npj | robotics

Article

https://doi.org/10.1038/s44182-025-00043-2

Learning stable bipedal locomotion skills for quadrupedal robots on challenging terrains with automatic fall recovery

Erdong Xiao, Yinzhao Dong, James Lam & Peng Lu ✉

Reinforcement learning has made remarkable strides in advancing quadrupedal locomotion. However, achieving bipedal locomotion for quadrupedal robots remains extremely challenging due to less contact with the surface. Additionally, during the transition from quadrupedal to bipedal locomotion, the body axis shifts from horizontal to vertical, and the center-of-mass rises suddenly. Here, we present TumblerNet, a deep reinforcement learning controller that enables robust bipedal locomotion for quadrupedal robots. Our proposed framework features an estimator that estimates the center-of-mass and center-of-pressure vector and rewards based on this vector, which allows the learning controller to monitor and maintain the balance of the robot during bipedal locomotion. As such, the proposed framework, although only trained on flat ground in simulation, can be directly deployed in a real robot on various terrains without additional training. The proposed framework exhibits exceptional robustness against various challenging terrains (uneven and soft terrains) and external disturbances, with automatic fall recovery.

Quadrupedal robots have the potential to execute tasks in challenging and hazardous environments, such as mountains and underground tunnels. The prerequisite for executing such tasks is the locomotion capability on rough and challenging terrains. Over the past few years, quadrupedal robots have made tremendous progress in terms of locomotion, transitioning from walking in controlled lab environments to traversing various real-life terrains, including challenging terrains, such as mountains. This evolution in locomotion lays a solid foundation for these robots to be widely deployed in various challenging missions.

Despite the progress, achieving bipedal locomotion for quadrupedal robots remains extremely challenging. Bipedal locomotion is necessary when quadrupedal robots traverse narrow spaces, where they must crab with two legs to pass through. Walking like a biped may be the most difficult challenge for a quadruped robot in terms of locomotion. It means that quadrupedal robots will lose their inherent stability with four legs. With fewer legs, the robot has less support from the ground, making it more difficult to balance. Existing gait optimization and control methods for quadrupedal robots will lose efficacy. Many gaits that were initially designed for quadrupedal robots, such as trotting, pacing, and bounding, do not apply anymore. All feasible region-based methods that are based on three or diagonal legs are also not applicable. Furthermore, changing from quadrupedal to bipedal locomotion will result in a dramatic change in the center of mass (CoM), which can significantly impact stability.

Walking with two legs for a quadrupedal robot can be even more challenging than that for a bipedal robot. Firstly, the structure of quadrupedal robots is not optimized for bipedal locomotion, unlike bipedal robots that are specifically designed for walking on two legs. Secondly, bipedal robots have more degrees of freedom in the leg, allowing for enhanced stability on uneven terrains. The feet of bipedal robots have several degrees of freedom to provide line-wise or plane-wise contact with the surface, benefiting stability and weight distribution. In contrast, the feet of a quadrupedal robot only provide point-wise contact. Additionally, the motors of quadrupedal robots are designed for walking with four legs, and switching to bipedal locomotion requires the remaining two motors to bear the load of the entire robot, posing challenges to the remaining motors.

Conventional methods, especially optimization-based methods, have made significant advancements in the planning and control of quadrupedal robots[1–6]. Typically, they first plan the motion of the quadrupedal robot based on a simplified centroidal dynamics model that captures the main dynamics of the complex robot[2,4,7]. Then, model predictive control is employed to track the planned commands. Although these methods are efficient in generating dynamic gaits, they are still confronted with challenges, particularly when walking on complex terrains[8]. When dealing with challenging terrains, these methods either assume that prior terrain information is available[9] or rely on deep neural networks to generate a contact sequence that provides safe foothold locations[10].

Department of Mechanical Engineering, University of Hong Kong, Pokfulam, Hong Kong, China. ✉e-mail: lupeng@hku.hk

In contrast, reinforcement learning provides an alternative solution to quadrupedal locomotion without prior information about the terrain and has made tremendous progress over the past few years. Reinforcement learning has enabled quadrupedal robots to perform agile and dynamic motions[11,12]. Moreover, these learning-based methods have greatly enabled quadrupedal robots to walk on challenging terrains by using a teacher and student policy[8,9,13,14,14–17].

However, all these approaches consider scenarios where all four legs of the robot are functioning properly[8,9,11–15,17,18]. Tripod walking has recently been achieved using deep reinforcement learning[19]. However, none of them considered bipedal locomotion, which poses a significant challenge for quadrupedal robots. While a wheel-legged robot has achieved stand-up movement, it moves on wheels instead of feet[20]. Several studies have enabled quadruped robots to stand with two legs, but they are unable to perform bipedal locomotion[21,22]. An upside-down and hand-supported motion has been achieved using a parkour framework[23]. However, they did not demonstrate the locomotion of such a gait over uneven terrain. A similar bipedal standing motion has been achieved by ref. 24. However, they also did not perform bipedal walking. Therefore, the grand challenge of bipedal locomotion for quadrupedal robots remains open. There are significant changes for the controller when the robot changes from quadrupedal to bipedal locomotion mode. Firstly, the body axis changes from nearly horizontal to nearly vertical. Secondly, the CoM rises abruptly during the transition. Finally, the number of effective control inputs decreases from four to two. These significant changes pose enormous challenges to the stability of the controller, which have yet to be addressed.

Here, we present TumblerNet, a deep reinforcement learning framework that can achieve robust bipedal locomotion for quadrupedal robots. The framework is robust against strong external disturbances and challenging terrains. It can even recover from falling completely by itself without designing an additional recovery controller.

The key feature of our approach is a learning-based framework that can continuously monitor and maintain the balance of the robot. Conventional methods use a centroidal dynamics model to generate feasible trajectories[6]. However, these methods do not consider challenging terrains. Furthermore, they also do not monitor the balance of the robot during locomotion. Reinforcement learning-based methods[8,9,13–16] train the robot on various terrains in simulation and then deploy them in real practice, also without monitoring the balance. Monitoring the balance of the robot is very important for quadrupedal robots, especially over challenging terrains. This is particularly evident in bipedal locomotion, where inherent stability is lost compared to quadrupedal locomotion.

Our framework has two essential components that enable monitoring and maintaining the balance of a quadrupedal robot during bipedal locomotion. The first essential component is an Estimator Net that estimates the CoM-CoP vector, the robot's morphology, and linear velocities in real time. The CoM-CoP vector is an indicator of the robot's balance, and its estimation allows us to determine whether the robot is standing or falling in the bipedal locomotion mode. The CoM-CoP estimator takes as input the body angular velocity and projected gravity and outputs the real-time CoM-CoP vector. Ablation studies demonstrate that this CoM-CoP estimation network is essential for maintaining an upright position. Additionally, the Estimator Net uses privileged information to estimate linear velocity and the robot's morphology, which enables the robot to track velocity commands and achieve a universal controller for different morphologies of robots.

Another important component of our framework is the rewards, which are based on the CoM-CoP vector, that can significantly enhance the robustness of the learning controller. Designing the rewards for a quadrupedal robot to perform bipedal locomotion is distinct from that for quadrupedal locomotion, as the direction of the body axis changes significantly. By designing orientation, contact force, feet air time, and base height rewards, the quadrupedal robot can gradually learn to stand like a biped. However, this is not sufficient to perform stable bipedal locomotion. Therefore, more importantly, we have designed three reward terms that are based on the CoM-CoP vector, namely the pendulum angle, the angular acceleration of the pendulum angle, and the handle length between the horizontal projection of the CoM and the CoP. Analysis shows that these rewards significantly enhance the robustness of bipedal locomotion. The Estimator Net is trained in parallel with the reinforcement learning controller. By doing this, the CoM-CoP is estimated and controlled in a closed-loop fashion. As such, our proposed framework demonstrates exceptional robustness against various challenging terrains, external disturbances, and even falling.

Our proposed framework is so robust that the network trained solely on a flat surface in simulation can be directly transferred to challenging terrains in real practice without additional training. This is remarkable, given that challenging terrains have always been a significant obstacle for quadrupedal locomotion, let alone bipedal locomotion for a quadrupedal robot. We conducted extensive experiments over various challenging terrains, including uneven terrains, soft pads, rocky grounds, grass fields, and even a sandy beach. Quadrupedal locomotion on a sandy beach is extremely challenging due to the deformable and soft properties of the sand[25]. Existing learning-based approaches propose to model the terrain for quadrupedal locomotion training. However, our proposed framework enables bipedal locomotion on a sandy beach without modeling the sandy beach. All these experiments fully demonstrate the effectiveness and robustness of our proposed framework.

Furthermore, our proposed framework also exhibits strong robustness against external disturbances, such as pushing and kicking. More surprisingly, our proposed framework can enable automatic fall recovery even when the robot is pushed over. To recover from falling, existing learning-based methods[8] propose to train another recovery controller. In contrast, our framework can enable robots to quickly recover to an upright position without designing an additional recovery controller. This further highlights the strength and effectiveness of our proposed framework.

Results

Movie 1 summarizes the results of the presented work. The policy is trained on a flat surface in a simulation environment. We directly deploy the trained reinforcement learning controller on the real robot without fine-tuning.

Bipedal locomotion with front and hind legs

We train our deep reinforcement learning policy using proximal policy optimization (PPO) in Isaac Gym[26], with 1024 agents in parallel. The training takes about 2 h on an NVIDIA RTX 3060 Laptop GPU. The trained policy is then deployed in Gazebo to validate velocity tracking during bipedal locomotion with hind legs, where true velocity and CoM-CoP are available. The policy accepts user-specified linear and angular velocity commands. We command the quadrupedal robot to move forward, sideward, and rotate simultaneously (Fig. 1a), and as shown in Fig. 1c, it follows all commands concurrently during bipedal locomotion.

