Unified motion planning method using random network and gradient method for multifunctional underwater robots

Koichiro Shiraishi[1] and Hajime Kimura[2]

[1]Department of Maritime Engineering, Graduate School of Engineering, Kyushu University, Fukuoka, Japan
(E-mail: k-shiraishi@system.nams.kyushu-u.ac.jp)
[2]Department of Marine Systems Engineering, Faculty of Engineering, Kyushu University, Fukuoka, Japan
(E-mail: kimura@nams.kyushu-u.ac.jp)

Abstract: This paper deals with motion planning for a multifunctional underwater robot that can perform various tasks such as swimming, walking and grasping objects. The authors have developed a unified motion planning method that can generate motion planning for a variety of movement using a single algorithm. Under this method, motion planning problems are modeled as finite-horizon Markov decision processes, and optimum motion planning is achieved by dynamic programming. However conventional dynamic programming is sometimes considered to have limited applicability because of ”the curse of dimensionality.” To avoid this issue, we propose two efficient approaches. One is an application a random network as a state transition network to suppress the explosion in the number of states. The other is a modification using a gradient method to improve the found motion in the random network. The effectiveness of the proposed method is demonstrated through numerical simulations involving two types of tasks for multifunctional robots. One is a reaching task, and the other is a thrust force generation task.

Keywords: Multifunctional underwater robots, Unified motion planning, Dynamic programming, Gradient method, Random network

1. INTRODUCTION

The field of underwater robotics is currently enjoying a period of high interest and growth. Applications for such robots include the exploration of deep sea environments, cable maintenance, monitoring of subsurface structures, and biological surveys [1]. The authors have been developing a multifunctional underwater robot that can perform various tasks such as swimming, walking and grasping objects [2]. This robot is equipped with a redundant degree of freedom manipulator, and has two outstanding features; One is its robustness in the event of troubles, and the other is its multifunctionality, which enables its use not only as an arm, but also as a finger for gripping objects, or as a fin for swimming.

This robot has a serious problem that corresponding motion planning algorithms must be developed for each individual task. Additionally, a work of generation of the motion planning algorithms requires considerable time. The authors have developed a unified motion planning method that can generate motion planning for a variety of tasks using a single algorithm [3]. Under this method, motion planning problems are modeled as finite-horizon Markov decision processes, optimum motion planning is achieved through dynamic programming. Applying dynamic programming, the state space needs to be discretized. Conventional dynamic programming is used to discretize the state space with full grid points. However, due to the exponential growth in the full grid discretization as the number of state variables increases, dynamic programming is still commonly considered to be computationally intractable. This paper presents a new approach that can avoid ”the curse of dimensionality”- the term used to describe the exponential increase in the number of states with the number of state variables. The proposed method generates a random state transition network by discretizing the state space at random to prevent an exponential increase in the number of states. The random network allows utilizing the dynamic programming in high dimensional problems, and to reduce a computation time. Optimum motion planning can be similarly achieved using dynamic programming in such a random network. This approach has a drawback that the solution using the random network is inferior the one using the lattice network. Therefore, this paper also presents the approach using a gradient method to improve the found motion in the random network.

The effectiveness of the proposed method is demonstrated through numerical simulations of two types of tasks for multifunctional robots. One is a reaching task composed of manipulator posture planning to minimize the robot’s energy consumption caused by the fluid drag force [4]. The other is a thrust force generation task in which the robot is driven forward by the fluid drag force acting on a manipulator. The experimental results show that the proposed method using random network enables application to high-dimensional problems, and reduction in the amount of computation time required. The method using the gradient method enables to obtain the solution which is near the obtained one using lattice network.

2. MOTION PLANNING PROBLEM

In this study, the authors dealt with two motion planning problems for a multifunctional underwater robot. One was a reaching task requiring a path planning task for the robotic manipulator, and the other was a thrust force generation task for the robot it self. The reaching task entailed obtaining an optimum sequence of postures in fluid, and optimum planning was used to minimize energy consumption caused by the fluid drag force. The authors considered a movable underwater robot equipped with a manipulator in the thrust force generation task. This robot generates fluid drag force to move forward by moving the manipulator. In this task, the objective function is to maximize the distance advanced.

