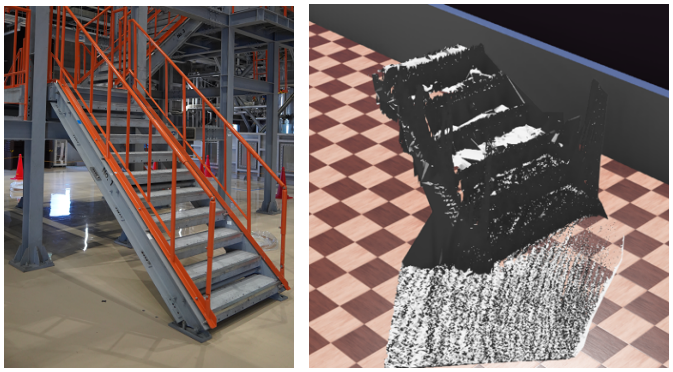
*2) Single-step environment:* To confirm the generalization performance of the action-value function, which learned through RL in the training environment, a single-step stair was prepared for the Webots. Fig. 6(a) shows that the single-step stair, and its height was such that the rescue robot could not climb over it when running with its flippers fixed at 45°. In contrast to the training environment, this stair was fixed step height and depth.

*3) Multi-step environment:* As a second environment, multi-step stairs were prepared in Webots, as shown in Fig. 6(b). Multi-step stairs were constructed by measuring the actual stairs in Building 10 of Tokyo Polytechnic University Atsugi Campus. Similar to the single-step environment, the stairs had fixed step height and depth.

*4) LiDAR data environment:* In this environment setting, the actual stairs were scanned with LiDAR and reconstructed in Webots. Fig. 7(a) shows that the actual stairs was built at the Naraha Center for Remote Control Technology Development (NARREC) of Japan Atomic Energy Agency (JAEA)[22]. The stairs were scanned in advance by LiDAR, and the stairs object reconstructed from the point cloud data in Webots is shown in Fig. 7(b). The number of stairs was also fixed at seven based on actual stairs.

### B. Evaluation Factor

In this experiment, the robot could climb steps and stairs, and the amount of change in posture angle could be determined, based on which we calculated angle data of the robot body in Webots for evaluation. The amount of change in the posture angle implies the vibration of the robot body when it is moved. A small amount of change in posture angle indicates that there are few movements that involve sudden changes. Vibration evaluation contributes to improved driving stability, less hazardous travel, and fewer hardware failures. Attitude angle change amount is calculated as follows:



(a) Mockup stairs (b) Reconstructed stairs

Fig. 7. LiDAR data environment is generated for physics simulation based on pointcloud data. The point cloud data is measured in an actual stair-way using Velodyne VLP-16 sensor. The stairs are built as a mockup based on stairs of the Fukushima-daiichi nuclear power station.

$$\frac{\Delta\eta}{\Delta t} = \frac{\sqrt{(\phi_{t+\Delta t} - \phi_t)^2 + (\theta_{t+\Delta t} - \theta_t)^2}}{(t + \Delta t) - t}, \tag{8}$$

where, $\phi$ is roll angle of the robot body, $\theta$ is pitch angle of the robot bod,. $t$ is time, and $\Delta t$ is min time as sampling time; however, it is dependent on the calculation of the time step of simulation setup. A yaw angle $\psi$ was not considered in this evaluation because yaw angle has probability to adjust owing to control of the moving direction of the rescue robot. $\omega = \Delta\eta/\Delta t$ means pseudo angler velocity. The posture angle was observed every 0.1 s.

### C. Results

*1) Training environment:* The result of RL in random stairs (Fig. 5) is shown in Fig. 8. The order of climbing and descending stairs, which changed every 100 episodes, is presented in Table I. In Fig. 8, the number of steps on the vertical axis is the number of actions required by the rescue robot from the start point to the goal point. The number of episodes on the horizontal axis is the iteration of learning, with movement from the start to the goal being one episode. From the downward trend of the learning curve, the rescue robot learns how to move using its flippers even in environments with random steps and switches between going up and down.

*2) Single-step environment:* First, an image of the rescue robot's behavior is explained in Fig. 9, 11, 10 and 12 in single-step environment. As shown in Fig. 9, in the fixed flipper condition, the rescue robot could not climb a single-step and tipped over. Whereas, automatic control flipper condition could realize climbing of a single-step utilizing rear flippers, as shown in Fig. 10. Both the descending results indicated that the rescue robots could descend without falling because single-step descending is not a difficult situation, as shown in Fig. 11 and 12.

TABLE I. CHANGE IN TYPE OF STAIRS

| Episode num. | Type |
|---|---|
| 1 | Climb |
| 101 | Climb |
| 201 | Climb |
| 301 | Descend |
| 401 | Climb |
| 501 | Descend |
| 601 | Climb |
| 701 | Descend |
| 801 | Climb |
| 901 | Climb |

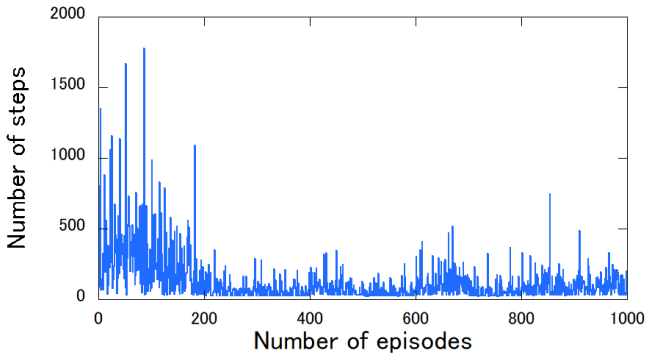



Fig. 8. Result of learning curve in training environment. At the beginning of learning, number of steps has large value. As the episode progresses, the number of steps reduces. Moreover, when the type of stairs changes indicated in TableI, a convergence tendency is observed. Thus, it is considered that the training environment can be learnt.