It should be noted that our policy is not limited to bipedal locomotion with hind legs. By changing the contact penalty for front legs to hind legs, the robot can perform bipedal locomotion with front legs. The training procedure is the same as bipedal locomotion with hind legs. The quadrupedal robot performing bipedal locomotion while tracking linear and angular velocities is shown in Fig. 1b.

The real experiments of the quadrupedal robot performing bipedal locomotion with front or hind legs can be found in Movie 1.

The ability to perform bipedal locomotion with front or hind legs demonstrates the flexibility of our proposed approach. Additionally, the tracking performance of the robot shows that our controller can successfully track simultaneous linear and angular velocity commands.

Analysis and evaluation of the estimators

In this subsection, we will evaluate the performance of the proposed Estimator Net and analyze the key information necessary for achieving a satisfactory velocity and CoM-CoP estimation. Note that the estimators are trained in parallel with the deep reinforcement learning controller using PPO[27] instead of being trained separately. We perform the evaluation in the

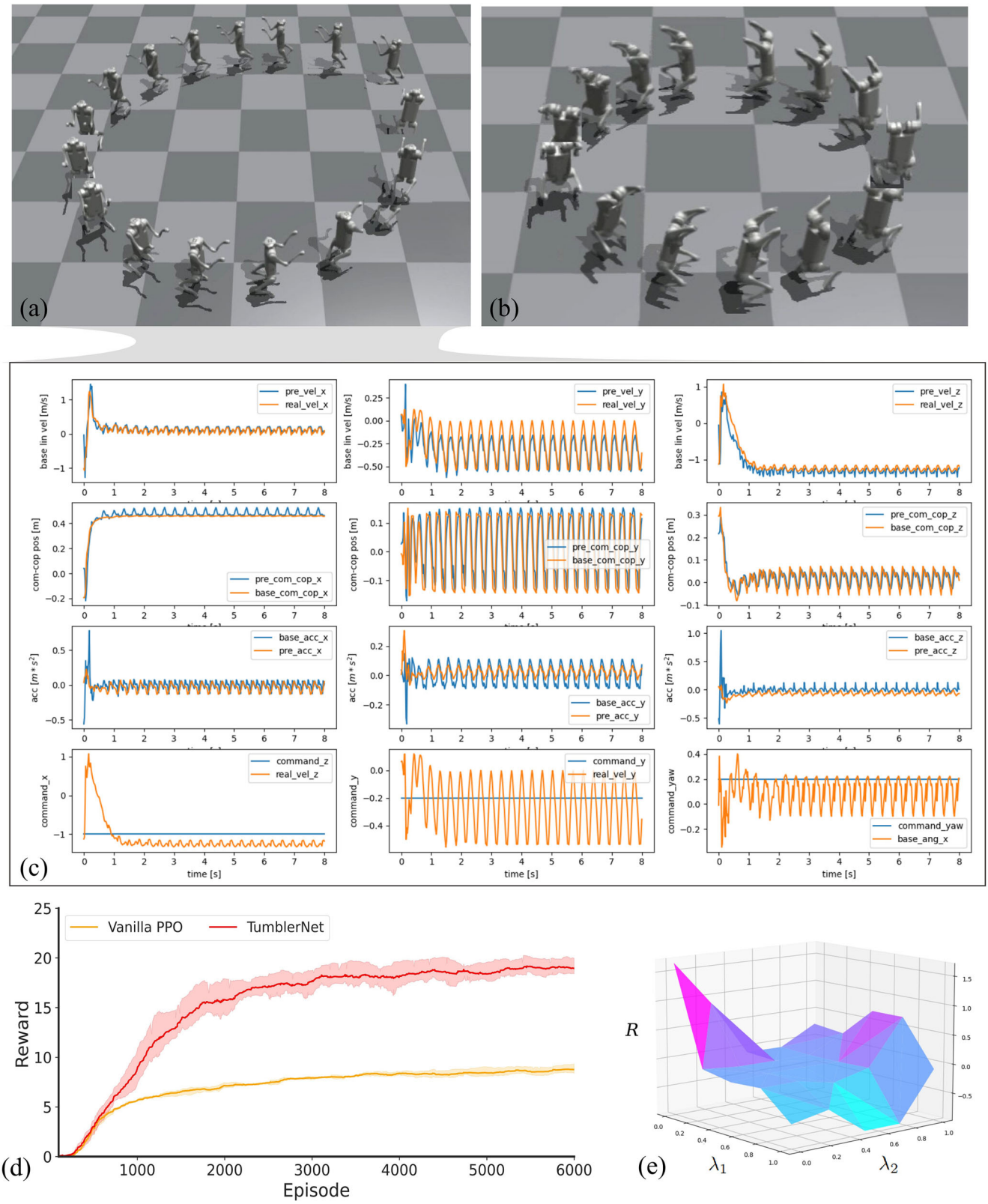

Fig. 1 | Motion and reward analysis of bipedal locomotion for a quadrupedal robot. a A quadrupedal robot performing bipedal locomotion in a circular motion using its hind legs. **b** A quadrupedal robot performing bipedal locomotion in a circular motion using its front legs. **c** The results for the bipedal locomotion using the hind legs: velocity estimates, CoM-CoP estimates, accelerations, and velocity tracking. "pre" denotes estimated values while "real" denotes true values. **d** The average rewards for the proposed TumblerNet and vanilla PPO. **e** The reward hyperplane.

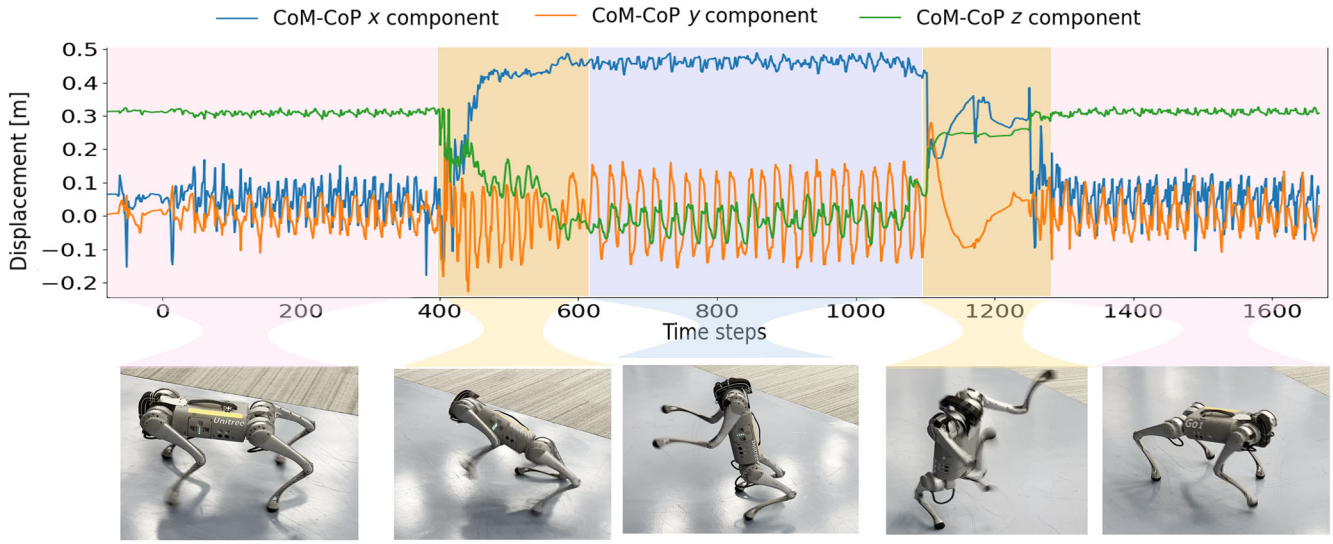

Fig. 2 | A quadrupedal robot transits from quadrupedal to bipedal locomotion and then switches back to quadrupedal locomotion. The two locomotions use the same deep reinforcement learning framework, with only a difference in the reward for lifting the legs. It can be readily observed that the CoM changes significantly during the transitions.

Gazebo simulation, in which the ground truth of the linear velocity and CoM-CoP can be obtained. Fig. 1c shows the results of the estimated and true velocities and CoM-CoP. It is seen that both information can be successfully estimated even though the ground truth data is oscillatory. The estimation can track the oscillatory curves closely, which demonstrates the effectiveness of our proposed Estimator Net. For the analysis on the morphology estimation, please refer to ref. 16.

The ability to estimate the CoM-CoP vector using the IMU and joint encoders provides an important implication for bipedal locomotion. As the CoM-CoP is an indicator of the balance of bipedal locomotion, being able to estimate the CoM-CoP implies that we could control it in a closed-loop fashion. This provides a theoretical explanation for the exceptional bipedal locomotion performances in the following sections.

We perform a comparison study to evaluate the importance of our proposed estimator. To this aim, we design and train another controller without the estimator (Vanilla PPO) to check its effects on the final obtained reward. The resultant average rewards of these two controllers (Fig. 1d) clearly show the importance of the proposed estimators.

Transition between quadrupedal and bipedal locomotion

While bipedal locomotion can be achieved using our proposed estimator and controller, it is worth investigating whether the same framework can be applied to quadrupedal locomotion.

To test this, we use the same rewards (with the exception of those penalizing the contact forces of the front legs) and train a deep reinforcement learning controller for quadrupedal locomotion. The final results showing the transition can be found in Fig. 2. It is interesting to observe that our controller can enable the robot to transition from quadrupedal to bipedal locomotion and from bipedal to quadrupedal locomotion using a similar reward design. This task is challenging as the CoM of the robot changes significantly during the transition, which is visible in the curves of Fig. 2.