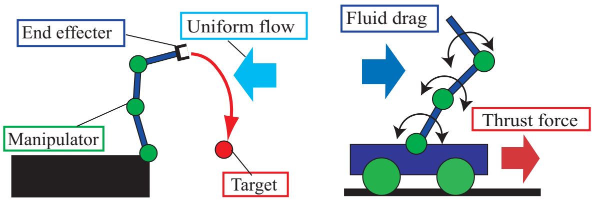

Fig. 1 Reaching task and thrust force generation task

2.1 Manipulator Coordinate System

Figure 2 shows the coordinate system and parameters of the manipulator. x and y represent the horizontal and vertical directions, respectively. l_i is the length of the ith link, and θ_i is its angle. The links are assumed to be columnar in shape. The joints are also to be small enough in relation to the links to enable omission of the fluid drag forces acting on the joint in the calculation.

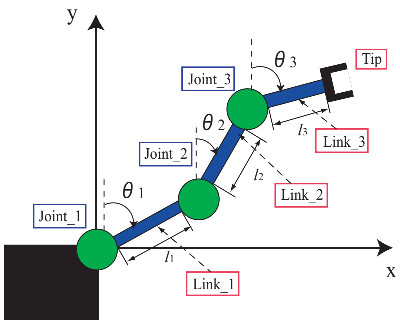

Fig. 2 Configuration of the three-link manipulator

$\boldsymbol{S}$ is defined as the manipulator posture, which is a vector composed of the angles of the links, given by

$$\boldsymbol{S} = (\theta_1, \theta_2, \cdots, \theta_i, \cdots, \theta_n). \tag{1}$$

The point of the nth link's tip becomes the position of the end effecter. The point is given by

$$x_{Tip} = \sum_{i=1}^{n} l_i \sin\theta_i, \quad y_{Tip} = \sum_{i=1}^{n} l_i \cos\theta_i. \tag{2}$$

The Coordinates of the other links can be provided by using Eq.(2). The notation $\boldsymbol{P}$ represents the sequence of manipulator postures by which the target posture is achieved from initial posture. It is given by

$$\boldsymbol{P} = (\boldsymbol{S}_{init}, \boldsymbol{S}_2, \cdots, \boldsymbol{S}_j, \cdots, \boldsymbol{S}_{end}), \tag{3}$$

where $\boldsymbol{S}_{init}$ is the initial posture when the manipulator begins to move is an initial posture, $\boldsymbol{S}_{end}$ is the target posture at which the end effecter reaches the target coordinates, j is the number of posture changes, and $\boldsymbol{S}_j$ is the jth manipulator posture.

2.2 Fluid Drag Force

The fluid drag force is given by

$$f = C_d \frac{1}{2} \rho D u|u| + C_m \rho \frac{\pi}{4} D^2 \frac{du}{dt}, \tag{4}$$

where D is the diameter of the column, C_d is the drag coefficient, C_m is the added mass coefficient, u is the velocity of a link moving through the fluid, and ρ is the density of the fluid. Since the velocity of the link is quite small, the second term on the right of Eq.(4) can be omitted. The fluid drag force acting on each link is shown as follows using Eq.(4)

$$F_{x,i} = F_{x,i+1} + f_i \cos\theta_i, \tag{5}$$

$$F_{y,i} = F_{y,i+1} - f_i \sin\theta_i, \tag{6}$$

$$M_i = M_{i+1} + T_i + F_{x,i+1} l_i \cos\theta_i - F_{y,i+1} l_i \sin\theta_i, \tag{7}$$

where f_i and T_i are given by

$$f_i = \int_0^{l_i} C_d \frac{1}{2} \rho D_i u_i |u_i| dr_i,$$

$$T_i = \int_0^{l_i} C_d \frac{1}{2} \rho D_i u_i |u_i| r_i dr_i.$$

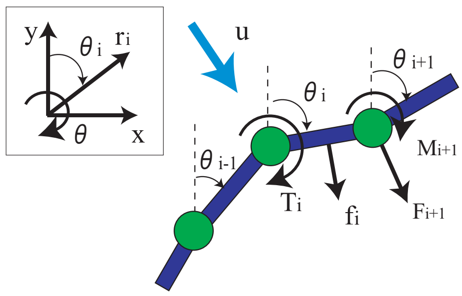

Fig. 3 Forces acting on a link

2.3 Reaching Task

As multifunctional underwater robots are operated in submarine environments, it is necessary to consider the influence of fluid drag forces caused by sea currents. Accordingly, the authors considered a motion planning problem to minimize the amount of energy consumed by such forces. This energy is determined by the torque acting on the joints and the angles of the links that move over a certain period. In this motion planning problem, the cost function is formulated using the amount of energy consumed. When the manipulator posture changes from $\boldsymbol{S}_A$ to $\boldsymbol{S}_B$, the cost function is shown approximately as follows:

