Adaptive Quadruped Locomotion of a Rat Robot Based on a Hierarchical Reinforcement Learning Framework

Zitao Zhang[1], Yuhong Huang[2], Zijian Zhao[1], Zhenshan Bing[2] and Kai Huang[1,*]
[1]Sun Yat-sen University;[2]Technical University of Munich

***Abstract*—Small robots encounter considerable difficulties in learning effective motions on complex terrains owing to their underactuated nature and limited perception. It is challenging to understand the nonlinear dynamics of such robots walking on complex terrain. In this paper, we present a novel framework for robot motion generation that implements reinforcement learning, based on simplified exploration of the robot's action and observation space. Our framework controls the robot's actions using normalized signals and hierarchical mappings on mathematical space, which facilitates the learning process. We gather and analyze perception information based on changes in time slices to monitor environmental changes during robot walking. Our proposed framework demonstrates a significant reduction in training convergence time, from millions to hundreds of thousands, compared to commonly used reinforcement learning frameworks. We evaluate the efficacy of our approach on a varied set of simulated terrain scenarios, which include various obstacles and terrain undulations. Our results show that our approach effectively achieves efficient motions on complex terrains designed for small-sized robots.**

***Index Terms*—motion generation, reinforcement learning, hierarchical mappings, time slices**

I. INTRODUCTION

The rat robot is a compact quadrupedal robot designed to closely resemble an actual mouse [1], which is promising in bionomics research and disaster response. For one thing, the drive structure of quadrupedal robots make them more competitive than wheeled robots in rugged terrains and complex environments. For another thing, in contrast to larger dog-sized quadrupedal robots such as ANYmal [2], small robots [3] exhibit increased flexibility, making them well-suited for navigating constricted spaces due to their smaller size, lower weight, and reduced cost.

Reinforcement learning in quadruped locomotion has gained attention as an alternative approach for conventional methods. On the one hand, conventional systems acquire manual effort to model both the robotics system and the target task scenario, often leading to laborious tuning. On the other hand, reinforcement learning is a viable approach for generating robot gait for complex terrain by analyzing the robot's status during walking. However, implementation of reinforcement learning for small-sized quadrupedal robots like rat robots has not been widely studied. When it comes to rat robots, the limited resource allocation and streamlined structure could hardly support the effectiveness of previously proposed common reinforcement learning methods. Therefore, this paper focuses on the motion generation of the rat robot in a source-limited scenario.

*Corresponding author: Kai Huang. Email: huangk36@mail.sysu.edu.cn

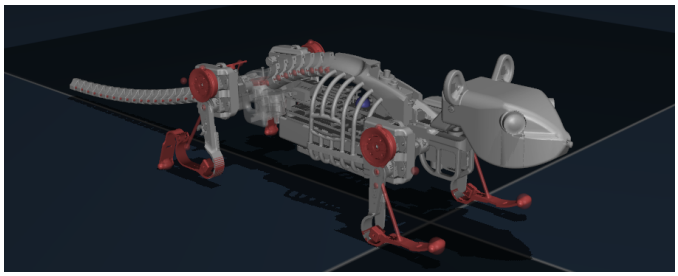


Fig. 1. The rat robot is designed with soft actuated components and supports more flexible robot motions.

Implementation of reinforcement learning for small-sized quadrupedal robot locomotion is challenging due to varied action scales and limited perceptual information. For one thing, various structure designs of robots endow them with complex nonlinear dynamics. Specifically, soft actuated legs of the rat robot, which possess an infinite number of digrees of freedom(DOF) theoretically [4], render kinematic modeling challenging and result in higher control errors. For another, a huge high-dimensional action space of multi-joint robots makes rewards sparse because direct control signals to the robot typically fall within a small time scale. For example, a 60 Hz motor signal takes dozens of executions to allow the robot to achieve a tiny movement of lifting a front claw. As a result, *how to simplify exploration of the robot's action is a fundamental problem for reinforcement learning of the rat robot*.

In addition, low perception efficiency poses another challenge, coming from absence of high-precision sensors. Due to limitations in size, weight, area, and power (SWAP), conventional sensors are frequently unfeasible for small-sized robots [5], which is critical for common-size robotics locomotion. Lacking joint angle sensors or touch sensors like the MIT Cheetah [6] and ANYmal robots [2], the rat robot is equipped with an inertial measurement unit (IMU), which has high data noise together with low power consumption and small size. Light sensors acquire for additional processing to attain comparable levels of perception to higher-quality counterparts

[7]. Therefore, *how to extract valid information from limited noisy IMU data for perception and self-learning is a problem to be solved*.

To tackle the above challenges, we propose a novel framework for terrain-aware motion generation of the rat robot, based on reinforcement learning. This framework is aimed to simplify exploration of the robot's action and observation space. The rat robot's motions utilize normalized signals and hierarchical mappings in mathematical space, which significantly improve training speed while maintaining motion flexibility. We extract states that integrate with the robot's self-action status by filtering time-series observations during interaction with the environment. The compression of state space by these effective states leads to improved training speed. The Main contributions of this work are summarized as follows:

- We propose a novel hierarchical motion generation approach that controls the robot's actions based on normalized signals and hierarchical mappings on mathematical space to simplify the learning process.
- We propose a novel training method with time cluster updating that incorporates processing of perception information integrated with the robot's self-action status based on changes in time slices, which reinforces perceptual feedback thus contributing to improved training efficiency.