Generalization to different morphologies

Our framework can be generalized to robots with different morphologies. For the estimation of the robot's morphology, we implement the Domain Randomization (DR) method to randomly generate different robots' URDF files (and load them into IsaacGym) in the training stage, and use the mass and size parameters acquired from those URDF files as ground truth. In the deployment stage, the estimator has already learned to estimate the robot's morphology based on historical observations, which enables our controller to be universal in controlling different morphologies of robots using a single controller.

We validated the universality of our framework using various robots, such as Anymal, mini-cheetah, Go2, Lite3, Spot, and Alingo. For the details, please see Movie S1. Furthermore, we have validated using another real robot, Lite3. As can be seen in Movie S2, our framework can enable Lite3 to perform bipedal locomotion as go1 using the same controller.

Robustness over challenging terrain

Locomotion over uneven terrains remains a challenge for quadrupedal locomotion[8,9], let alone bipedal locomotion for a quadrupedal robot. The reason why locomotion over challenging terrains is challenging is that uneven footholds lead to constant tilting of the CoM-CoP, ultimately jeopardizing stability. Our learning-based controller, even trained only on flat ground in simulation, can be directly transferred to real-life challenging terrains without fine-tuning. This is different from existing learning-based controllers for challenging terrains, which train the robot to walk on various terrains before deployment in real-life terrains[8,9,15,16].

We first validate the robustness of our proposed framework in indoor uneven terrains. We construct an uphill and downhill terrain connected with a soft pad (Fig. 3a). This is to test the robustness against various terrains with varying degrees of softness. The robot can perform bipedal locomotion while successfully passing these terrains. We then construct uneven terrains with multiple planks (Fig. 3b). It is noted that these uneven terrains pose a great challenge to the stability of the approach. Our quadrupedal robot successfully walks through these uneven terrains without falling, showcasing its strong robustness against irregular terrains.

To further validate the robustness against various terrains, we perform bipedal locomotion in various natural environments, including a grass field and rocky ground (Fig. 3c). In Movie 1, the robot walks with its hind legs over an uneven grass field. Despite the uneven terrain and reduced friction, the robot can maintain its bipedal locomotion without falling. For the rocky ground (Fig. 3d), our approach also enables the robot to traverse it successfully.

Finally, to further test the robustness of our approach against challenging terrains, we let our robot walk on a sandy beach (Fig. 3e). Walking on sand is extremely challenging for quadrupedal robots[25] due to the soft and deformable nature of the terrain, let alone quadrupedal robots performing bipedal locomotion. Existing research has designed a terrain model

Fig. 3 | The quadrupedal robot is performing bipedal locomotion on challenging terrains. Indoor environments: **a** a challenging terrain composed of an uphill, a soft pad, and a downhill; **b** an uneven terrain made by planks. Outdoor environments: **c** a grass field; **d** a rocky field; **e** a sandy beach. The sandy beach is extremely challenging due to the soft and deformable properties of the sand.

for the sandy beach to train quadrupedal locomotion on such a terrain[25]. Our approach successfully enables the robot to perform bipedal locomotion without falling in the sand despite the foot being stuck in the sand and leaving footprints (Movie 1). It should be noted that our approach does not require the modeling of the sand, which demonstrates the superiority of our approach. The constant tilting of the CoM-CoP due to the slippery and deformed sand is estimated in real time by our CoM-CoP estimator and compensated by the reinforcement learning controller.

Robustness against unknown external disturbances

Locomotion under external disturbances is challenging because the CoM-CoP is suddenly disturbed.

If the disturbance is overly large, robots need to extend their legs to keep the CoM within a stable region to regain balance. External disturbances are usually unpredictable and very difficult to address, which poses a great challenge to the stability and robustness of the controller. Our controller can also fight against unknown external disturbances. We perform two types of tests to showcase its robustness. In the first scenario (Fig. 4a), we use a stick to hit the robot. The robot is able to keep its upright position without falling. In the second scenario (Fig. 4b), we kick the robot constantly. In both cases, the controller can maintain its bipedal locomotion.

To systematically evaluate the disturbance rejection ability of the proposed controller, we perform a test that pushes the robot from all directions, and we record the successful rejection. The results (Fig. 4c) show that the robot can reject disturbances from all directions. We further perform a test to demonstrate the importance of the proposed CoM-CoP estimator on disturbance rejection by removing the CoM-CoP estimator from the neural network. The comparison result (Fig. 4c) shows that the robot can resist larger disturbances due to the CoM-CoP estimator. Our proposed rewards (variable height inverted pendulum model (VHIP) and cart table) are based on the CoM-CoP. As our estimator is trained in parallel with the controller, the rewards could also affect the CoM-CoP estimation. To analyze which dynamics model is critical to the CoM-CoP estimation, we also perform an ablation study by removing either the VHIP or cart table

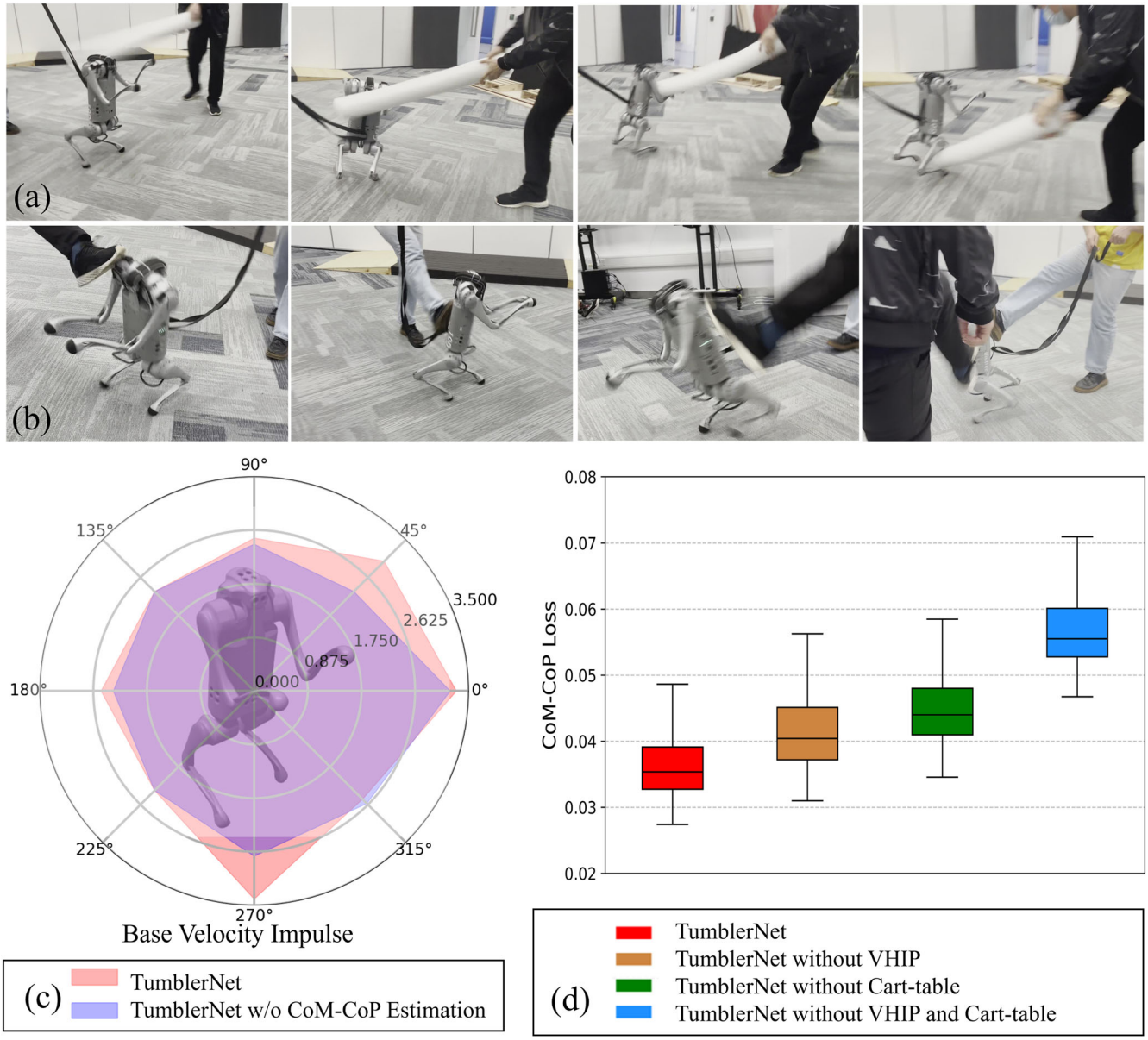

Fig. 4 | Experiment and analysis on disturbance resistance of the bipedal locomotion controller. a The robot is being struck on different body parts to evaluate its resilience to disturbances. **b** The robot is being kicked all over its body. **c** The radar plot of the robot shows its ability to reject disturbances. **d** The loss of the CoM-CoP estimation using different rewards. It is seen that with the VHIP and cart-table rewards, the estimation loss is the lowest.

model. The result (Fig. 4d) shows that both of these two models contribute to the low estimation loss, and the cart table plays a slightly more important role.

If the external disturbances are too large or the robot gets tripped by a tree or pad, the robot may still fall over. However, the robot can quickly recover to its upright position automatically. This can be observed in Fig. 5.