$$Cost(\boldsymbol{S}_a, \boldsymbol{S}_b) = \sum_{i=1}^{n} \int_{\theta_{S_a,i}}^{\theta_{S_b,i}} M_i d\theta_i + \Delta E \tag{8}$$

where n is the number of the links, $\theta_{S,i}$ is the angle of the ith link in the manipulator posture S, and ΔE denotes the energy consumption caused by a mechanical friction. The total cost is provided using Eq.(8) as

$$Total\ Cost(\boldsymbol{P}) = \sum_{i=1}^{|\boldsymbol{P}|-1} Cost(\boldsymbol{S}_i, \boldsymbol{S}_{i+1}). \tag{9}$$

In the reaching task, the motion planning problem is formulated as an optimization problem involving the minimization of the total cost in Eq.(9).

$$Optimal\ Planning = \min_{\boldsymbol{P} \in \boldsymbol{P}_{All}} Total\ Cost(\boldsymbol{P}), \quad (10)$$

where $\boldsymbol{P}_{All}$ denotes the set of all executable motion planning in the motion planning problem. The motion planning $\boldsymbol{P}$ to satisfy Eq.(10) is the best operation plan. In this paper, the positive energy obtained from the current is not considered.

2.4 Thrust Force Generation Task

In this section, the authors describe a motion planning problem involving the generation of thrust force for a movable underwater robot. The robot is able to move forward using the fluid drag force that acts on the manipulator. In this paper, it is assumed that the body of the robot is constrained by a rail. As a preliminary experiment, the one-dimensional motion is considered. For simplicity, the effect of the added mass related to the fluid is assumed to be negligible, and it is considered that fluid drag forces acts on the main body and on each links. The motion equation for the body is given by

$$M\ddot{x} = F_{body} + \sum_{i=1}^{n} F_{x,i}, \quad (11)$$

where M is the mass of the movable underwater robot. F_{body} denotes the fluid drag force acting on the body of the robot except for the force on the links. $F_{x,i}$ denotes the fluid drag force of axial component x that acts on the ith link. n is the number of links of the manipulator. $L(\boldsymbol{S}_i, \boldsymbol{S}_{i+1})$ denotes the distance which the robot advances when the posture changes from $\boldsymbol{S}_A$ to $\boldsymbol{S}_B$ In the thrust force generation task, the motion planning problem is formulated as an optimization problem involving the maximization of the distancethat the robot advances.

$$Optimal\ Planning = \max_{\boldsymbol{P} \in \boldsymbol{P}_{All}} L(\boldsymbol{S}_i, \boldsymbol{S}_{i+1}), \quad (12)$$

The motion planning $\boldsymbol{P}$ to satisfy Eq.(12) is the best operation plan.

3. UNIFIED MOTION PLANNING METHOD

In this section, we describe the motion planning for the multi-DOF underwater manipulator using dynamic programming with the random network and the gradient method. First, motion planning problems are modeled as finite-horizon Markov decision processes under this method, and optimum motion planning for various tasks can be obtained using dynamic programming. Next, the random networks are used as the state transition network to avoid the curse of dimensionality. Finally, the author modifies the obtained motion using the gradient method.

