Learning Footstep Planning using Deep Reinforcement Learning for Dynamic Walking

1st Jasper Güldenstein
Department of Informatics
University of Hamburg
Hamburg, Germany
jasper.gueldenstein@uni-hamburg.de

2nd Jianwei Zhang
Department of Informatics
University of Hamburg
Hamburg, Germany

***Abstract*—Selecting and executing dynamically feasible footsteps on a bipedal robot while efficiently navigating toward a goal pose is a complex optimization problem. Many approaches have been proposed, such as end-to-end planning, velocity tracking using machine learning, and discretization methods. This paper proposes a policy trained using deep reinforcement learning that selects footsteps to be executed by a walking engine. It outperforms a previously successfully used baseline in a simulated environment.**

***Index Terms*—Footstep planning, Humanoid robots, Bipedal robots, Legged robots, Robot learning, Deep reinforcement learning**

I. Introduction

Selecting and placing dynamically stable footsteps is vital for the robust navigation of legged robots. Furthermore, the footstep placement must be selected to navigate towards a defined goal pose efficiently. Many approaches have been proposed to solve this problem. Haarnoja et al. [1] proposed an end-to-end learned approach to soccer-playing robots. However, this approach relies on external motion tracking and, due to its end-to-end nature, does not allow the use of navigation capabilities only. Lee et al. present an approach for a quadruped robot only relying on proprioceptive sensor information which generates foot trajectories in Cartesian space [2] in unknown environments. The controller is trained in simulation and deployed on multiple similar real robots. While this approach shows impressive performance, the policy is controlled with commanded velocities, requiring additional planning for efficient navigation toward a goal pose. Footstep planning is calculating a set of steps to allow a legged robot to walk toward a goal pose while avoiding obstacles. Finite transition sets have been proposed to solve this problem [3], [4]. Often, these approaches do not consider the robot's dynamic state.

The approach presented in this paper learns which footsteps are dynamically stable for a given walking engine [5] on a humanoid robot used in the RoboCup KidSize league [6]. We use deep reinforcement learning in simulation to generate footsteps, allowing the robot to navigate toward a goal pose while remaining stable.

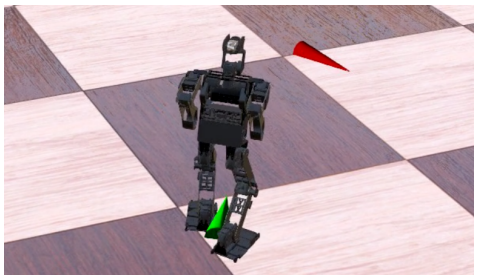

Fig. 1: Visualization of the simulated robot executing navigation using the policies actions. The green arrow shows the robot's current pose. The goal pose is marked as a red arrow.

II. Approach

Our approach uses a reinforcement learning setup training a policy using PPO [7] implemented in stable baselines 3 [8]. The following sections describe the setup briefly.

a) Environment: The environment uses Webots [9] as a rigid body simulator. A visualization of the environment is presented in Figure 1. At the beginning of an episode, the goal pose is sampled from a normal distribution around the robot's starting pose. In each step of the episode, the robot walks a double step as the policy selects. The joint positions commanded to PID controllers in the simulated actuators are generated by the walking engine [5] throughout the double step. The episode ends when the robot reaches the goal pose or falls.

b) Observation: The agents's observation consists of the goal pose relative to the robot and the robot's alignment towards the goal. The exact encoding of this information is presented in Figure 2. Furthermore, two previously taken actions by the policy are provided. Gaussian noise modeling the localization pipeline's performance is applied to the measurement for some evaluations.

c) Policy: The policy presented in Figure 2 consists of an MLP with two hidden layers of 64 neurons each. Its outputs are the parameters of a beta distribution for each variable and, therefore, normalized. A distribution is required for the stochastic on-policy training of PPO [7]. The inference is deterministic as the maximum of the distribution is selected.