Observation input analysis/CoM-CoP estimation

This section will analyze the effects of different inputs on bipedal locomotion. To achieve this, we remove different terms from the input of the neural network, including the IMU information (ω_t and g_t), joint encoder information (q_t and $\dot{q}_t$), and the previous action a_{t-1}. The effect of the velocity command is obvious, which allows the user to give different velocity commands. Thus, it is not considered.

The experimental result shows that removing the IMU feedback from the input, which is ω_t and g_t, will affect the estimation of the CoM-CoP, resulting in the robot exhibiting repeated falling and recovering even when no user commands are being sent. In contrast, removing the motor encoder's feedback, which is q_t and $\dot{q}_t$, the front swing legs are crossed (Movie S3). Solely removing the previous action a_{t-1} has no impact on the locomotion. But if we remove a_{t-1} and the motor encoder's feedback simultaneously, the robot will fail to stand up. The behaviors of different controllers with various combinations of inputs are summarized in Table S1.

This phenomenon reveals that the IMU plays an important role in the stabilization capacity of our control policy. The joint encoder information determines the joint position, and knowing the joint position feedback is a crucial condition of bipedal locomotion.

Reward analysis

To analyze the reward effect in our network, we remove the smoothness reward $\boldsymbol{r}_3$ (v_x^2, $||\boldsymbol{w}_{yz}||^2$, $||\boldsymbol{a}_t - \boldsymbol{a}_{t-1}||^2$, $\sum_{j=0}^{12} ||\boldsymbol{q}_{t,j} - \boldsymbol{q}_j||$), bipedal encouragement reward $\boldsymbol{r}_4$ ($||\boldsymbol{G}_{xy}||^2$, $||\boldsymbol{F}_{FR}||$, $||\boldsymbol{F}_{FL}||$, $\sum_{f=0}^{4}(t_{air,f} - 0.5)$, $(h - h^*)^2$), and stabilization reward $\boldsymbol{r}_5$ ($||\boldsymbol{\theta}||$, $||\dot{\boldsymbol{\theta}}||$, $||\boldsymbol{c}_{xy}||$) respectively, and perform ablation experiments. In the reward ablation experiments, it is observed that without the stabilization terms $\boldsymbol{r}_5$, the robot is able to stand up but not able to follow the desired command, which also demonstrates the effectiveness of

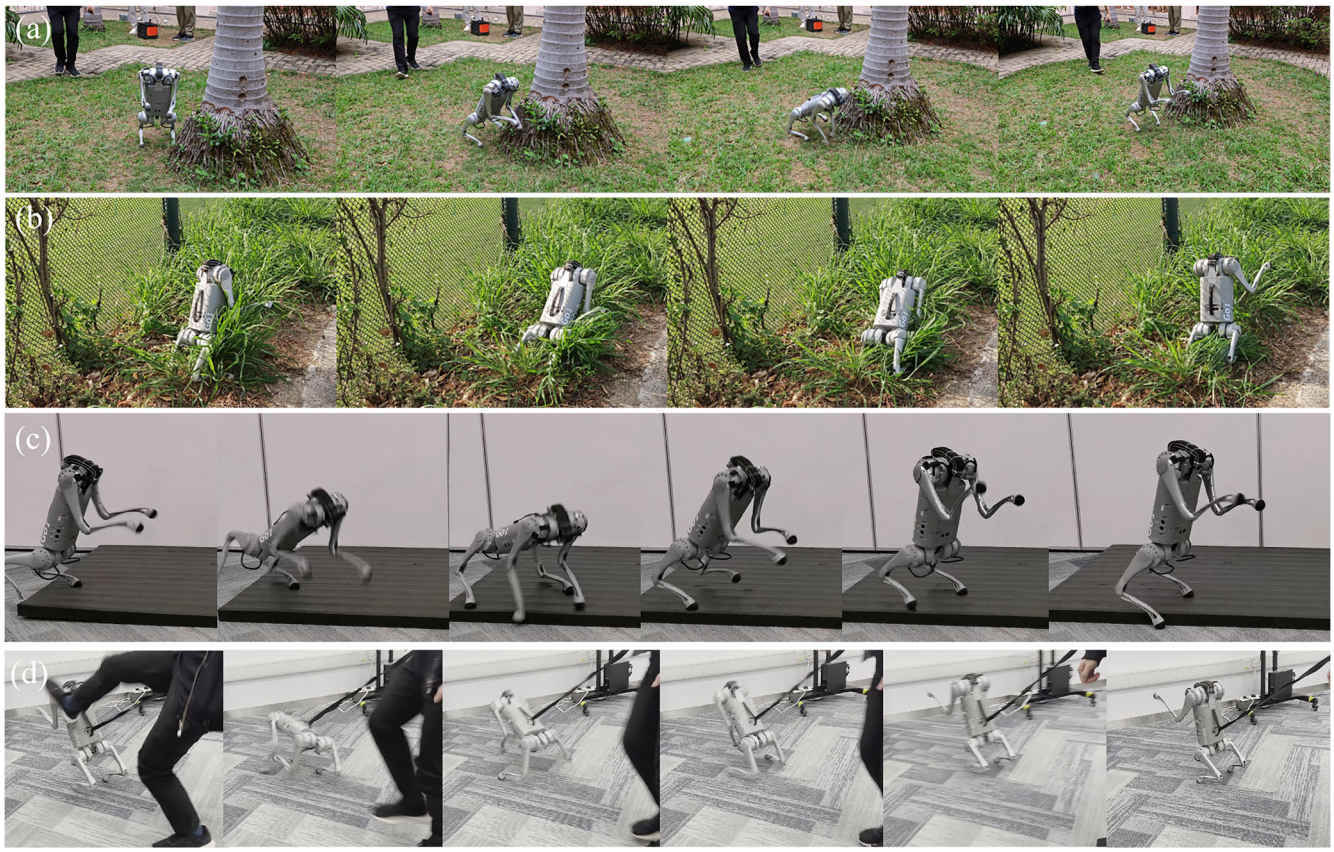

Fig. 5 | Automatic fall recovery in different scenarios: a stumbled by a tree; **b** stumbled by dense grass; **c** stumbled by a pad; **d** kicked by a human.

our proposed rewards. If we remove the smoothness reward r_3, the robot will stand with crossed front legs, and the body will have drastic jittering movements. If the bipedal encouragement reward r_4 is removed, the robot is not able to switch to the bipedal position.

We use the concept of hyperplane[28] to validate the effect of reward. The hyperplane is defined by $\theta^{\text{avg}} = \lambda_1\theta^{\text{wosmooth}} + \lambda_2\theta^{\text{wostable}} + (1-\lambda_1-\lambda_2)\theta^{\text{tumbler}}$ where θ^{wosmooth} stands for the parameters in both the actor and the estimator net for the policy training without smoothness terms, and θ^{wostable} represents the parameters for the policy training without stabilization terms. We average them with the parameters of TumblerNet (denoted as θ^{tumbler}), then get the average policy and build the hyperplane to analyze the policies. The metric is achieved by replaying the policy in the IsaacGym and calculating the mean reward for one second based on the normal TumblerNet reward setting. The final reward hyperplane is shown in Fig. 1e.

Comparison with existing algorithms

To evaluate the robustness and generalization capability of our locomotion algorithm, we conduct a comparative experiment measuring the success rate of walking across four different terrains in IsaacGym: a plane, a slope with 20 degrees, and two uneven trimesh terrains (one with height within [−0.04, 0.04] m and the other one with height within [−0.08, 0.08] m). Specifically, we select three representative algorithms: Extreme Parkour[23], Mujoco Playground[24], and open-source Unitree RL Gym[29]. Each algorithm is trained under a velocity command tracking task and deployed in the IsaacGym simulation environment. Each time, 100 agents are sent with the same walking forward velocity command of 0.8 m/s on the same uneven terrains for up to 1000 time steps. If contact occurs between the thigh, calf (excluding the toe), trunk of the robot, and the ground, the robot will be reset and regarded as a task-unsuccessful agent. At intervals of every 200 time steps, we record the proportion of successful agents, defined as those that remained upright and continued progressing. This success rate serves as a measure of the algorithm's reliability and adaptability as the walking duration increases. We repeat the trials ten times for each algorithm to account for stochasticity, and report the mean success rate along with its variation error bars.

The results are shown in Fig. 6a. All algorithms can perform well in the plane. However, the performance of the baseline algorithms[23,24,29] significantly drops in other uneven terrains. In contrast, our approach maintains a higher success rate over all terrains, demonstrating the effectiveness of our framework in achieving stable and robust locomotion over challenging terrains.

We repeated the comparison on 4 and 8 cm terrains to evaluate velocity tracking across methods. As shown in Fig. 6b, our method achieves the lowest tracking error and minimal oscillation. In contrast, the other methods exhibit sudden drops or rapid increases in forward velocity due to falls or foot trapping on the terrain.

The analysis in Fig. 1e and **Reward analysis** session shows that reward modules r_1 and r_2 are essential for foundational quadrupedal motion, while r_4 enables bipedal transitions. Tasks cannot be performed without these three modules. An ablation study on reward modules r_3 and r_5 examines their contributions to success rates. Comparing methods with and without these modules, repeated across three seeds for stochasticity. The result reported in Fig. 7 reveals that both r_3 and r_5 significantly enhance performance on uneven terrain. Specifically, without r_5, the robot leans forward excessively, increasing the risk of falling, and without r_3, unnecessary tremors destabilize it. Removing both severely hinders forward movement.