3.1 Modeling Motion Planning Problem

The state transitions of the manipulator are represented as a network, as shown in Fig.4. In this network, each node denotes a manipulator posture, and the directions

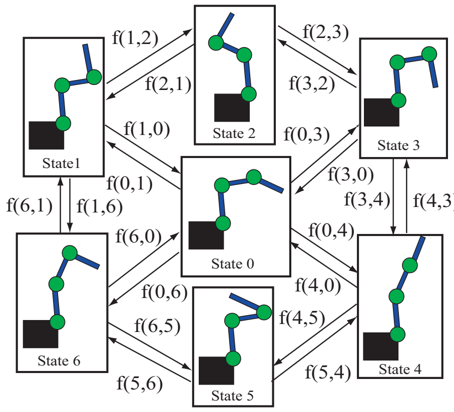

Fig. 4 State transition network

of the arrows denote the direction of the posture change. $f(s_i, s_{i+1})$ is the cost function, when the robot posture changes from s_i to s_{i+1}. This representation allows the motion planning problem to be considered as a graph search problem. Therefore, a variety of solution methods for the graph theory have applicability to these problems.

Assuming that the Markov property is confirmed in the state transition network, the motion planning problem is considered as a Markov decision process, which forms a framework for planning that effectively captures the essence of purposeful activity in various situations. A Markov decision process is defined by its state and its action sets, and by the one-step dynamics of the environment [5], [6]. Markov decision process models an agent that interacts with a stochastic environment. State s is defined as the manipulator posture, and action a represents the motion of each link in a posture change. Reward $R(s'|s,a)$ is the value of the reward function or the cost function in the state transition. Value function $V(s)$ denotes the expected total reward starting at state s according to the strategy. In the Markov decision process, the Bellman equation is shown by

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} p(s'|s,a) \left(R(s'|s,a) + \gamma V^*(s') \right), \quad (13)$$

where $\boldsymbol{S}$ is the state set, $\boldsymbol{A}$ is the action set, and γ is the discount factor. The optimum policy is obtained by solving Eq.(13) by using standard dynamic programming, and represents optimum motion planning for various tasks. In other words, the optimum motion planning is obtained by the reward is set according to tasks and the Bellman equation is solved In this paper, the standard value iteration method is used to solve the Bellman equation.

3.2 Discretization of State Space

State space should be discretized in standard dynamic programming. In the conventional technique, state space is discretized in the form of a lattice, but this technique suffers from the "curse of dimensionality." - a phenomenon by which the number of states increases in exponentially. A random network that discretizes the state space at random has been developed as a method to avoid this issue [7], [8]. The random network is easily applicable in the proposed method, and various successful researches using the network are reported. In this study, a random network was used as the state transition network, which enables to obtain optimum motion planning using dynamic programming. The generation process of the random network is as follows. First, coordinate points in the n dimensional space are allocated at random using uniform random numbers. The coordinate points are defined as the angles of the n link manipulator, and means the nodes in the networks. The specified numbers of coordinate points are produced in this way. Next, connecting each coordinate point, the random network is produced. This approach enables to discretize the state space at the arbitrary number of states, and to avoid the explosion of memory storage requirements. Therefore, the proposed method using the random network may have applicability to high dimensional problems.

3.3 Modification Method

The proposed method with the random network has two advantages including applicability to the high-dimensional problems and reducing the computation time. However, the method has a serious shortcoming. The shortcoming is that the quality of the solution using the random network is inferior to the solution using the lattice network. That is, the quolity of the solution depends on the initial design of the random network. In addition, the random network tends to be a sparse network, which becomes a factor for the deterioration of the ability to express the optimum motion planning. Therefore, it is necessary to develop a modification method to improve the quality of the obtained motion planning in the random network approach. This paper proposes the modification method using a gradient method. That is, the locations of the nodes in the solution are modified using the gradient method to improve the quality of the solution. Accordingly, this approach enables to obtain the equibalent solution in the lattice network approach. In this paper, the conjugate gradient method which converges comparatively fast, was used [9].

4. EXPERIMENTS AND RESULTS

To confirm the validity of the proposed method, several experiments involving the reaching task for the multi-DOF underwater manipulator were conducted.