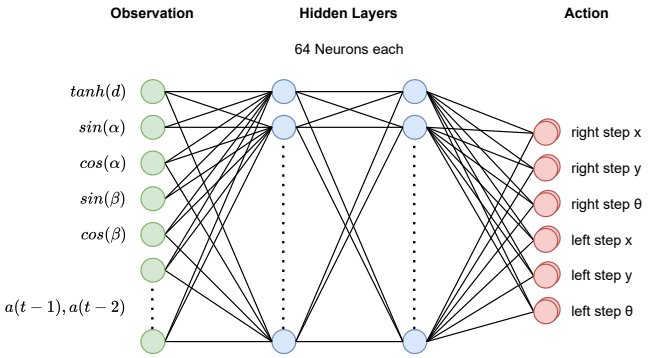

Fig. 2: Structure of the policy. d is the Euclidean distance between the robot and the goal pose. α is the angular distance between the robot and goal orientations. β is the angle distance between the robot orientation and a vector from the robot position to the goal position. The outputs are the two parameters of a beta distribution for each variable.

d) Action: The policy's output (i.e., the action) are the poses of the next two footsteps. It is scaled to the experimentally verified kinematic capabilities of the step length of the used robot.

e) Reward Function: Shaping rewards are used to guide the agent towards reaching the goal pose. The overall reward function is as follows:

$$r(t) = s_d \cdot r_d(t) + s_a \cdot r_a(t) + s_o \cdot r_o(t) + s_c \cdot r_c(t) \quad (1)$$

Table I shows the scalars and formulas for the individual reward terms.

Term	Scalar
$r_d(t) = exp(-0.5 \cdot d)$	$s_d = 0.3$
$r_a(t) = exp(-5 \cdot d) \cdot exp(-1 \cdot \lvert\alpha\rvert)$	$s_a = 0.3$
$r_o(t) = \begin{cases} exp\left(-1 \cdot \lvert\beta\rvert\right) & d \geq 0.1\,\mathrm{m} \\ 1 & d < 0.1\,\mathrm{m} \end{cases}$	$s_o = 0.2$
$r_c(t) = mean\left(exp\left(-3 \cdot \lVert a(t) - a(t-1)\rVert\right)\right)$	$s_c = 0.2$

TABLE I: Reward function terms. The variables d, α, and β are explained in Figure 2. The *distance reward* r_d encourages the agent to move close to the goal. The *angle reward* r_a incentivizes the agent to align with the goal's orientation when close in Euclidean distance. The *orient to goal reward* r_o encourages the agent to face the goal pose while navigating towards it. The *continuity reward* r_c penalizes large changes in footstep length from one action to another to reduce accelerations.

III. Evaluation

The policy was trained for 1,000,000 steps. It is compared to a baseline developed by the RoboCup Team Hamburg Bit-Bots [10]. We generated a set of 500 goal poses for evaluation. Navigation was performed by the policy and the baseline with and without artificial measurement noise. A trial is considered successful if the goal pose is approached to a distance of $0.05\,\mathrm{m}$ in the Euclidean distance and $10°$ angular distance. Furthermore, after $100\,\mathrm{s}$ of simulated time the attempt is ended. Table II presents the results.

	Approach	Success	Timeout	Fall	Success Time [s]
without noise	ours	**0.98**	**0.0**	0.02	**6.07 ±2.13**
	baseline	0.96	0.032	**0.008**	11.13 ±4.95
with noise	ours	**0.978**	**0.0**	0.022	**6.35 ±2.23**
	baseline	0.91	0.078	**0.012**	11.43 ±5.11

TABLE II: Results of our policy and baseline in the experiments with and without artificial noise added to the measurements. The standard deviation is given for the success time.