II. Related Work

While many problems in the field of robotic control can be naturally formulated as reinforcement learning problems, the domain of robotics differs significantly from most well-studied benchmark problems in reinforcement learning [8], necessitating the development of new targeted approaches.

Deep reinforcement learning has been extensively utilized for legged locomotion of quadrupedal robots in recent years [9], [10], [11], [12]. The model-free approach [13] employs actions such as torques and joint angles to enable end-to-end control of the robot. However, learning efficient gaits from scratch is challenging in quadrupedal robots due to their complex nonlinear dynamics and sparse rewards. Some End-to-end reinforcement learning methods seek critical control instructions from the network. Hwangbo et al. [14] uses supervised learning to train a neural network to build the mapping of the target joint position to the joint torque, on top of the strategy network for motion direction. However, it still faces challenges in learning effective gaits from scratch in quadruped robots due to complex nonlinear dynamics and sparse reward signals of quadrupedal robots.

Model-based reinforcement learning methods [15] typically utilize a predetermined model of system dynamics to enhance training. Lee et al. [9] employs a trajectory generator to obtain an a priori motion trajectory, which is then refined using reinforcement learning, resulting in a significant reduction in training difficulty. ETG-RL [16] utilizes an evolutionary trajectory generator to optimize the shape of the output trajectory for the given task, thus providing diversified motion priors to guide policy learning. Lee et al. [9] employs neural networks to extract ground information, which is not directly available to the robot's sensors, mined from the robot's proprioceptive information.

The application of reinforcement learning in rat robots is currently limited. Xie et al. [17] recently developed a motion generation strategy for rat robots to interact with animals, which selects pre-defined actions through a network. However, further work is needed on the generation of actions for rat robots. The implementation of reinforcement learning on small-sized quadruped robots requires a high level of perception despite limited sensor information, where further research is needed. Full-sized robots like ANYmal utilize machine learning technology to build an altitude map of the surrounding environment by mining data from tactile sensors. However, small-sized robots face additional challenges due to equipment limitations. Maurice et al. [18] implemented linear policies to enable a small middle-sized quadruped robot (with a height of 200mm) to navigate uneven terrain, demonstrating the potential for further processing of IMU data in the locomotion system of small-sized robots.

To summarize, on one hand, although current reinforcement learning methods have shown promising results in various tasks, it is acknowledged that the training speed is often slow, with convergence typically occurring at around 10 million iterations. This is comparable to exhaustively traversing the entire action space, which not only prolongs the training process but also limits the exploration of the robot's kinematics and dynamics. On the other hand, motion control of small mouse robots is hindered by limited perception capabilities and structural features, and the existing research in this area is still limited. Therefore, there is a need for an efficient deep adaptive control framework specifically designed for small-sized rat robots.

III. Framework Overview

In this section, we give an overview of our proposed framework for the rat robot, which is shown in Fig. 2.

One aspect of our work involved redesigning decision-making actions for rat robots in order to simplify the action space. Multi-joint quadruped robots have a large, high-dimensional action space due to their high degree of freedom and the continuity of control signals. In policy learning, obtaining occasional positive reward feedback requires the robot to go through numerous exploration processes due to the sparse nature of environmental feedback rewards. Algorithm convergence may be hindered by an excessively complex continuous action space that prevents the accumulation of positive reward information by the learning process. In the part of hierarchical motion design (Section IV), we firstly map the motor control signal Φ to coordinates in end-effector space of four legs through a mathematical kinematic model. Then, around the motion center, they are mapped to polar coordinate system as (θ, ρ), where actions are redefined.

We calculate the phase θ based on the current time t and gait pattern to represent the periodic state of each leg in different

