DreamWaQ++: Obstacle-Aware Quadrupedal Locomotion With Resilient Multi-Modal Reinforcement Learning

I Made Aswin Nahrendra[1,2†], Byeongho Yu[3], Minho Oh[1,3], Dongkyu Lee[1,3], Seunghyun Lee[1], Hyeonwoo Lee[1], Hyungtae Lim[4], Hyun Myung[1*]

***Abstract*—Quadrupedal robots hold promising potential for applications in navigating cluttered environments with resilience akin to their animal counterparts. However, their floating-base configuration makes them susceptible to real-world uncertainties, presenting substantial challenges in locomotion control. Deep reinforcement learning has emerged as a viable alternative for developing robust locomotion controllers. However, approaches relying solely on proprioception often sacrifice collision-free locomotion, as they require front-foot contact to detect stairs and adapt the gait. Meanwhile, incorporating exteroception necessitates a precisely modeled map observed by exteroceptive sensors over time. This work proposes a novel method for fusing proprioception and exteroception through a resilient multi-modal reinforcement learning framework. The proposed method yields a controller demonstrating agile locomotion on a quadrupedal robot across diverse real-world courses, including rough terrains, steep slopes, and high-rise stairs, while maintaining robustness in out-of-distribution situations.**

***Index Terms*—Legged robots, control, multi-modal perception, reinforcement learning**

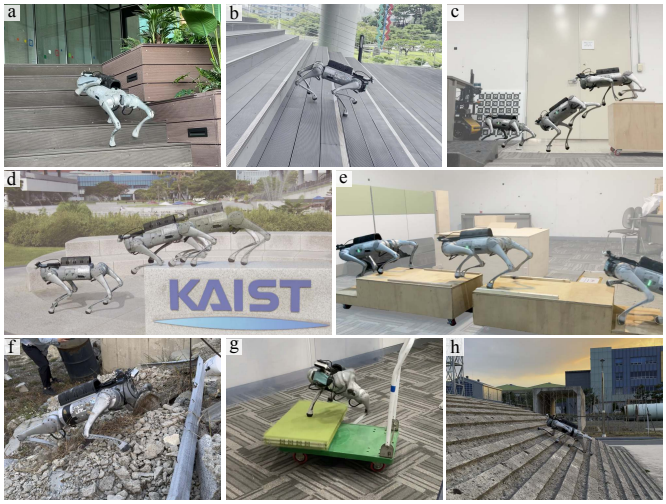


Fig. 1. The locomotion controller trained using DreamWaQ++ allows a quadrupedal robot to perform agile and resilient locomotion over various obstacles and terrains. The controller exhibits versatile gaits such as (a) ascending and (b) descending over a flight of stairs, (c) performing a leap motion, (d) probing when faced with an uncertain dip, (e) crossing a gap, (f) adapting to unseen deformable disastrous terrain, (g) balancing on movable platforms, and (h) climbing a 35° slope. Note that all these behaviors are embodied in a single neural network without specialized training for a particular scenario.

I. Introduction

In the past decade, quadrupedal robots have revolutionized robotic applications in real-world environments thanks to their capability to traverse cluttered spaces, enabling diverse applications spanning exploration and inspection [1]–[4]. The growing interest in quadrupedal robot applications has been accompanied by advancements in control algorithms, which have evolved from traditional model-based control [5]–[9] to data-driven approaches such as deep reinforcement learning (RL) [10]–[20].

Traditional model-based control pipelines for legged robots typically rely on a complex cascaded structure [5] that comprises accurate state estimation [21]–[24], terrain mapping [25]–[29], and a whole-body controller optimizing the robot's foot trajectory [6]–[9]. However, these pipelines can be computationally intensive for real-time inference and often require strict assumptions, such as collision-free and non-slip conditions. Although simplified models are sometimes used to reduce problem complexity, they potentially aggravate the performance.