*3) Multi-step environment:* Fig. 13 and 14 show the robot's movements under multi-step environment with fixed flipper configuration. Under the fixed flipper condition, the rescue robot could climb and descend steps. However, when the rescue robot climbed the stairs, the flippers were fixed; thus, it is expected that there will be a large impact when the robot lands. It is also expected that an impact will be generated when entering the descending stairs because the flippers were fixed. Fig. 15 and 16 show the robot's movements under multi-step environment with automatic flipper control configuration using the proposed method. When the rescue robot climbed the stairs, it used flippers to increase the number of points it touched. Furthermore, as the flippers were in front of the robot when it finished climbing the stairs, it is expected that the impact of landing would be alleviated. When the rescue robot entered the descending stairs, the flipper protruded in front of the robot body, which is considered to reduce the impact upon entry. Under these experimental conditions, the rescue robot could run with or without flipper control.

*4) LiDAR data environment:* The rescue robot's behavior is explained in Fig. 17, 18, 19 and 20 in a LiDAR data environment. Fig. 17 and 18 show the robot's movements under LiDAR data environment with fixed flipper configuration. Under these experimental conditions, the rescue robot with fixed flippers could not climb the stairs and overturned. On the descending stairs, the rescue robot with fixed flippers could descend, but it appeared to be slipping while running down the stairs. Fig. 19 and 20 show the robot's movements under LiDAR data environment with automatic flipper control configuration using the proposed method. The rescue robot whose flippers were controlled using the proposed method could climb the stairs in the LiDAR data, and when it finished climbing, it moved its flippers slightly forward and downward to soften the impact of landing. On descending stairs, when the rescue robot entered the stairs, it moved the rear flipper downwards to soften the impact when entering the stairs. Under the flipper control conditions, it is considered that the contributing factors were that the robot moved without slipping on the stairs and reduced the impact upon entry.

TABLE II. SUMMERY OF RUNNING RESULTS

| Environments | Fixed flipper | Automatic control |
|---|---|---|
| Single-step (climb) | failed | success |
| Single-step (descend) | success | success |
| Multi-step (climb) | success | success |
| Multi-step (descend) | success | success |
| LiDAR data (climb) | failed | success |
| LiDAR data (descend) | success | success |

### D. Summary of Results

In summary, we concluded that the rescue robot could travel through each environment, as presented in Table II. Under the fixed flipper conditions, the rescue robot rolled over and could not climb the stairs in environments when climbing single-step and in case of LiDAR data. Moreover, slipping occurred during movement in the case of stairs, resulting in unintended control. However, the rescue robot that implemented the flipper control law using reinforcement learning could run under all experimental conditions.

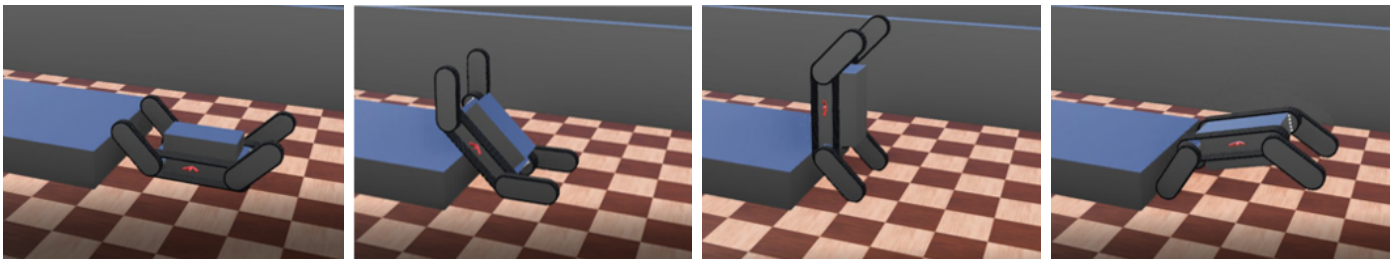
Fig. 21 shows the time series data of the angular velocity measured under each experimental condition.

### E. Discussions

To quantitatively demonstrate the usefulness of automatic flipper control using the proposed method, this study used the expected value and variance of the occurrence probability distribution for the evaluation index $\Delta\eta/\Delta t$, which is defined as (8). Considering $\Delta\eta/\Delta t$ observed time series data $\Delta\eta/\Delta t$ were replaced for the distribution. It can be expressed as a probability distribution by taking the possible angular velocity as frequency of the probability of occurrence $P(\omega_i)$. Fig. 22 presents the angular velocity distribution for each experimental result. When calculating the distribution, it was experimentally clear that angular velocities of 0.1 [deg/s] or less in the time series data were vibrations of the rescue robot while it was running; thus, the distribution was calculated by filtering data with $\Delta\eta/\Delta t$ greater than 0.1 [deg/s].

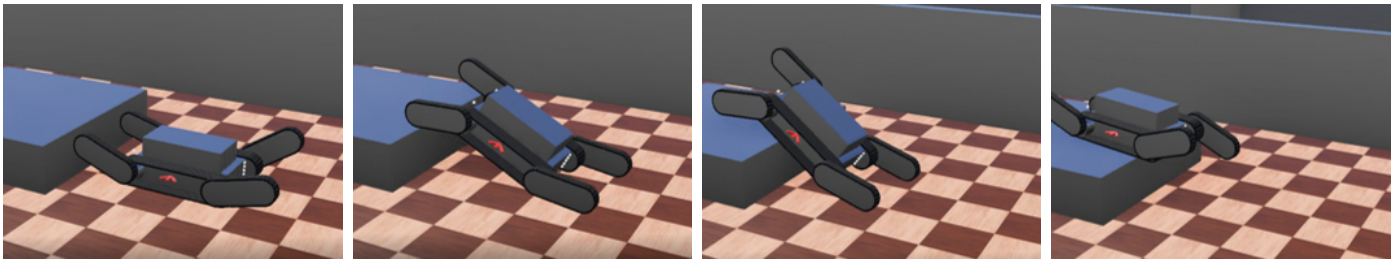
In the Fig. 22, single-step environment exhibited no significant difference in the shape of the distribution; however, other distributions, the size and spread of the peak differed between the fixed and automatically controlled flipper conditions. Therefore, in this study, to quantitatively evaluate the differences in these distributions, the expected value and variance of the distributions were calculated and compared for each experimental result.