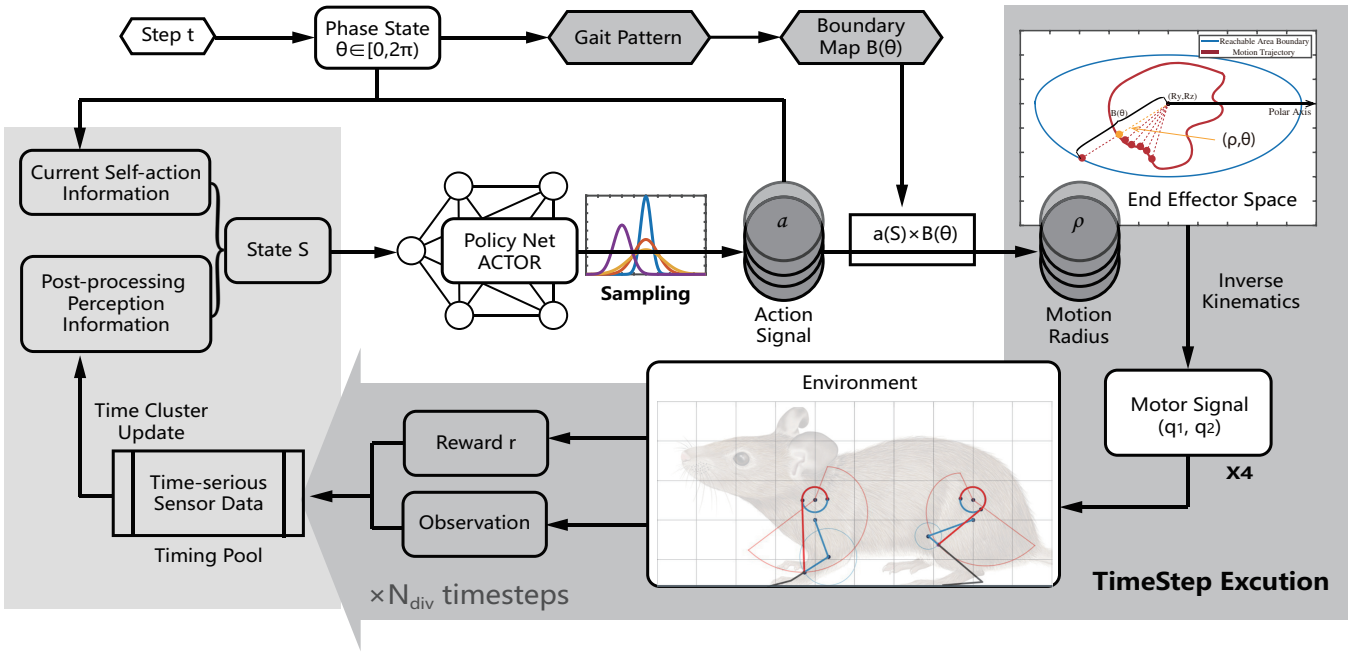


Fig. 2. Framework for robot gait training on complex terrains. The part of motion design maps the control signals of each leg to the angular values of the two drive motors through a hierarchical mapping, which is illustrated in Section IV. During the training process, multi-timestep running feedbacks are collected to generate state nodes for training by process of time-serious information, which is illustrated in Section V.

gaits. The location of the end point drop can be determined for any phase θ by specifying the polar radius ρ. We also define a maximum range of motion, which represents the maximum calculable polar radius. The action signal is defined as the ratio between the polar radius of the end point's landing position and the maximum range of motion, with a range of 0 to 1. At each time step t, the landing points of the end points of the four legs are determined using a 4-dimensional action signal $\mathbf{a}_t$. Next, inverse kinematics calculations are performed to obtain eight motor signals for direct robot control. This approach based on a mathematical model improves training speed while retaining motion flexibility.

Another aspect of our work involves filtering high-value information from a limited amount of observations to reduce the complexity of environmental feedback information. On one hand, robots in complex environments are equipped with powerful sensing devices, while small robots like rat robots can only rely on light sensors that provide high-noise information. On the other hand, environmental observations exhibit temporal correlations, and valuable motion feedback information is embedded in the time sequence. In the section on time cluster updates (Section V), we create a timing pool to accumulate environmental feedback over a period of time. Following a post-filtering process, valid information is extracted from the time series data. Finally, the extracted data is integrated with the robot's current motion information in a Markov node, and used to update the policy.

IV. Motion Generation Based on a Hierarchical Mapping

This section elucidates the design of motion for the rat robot. A hierarchical framework is constructed to generate motion signals by sequentially querying the planning policy θ_P, taking into consideration the training efficiency and exploration of the robot's kinematics and dynamics.

In the end-effector coordinate space, we defined the motion representation in polar coordinates (θ, ρ). Gait patterns are encoded using the phase θ, and the target landing position of the end-effector can be determined by specifying the radial distance ρ. The top-level motion command of the control system is defined as the ratio between the radial distance of the target landing position of the end-effector and the maximum range of motion.

The rat robot based on our previous work is a type of small-scale bio-inspired robot, designed with soft actuated components that differ from dog-size quadrupedal robots. The bio-inspired legs of our robot are designed to closely mimic the body structure and movement pattern of natural mice, while the flexible structures pose challenges to traditional quadrupedal motion control methods, particularly when reinforcement learning is implemented. As Figure 1 shows, each leg of the rat robot is driven by tendon ropes with two motors on the hip. Each motor has a rotation range of 180 degrees. As a result, the direct control signal in a single time step is denoted as

$$\mathbf{q} \in \mathbb{R}^8_{clip}, \mathbb{R}^8_{clip} := [-\frac{\pi}{2}, \frac{\pi}{2}] \tag{1}$$

which consists of 8 motor signals, each two working together

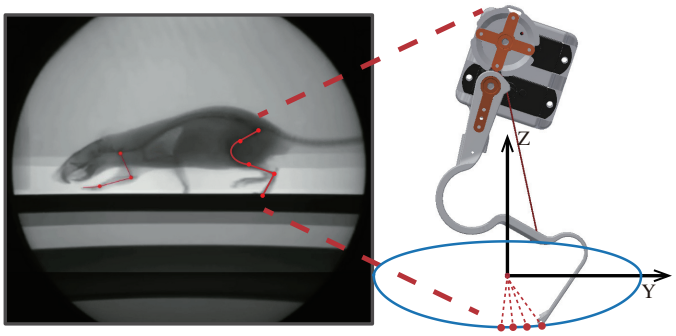


Fig. 3. Structure of the bionic leg, consisting of an elastic limb, a driving tendon and two actuating motors. End point of the limb moves in the 2-Dimension end-effector space, driven by two actuators. For the flexible leg driven by dual motors, the end-effector space is located in the Y-Z plane of its own Cartesian coordinate system. Schematic diagram of the leg movements is cited from [19].

for a leg, and the rat robot gait can be specified by a sequence $\Phi = \{\mathbf{q}\}$.

