Deep Neural Network-based Fast Motion Planning Framework for Quadrupedal Robot*

Jinhyeok Jang[1], Heechan Shin[2], Minsung Yoon[2], Seungwoo Hong[3], Hae-Won Park[3] and Sung-Eui Yoon[2]

***Abstract*— We present a motion planning framework that generates the motion of a quadrupedal robot in a short time using a deep neural network. Our planner gets the initial robot state, target goal pose, and terrain heightmap as input and generates a trajectory of a quadrupedal robot. The planner contains deep neural networks that extract features from input. These features guide the planner to generate a precise trajectory. We achieved the planning time within 230ms for 2 seconds long trajectory over various terrain types.**

I. INTRODUCTION

The quadrupedal robot's motion planning is a challenging problem due to the indirect control of the base body trajectory, which is determined by the contact of the feet. There are many constraints for foot contacts, so that traditional methods decouple the planning of the base body trajectory and footstep. However, these approaches loosen the relationship between base body trajectory and footstep. Since the base body trajectory and footstep should be optimized simultaneously, some of the prior studies try to use the trajectory optimization (TO) method to this problem (e.g. [1], [2]). These prior methods find a general solution for various terrain, but they have a high computational complexity that makes the planning time long. Recent researchers apply deep neural networks to generate quadrupedal robot's motions faster [3]–[7]. Still, they suffered from the input ambiguity problem that the network generates average motions such as skating.

We propose a motion planning framework that generates a motion of a quadrupedal robot using a deep neural network. Our model generates the base body trajectory and footstep together within 230ms. Specifically, our approach handles the input ambiguity problem by the novel network structure, which generates a distinguishable feature from input data.

II. METHODOLOGY

A. Problem Definition

Let's consider a quadrupedal robot on a terrain. Our target is finding a trajectory of four-legged locomotion to the goal given terrain and initial robot state. The target terrain is acquired as a heightmap and the goal indicates the target pose of the base body. The overall robot state is represented by base body position, rotation, linear velocity, angular velocity, and foot position, foot reaction force, and contact information per foot. The initial robot state represents the robot at the start point, $t = 0$.

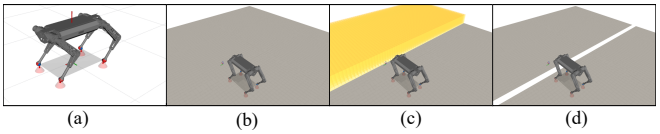

Fig. 1. Target robot (a) and trained terrain data types: flat terrain (b), step terrain (c), and gap terrain (d).

B. Network Structure

Our model is trained as a supervised manner using demonstration trajectories, generated by optimization-based planner, *towr* [2]. Every trajectory is 2.0 seconds long and sampled on three types of terrains: flat, step, and gap (Fig.1). Each terrain type has 6,000 pairs of motion and terrain; therefore there are 18,000 dataset.

Our motion planning framework consists of three networks: terrain encoder, state feature network, and trajectory generation network. The *terrain encoder* is pre-trained as an autoencoder in the end-to-end manner. The encoder learns the extraction of the terrain information from heightmap. The extracted terrain information, initial robot state and goal pose are fed to the *state feature network*. This network is designed to generate distinguishable feature from input data to solve the ambiguity. Also, input data are rearranged based on the data type to get a highly related feature between them. The CNN layers then extract the state feature $\mathfrak{R}_{state}$ that implies the relationship among input data. The *trajectory generation network*, composed of MLP layers, finds a proper trajectory to the goal pose using the state feature.

III. EXPERIMENTS

In this section, we conduct experiments that show the performance of the network, effects of the state feature, and the generality of our planner. Our target robot has a single underactuated base body with four legs, where each leg has 3 joints (Fig. 1).

A. Network Performance

We show various error values and planning time. The error values re calculated against the reference trajectory, generated by *towr* (Table I). We measure the RMSE for all variables and MSE for rotations, and distance error of the base body and feet position for 1,000 samples that have never been used for training. The base body and foot

* The extended version of this research has been submitted to IROS 2021.

[1]Jinhyeok Jang (mrjohd@kaist.ac.kr) is at Robotics Program, [2]Heechan Shin (shin_heechan@kaist.ac.kr), Minsung Yoon (minsung.yoon@kaist.ac.kr), and Sung-Eui Yoon (corresponding author, sungeui@gmail.com) are at School of Computing, and [3]Seungwoo Hong (seungwoohong@kaist.ac.kr), Hae-Won Park (haewonpark@kaist.ac.kr) are at Department of Mechanical Engineering, Korea Advanced Institute of Science and Technology (KAIST)

TABLE I

PERFORMANCE METRICS FOR OUR COMPUTED TRAJECTORY VERSUS REFERENCE.