Discussion

The presented learning-based controller significantly advances bipedal locomotion for quadrupedal robots. The feature that monitors and maintains the CoM and center of pressure brings a new perspective on the design of learning-based controllers for quadrupedal robots over various challenging terrains. The validation on various challenging terrains fully demonstrates the robustness of the proposed framework. Furthermore, the

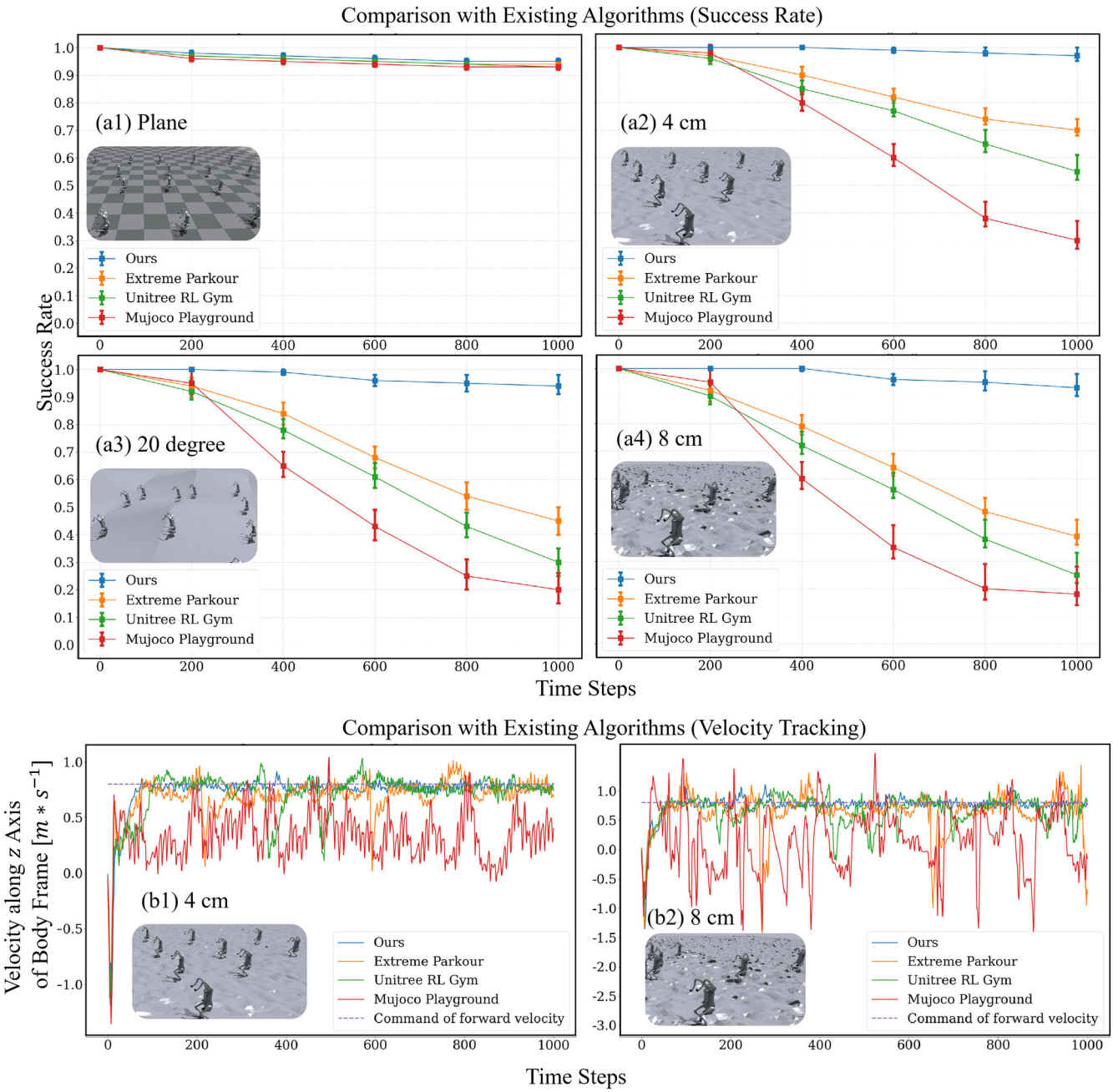

Fig. 6 | The comparison with existing algorithms, measuring the success rate of walking across and the velocity tracking effect. a The comparison with existing algorithms, measuring the success rate of walking across **a1** a flat plane. **a2** a 4 cm uneven trimesh terrain (with a maximum height of 0.04 m and a minimum height of −0.04 m). **a3** a 20-degree slope terrain (with a 20-degree slope angle). **a4** An 8 cm uneven trimesh terrain (with a maximum height of 0.08 m and a minimum height of −0.08 m). **b** The comparison with existing algorithms, measuring the velocity along the forward direction (body's Z axis) on **b1** a 4 cm uneven trimesh terrain. **b2** An 8 cm uneven trimesh terrain.

proposed framework can reject unknown external disturbances, such as pushing and kicking. Unknown external disturbances are also challenging for learning-based controllers. Our framework can reject these external disturbances without prior information, which further exhibits the robustness of the proposed framework.

Quadrupedal locomotion on a soft and deformable terrain, such as a sandy beach, is of great challenge, let alone bipedal locomotion on such a terrain. The deformable property of the terrain significantly jeopardizes the stability of the locomotion as it constantly changes the support location of the feet, thus tilting the CoM and CoP of the robot. Terrain modeling is a potential solution to this challenge. However, different terrains have different properties, and it requires a large amount of effort to generalize to various terrains. In contrast, we approach the problem from a different perspective. Instead of modeling the terrain, we monitor the tilting of the CoM-CoP vector and change the controller input to maintain the upright position. This waives the trouble of modeling various terrains. The successful demonstration of bipedal locomotion on a sandy beach adequately shows the effectiveness of the proposed framework.

Fall recovery is essential for robots as falling is inevitable due to challenging terrains or external disturbances. State-of-the-art fall recovery requires training an additional recovery policy controller, as the behavior is distinct from normal locomotion[11]. This is more advantageous than conventional methods that rely on well-tuned joint trajectories. However, it still needs to design an additional controller and activate the controller. Our

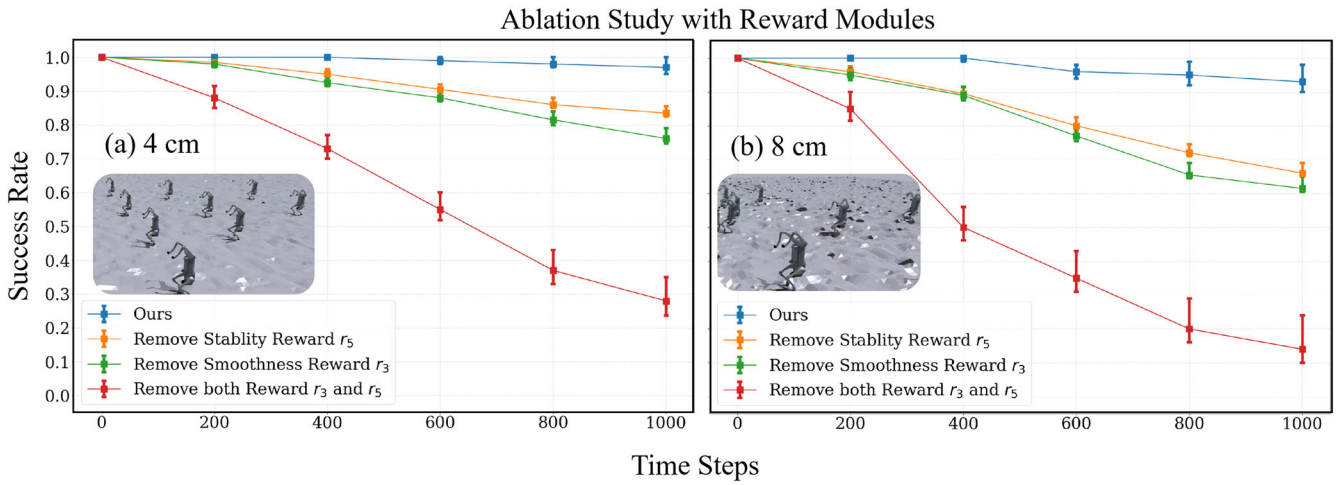

Fig. 7 | The ablation study with reward modules, measuring the success rate of walking across. a a 4 cm uneven trimesh terrain (with a maximum height of 0.04 m and a minimum height of −0.04 m). **b** An 8 cm uneven trimesh terrain (with a maximum height of 0.08 m and a minimum height of −0.08 m).

framework can automatically detect the falling and recover to its upright position without designing additional controllers and falling detection strategies. This is all due to the closed-loop control of the CoM-CoP vector by the proposed framework.

The proposed framework can also be applied to bipedal and humanoid robots, which is expected to significantly enhance the robustness of the locomotion against various challenging terrains and external disturbances. Moreover, the automatic fall recovery property of the framework also waives the trouble of designing additional recovery controllers. Future work would explore the limits of bipedal locomotion for quadrupedal robots. For instance, we would study whether the quadrupedal robots can achieve parkour when performing bipedal locomotion.

Methods

Overview of the framework

This section will detail our proposed method. An overview of our proposed framework is given in Fig. 8.