Expected value of angular velocity $\omega_i$ in each experimental conditions is defined as follows:



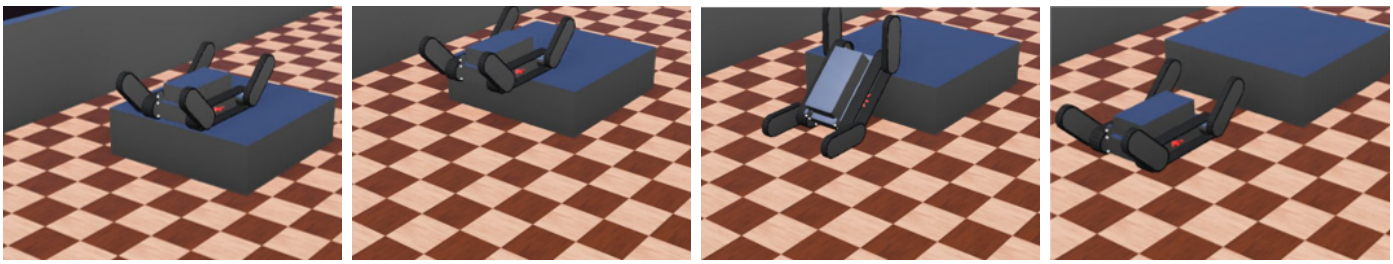
(a) (b) (c) (d)

Fig. 9. Rescue robot climbing operation result under the single-step environment with fixed flipper condition.



(a) (b) (d) (f)

Fig. 10. Rescue robot climbing operation result under the single-step environment with automatic control flipper condition.



(a) (b) (c) (d)

Fig. 11. Robot operation descending result under the single-step environment with fixed flipper condition.

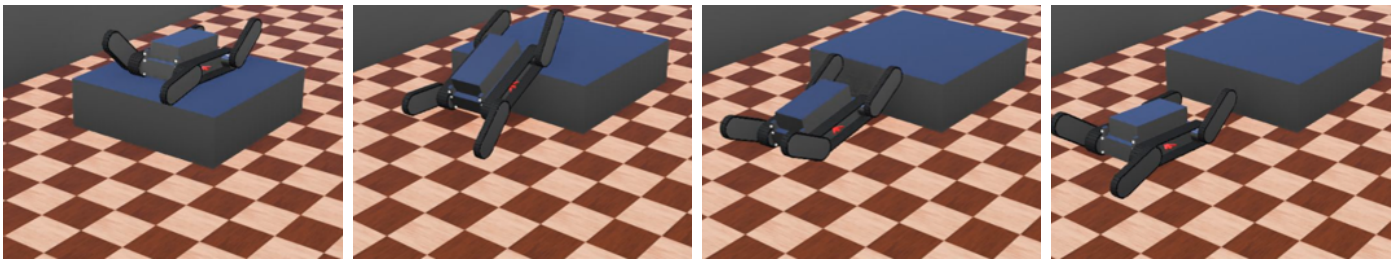
$$E(\Omega) = \sum_i \omega_i P(\omega_i). \tag{9}$$

Here, $\Omega$ is the set of $\omega_i$. Note that, in this case $E(\Omega) = \bar{\omega_i}$ because the expected value was not calculated from a classified distribution, but from the measured time-series angular velocity data. $\bar{\omega_i}$ is average of $\omega_i$. Variance of calculated angular velocity $\omega_i$ is defined as following:

$$V(\Omega) = \sum_i (\omega_i - \mu)^2 P(\omega_i). \tag{10}$$

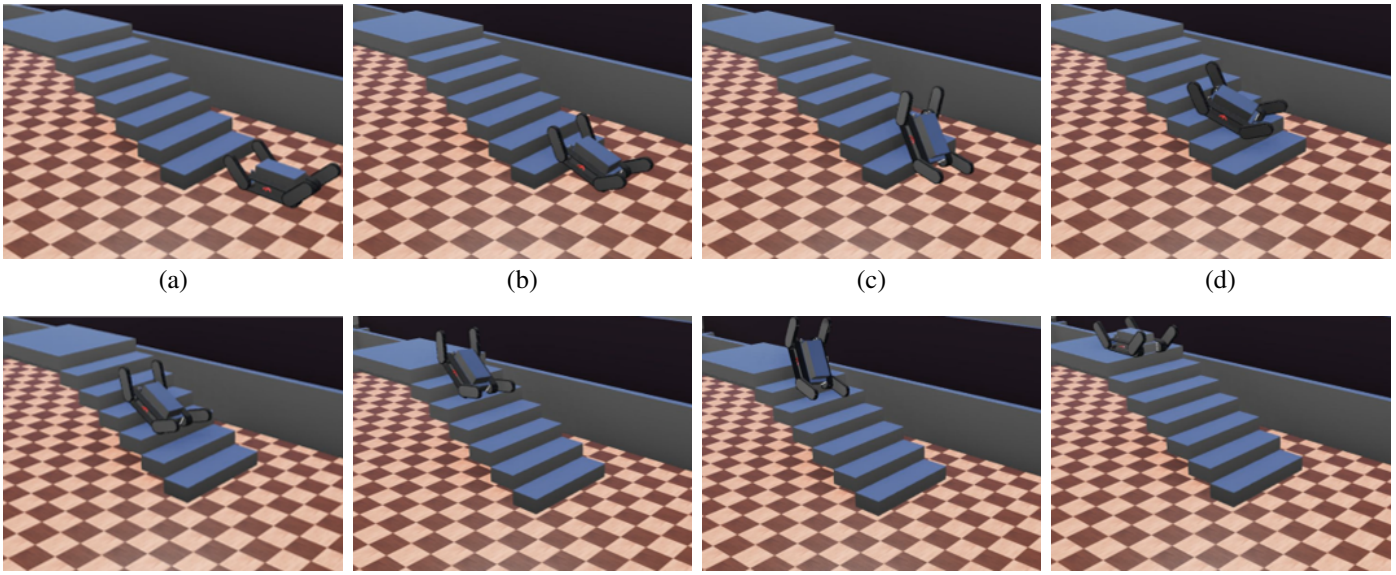
Here $\mu = E(\Omega)$ is defined for above equation. The expected values and variances for each experimental condition are presented in Table III. Note that climbing in the single-step and LiDAR data environment under the fixed flipper condition was not possible owing to a fall; thus, this was excluded from the discussion but will be listed in parentheses in the Table III. In the single- and multi-step environments, the expected value of automatic control of the flipper shifted to be smaller than the expected value of fixed flipper. Similarly, the variation was also narrower. This is because the rescue robot equipped with a flipper that was automatically controlled by the proposed method had a lower angular velocity when climbing up and down stairs than in the case of a fixed flipper. Thus, the vibration decreased. In addition, by narrowing the validation, it is suggested that the number of types of angular velocity was reduced and the vibration became less complex.