RMSE	MSE_f	MSE_q	MSE_y	$dist_{base}$	$dist_{foot}$
0.247	0.00154	0.00114	0.0254	0.0229	0.0308

TABLE II

PLANNING TIME OF REFERENCE PLANNER AND OURS

	Flat terrain	Step terrain	Gap terrain
towr	859	2479	8029
Our model (improvement)	220 (3.90)	222 (11.2)	213 (37.7)

[ms]

TABLE III

ABLATION STUDY ON STATE FEATURE ($\mathfrak{R}_{state}$).

	RMSE	MSE_f	MSE_q	MSE_y	$dist_{base}$	$dist_{foot}$
w/ $\mathfrak{R}_{state}$	0.247	0.00154	0.00114	0.0254	0.0229	0.0308
w/o $\mathfrak{R}_{state}$	0.389	0.0189	0.0171	0.131	0.0746	0.0746

TABLE IV

SKATING METRIC.

towr	w/o $\mathfrak{R}_{state}$	w/ $\mathfrak{R}_{state}$
0.040	0.074	0.050

position distance errors are about 0.916% and 1.23% error about the terrain size, which indicates that our prediction is close to the reference trajectory. Table II shows the difference of the planning time between *towr* and our model. *towr* shows varying planning time for different terrain types while our model generates a trajectory in the almost same time (<230ms) regardless of the terrain type. Therefore our motion planning framework can generate a similar trajectory to that generated by an optimization-based method with a much faster time, e.g. 3.90 times for flat, 11.2 times for step, 37.7 times faster for gap terrain type, respectively.

B. Ablation Study

In this section, we show the effect of the state feature on solving the input ambiguity problem.

Table III shows the error w/ and w/o using state feature network. These results show that our model can generate 1.57 15 times precise trajectory by using the state feature network. Especially, the rotation error which changes in a small range so that sensitive than other shows the highest improvement. Table IV shows the skating metric of the trajectory of the reference (*towr*), a model with state feature, and a model without state feature. The used skating metric is introduced from [3] that clamped at 1 if the feet are always on the ground. Based on the results, the state feature can reduce the skating motion which shows a more similar skating metric to the reference trajectory. Overall, these results show that the state feature $\mathfrak{R}_{state}$ ameliorates the input data ambiguity problem and helps to generate a precise trajectory.

C. Test on Unseen Terrain

We trained our model only with flat, step, and gap terrain. To show the generality of our planner, we test it to an unseen terrain type, which has never been used for training.

Fig. 2 shows a quadrupedal robot walking on the unseen terrain. The most significant difference of unseen terrain over the trained ones is continuity. Trained terrains contain discontinuous parts, e.g. step terrain with step block, gap terrain with an empty gap. On the other hand, unseen terrain is generated by using multiple trigonometric functions. Fig. 2 shows that our model can handle terrains with different characteristics to the trained ones.

(a) (b) (c)

Fig. 2. The sequence of the robot walking on the unseen terrain. The robot moves in order of (a), (b), (c).

IV. CONCLUSION

We have proposed a motion planning framework for a quadrupedal robot using a deep neural network. It finds a trajectory of the base body and footstep simultaneously and fast. Our model can generate a trajectory for flat, step, gap, and even unseen terrain while avoiding the ambiguity problem via our state feature network.

In the future, physical constraints can be modeled as a soft constraint and modeled as a loss function, which helps our model better respects physical characteristics.

ACKNOWLEDGMENT

This research was supported by the Defense Challengeable Future Technology Program of Agency for Defense Development, Republic of Korea.

REFERENCES

[1] J. T. Betts, "Survey of numerical methods for trajectory optimization".

[2] A. W. Winkler, C. D. Bellicoso, M. Hutter, , and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based endeffector parameterization", in *IEEE Robotics and Automation Letters*. 2018, p. 1560–1567, IEEE.

[3] H. Zhang, S. Starke, T.Komura, and J. Saito, "Mode-adaptive neural networks for quadruped motion control", *ACM Transactions on graphics*, vol. 37, no. 3, pp. 1–11, 2018.

[4] V. Tsounis, M. Alge, J. Lee, F. Farshidian, and M. Hutter, "Deepgait: Planning and control of quadrupedal gaits using deep reinforcement learning", *IEEE Robotics and Automation Letters*, 2020.

[5] D. Holden, J. Saito, T. Komura, and T. Joyce, "Learning motion manifolds with convolutional autoencoders", *SIGGRAPH Asia 2015 Technical Briefs*, vol. 18, pp. 1–4, 2015.

[6] D. Holden, J. Saito, and T. Komura, "A deep learning framework for character motion synthesis and editing", *ACM Transactions on graphics*, vol. 138, 2016.

[7] T. Komura D. Holden and J. Saito, "Phase-functioned neural networks for character control", *ACM Transactions on Graphics*, , no. 42, 2017.