The motor control of the rat robot is determined by the motor signals of the joints, denoted as $\boldsymbol{q}$ in Equation 1. Each set of leg joints is underactuated and controlled by two motors. In most reinforcement learning methods, the motor rotation angle is defined as an action. However, in the context of rat robots, such an action space setup results in discontinuous actions in the end-effector space. This is because numerically similar motor values often correspond to very different locations when mapped to the end-effector space, posing a significant challenge for learning intelligent behaviors. In our work, motion planning is performed in the end-effector space and incorporates gait phase information through coordinate transformation.

Initially, in the end-effector workspace, we define a maximum reachable motion space $\mathbf{S}_r$ based on the original motion trajectory in our previous work [20], which balanced the step length and foot clearance.

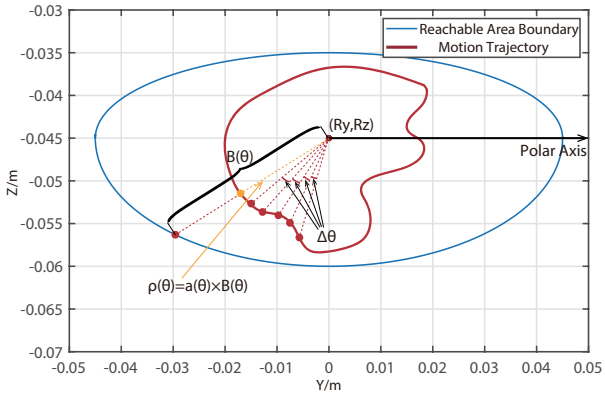


Fig. 4. Motion Signal of Four Legs in Scene Board. $\Delta\theta$ is related to factors including control frequency, response time, and physical constraints.

To integrate gait information into the motion signals, the end-effector space is converted from the Cartesian coordinates (y, z) to the polar coordinates (ρ, θ), with the center of motion (C_y, C_z) as the pole. The maximum range of motion $B(\theta)$ for any phase θ can be obtained by

$$B(\theta) \leftarrow \partial \mathbf{S}_r. \tag{2}$$

At any phase θ, the location of the end point drop can be determined by specifying the polar radius ρ. Now we can define the motion signal of four legs as the ratio between the polar radius of the end point's landing position and the maximum range of motion with format

$$\mathbf{a}_t \in \mathbb{R}^4_{clip}, \mathbb{R}^4_{clip} := [0, 1] \tag{3}$$

where the four components represent four legs for the robot.

By adjusting the motion signal $a_t \in [0, 1]$, the corresponding motion position at phase θ can be controlled by

$$\rho(\theta) = a_t B(\theta), \ (\rho, \theta) \in \mathbf{S}_r \tag{4}$$

Here, phase state θ is associated with time. In a complete motion cycle T, the phase state of leg motion is

$$\theta_i(t) = 2\pi * (t/T + \ \phi_i), t \in [0, T) \tag{5}$$

where i from 1 to 4 represent the left front leg, right front leg, left hind leg, and right hind leg, respectively, and ϕ_i is the phase bias of each leg. In the Trotting gait, which is applicable to low and medium speed movements, they are$\phi_1 = 0, \phi_2 = 0.5, \phi_3 = 0, \phi_4 = 0.5$.

a_t can be obtained from the motion generation network:

$$\mathbf{a}_t \leftarrow A(O, \theta_P) \tag{6}$$

where $A(O, \theta_P)$ is the motion generation network and O is the observation of the environment, which can be a high-dimensional vector. By controlling the size of $\mathbf{a}_t$, we can generate motion trajectories of arbitrary shapes within the reachable region. When $a_t = 1.0, \forall\theta \in [0, 2\pi)$, the motion trajectory degenerates into a pre-defined smooth closed curve.

Back to the Cartesian coordinate system(y, z) in end-effector space, each motion track point can be obtained by coordinate transformation:

$$\begin{aligned} y &= C_y + \rho(\theta)\cos\theta, \\ z &= C_z + \rho(\theta)\sin\theta. \end{aligned} \tag{7}$$

where C_y and C_z are the centor of motion. In equation 7, (y, z) represents a point on the trajectory of a single limb.

Subsequently, the coordinates of the end-effector space point are mapped back to the rotation signals of the two motors at the upper and lower hip joints through mathematical inverse kinematic calculations:

$$\begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = M_{inv} \begin{bmatrix} y \\ z \end{bmatrix}. \tag{8}$$

Here, q_1 and q_2 represent the rotation signals of the two motors, while M_{inv} is the inverse kinematics matrix obtained through a mathematical bijection established in our previous work [20]. This mathematical model allows for design motions in the end-effector space of the robot quadruped, which has a direct physical meaning for the motion of the robot.

After the aforementioned steps, the motion of the rat robot can be determined by a compact four-dimensional vector. Information of the motion range and gait phase improves the training efficiency, while the action signals defined through hierarchical analytical mapping provide freedom for exploration.