In the LiDAR data environment, although there was no large difference in value, both the expected and validation values were larger in the automatic flipper condition than in the fixed flipper condition. This is thought to be owing to the fact that the shape of the environment was composed of data obtained from LiDAR and had a complex shape that included unevenness. Although the usefulness of the proposed method cannot be clearly observed from the numerical values, considering that the rescue robot could run the stairs, it can be said that the proposed method can contribute to running the stairs compared to the fixed flipper condition.



(a) (c) (e) (f)

Fig. 12. Rescue robot descending operation result under the single-step environment with automatic control flipper condition.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 13. Rescue robot climbing operation result under the multi-step environment with fixed flipper condition.

TABLE III. COMPARISON OF EXPECTED VALUES AND VALIDATIONS. THE NUMBERS IN PARENTHESES ARE FOR REFERENCE ONLY

| | $E(\Omega)$ | | $V(\Omega)$ | |
|---|---|---|---|---|
| Environments | Fix | Auto | Fix | Auto |
| Single-step (climb) | (4.96) | 4.24 | (12.54) | 7.17 |
| Single-step (descend) | 6.67 | 4.68 | 32.30 | 14.57 |
| Multi-step (climb) | 5.01 | 3.30 | 10.68 | 5.44 |
| Multi-step (descend) | 5.41 | 3.10 | 11.71 | 4.77 |
| LiDAR data (climb) | (8.73) | 1.81 | (1609.12) | 11.01 |
| LiDAR data (descend) | 3.41 | 4.09 | 3.42 | 8.24 |

## V. CONCLUSION

This study described that tele-operation of the rescue robot is difficult for human operator in disaster response situations. Thus, automatic flipper control using RL and physics simulation, was proposed. In the proposed method, the rescue robot's flipper control was trained using stairs of random heights in advance through simulation to provide generalization performance. Thereafter, as an evaluation of the proposed method, it was confirmed that the running performance of the learning resulted in a realistic staircase environment obtained using LiDAR from the actual environment, including a single step and multiple steps. As the experimental results, compared to a rescue robot with a fixed flipper condition, the flipper control using the proposed method tended to have less vibration during movement, suggesting that it reduced risks at the store and damage to the rescue robot itself.

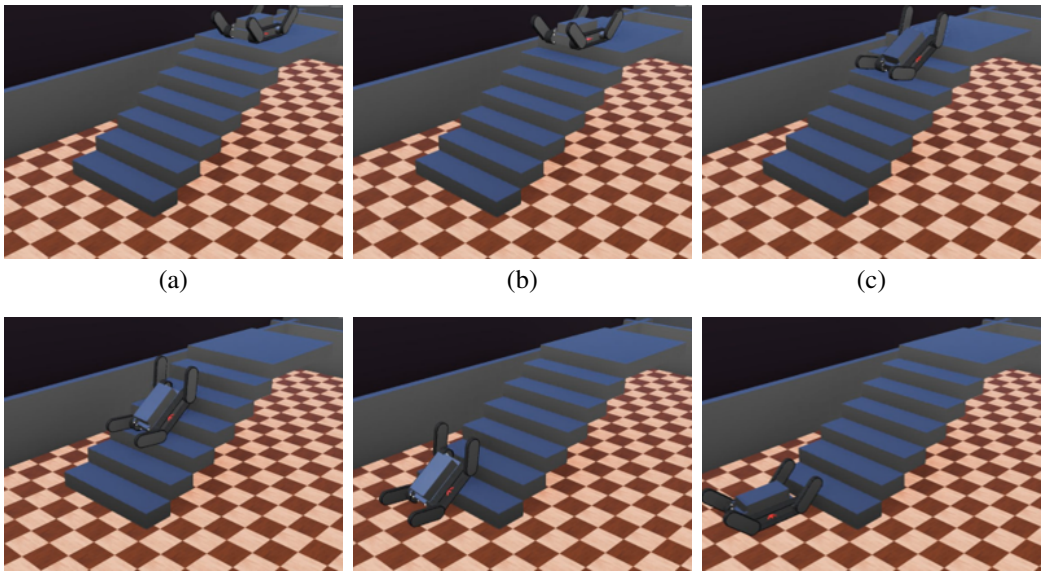
In the future works, it will be necessary to transfer the learning results using the proposed method in this study to an actual rescue robot and verify whether it is possible to travel as expected. Furthermore, although this study evaluated artificial single- and multi-step environments, it is thought that the environment shape obtained by LiDAR is the most effective learning environment for actual tasks. Therefore, it is also important to verify the effectiveness of the proposed method through computer simulations using environmental shapes measured by LiDAR in various other environments.

## ACKNOWLEDGMENT

We would like to thank Editage (www.editage.jp) for English language editing.