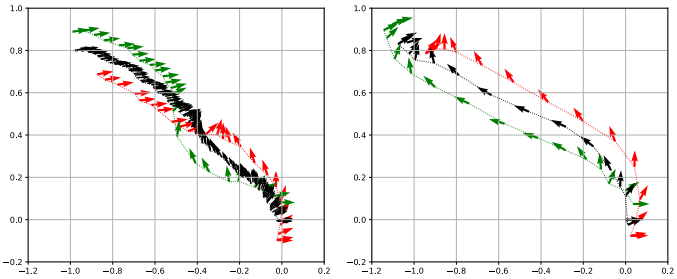

Fig. 3: Footstep positions produced by the baseline (left) and our approach (right). The left and right footsteps are marked as green and red arrows, respectively. The pose of the robot's torso projected on the ground is marked in black. Our policy plans much larger steps and approaches the goal pose more effectively than the baseline.

The policy outperformed the baseline and showed higher robustness to measurement noise. Time for successful navigation was $54.5\,\%$ and $55.6\,\%$ of the baseline approach. However, the robot fell 1.83 to 2.5 times more often when controlled by the policy. The cases in which the policy-controlled robot fell were still few with $2\,\%$ and $2.2\,\%$, respectively. The robot platform is capable of standing up, which was not considered in these experiments. Figure 3 shows an example of the produced steps.

IV. Conclusion and Future Work

This work outlines foundational work for developing a deep reinforcement learned policy that exploits a walking engine's capabilities while remaining stable. Significantly better results compared to a baseline were achieved in a simulated environment. Further work is required to transfer the results to the real world. We plan to incorporate domain randomization, such as varying the robot model's parameters and artificial disturbances. More accurate modeling of the robot's actuators could also ease transfer. Furthermore, including proprioceptive sensor information in the policy input (e.g., IMU or derived orientation measurement) may improve performance. Obstacle avoidance could be approached by introducing intermediate goals on a conventionally computed path. Furthermore, stabilization techniques applied by the walking engine, such as elongating or ending a step prematurely based on IMU information, have proven effective in real-world applications. Integrating this into the policy is nontrivial but planned.

REFERENCES

[1] T. Haarnoja, B. Moran, G. Lever, S. H. Huang, D. Tirumala, J. Humplik, M. Wulfmeier, S. Tunyasuvunakool, N. Y. Siegel, R. Hafner, M. Bloesch, K. Hartikainen, A. Byravan, L. Hasenclever, Y. Tassa, F. Sadeghi, N. Batchelor, F. Casarini, S. Saliceti, C. Game, N. Sreendra, K. Patel, M. Gwira, A. Huber, N. Hurley, F. Nori, R. Hadsell, and N. Heess, "Learning agile soccer skills for a bipedal robot with deep reinforcement learning," *Science Robotics*, vol. 9, no. 89, Apr. 2024.

[2] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, Oct. 2020.

[3] J. Garimort, A. Hornung, and M. Bennewitz, "Humanoid navigation with dynamic footstep plans," in *2011 IEEE International Conference on Robotics and Automation*. Shanghai, China: IEEE, May 2011, pp. 3982–3987.

[4] A. Hornung, A. Dornbush, M. Likhachev, and M. Bennewitz, "Anytime search-based footstep planning with suboptimality bounds," in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. Osaka, Japan: IEEE, Nov. 2012, pp. 674–679.

[5] M. Bestmann and J. Zhang, "Bipedal walking on humanoid robots through parameter optimization," in *RoboCup 2022: Robot World Cup XXV*, Jul. 2022.

[6] M. Bestmann, J. Güldenstein, F. Vahl, and J. Zhang, "Wolfgang-op: A robust humanoid robot platform for research and competitions," in *IEEE Humanoids 2021*, Jul. 2021.

[7] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[8] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021.

[9] Cyberbotics Ltd., "Webots," http://www.cyberbotics.com, Open-source Mobile Robot Simulation Software, Accessed on 04.10.2024.

[10] Hamburg Bit-Bots, "bitbots_main," https://github.com/bit-bots/bitbots_main, Accessed on 02.10.2024.