V. TRAINING ARCHITECTURE WITH TIME CLUSTER UPDATING

Our training architecture integrates the approach of motion design and time cluster updating within a generic reinforcement learning framework. Time-series processing has been used to reduce the dimensionality of complex environment observations.

In addition to structural peculiarities, limited sensor equipment affects perception. The rat robot has only one IMU at its center, and global environment information is concealed. As a result, the immediate observation of the environment is constrained to $o_t = \{v, g\}$, where $v \in \mathbb{R}^3$ represents velocity and $g \in \mathbb{R}^3$ represents angular velocity.

In reality, the problem of reward sparsity resulting from weak feedback in robot movements makes training challenging in practice. The control effect of an individual motion signal depends on factors such as control frequency, response time, and physical constraints. In a continuous control scenario, a motor signal within a timestep results in a limited change in pose. Consequently, utilizing a single motor signal as a motion on a Markov node would significantly expand the search space of the Actor network, thereby impeding the robot's motion learning process.

To address the aforementioned problems, we create a sequence of motor control signals spanning multiple timesteps, referred to as a "time cluster", to enable the robot to receive adequate feedback from the environment.

Algorithm 1 RL with Rat Robot

Require:S_0,State with initial environment
β, experience replay buffer
α, learning rate
N, the number of time steps a step cluster contains
$\epsilon \sim \mathcal{N}(0, 1)$, Gaussian noise for exploration

Initiate the policy
while Not Converged **do**
  Get action signal $a_t \leftarrow \pi_\theta(s) + \epsilon$
  for $i = 1$ to N **do** ▷ Generate One-Step time cluster
    Get motion radius $\rho(\theta) = a_t \times B(\theta)$
    Transfer to motor signal $(q_1, q_2) \leftarrow (y, z) \leftarrow \rho_t(\theta)$
    Append IMU Data into Timing Pool
  end for
  Generate State S of current time cluster
  Append the transition (s_t, a_t, r, s_{t+1}) into β
  if β is full **then**
    Update Policy
    Reset β
  end if
end while

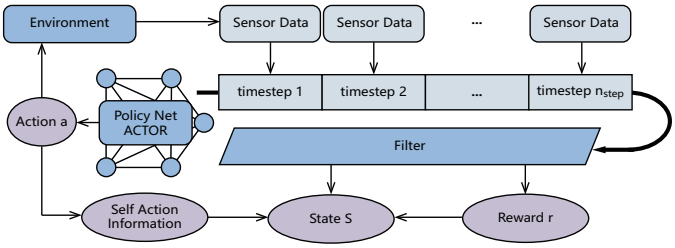


Fig. 5. Process of State Node Generation within a Time Cluster

In our training framework, an RL step comprises a segment of end-effector trajectory that spans multiple timesteps. In any arbitrary gait, a complete motion cycle of length T can be divided into multiple trajectory segments, and the number of timesteps in each segment is calculated by

$$n_{step} = (fT_0N_{div})^{-1} \tag{9}$$

where n_{step} represents the number of timesteps in one RL step, f denotes the frequency of periodic motion, T_0 stands for the minimum control interval (set to $0.002s$ in our simulation), and N_{div} refers to the number of period divisions. For instance, with $N_{div} = 8$, a single time cluster comprises $1/8$ of the motion trajectory of T.

The process of generating state nodes within a time cluster is illustrated in Fig. 5. During the execution of motions within a time cluster, sensor data is collected into the Timing Pool, where it is processed to obtain valid sensory information with filtered noise.

We use the mean filter to process the sensor information. The processing of the three-axis velocity information is as follows:

$$\mathbf{V}^i = \sum_{k=0}^{n_{step}} \frac{v_k^i}{n_{step}} \tag{10}$$

where i from 1 to 3 corresponds to three directions, and v_k^i is the kth sensor data of velocity within a time cluster.

In a continuous run, the computation at the end of a time cluster needs to be reduced in order to balance the computation time within each timestep for real-world execution. We modify Equation 10 into an incremental form in order to update the amount of state information in real time:

$$\mathbf{V}_k^i = \frac{k-1}{k}\mathbf{V}_{k-1}^i + \frac{1}{k}v_k^i. \tag{11}$$

The processing of the angular velocity follows the same approach. Note that the selection of filter types is not the focus of our research, where other filters can be used as well.

After generation of a Timing Cluster, the processed perceptual information together with the information of its own action in the current RL step form the State s in the Markov node

$$s_t = \left(a_t, i_C, r, \vec{V_{vel}}, \vec{V_{gyro}}\right) \tag{12}$$

where i_C is the index of the phase segment of the time cluster in the entire motion cycle T, r is the reward value, $\vec{V_{vel}} \in \mathbb{R}^3$