## REFERENCES

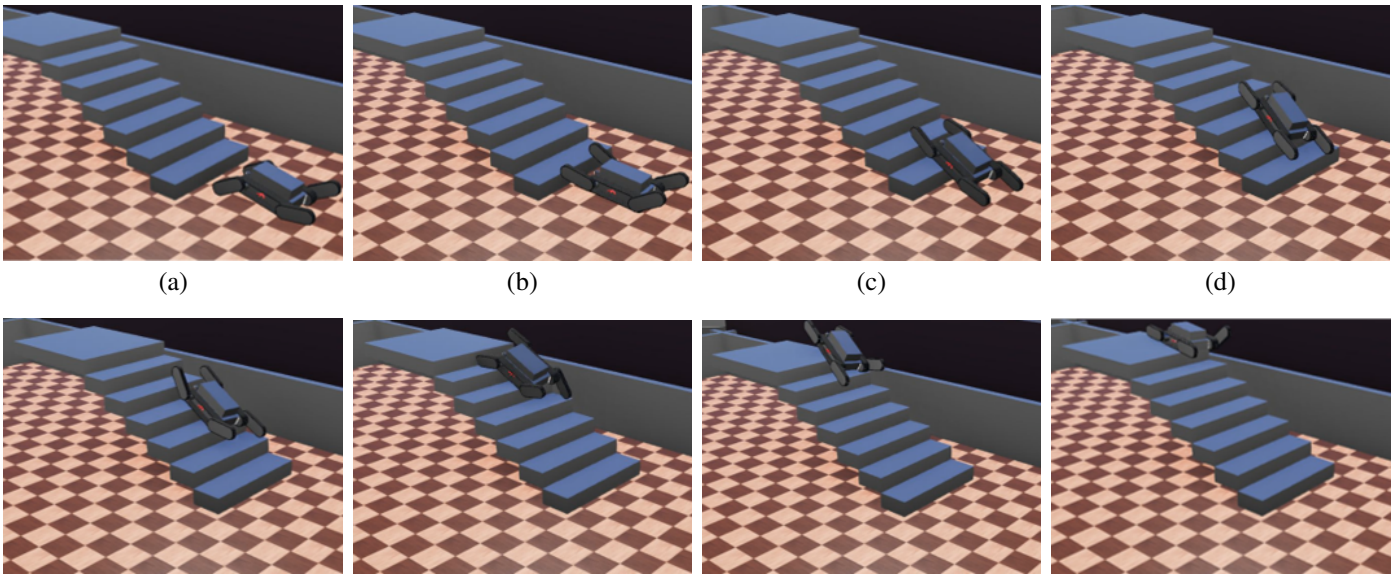
[1] R. R. Murphy, "Trial by fire," IEEE Robot. Autom. Mag., Vol. 11, pp. 50-61, 2004.



(a) (b) (c)

(d) (e) (f)

Fig. 14. Rescue robot descending operation result under the multi-step environment with fixed flipper condition.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 15. Rescue robot climbing operation result under the multi-step environment with automatic control flipper condition.

[2] K. Kawabata, "Toward technological contributions to remote operations in the decommissioning of the Fukushima Daiichi Nuclear Power Station," Japanese J. Appl. Phys., vol. 59, no. 5, 050501, 2020.

[3] L. Battistuzzi, C. T. Recchiuto and A. Sgorbissa, "Ethical concerns in rescue robotics: a scoping review," Ethics Inform. Technol.. vol. 23, no. 4, pp. 863-875, 2021.

[4] K. Nagatani, S. Kiribayashi, Y. Okada, S. Tadokoro, T. Nishimura, T. Yoshida, E. Koyanagi and Y. Hada, "Redesign of rescue mobile robot Quince," In Proceedings of the 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics, pp. 13-18, 2011.

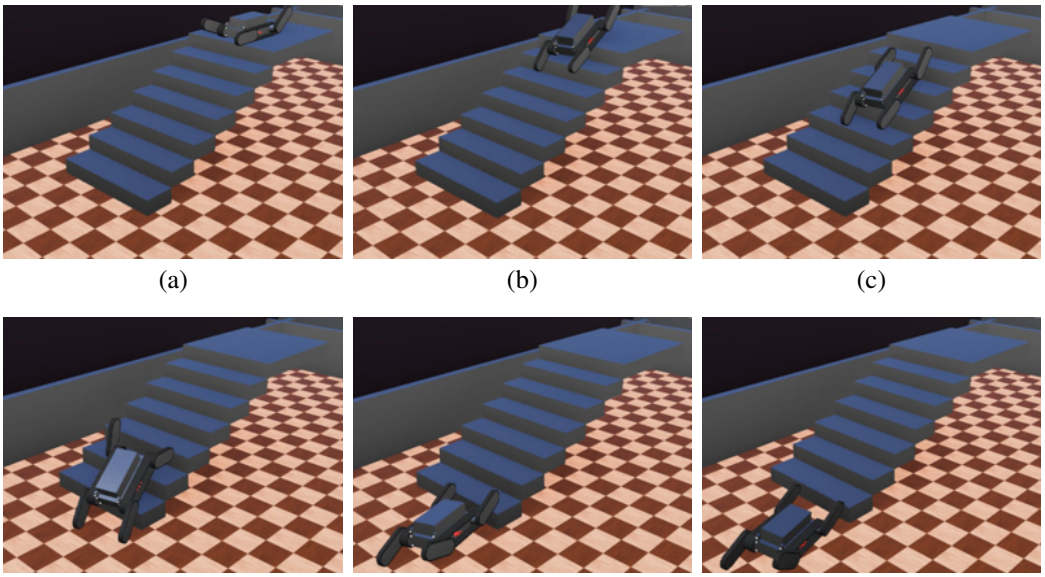
[5] T. Ito, H. Kono, Y. Tamura, A. Yamashita and H. Asama, "Recovery Motion Learning for Arm Mounted Mobile Crawler Robot in Drive System's Failure," In Proceedings of the 20th World Congress of the International Federation of Automatic Control (IFAC2017), pp. 2365-2370, 2017.

[6] K. Ohno, S. Morimura, S. Tadokoro, E. Koyanagi and T. Yoshida, "Semi-autonomous control system of rescue crawler robot having flippers for getting Over unknown-Steps," In Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3012-3018, 2007.

[7] Y. Okada, K. Nagatani and K. Yoshida, "Semi-autonomous operation of tracked vehicles on rough terrain using autonomous control of active flippers," In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2815-2820, 2009.

[8] E. Rohmer, T. Yoshida, K. Ohno, K. Nagatani, S. Tadokoro and E. Konayagi, "Quince : A Collaborative Mobile Robotic Platform for Rescue Robots Research and Development," The Abstracts of the international conference on advanced mechatronics : toward evolutionary fusion of IT and mechatronics : ICAM , vol. 5, pp. 225-230, 2010.

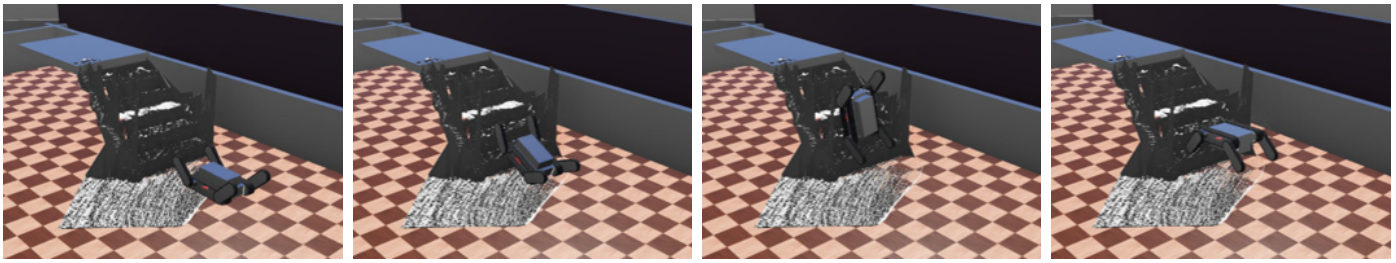
[9] E. Rohmer, K. Ohno, T. Yoshida, K. Nagatani, E. Konayagi and S.