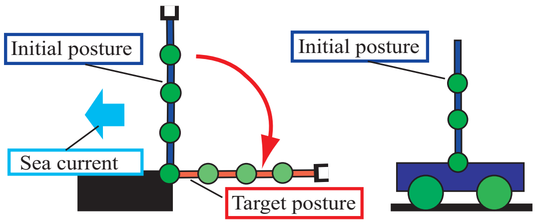

Fig. 5 Settings of the experiment for the reaching task

4.1 Experiment 1: Reaching Task

The aim of Experiment 1 was to verify whether the proposed method could achieve motion planning to minimize the energy consumed by fluid drag force. In the experiment, a 4-DOF manipulator was used. The parameters of the experiment were set as follows: $l_i = 0.8[m], D_i = 0.2[m], C_d = 1.17, \rho = 1.023, |u| = 2.0[m/s]$, and $\theta_{limit} = \pm 1.571[rad]$, where θ_{limit} is the angle within which the links of the manipulator can rotate. To enable comparison of the results, three motion planning sets were used: Planning 1 was generated through the proposed method using the lattice network, and Planning 2 was generated through the proposed method using the random network. The lattice network was provided by discretizing the state space in the form of a lattice. Each dimension of the state space was divided evenly into 13 parts. The number of nodes was $13^4 = 28,561$, and the number of links per a node was 80 in this network. On the other hand, the random network was provided by discretizing the state space at random. The number of nodes was $10,000$, and the number of links per a node was 40 in this network. Simulation was executed 10 times, and the performance was evaluated from an average of the results.

Figures 6 and 7 show the motion obtained in Experiment 1. Figures 10 and 12 show time series graphs for the cost. All planning sets were able to achieve the motion required to reach the target posture from the initial posture. Table 1 shows the cost and the computation time for Experiment 1 for the achieved motion and Table 2 shows the cost and the computation time for Experiment 1 using the modification method. The cost of Planning 1 was smaller than that of Planning 2, although its computation time is the longest. The computation time of Planning 2 is $1/5$ that of Planning 1. In Planning 2, the cost waw improved using the modification method. The cost of Planning 1 was not improved since the solution might be the local optimum solution. In addition, the difference of the improved cost of Planning 1 and that of Planning 2 was very small. The authors also noted that the motion of the best solution shown in Figures 6 and 7 takes advantage of the uniform flow.

Table 1 Simulation results of Experiment 1

	Cost $[N \cdot m]$	Computation Time $[s]$
Planning 1	0.311 ± 0.0	774.6 ± 0.0
Planning 2	0.536 ± 0.031	150.7 ± 14.1

Table 2 Simulation results of Experiment 1 applied the modification method

	Cost $[N \cdot m]$	Computation Time $[s]$
Planning 1	0.291 ± 0.0	1532.5 ± 0.0
Planning 2	0.338 ± 0.018	274.2 ± 76.4

4.2 Experiment 2: Thrust Force Generation Task

The aim of the Experiment 2 was to verify whether the proposed method could achieve the generation of thrust force motion. In the experiment, a movable underwater robot equipped with a 3-DOF manipulator was used. The parameters of the experiment were set as follows: $l_i = 0.5[m], D_i = 0.2[m], C_d = 1.17, M = 5.0[kg], \rho = 1.023, \theta_{limit} = \pm 1.047[rad]$, and $\dot{\theta} = 0.034$ $[rad/s]$, where $\dot{\theta}$ is the velocity of the links of the manipulator. The height and the width of the body were $0.2[m]$ and $0.3[m]$, respectively. In the lattice network, the number of nodes was $11^3 = 1,331$, and the number of links per a node is 26. In the random network, the number of nodes is 600, and the number of links per a node was 50. To enable comparison of the results, two sets of motion planning were used: Planning 3 was generated through the unified motion planning using the lattice network, and Planning 4 was generated through the unified motion planning using the random network.

Figures 8 and 9 show the motion planning obtained by simulation of the thrust force generation task. In the both cases, the motion of the manipulator consists of a flutter kick for swimming. Table 3 shows the average velocity and the computation time for Experiment 2 without the modification method, Table 4 shows the average velocity and the computation time for Experiment 2 applied the modification method. Planning 3 is superior to Planning 4 in terms of average velocity, however the computation time of Planning 4 is half that of Planning 3. In Planning 3 and Planning 4, the costs were improved using the modification method.