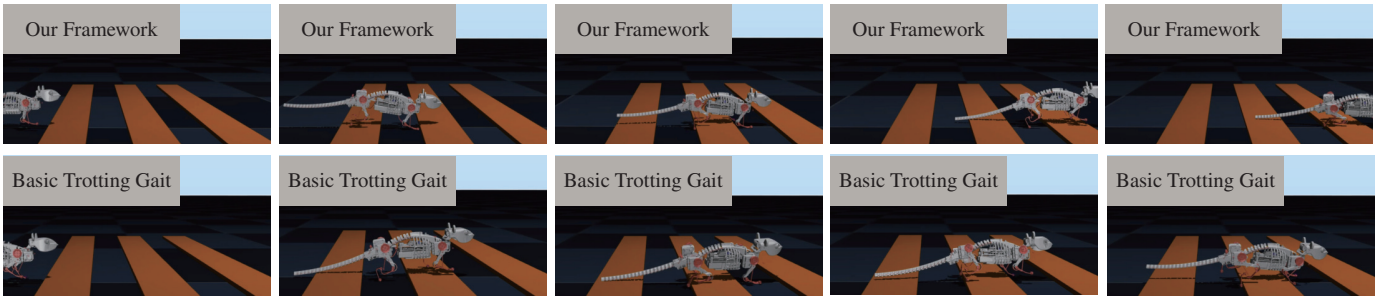


Fig. 6. Montage of walking status on "Planks": the first line shows the movement of the rat robot controlled with our reinforced gait. The robot touches the middle plank at 3 s and it takes less than 6 s to pass. The second line shows the movement of the rat robot controlled with a simple trotting gait. The robot has got trapped at the middle plank since 3 s, failing to overcome the obstacle.

is the filtered velocity, and $\vec{V_{gyro}} \in \mathbb{R}^3$ is the filtered angular velocity.

After post-processing of data in the time cluster, the reward of one single time step can be defined as

$$r = K_{vel}\vec{V_{gyro}} \cdot \vec{u_{dir}} \tag{13}$$

where $K_{vel}$is the weighting factor(as a hyper-parameter.), $\vec{u_{dir}}$ is a unit vector for direction. Our work of reward shaping did not focus on specific body structure or character of environments. Above design of the reward applies for other tasks of quadruped robots as well while remaining room for improvement in specific scenarios.

We parameterize the Control policy as a Gaussian distribution with a diagonal covariance matrix.

$$\pi_\theta\left(a \mid s_t\right) = \mathcal{N}\left(a \mid \mu_\theta\left(s_t\right), \sigma_\theta\right). \tag{14}$$

A deep neural network (DNN) structure is employed to generate the mean $\mu_\theta\left(s_t\right)$ of the distribution, with inputs from both exteroceptive measurements and self-action information. Additionally, the standard deviation σ_θ is generated by a separate and independent network layer, which facilitates exploration during training. The complete algorithm is outlined in Algorithm 1.

VI. EXPERIMENTS

This section provides a detailed description of the experimental setup employed to assess the performance of the proposed framework. The motion of the rat robot is analyzed to demonstrate the effectiveness of our proposed method. Subsequently, we elaborate on the efficiency of the algorithm. The video of experimental results is on [21].

A. Experimental Setup

Four terrain scenarios were constructed for the purpose of training the rat robot to navigate, as illustrated in Figure 7. The size of the terrain scenarios is designed to simulate the real-world environment encountered by small robots.

The rat robot is a small quadrupedal robot that has dimensions of 40 cm $\times$ 25 cm and features 8 degrees of freedom for control. Each leg of the robot has a step length of 9 cm and a foot clearance of 1.5 cm. Using the physical platform utilized in our prior research, as depicted in Figure 1, a digital twin was created in the MuJoCo environment.

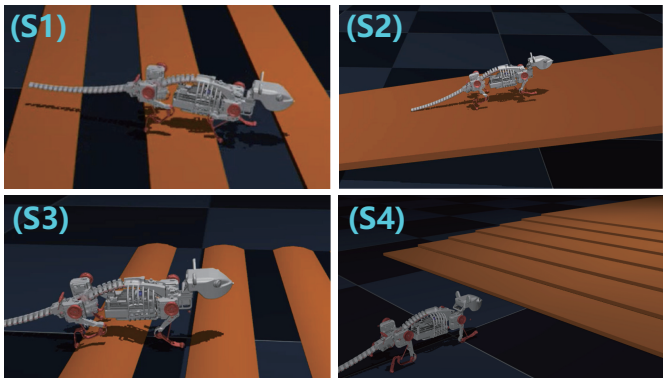


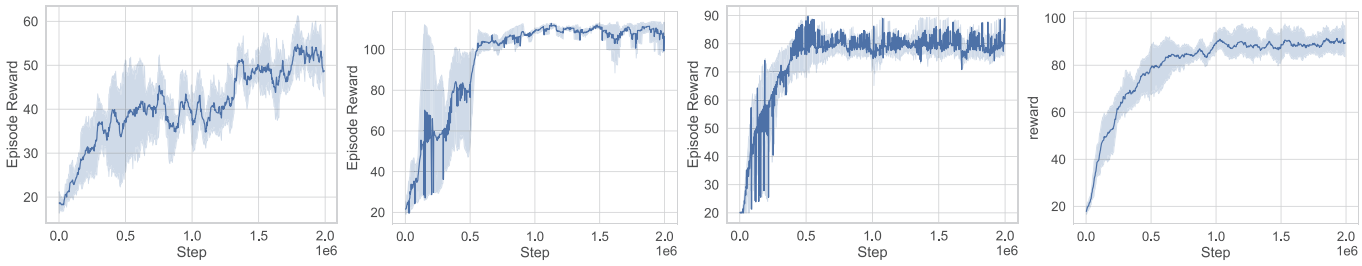
Fig. 7. The proposed approach can adapt the rat robot to all 4 test scenarios. (S1)Planks: Gallop over planks with wide gap(width=10 cm).(S2)Uphill: a 10-degree slope. (S3)Logs: Upon a pile of logs. (S4)Stairs: Micro stairs between two platforms.