Problem Formulation

Our problem is formulated as a partially observable Markov decision process (POMDP), denoted as a 7-tuple $\mathcal{M} = \{\mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \Omega, \gamma\}$, where $\mathcal{S}$, $\mathcal{O}$ and $\mathcal{A}$ are the set of states, observations, and actions, respectively. For each state $\boldsymbol{s}_t \in \mathcal{S}$, the learning agent interacts with the environment with an action $\boldsymbol{a}_t \in \mathcal{A}$ and receives a reward $\mathcal{R}(\boldsymbol{s}_t, \boldsymbol{a}_t)$, leading to the transition of the environment to the next state $\boldsymbol{s}_{t+1}$ with the probability $\mathcal{T}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t)$. Meanwhile, the observation $\boldsymbol{o}_{t+1} \in \mathcal{O}$ depends on the new state $\boldsymbol{s}_{t+1}$ and the action $\boldsymbol{a}_t$ with a conditional probability $\Omega(\boldsymbol{o}_{t+1}|\boldsymbol{s}_{t+1}, \boldsymbol{a}_t)$. The objective is to determine the optimal policy π^* that maximizes the accumulated rewards of this POMDP $\mathcal{M}$, considering a discount ratio γ, i.e.:

$$J_{\mathcal{M}}(\boldsymbol{\pi}) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{T-1}(\gamma^t \mathcal{R}(s_t, a_t))\right]. \tag{1}$$

Observation and State Space. The robot observation $\boldsymbol{o}_t$ in our control scheme can be collected from the proprioceptive sensors, including body angular velocity $\boldsymbol{\omega}_t \in \mathbb{R}^3$, projected gravity $\boldsymbol{g}_t \in \mathbb{R}^3$, body linear velocity command $\boldsymbol{v}_t^* \in \mathbb{R}^3$, joint angles $\boldsymbol{q}_t \in \mathbb{R}^{12}$, joint angular velocities $\dot{\boldsymbol{q}}_t \in \mathbb{R}^{12}$, and the action of the last step $\boldsymbol{a}_{t-1} \in \mathbb{R}^{12}$, which can be defined as:

$$\boldsymbol{o}_t = \left(\boldsymbol{\omega}_t, \boldsymbol{g}_t, \boldsymbol{v}_t^*, \boldsymbol{q}_t, \dot{\boldsymbol{q}}_t, \boldsymbol{a}_{t-1}\right). \tag{2}$$

In addition to the proprioception information of the robot, the state space also includes the privileged information $\boldsymbol{x}_t$ defined as:

$$\boldsymbol{x}_t = (\boldsymbol{v}_t, \boldsymbol{c}_t, \boldsymbol{h}_t^{feet}, \boldsymbol{b}_t^{feet}, \boldsymbol{h}_t, \boldsymbol{f}_t, \boldsymbol{\mu}, \boldsymbol{K}_{PD}) \tag{3}$$

where $\boldsymbol{v}_t \in \mathbb{R}^3$, $\boldsymbol{c}_t \in \mathbb{R}^3$, $\boldsymbol{h}_t^{feet} \in \mathbb{R}^4$, $\boldsymbol{b}_t^{feet} \in \mathbb{R}^4$, $\boldsymbol{h}_t \in \mathbb{R}^{187}$, $\boldsymbol{f}_t \in \mathbb{R}^2$, and $\boldsymbol{\mu} \in \mathbb{R}^1$ denote the body velocity, the subtraction vector between CoM and CoP, the feet height, the contact boolean of all feet, the egocentric height map of the robot body, the disturbance force projection in x-o-y plane, and the body friction coefficient, respectively. $\boldsymbol{K}_{PD} \in \mathbb{R}^{14}$ includes the proportional gain $\boldsymbol{K}_p \in \mathbb{R}^1$, the derivative gain $\boldsymbol{K}_d \in \mathbb{R}^1$, and the motor strength of each joint $\boldsymbol{\alpha} \in \mathbb{R}^{12}$.

The entire state vector $\boldsymbol{s}_t$ can be organized in the combination of the observable part $\boldsymbol{o}_t$, and the unobservable part $\boldsymbol{x}_t$:

$$\boldsymbol{s}_t = (\boldsymbol{o}_t, \boldsymbol{x}_t) \tag{4}$$

Action space. The actions $\boldsymbol{a}_t \in \mathbb{R}^{12}$ carried out by our neural network-based controller are the displacement computed by the desired positions $\boldsymbol{q}_d$ and the initial joint position $\boldsymbol{q}_o$ of all joints, as follows:

$$\boldsymbol{a}_t = \boldsymbol{q}_d - \boldsymbol{q}_o \tag{5}$$

where $\boldsymbol{q}_d$ and $\boldsymbol{q}_o$ stand for the desired joint position and the initial joint position of the robot, respectively. The final torque command for the motors is computed by:

$$\begin{aligned} \boldsymbol{\tau}_t &= \boldsymbol{\alpha} \cdot \boldsymbol{K}_p(\boldsymbol{q}_d - \boldsymbol{q}_t) + \boldsymbol{\alpha} \cdot \boldsymbol{K}_d(\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}}_t) \\ &= \boldsymbol{\alpha} \cdot \boldsymbol{K}_p(\boldsymbol{q}_o + \boldsymbol{a}_t - \boldsymbol{q}_t) - \boldsymbol{\alpha} \cdot \boldsymbol{K}_d \dot{\boldsymbol{q}}_t \end{aligned} \tag{6}$$

where the target joint velocity $\dot{\boldsymbol{q}}_d$ is set as $\boldsymbol{0}$, to determine the torque value required to be applied. In the learning simulation and the real-world environment, we can specify $\boldsymbol{K}_p$ and $\boldsymbol{K}_d$.

Reward function. The detailed expression of each reward term is shown in Table S2. The stabilization reward weights tuning ranges are initially calculated based on their proportions to the major tracking terms, and the final values are determined based on experimental trials. The total reward at given state $\boldsymbol{s}_t$ and action $\boldsymbol{a}_t$, $\mathcal{R}_t(\boldsymbol{s}_t, \boldsymbol{a}_t)$, can be obtained as the summation of the dot product of the different classes of

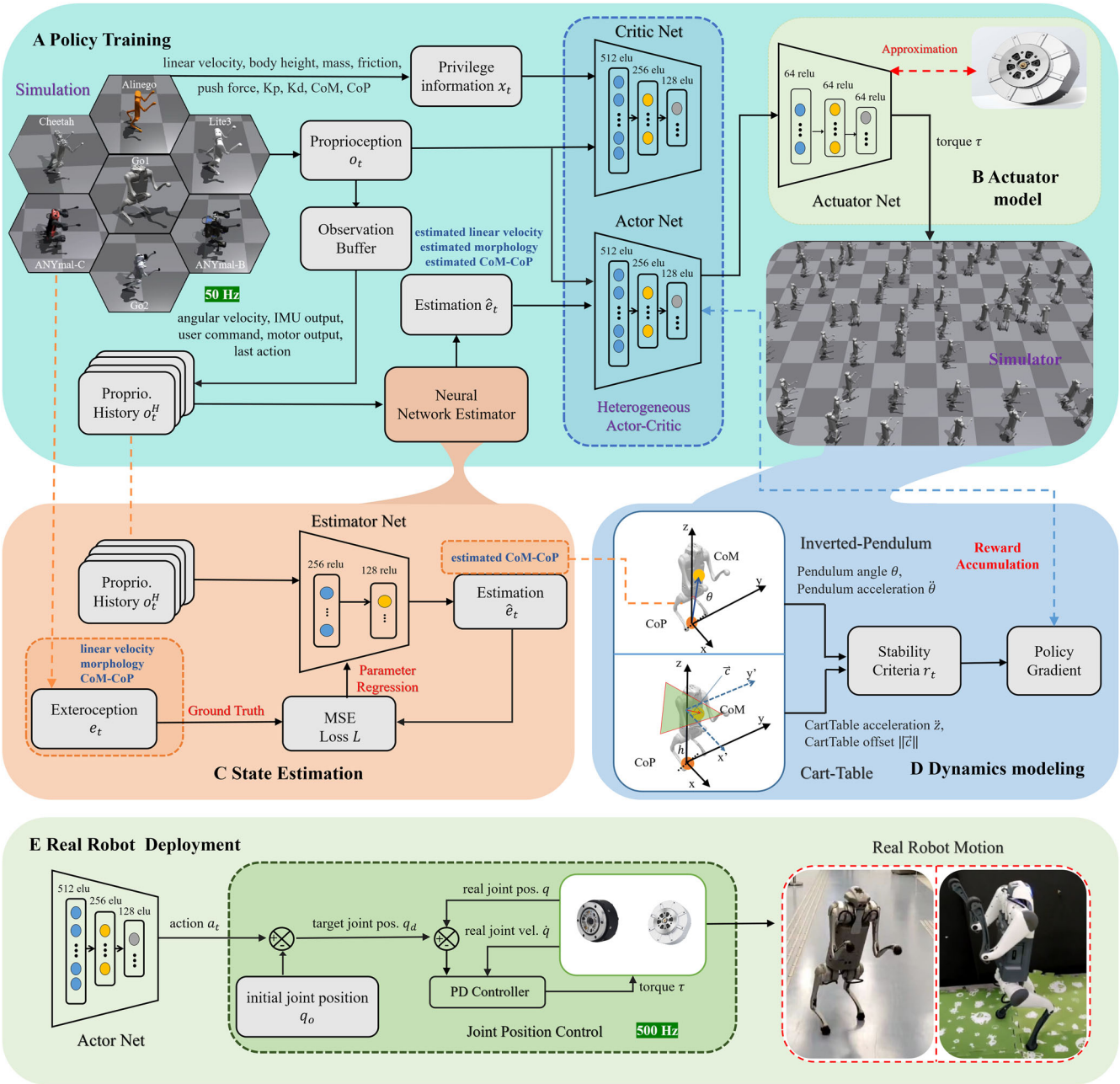

Fig. 8 | Training and deployment framework of the TumblerNet. Framework of the TumblerNet, Session a (white): The workflow of the TumblerNet, from left to right, is the forwarding direction, while from the right to the left is the backward and policy updating direction. Session b (orange): The design of the Neural Network-based state estimator. Session c (blue): The design of the model-based stability reward of the reinforcement learning. Session d (green): The transfer of the trained policy to the real robot.

reward we designed $\boldsymbol{r}_i$ and the corresponding weight we tuned $\boldsymbol{w}_i$ as follows:

$$\mathcal{R}_t(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_i (\boldsymbol{w}_i{}^T \boldsymbol{r}_i) \tag{7}$$

Dynamics modeling

We implement two simplified dynamics models to introduce the physical knowledge to our deep neural network: the variable height inverted pendulum and the cart-table.