As opposed to model-based control, deep RL methods reformulate the optimization problem into offline optimization during training by learning a decision-making policy that implicitly plans future control actions based on given observations. Notably, a blind locomotion controller, relying only on proprioception, demonstrates impressive robustness across various terrain profiles [10]–[14]. However, the resilience of blind locomotion controllers is inherently limited, as they require collisions between the robot's legs and the environment to sense obstacle properties and adjust the gait accordingly.

To advance blind locomotion controllers, an efficient fusion of proprioception and exteroception to learn a robust

[1]Urban Robotics Lab., School of Electrical Engineering, KAIST, Daejeon, Republic of Korea.
[2]KRAFTON, Seoul, Republic of Korea.
[3]URobotics, Seoul, Republic of Korea.
[4]Laboratory for Information and Decision Systems (LIDS), MIT, Cambridge, MA, USA.
[†]Currently at [2]. This work was done during his time at [1].
[*]Corresponding author: Hyun Myung (hmyung@kaist.ac.kr)
Project page: https://dreamwaqpp.github.io
This work was supported in part by Korea Evaluation Institute of Industrial Technology (KEIT) funded by the Korea Government (MOTIE) under grant No. 20018216, "Development of Mobile Intelligence SW for Autonomous Navigation of Legged Robots in Dynamic and Atypical Environments for Real Application", and in part by the R&D Program for Forest Science Technology (Project No. RS-2025-25424472) provided by Korea Forest Service (Korea Forestry Promotion Institute). The students are supported by BK21 FOUR.

quadrupedal locomotion controller is actively studied in the legged robotics community [15]–[20]. Naturally, animals have an agile locomotion behavior, owing to their ability to observe the terrain ahead using their eyes and quickly plan their effective gait for traversing the terrain. Therefore, incorporating exteroception for the gait planning of legged robots is paramount for eliciting agile behaviors [30]–[33].

We propose DreamWaQ++, an obstacle-aware quadrupedal locomotion controller that specifically aims to tackle the following challenges: 1) a resilient controller with multi-modal perception capability and sensor-agnostic nature that can be integrated with various options of exteroceptive sensors, 2) an efficient control framework that enables real-time control and fast adaptation, 3) an efficient reinforcement learning (RL) pipeline with a single-stage learning procedure. By employing DreamWaQ++ on a Unitree Go1 [34], we demonstrated remarkable performance in traversing various challenging environments[1]as shown in Fig. 1.

This paper is an evolved version of the conference paper [13]. We have extended the work by providing substantial improvement in the controller's performance and robustness in the following aspects:

- Incorporation of a novel multi-modal perception module that fuses proprioception and exteroception. The module enables the controller to adapt to various terrains and obstacles, including stairs, gaps, and deformable terrains.
- Enhanced robustness via a multi-modal deep RL framework that enables the controller to efficiently leverage the multi-modal perception module for learning a robust locomotion policy in a single-stage learning procedure.
- Improved agility via skill discovery objectives that encourage the controller to learn versatile gaits for traversing diverse terrains and obstacles.
- Extensive evaluation on various challenging environments and robotics platforms, highlighting the robustness and scalability of the proposed controller.
- Ablation studies to analyze the framework's components and their contributions to the controller's performance, opening up new research directions for future work.

II. Related Works

In recent years, research on legged locomotion has made significant progress, largely driven by advancements in simulation-based policy learning through deep reinforcement learning. Earlier works in the field primarily focused on blind locomotion strategies, aiming to ensure reliable sim-to-real transfer by minimizing actuator model discrepancies [35] and incorporating adaptation mechanisms to handle environmental variations [10]–[13]. These approaches relied predominantly on proprioceptive inputs to control the robot's movement. However, due to their limited perceptual capabilities, these methods often struggle in more complex dynamic environments.