The simulations are carried out using the MuJoCo robot simulation platform [22](Version 2.1.0), operating on a Ubuntu 18.04 system. The training machine is a server equipped with dual Intel Xeon Gold 6234 Processors and 256GB of memory. Each training session is allocated one physical core and requires less than 2GB of memory.

B. Analysis of Motion

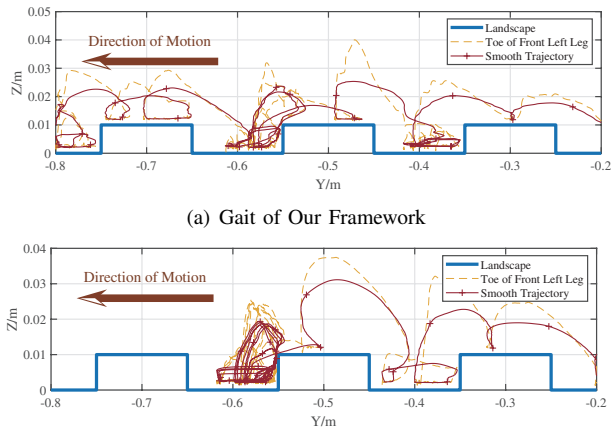
Our rat robot successfully completed all four tasks. Fig. 6 presents a montage of the rat robot walking forward in Scenario Planks. The first row displays the gait generated by our framework, while the second row shows the basic trotting gait. The end point trajectory of the rat robot is depicted in Fig. 9. Scenario Planks is selected as the analysis sample due to its higher difficulty level, as evident from the training curves in Fig. 8.

It is important to note that the robot relies solely on the IMU at the center of its body for perception, and does not possess any other perceptual capabilities or access to global environmental information. During the early stage, the robot



(a) Planks (b) Uphill (c) Logs (d) Stairs

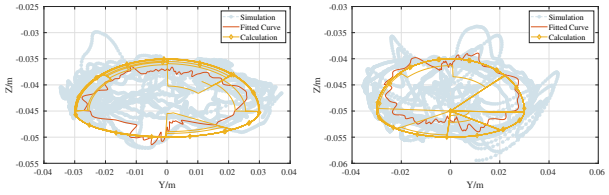
Fig. 8. Training curves on 4 simulation tasks. The solid lines represent the average score, while the shaded areas represent a standard deviation.



(a) Gait of Our Framework

(b) Basic Trotting Gait

Fig. 9. End point trajectory of the rat robot's leg in (S1)Planks. Protrusions of the blue line named Landscape indicate the obstacles that the robot encounters while moving forward. The robot successfully overcame the obstacle after adaptive adjustment of our framework, but got stuck at the second plank with basic trotting gait.



(a) Front Left Leg (b) Hind Left Leg

Fig. 10. Motion curves of leg endpoint in (S1)Planks, with local coordinate system centered on the hip.

moves on flat ground with an adaptive gait, without sensing the obstacle ahead, until it touches the plank. In the next stage, the robot becomes trapped between the planks due to its initial gait, which prompts it to change its motion signals in order to adapt to the unknown landscape. Once the robot fully senses the environmental information, it regains the ability to move forward. In the third stage, the robot successfully climbs over the first plank using a new adaptive gait. At each plank gap, there is a transition from perception to motion generation. After successfully climbing over three planks, the robot returns to the flat ground, reverting to its initial gait.

Figure 10 displays the motion curves of the front and hind legs during the process of climbing over the first plank. The motion signals output by the neural network undergo dynamic adjustments during interactions with the environment. The motion range of the robot's end-effectors encompasses a significant motion trajectory (as depicted by the fitted curves) and several stray tuning movements, covering most of the feasible region.

C. Algorithm Efficiency

The training curves depicted in Figure 8 demonstrate that episode rewards converged within hundreds of thousands of step iterations, surpassing the convergence speed of most widely-used reinforcement learning methods. It is well-known that conventional reinforcement learning methods like PPO tend to be slow in training, despite their good performance in terms of converged steps which can reach the order of tens of millions.

While certain methods, such as ETG-RL [16], achieve convergence within a million steps, the total number of iterations still amounts to tens of millions when incorporating a lengthy pre-training process. In contrast, our approach can effectively obtain gait patterns within hundreds of thousands of step iterations by initiating from scratch.

The experimental results demonstrate that the neural network strategy is capable of learning effective gait through reinforcement learning (RL), facilitated by our model-based motion generation and post-processing of time-series information and state integration, specifically tailored for small robots. Our training did not yield optimal results due to the absence of complex reward shaping and network design. However, the concept of motion generation and time cluster update is not limited to specific tasks and can be equally applicable to other small robots as well.

It is worth noting that previous works on reinforcement learning for quadrupedal locomotion, such as Deep Gait [23] and ETG-RL [16], are not directly applicable to our rat robot due to differences in body size and perceptual ability. On one hand, the physical structure of tiny robots, in contrast to dog-size ones, requires a distinct motion design approach. Our rat robot is constructed using flexible materials and its limbs are actuated by rope tendons, as opposed to rigid body-based motor control. On the other hand, our rat robot relies on an

Inertial Measurement Unit (IMU) for perception, without the use of sophisticated sensors like tactile sensors or LIDAR, which are commonly employed in dog-size robots.