Variable height inverted pendulum model. The VHIP, shown in Fig. S1a, is a commonly used simplified model for legged robots. It represents the robot's leg as an inverted pendulum, where the CoM is localized above the foot. The height of the CoM is variable and can change dynamically as the robot moves. This model assumes that the leg is massless and the foot is fixed on the ground, neglecting the dynamics of the leg itself. By considering the dynamics of the CoM height and the forces acting on the foot, this model allows for the analysis and control of the robot's stability and locomotion. θ is defined as the angle between the vector CoM-CoP and the vertical z-axis.

CoM denotes the center of mass of the robot, where its position can be calculated as follows:

$$\boldsymbol{p}_{CoM} = \sum_{l_i \in \mathbb{L}} \frac{m_i \boldsymbol{p}_{l_i}}{\sum_i m_i} \tag{8}$$

CoP position falls into the support pattern of the contact position $\boldsymbol{C}_i \in \mathbb{R}^3$, here mainly a line segment or a single point:

$$\boldsymbol{p}_{CoP} = \sum_{c_i \in \mathbb{C}} \frac{\boldsymbol{f}_i \cdot \boldsymbol{n}}{\sum_i (\boldsymbol{f}_i \cdot \boldsymbol{n})} \boldsymbol{C}_i \tag{9}$$

where $\boldsymbol{p}_i \boldsymbol{p}_{CoM}$ and $\boldsymbol{p}_{CoP}$ stand for the i-th Cartesian position of the links, the CoM and the CoP of the robot in the body frame, respectively.

Cart-Table. Similar to the VHIP, the cart table model (shown in Fig. S1b) is another simplified model frequently used for legged robots. This model represents the robot's leg as a massless rod attached to a cart. The cart represents the robot's body, which can move freely along a horizontal plane. In this model, h and c_{xy} denote the vertical height from the horizontal plane determined by CoP to the horizontal plane determined by CoM, and the horizontal offset vector between the CoP and CoM, respectively. (CoM and CoP here have the same definition as the VHIP model.) This model assumes that the leg can swing freely without any constraints, neglecting the leg's interaction with the environment. By considering the dynamics of the cart and the forces acting on the leg, this model enables the analysis and control of the robot's balance and gait planning.

Reward Design

The reward design is based on five criteria: tracking commands tasks reward $\boldsymbol{r}_1$, avoiding collision (including self-collision and the collision with the environment) $\boldsymbol{r}_2$, guaranteeing smoothness of the motion $\boldsymbol{r}_3$, encouraging the robot to perform bipedal locomotion $\boldsymbol{r}_4$ and the stability based reward $\boldsymbol{r}_5$ as described in Eqs. (10), (11), and (12). The reward scale setting is shown in Table S2.

VHIP angle. A direct stability criterion that can be implemented in the reward function is the pendulum angle, which can be obtained by

$$\theta = \arccos \frac{||\boldsymbol{p}_{CoM,z}||}{||\boldsymbol{p}_{CoM} - \boldsymbol{p}_{CoP}||} \tag{10}$$

VHIP angular acceleration. When the quadrupedal robot tends to fall, the acceleration of θ abruptly increases. Therefore, in addition to circumventing large θ angle, we involve the penalization of $\ddot{\theta}$ as well. By ignoring the base linear acceleration, which is much smaller than the acceleration of gravity, it is trivial to have the equation of motion for VHIP

$$\ddot{\theta} = -\frac{g}{||\boldsymbol{p}_{CoM} - \boldsymbol{p}_{CoP}||} \sin\theta \tag{11}$$

where g is the gravitational acceleration.

Cart-Table handle length. If the CoM projection is far away from the CoP in the horizontal plane, the momentum that causes the falling tendency would relatively increase. Therefore, we want to reduce this handle length by introducing a reward:

$$||\boldsymbol{c}_{xy}|| = \sqrt{||\boldsymbol{p}_{CoM} - \boldsymbol{p}_{CoP}||^2 - h^2} \tag{12}$$

where h is the height of the robot, which is defined as the vertical distance (along the z-axis) between CoM and CoP.

Deep reinforcement learning controller design

Heterogeneous Actor-Critic. To train robots for complex locomotion capabilities, our framework makes use of the strengths of both actor and critic networks to improve the learning process. The actor network generates control policies, while the critic network evaluates the quality of these policies. By incorporating heterogeneous networks, this method can help the agent handle the high-dimensional state and action spaces of legged robots effectively, enabling more efficient exploration and learning.

Actor Net. The actor net $\boldsymbol{\pi}(\boldsymbol{a}_t|\hat{\boldsymbol{v}}_t, \boldsymbol{o}_t, \hat{\boldsymbol{c}}_t)$, is utilized to determine the action $\boldsymbol{a}_t \in \mathbb{R}^{12}$ by taking the proprioception $\boldsymbol{o}_t \in \mathbb{R}^{45}$ and the output of an estimator net $\hat{\boldsymbol{e}}_t \in \mathbb{R}^6$ as input (Fig. 8), which is utilized for predicting the body linear velocity $\hat{\boldsymbol{v}}_t \in \mathbb{R}^3$ and the connection vector (which is also the pendulum's body) between CoM and CoP, $\hat{\boldsymbol{c}}_t \in \mathbb{R}^3$, which is defined as:

$$\hat{\boldsymbol{c}}_t = \hat{\boldsymbol{o}}_t^{com} - \hat{\boldsymbol{o}}_t^{cop} \tag{13}$$

Critic Net. The critic net $\boldsymbol{V}(\boldsymbol{s}_t)$ is responsible for providing feedback to the actor net by estimating the expected value function. Unlike the actor net, the critic net does not receive estimated information from the estimator net. Instead, it directly receives ground truth values from the environment. In addition, it can also receive privileged information about the robot and exteroceptive information about the disturbances.

Learn to estimate the exteroception information. One of the key factors that contributes to the success of our proposed neural network framework is our proposed estimator net. On the one hand, it is necessary to estimate some information available in the simulation, but difficult to obtain in real practice. For instance, the CoM and CoP are very important for the stability of legged robots. However, their true values are difficult to obtain. In this paper, we propose to estimate the connection vector between CoM and CoP using a deep neural network. The estimated values will be used by the actor net. On the other hand, without this estimator net, the performance will degrade, which we will show in Section Experiments. In addition to estimating the CoM-CoP, we will also estimate the linear velocity of the robot, as this information is also important but not directly available. Our proposed estimator net, the structure of which is shown in Fig. 5c, is as follows:

Estimator Net. The estimator net $\boldsymbol{E}_{\boldsymbol{\theta}}(\boldsymbol{o}_t^H)$ is implemented to estimate the body velocity and CoM-CoP simultaneously, which takes the history of the observation $\boldsymbol{o}_t^H$ as input and outputs the estimated body velocity $\hat{\boldsymbol{v}}_t$ and the estimated gap between CoM and CoP $\hat{\boldsymbol{c}}_t$. Lastly, the estimation of morphology observation $\hat{\boldsymbol{o}}_t^{mor} \in \mathbb{R}^{9|16}$. The robot's morphology contains 4 dimensions of mass (the mass of the robot's body, hip, thigh, and calf), and 5 dimensions of size (the length and width of the robot's body, length of the hip, thigh, and calf).

$$\boldsymbol{e}_t^{\boldsymbol{\theta}} = \boldsymbol{E}_{\boldsymbol{\theta}}(\boldsymbol{o}_t^H) = (\hat{\boldsymbol{v}}, \hat{\boldsymbol{c}}, \hat{\boldsymbol{o}}_t^{mor}) \tag{14}$$

Backward loss and Concurrent update. The estimator net is optimized by regression loss $loss_{reg}$ to reduce the mean squared error, as follows:

$$loss_{reg} = MSE(\hat{\boldsymbol{v}}_t, \boldsymbol{v}_t) + MSE(\hat{\boldsymbol{c}}_t, \boldsymbol{c}_t) + MSE(\hat{\boldsymbol{o}}_t^{mor}, \boldsymbol{o}^{mor}) \tag{15}$$

The training of the estimator net is not independent of the training of the actor net. During training, the policy gradient $loss_{policy}$ can be back-propagated not only to the actor net but also to the estimator net, as indicated by the red dashed line. Thus, the estimator net is not only updated by a regression loss $loss_{reg}$ but also updated by the PPO[27], aiming to improve locomotion performance as follows:

$$Loss = \beta \cdot loss_{reg} + (1 - \beta) \cdot loss_{policy} \tag{16}$$

where $\beta = 0.5$ means the regression loss weight is the same as the weight of policy loss.

Sim2real transfer

Domain randomization. DR is a technique used in training legged robots where the robot's environment is deliberately varied during the training process. This approach aims to improve the robot's generalization and adaptability by exposing it to a wide range of simulated environments with varying physical properties. By randomizing factors, such as friction, gravity, terrain, and lighting conditions, the robot learns to adapt its control policies to handle different scenarios. This technique helps the

robot overcome the challenge of transferring learned behaviors from simulation to the real world, as it encourages the robot to learn robust and flexible policies that can handle unforeseen environmental variations. It allows for better preparation and testing of legged robots in diverse and unpredictable real-world environments, increasing their ability to navigate and perform tasks in various conditions.