(a) (b) (c)

(d) (e) (f)

Fig. 16. Rescue robot descending operation result under the multi-step environment with automatic control flipper condition.



(a) (b) (c) (d)

Fig. 17. Rescue robot climbing operation result under the LiDAR data environment with fixed flipper condition.

Tadokoro, "Integration of a sub-crawlers' autonomous control in Quince highly mobile rescue robot," In Proceedings of the 2010 IEEE/SICE International Symposium on System Integration, pp. 78-83, 2010.

[10] K. Chen, M. Kamezaki, T. Katano, T. Kaneko, K. Azuma, T. Ishida, M. Seki, K. Ichiryu and S. Sugano, "A semi-autonomous compound motion pattern using multi-flipper and multi-arm for unstructured terrain traversal," In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2704-2709, 2017.

[11] M. Kamezaki, T. Katano, K. Chen, T. Ishida and S. Sugano, "Preliminary study of a separative shared control scheme focusing on control-authority and attention allocation for multi-limb disaster response robots," Adv. Robot., 34(9): 575-591, 2020.

[12] M. Pecka, V. Šalanský, K. Zimmermann and T. Svoboda, "Autonomous flipper control with safety constraints." In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2889-2894, 2016

[13] H. Pan, X. Chen, J. Ren, B. Chen, K. Huang, H. Zhang and H. Lu, "Deep Reinforcement Learning for Flipper Control of Tracked Robots in Urban Rescuing Environments." Remote Sensing, 15(18): 4616, 2023.

[14] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," MIT press, 1998.

[15] C. J. C. H. Watkins and P. Dayan, "Q-Learning," Mach. Learn., vol. 8 pp. 279-292, 1992.

[16] Cyberbotics: Robotics simulation with Webots, https://cyberbotics.com/ (Access 2023-09-14).

[17] A. Y. Ng, D. Harada and S. Russell, Policy invariance under reward transformations: Theory and application to reward shaping. In Proceedings of the Sixteenth International Conference on Machine Learning, pp. 278-287, 1999.

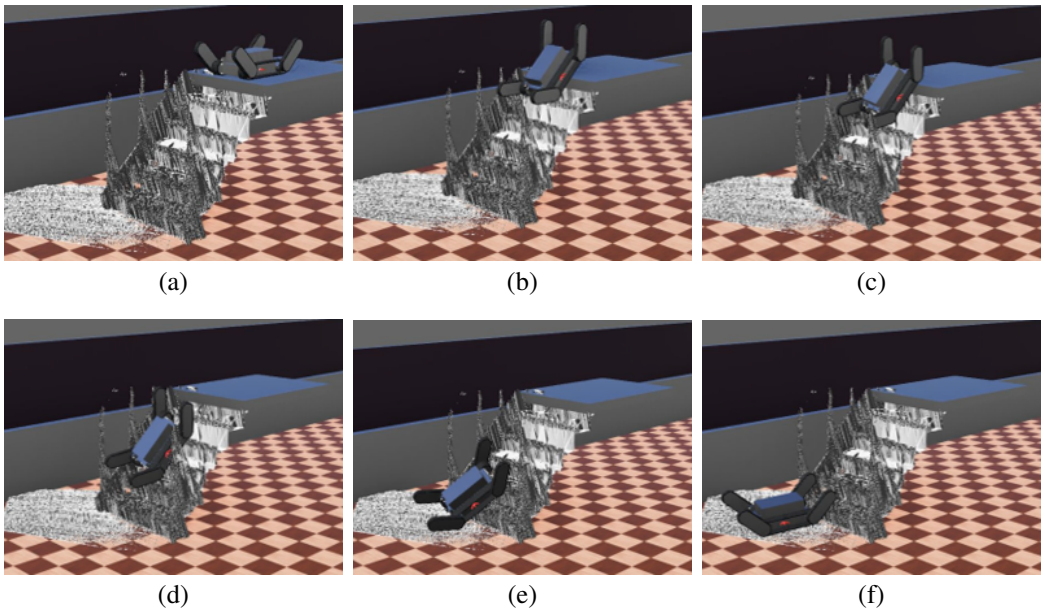
[18] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," J. Mach. Learn. Res., vol. 10 pp. 1633-1685, 2009.

[19] M. E. Taylor, "Transfer in Reinforcement Learning Domains," Studies in Computational Intelligence 216, Springer, 2009.

[20] H. Kono, Y. Sakamoto, Y. Ji, and H. Fujii, "Automatic Transfer Rate Adjustment For Transfer Reinforcement Learning," Int. J. Artif. Intell. Appl., vol. 11, no. 5/6, pp. 47-54, 2020.

[21] T. Takano, H. Takase, H. Kawanaka, H. Kita, T. Hayashi and S. Tsuruoka, "Transfer Learning based on Forbidden Rule Set in Actor-Critic Method." Int. J. Innov. Comput. Inform. Control, vol. 7, no. 5(B), pp. 2907-2917, 2011.