To address these limitations, exteroceptive perception has been incorporated into the locomotion pipeline to enable policies that are not only stable but also aware of the surrounding terrain geometry, particularly near the robot's legs [15], [16], [19], [36]. Recent studies have utilized raw egocentric depth vision for locomotion [17]–[20], [37]to mimic the locomotion abilities of animals. However, elevation map–based approaches [15], [16], [38] have proven to be superior, particularly in situations where depth vision is unreliable due to limited field of view (FoV).

In addition to exteroception, memory-based architectures such as long short-term memory (LSTM) and gated recurrent units (GRU) have become primary components in the success of recent perceptive locomotion controllers [12], [14], [15], [39] to mitigate partial observability. However, training recurrent network models often suffers from vanishing gradients due to the backpropagation through time (BPTT) mechanism [40]. As a workaround, variants of convolutional neural network (CNN) architectures have been used as a viable option to handle sequential data [10], [11]. However, CNNs are prone to inductive bias, which assumes that neighboring data points are more likely to be related than others. This inductive bias hinders a neural network's ability to freely learn the positional relationships between features in unstructured time-series data. More recently, attention-based sequence models, such as transformers [41], [42], have shown potential as viable alternatives for constructing memory in locomotion tasks [17].

Utilizing large and expressive models such as transformers is effective for learning complex policies in high-dimensional action spaces [42]. However, these models typically require large amounts of training data and are more susceptible to real-time inference constraints due to their computational overhead. To effectively balance perception, memory, and fast reaction speed, we propose a multi-modal mixer architecture that combines the strengths of both lightweight MLPs and attention-based mechanisms. This architecture enables efficient processing of both proprioceptive and exteroceptive inputs, allowing the policy to learn a robust and adaptable locomotion strategy.

However, memory alone is sometimes insufficient for achieving resilient locomotion behavior if the learned latent representation obtained from memory does not account for explorative behavior that promotes skill discovery. Without an adequate skill discovery strategy, the latent representation may lead the policy to overfit to a limited range of behaviors, resulting in a conservative policy that struggles to adapt to diverse environmental changes [43], [44]. Therefore, we propose two regularization techniques that balance accurate perception learning while also promoting explorative behavior.

III. The Proposed Learning Framework

A. Overview

DreamWaQ++ comprises a perception pipeline and a control pipeline, detailed throughout this section. All modules are jointly trained with a combination of objective functions that facilitate interaction between networks, promoting cooperative learning of informative latent features. This training approach resembles the context adaptation paradigm in a few-shot meta-RL setting [45], [46]. Specifically, the context encoder is trained on a large dataset of simulated quadrupedal locomotion