Table 3 Simulation results of Experiment 2

	Velocity $[m/s]$	Computation Time $[s]$
Planning 3	0.0465 ± 0.0	509.7 ± 0.0
Planning 4	0.0417 ± 0.0065	247.8 ± 3.45

Table 4 Simulation results of Experiment 1 applied the modification method

	Velocity $[m/s]$	Computation Time $[s]$
Planning 3	0.0724 ± 0.0	5.74 ± 0.0
Planning 4	0.0571 ± 0.0153	4.16 ± 2.291

5. CONCLUSIONS AND FUTURE WORKS

5.1 Conclusions

In this study, we developed the unified motion planning method for the multifunctional underwater robots. Under the proposed method, the motion planning problems were modeled as Markov decision processes, and optimum motion planning was obtained using the dynamic programming. The proposed method can generate motion planning for a variety of tasks using a single algorithm. In addition, we used the random network as the state transition network to enable reduction of the computation time involved. The method avoids the "curse of dimensionality," since the number of nodes is independent of the number of state variables. This paper also proposed the modification method to utilize the conjugate gradient method. The proposed method can solve the problem that the quality of the solution using the random network is inferior to the solution using the lattice network. The effectiveness of the proposed method was demonstrated through the numerical experiments involving two types of tasks for multifunctional robots.

5.2 Future Works

The random networks tend to be sparse networks in the high dimensional state space. This phenomenon deteriorate the performance of the obtained motion planning, since the sparse networks have inferior ability to express the motion planning. This also introduce a few improvement using the proposed modification method. In further works, random networks will be generated using a low-discrepancy sequece, which will make possible well-ballanced discretization. In the modification method, we will add the nodes to the trajectory of the motion planning and applying the gradient method, so that the perfomance of the proposed method may not deteriorate.

REFERENCES

[1] G. Antonelli, *Underwater Robots: Motion and Force Control of Vehicle-Manipulator Systems*, Springer, 2003.

[2] K. Shiraishi and H. Kimura, "On a Unified Motion Planning Method for a Multifunctional Underwater Robot," *Proceedings of the 26th Annual Conference of the Robotics Society of Japan*, (in Japanese) 2008.

[3] K. Shiraishi and H. Kimura, "A Unified Motion Planning Method for a Multifunctional Underwater Robot," *Proceedings of the 14th International Symposium on Artificial Life and Robotics 2009*, 2009.

[4] S. Ishibashi, M. Ito, and E. Shimizu, "Autonomous Motion Planning for a Manipulator Equipped on AUV in a Workspace Divided into Cubes," *Proceedings of the 13th International Offshore and Polar Engineering Conference*, pp. 231-238, 2003.

[5] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, 1996.

[6] Abhijit Gosavi, *Simulation-Based Optimization*, Kluwer Academic Publishers, 2003.

[7] C. G. Atkeson and B. J. Stephens, "Random Sampling of States in Dynamic Programming," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 38, pp. 924-929, 2008.

[8] S. M. LaValle, M. S. Branicky, and S. R. Lindemann, "On the Relationship Between Classical Grid Search and Probabilistic Roadmaps," *International Journal of Robotics Research*, Vol. 23, pp. 673-692, 2004.

[9] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery, *Numerical Recipes in C, Second Edition*, Cambridge University Press, 1992.

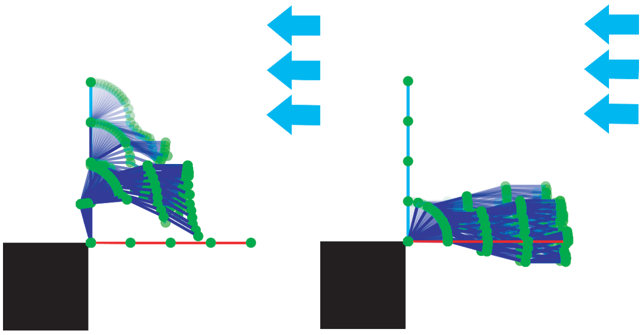

Fig. 6 Motion obtained by the proposed method with the lattice network (Planning 1)

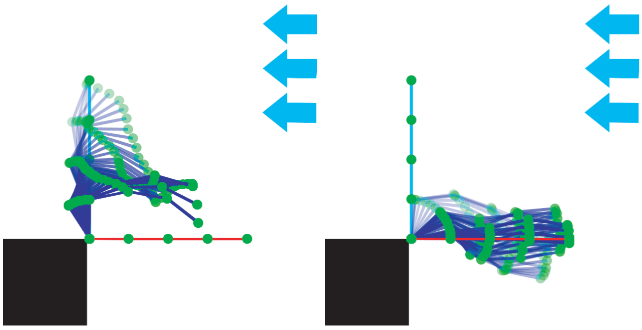

Fig. 7 Motion obtained by the proposed method with the random network (Planning 2)