Actuator adaptation. The trained controllers in simulation often cannot be directly transferred to real robots due to the sim-to-real discrepancies induced by the distinction in the actuator's nonlinear properties.

Actuator net. To solve this problem, we train an actuator network $\boldsymbol{A}_\theta(\cdot)$ to simulate the dynamics of the motor[11]. The pre-trained actuator network is used to generate the torque in simulation $\boldsymbol{\tau}^{sim}$ with the input being the joint state ($\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$) of the current step and the previous two steps. The predicted torque at a given time step t can be obtained as:

$$\boldsymbol{\tau}_t = \boldsymbol{A}_\theta\Big(\left[\Delta \boldsymbol{q} + \Delta \boldsymbol{M}\right]_{t-2\delta t:t},\ \dot{\boldsymbol{q}}_{t-2\delta t:t}\Big) \tag{17}$$

where δt is the control interval in simulation (5 m in our case), $\Delta \boldsymbol{q}_t$ is the position error, namely the difference between the desired position $\boldsymbol{q}_t^*$ and the current position $\boldsymbol{q}_t$, and $\Delta \boldsymbol{M}$ is the motor offset acting as noise to the measurement of $\Delta \boldsymbol{q}_t$. Finally, considering the latency, $\boldsymbol{\tau}^{sim}$ is taken as the torque at the previous ΔT second (measured from the real actuator and set as 12 ms), i.e., $\boldsymbol{\tau}^{sim} = \boldsymbol{\tau}_{t-\Delta T}$.

To train $\boldsymbol{A}_\theta(\cdot)$, we collect the data from all 12 actuators simultaneously, and it is carried out on the types of actuators mounted on Unitree Go1. The robots are commanded to track a series of random trajectories with different body heights, which aims to cover the overall actuator operation range.

Implementation details

The policy network and critic network are both MLPs with the layer sizes being [512, 256, 128, 12], and the estimator network is a 3-layer MLP with the feature [256, 128, 6]. Meanwhile, the actuator net structure is [64, 64, 64, 1]. We apply the PPO algorithm[27] to update $\boldsymbol{\pi}$ and $\boldsymbol{E}_\theta$.

The critical parameters of the robots, such as motor frictions, motor strength $\boldsymbol{\alpha}$, and motor offset $\Delta \boldsymbol{M}$, are randomized at the initialization stage and shown in Table S3.

The real hardware experiments are carried out on the Unitree Go1-NX quadruped with 12 actuated degrees of freedom and ~12 kg weights. In our work, the nominal $\boldsymbol{K}_p$ is set as 30.0, 30.0, 30.0 for hip, thigh, and calf joint, and $\boldsymbol{K}_d$ as 0.8 for all joints on each leg. A multi-threading deployment strategy is shown in Fig. 8, showcasing the frequency of the state reading from the real sensor (IMU, joint encoders, foot force sensor, etc.), the policy, and the adapter network forwards. The frequencies are determined according to the computational time of each thread, especially the forward time of the two deployed networks.

Data availability

All data supporting the findings of this study are available in this manuscript and its Supplementary Information and by request from the Authors.

Code availability

The code to reproduce the learning experiments can be found in: https://github.com/arclab-hku/bipedal_locomotion_for_quadrupedal_robots.

Received: 26 April 2025; Accepted: 10 July 2025;
Published online: 01 August 2025

References

1. Farshidian, F., Neunert, M., Winkler, A. W., Rey, G. & Buchli, J. An efficient optimal planning and control framework for quadrupedal locomotion. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 93–100 (ICRA, 2017).
2. Bledt, G. et al. Mit cheetah 3: Design and control of a robust, dynamic quadruped robot. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2245–2252 (IROS, 2018).
3. Neunert, M. et al. Whole-body nonlinear model predictive control through contacts for quadrupeds. *IEEE Robot. Autom. Lett.* **3**, 1458–1465 (2018).
4. Meduri, A. et al. Biconmp: A nonlinear model predictive control framework for whole body motion planning. *IEEE Trans. Robot.* **39**, 905–922 (2023).
5. Grandia, R., Jenelten, F., Yang, S., Farshidian, F. & Hutter, M. Perceptive locomotion through nonlinear model-predictive control. *IEEE Trans. Robot.* **39**, 3402–3421 (2023).
6. Abdalla, A., Focchi, M., Orsolino, R. & Semini, C. An efficient paradigm for feasibility guarantees in legged locomotion. *IEEE Trans. Robot.* **39**, 3499–3515 (2023).
7. Ding, Y., Pandala, A., Li, C., Shin, Y.-H. & Park, H.-W. Representation-free model predictive control for dynamic motions in quadrupeds. *IEEE Trans. Robot.* **37**, 1154–1171 (2021).
8. Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V. & Hutter, M. Learning quadrupedal locomotion over challenging terrain. *Sci. Robot.* **5**, eabc5986 (2020).
9. Miki, T. et al. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Sci. Robot.* **7**, eabk2822 (2022).
10. Villarreal, O., Barasuol, V., Wensing, P. M., Caldwell, D. G. & Semini, C. Mpc-based controller with terrain insight for dynamic legged locomotion. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2436–2442 (IEEE, 2020).
11. Hwangbo, J. et al. Learning agile and dynamic motor skills for legged robots. *Sci. Robot.* **4**, eaau5872 (2019).
12. Hoeller, D., Rudin, N., Sako, D. & Hutter, M. Anymal parkour: learning agile navigation for quadrupedal robots. *Sci. Robot.* **9**, eadi7566 (2024).
13. Kumar, A., Fu, Z., Pathak, D. & Malik, J. RMA: rapid motor adaptation for legged robots. In *Proc. Robot.: Sci. Syst. (RSS),* XVII (RSSF, 2021).
14. Nahrendra, I. M. A., Yu, B. & Myung, H. Dreamwaq: Learning robust quadrupedal locomotion with implicit terrain imagination via deep reinforcement learning. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 5078–5084 (IEEE, 2023).
15. Margolis, G. B. & Agrawal, P. Walk these ways: tuning robot control for generalization with multiplicity of behavior. In *Proc. Conference on Robot Learning*, 22–31 (PMLR, 2023).
16. Luo, Z. et al. Moral: learning morphologically adaptive locomotion controller for quadrupedal robots on challenging terrains. *IEEE Robot. Autom. Lett.* **9**, 4019–4026 (2024).
17. Shafiee, M., Bellegarda, G. & Ijspeert, A. Viability leads to the emergence of gait transitions in learning agile quadrupedal locomotion on challenging terrains. *Nat. Commun.* **15**, 3073 (2024).
18. Chen, D., Zhou, B., Koltun, V. & Krähenbühl, P. Learning by cheating. In Kaelbling, L. P., Kragic, D. & Sugiura, K. (eds.) *Proceedings of the Conference on Robot Learning*, vol. 100. In *Proc. Machine Learning Research*, 66–75 (PMLR, 2020).
19. Luo, Z., Xiao, E. & Lu, P. Ft-net: Learning failure recovery and fault-tolerant locomotion for quadruped robots. *IEEE Robot. Autom. Lett.* **8**, 8414–8421 (2023).
20. Vollenweider, E. et al. Advanced skills through multiple adversarial motion priors in reinforcement learning. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 5120–5126 (IEEE, 2023).
21. Fuchioka, Y., Xie, Z. & Van de Panne, M. Opt-mimic: Imitation of optimized trajectories for dynamic quadruped behaviors. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 5092–5098 (IEEE, 2023).
22. Li, Y., Li, J., Fu, W. & Wu, Y. Learning agile bipedal motions on a quadrupedal robot. In *Proc. IEEE International Conference on Robotics and Automation (ICRA),* pp. 9735–9742 (IEEE, 2024).

23. Cheng, X., Shi, K., Agarwal, A. & Pathak, D. Extreme parkour with legged robots. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 11443–11450 (IEEE, 2024).
24. Zakka, K. et al. Mujoco playground. *arXiv preprint arXiv:2502.08844* (2025).
25. Choi, S. et al. Learning quadrupedal locomotion on deformable terrain. *Sci. Robot.* **8**, eade2256 (2023).
26. Makoviychuk, V. et al. Isaac gym: high performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470* (2021).
27. Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).
28. Jin, Y., Liu, X., Shao, Y., Wang, H. & Yang, W. High-speed quadrupedal locomotion by imitation-relaxation reinforcement learning. *Nat. Mach. Intell.* **4**, 1198–1208 (2022).
29. Robotics, U. Unitree rl gym. https://github.com/unitreerobotics/unitree_rl_gym (2025). BSD 3-Clause License.

Acknowledgements

We would like to thank Prof. Auke Ijspeert for providing useful suggestions on improving the quality of this manuscript. We would like to thank Zeren Luo, Jiahui Zhang, Yidan Lu, and Xinqi Li for the experiment and hardware support. This work was supported by the General Research Fund under Grant 17204222, the Seed Fund for Collaborative Research, General Funding Scheme-HKU-TCL Joint Research Center for Artificial Intelligence, National Science Foundation China Grant 62273286, and Strategic Topics Grant STG1/E-401/23-N.

Author contributions

P.L. proposed the initial idea of the research. E.X. implemented all the software modules. The experiments were designed by P.L. and performed by E.X. with the help of Y.D. and P.L. The manuscript was written by P.L. and E.X. J.L. provided comments for improving the manuscript. P.L. provided the funding and supervised the research.

Competing interests

We declare that the Authors have no competing interests as defined by Nature Portfolio, or other interests that might be perceived to influence the results and/or discussion reported in this paper.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s44182-025-00043-2.

Correspondence and requests for materials should be addressed to Peng Lu.