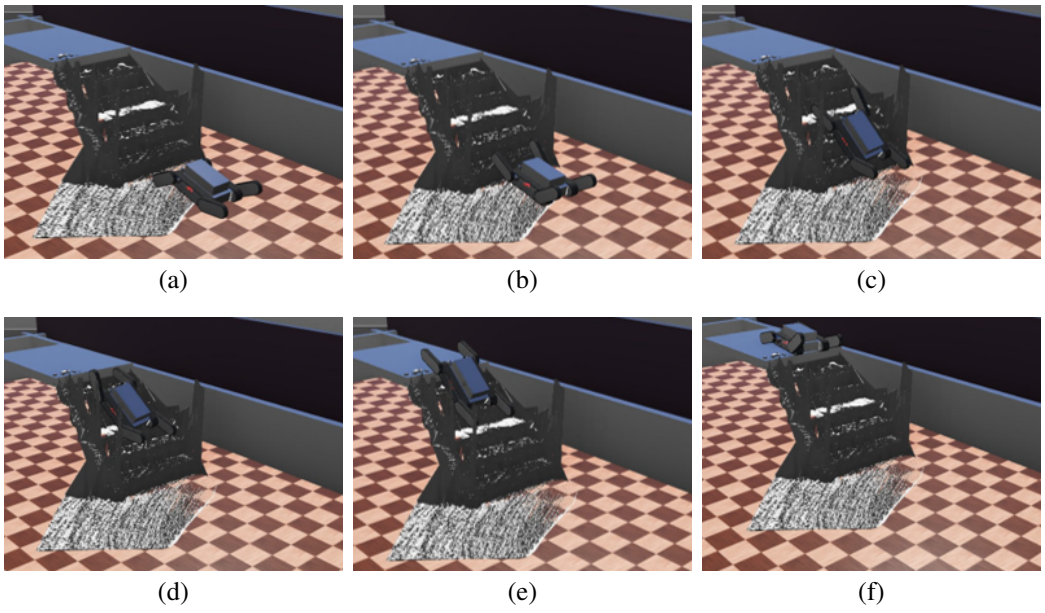
[22] Japan Atomic Energy Agency, Naraha Center for Remote Control Technology Development, https://naraha.jaea.go.jp/en/index.html (Access 2023-11-10).



(a) (b) (c)

(d) (e) (f)

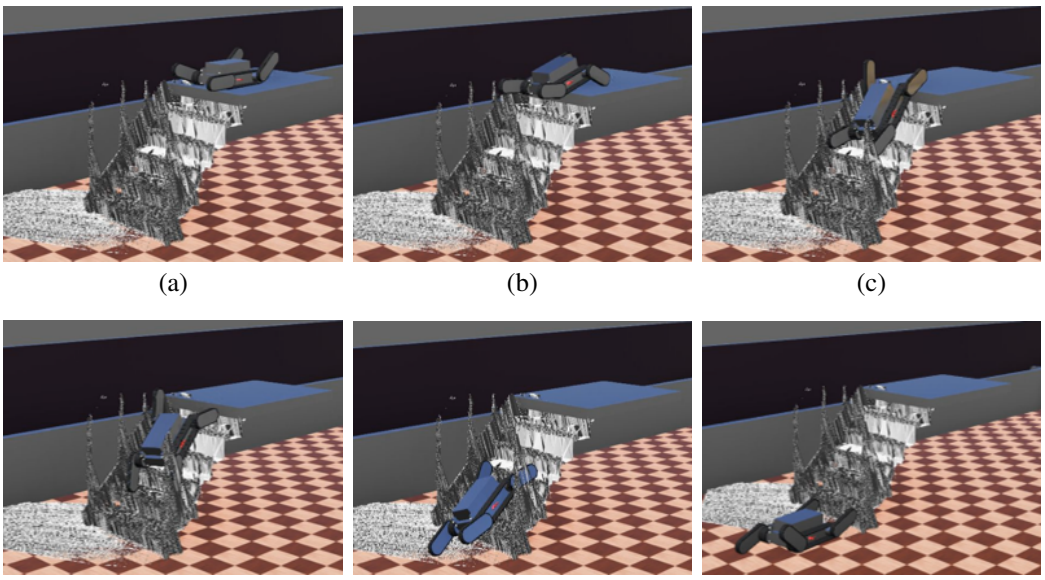
Fig. 18. Rescue robot descending operation result under the LiDAR data environment with fixed flipper condition.



(a) (b) (c)

(d) (e) (f)

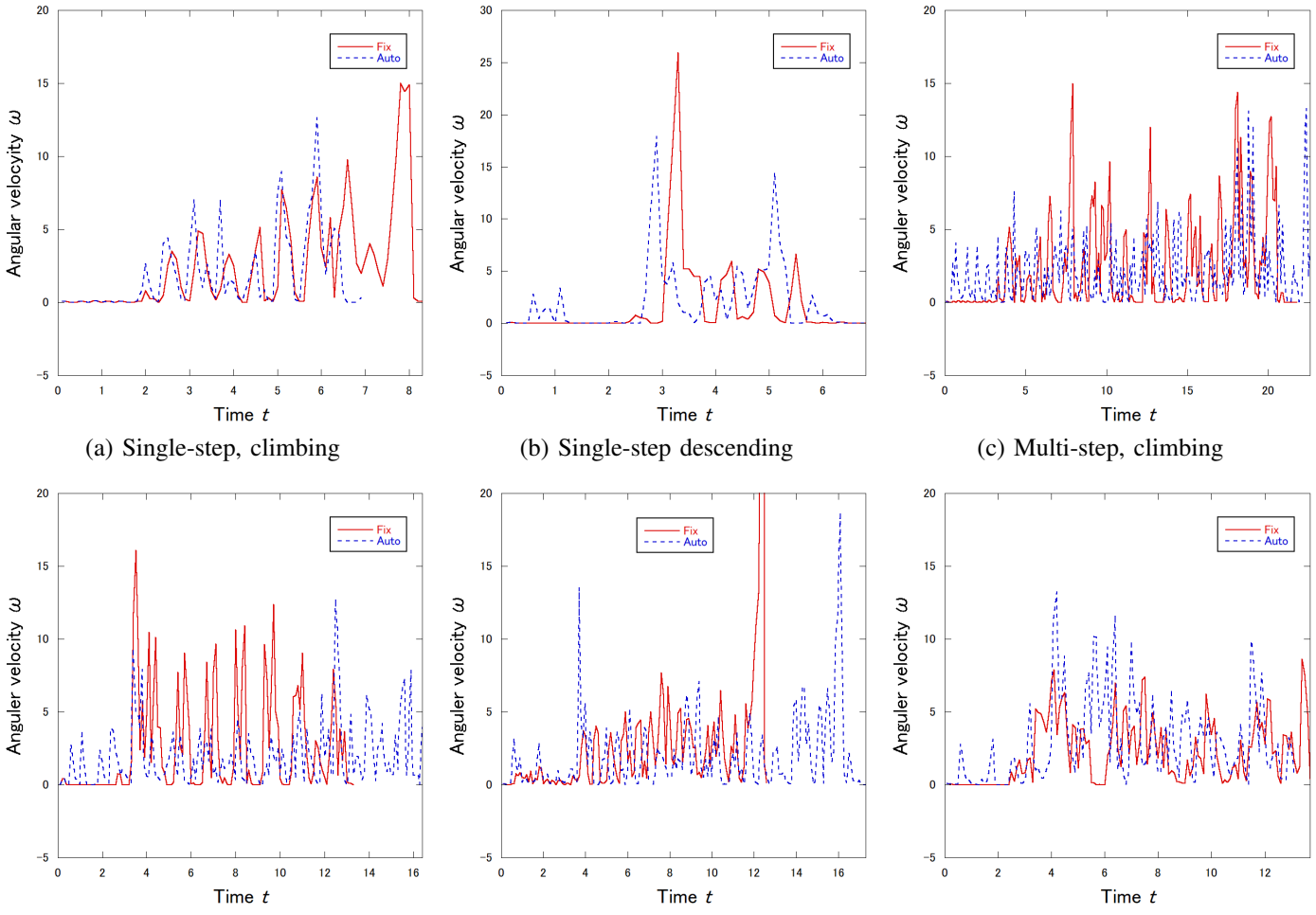
Fig. 19. Rescue robot climbing operation result under the LiDAR data environment with automatic control flipper condition.