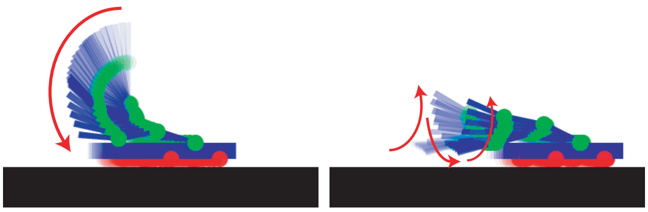

Fig. 8 Motion obtained by the proposed method with the lattice network (Planning 3)

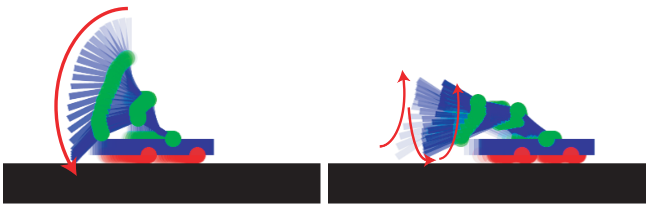

Fig. 9 Motion obtained by the proposed method with the random network (Planning 4)

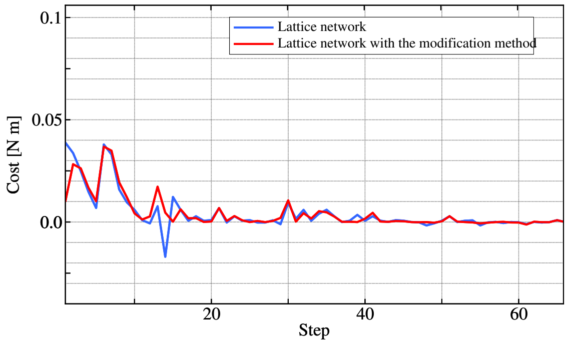

Fig. 10 Time series graph for the cost of Planning 1

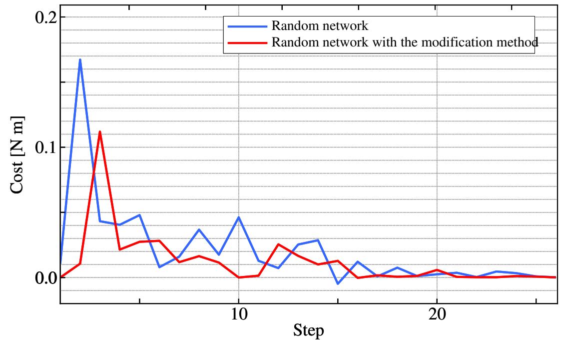

Fig. 11 Time series graph for the cost of Planning 2

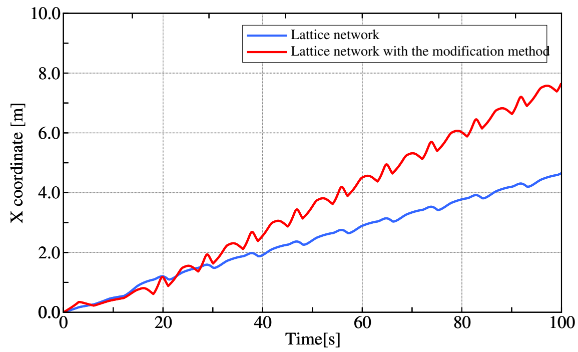

Fig. 12 Time series graph for the x-coordinate values of Planning 3

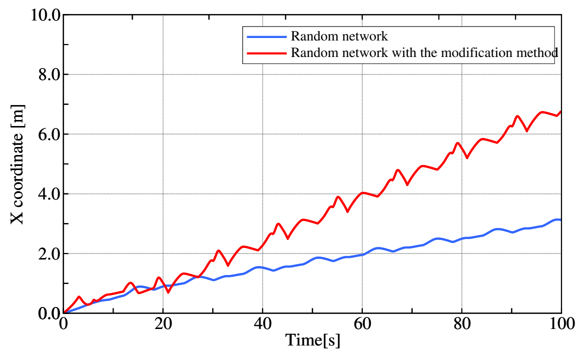

Fig. 13 Time series graph for the x-coordinate values of Planning 4