VII. CONCLUSIONS

This paper presents a hierarchical framework for autonomously generating robot gait for complex terrains. Our proposed approach combines the motion generation of our rat robot with model-based methods and hierarchical mapping, simplifying the training process while preserving exploration of the robot's kinematics and dynamics. Through post-processing of time-series data during the robot's interaction with the environment and time cluster updating that integrates self-action status and perceptual information from the environment, the initially large state space is compressed to enhance training speed. We conducted a series of case studies, and the results demonstrate the effectiveness and training efficiency of our proposed framework. The proposed method was successful in all challenging scenarios tested, and it can be applied to other robotic applications as well.

REFERENCES

[1] P. Lucas, S. Oota, J. Conradt, and A. Knoll, "Development of the neurorobotic mouse," in *2019 IEEE International Conference on Cyborg and Bionic Systems (CBS)*. IEEE, 2019, pp. 299–304.

[2] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. Hoepflinger, "Anymal - a highly mobile and dynamic quadrupedal robot," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 38–44.

[3] S. M. Neuman, B. Plancher, B. P. Duisterhof, S. Krishnan, C. Banbury, M. Mazumder, S. Prakash, J. Jabbour, A. Faust, G. C. de Croon, and V. J. Reddi, "Tiny robot learning: Challenges and directions for machine learning in resource-constrained robots," in *2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 2022, pp. 296–299.

[4] D. Rus and M. T. Tolley, "Design, fabrication and control of soft robots," *Nature*, vol. 521, no. 7553, pp. 467–475, May 2015. [Online]. Available: https://doi.org/10.1038/nature14543

[5] B. P. Duisterhof, S. Krishnan, J. J. Cruz, C. R. Banbury, W. Fu, A. Faust, G. C. H. E. de Croon, and V. Janapa Reddi, "Tiny robot learning (tinyrl) for source seeking on a nano quadcopter," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 7242–7248.

[6] S. Seok, A. Wang, M. Y. Chuah, D. Otten, J. Lang, and S. Kim, "Design principles for highly efficient quadrupeds and implementation on the mit cheetah robot," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 3307–3312.

[7] P. Fankhauser, M. Bjelonic, C. Dario Bellicoso, T. Miki, and M. Hutter, "Robust rough-terrain locomotion with a quadrupedal robot," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 5761–5768.

[8] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[9] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, p. eabc5986, Oct. 2020, publisher: American Association for the Advancement of Science.

[10] S. Gangapurwala, M. Geisert, R. Orsolino, M. Fallon, and I. Havoutis, "Real-time trajectory adaptation for quadrupedal locomotion using deep reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 5973–5979.

[11] Y. Ji, Z. Li, Y. Sun, X. B. Peng, S. Levine, G. Berseth, and K. Sreenath, "Hierarchical reinforcement learning for precise soccer shooting skills using a quadrupedal robot," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 1479–1486.

[12] J. Wang, C. Hu, and Y. Zhu, "Cpg-based hierarchical locomotion control for modular quadrupedal robots using deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7193–7200, 2021.

[13] T. Degris, P. M. Pilarski, and R. S. Sutton, "Model-free reinforcement learning with continuous action in practice," in *2012 American Control Conference (ACC)*, 2012, pp. 2177–2182.

[14] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[15] X. Li, W. Shang, and S. Cong, "Model-based reinforcement learning for robot control," in *2020 5th International Conference on Advanced Robotics and Mechatronics (ICARM)*, 2020, pp. 300–305.

[16] H. Shi, B. Zhou, H. Zeng, F. Wang, Y. Dong, J. Li, K. Wang, H. Tian, and M. Q.-H. Meng, "Reinforcement Learning With Evolutionary Trajectory Generator: A General Approach for Quadrupedal Locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3085–3092, Apr. 2022, conference Name: IEEE Robotics and Automation Letters.

[17] H. Xie, G. Jia, M. Al-Khulaqui, Z. Gao, X. Guo, T. Fukuda, and Q. Shi, "A motion generation strategy of robotic rat using imitation learning for behavioral interaction," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7351–7358, 2022.

[18] M. Rahme, I. Abraham, M. L. Elwin, and T. D. Murphey, "Linear policies are sufficient to enable low-cost quadrupedal robots to traverse rough terrain," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 8469–8476.

[19] A. Rohregger. (2021) Mouse gait: Visual analysis of front leg. Website. [Online]. Available: https://www.notion.so/Rodent-Gait-89ae86764cc243cfa1d58a04dace5a15

[20] Y. Huang, Z. Bing, F. Walter, A. Rohregger, Z. Zhang, K. Huang, F. O. Morin, and A. Knoll, "Enhanced quadruped locomotion of a rat robot based on the lateral flexion of a soft actuated spine," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 2622–2627.

[21] Z. Zhang. (2023) Experiment video. Website. [Online]. Available: https://github.com/RicardoZon/SMC2023

[22] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[23] 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*, vol. 5, no. 2, pp. 3699–3706, 2020.