(a) (b) (c)

(d) (e) (f)

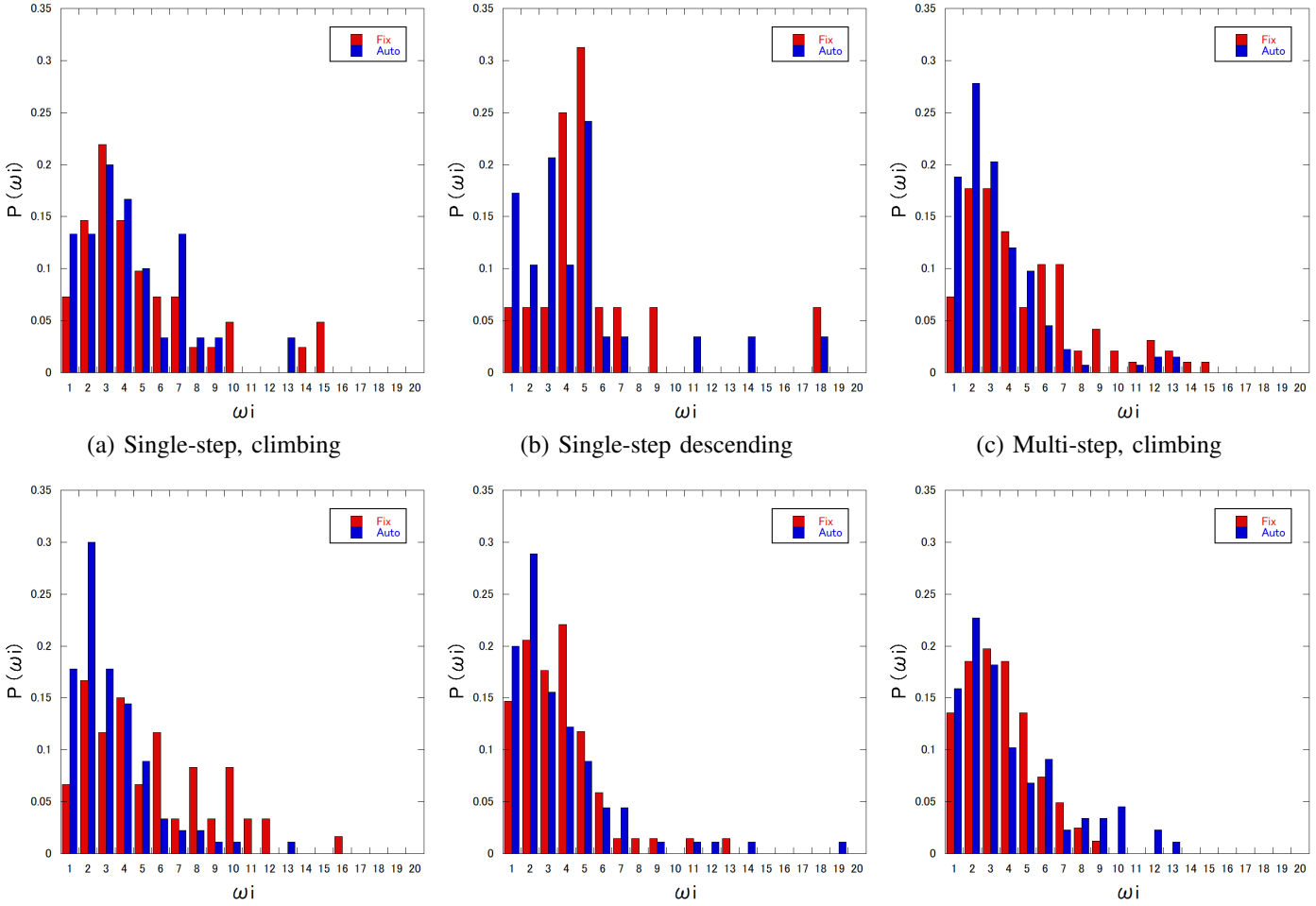
Fig. 20. Rescue robot descending operation result under the LiDAR data environment with automatic control flipper condition.




(a) Single-step, climbing (b) Single-step descending (c) Multi-step, climbing

(d) Multi-step, descending (e) LiDAR data, climbing (f) LiDAR data, descending

Fig. 21. Time series data of measures angular velocity under each experimental condition. In this figure, “Fix” is the condition for a fixed flipper, and “Auto” is a condition for automatic control of the flipper learned by reinforcement learning.




(a) Single-step, climbing (b) Single-step descending (c) Multi-step, climbing

(d) Multi-step, descending (e) LiDAR data, climbing (f) LiDAR data, descending

Fig. 22. Angular velocity distribution in each experimental result. In this figure, “Fix” is the condition for a fixed flipper, and “Auto” is a condition for automatic control of the flipper learned by reinforcement learning.