[1]https://youtu.be/DECFbMdpfps

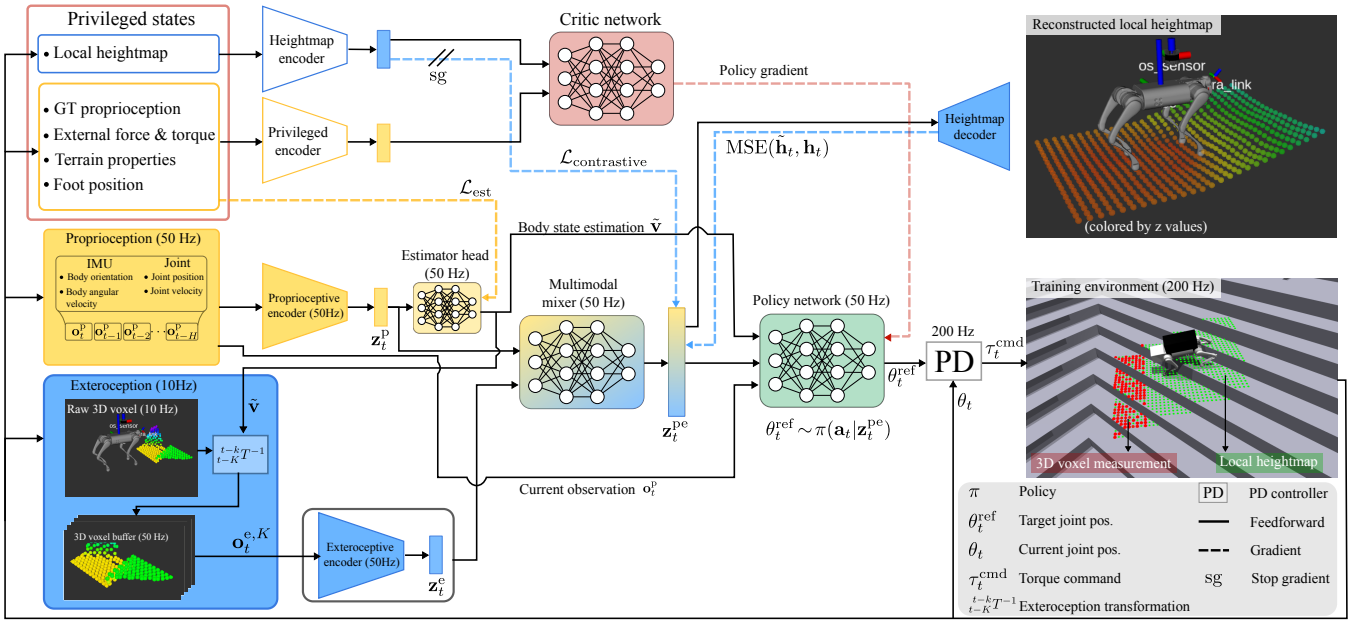


Fig. 2. Overview of DreamWaQ++. The encoder has a hierarchical structure that consists of low-level raw measurement encoders and a spatio-temporal mixer.

with extensive domain randomization, resembling the meta training phase. Subsequently, the learned context is used to condition the policy in real time, akin to the meta-testing phase. Simultaneously, the policy is trained to effectively control the robot, thereby mitigating the impact of poor context features and state estimation. All trained networks were deployed on a real-world quadrupedal robot with onboard sensors, without any fine-tuning. An overview of the entire framework is shown in Fig. 2.

The control problem is formulated in a partially observable Markov decision process (POMDP) setting with a goal to maximize the expected discounted future rewards, which in turn results in a policy:

$$\pi = \arg\max_{\mathbf{a}} E\left[\sum_{t=0}^{\infty} \gamma^t r_t\right], \tag{1}$$

where $\mathbf{a}$, γ, and r are the action, discount factor, and rewards, respectively. This objective is optimized using the proximal policy optimization (PPO) [47] algorithm, while also taking into account the auxiliary objectives related to the perception pipeline.

We adopt a privileged learning setting using an asymmetric actor-critic architecture trained using the proximal policy optimization (PPO) [47] algorithm. The actor, i.e. the policy receives partial and noisy observations ($\mathbf{o}_t^{\text{p}}$), akin to the real-world observations and the multi-modal context ($\mathbf{z}_t^{\text{pe}}$) as its input. The critic, by contrast, receives a privileged state that are accessible only in simulation. The policy network runs at a rate of 50 Hz, generating a target joint position that is tracked by a low-level PD controller, running at 200 Hz, to generate the joint torque commands.

B. Hierarchical Exteroceptive Memory

Employing exteroceptive measurements in a controller presents distinct challenges due to the low-frequency nature of the sensor, which operates at approximately two to five times lower than the control loop and proprioceptive sensors. This asynchrony introduces non-negligible delays into the control loop, yielding degraded performance.

We circumvent this issue using a memory structure that generates a denser point cloud, $\mathbf{o}_t^{\text{e},K}$, around the robot by concatenating points from the last K measurements, each transformed to the robot's current position using $SE(3)$ transformations. Unlike approaches such as [38], [48], which employ U-Net-based architectures for full scene reconstruction, we use autoregression solely to estimate the $SE(3)$ transformation of the robot's body frame over time. This method avoids computationally intensive reconstruction while still providing temporally dense exteroception for the controller.

The $SE(3)$ transformation is updated at each control loop iteration by combining the IMU's orientation measurement with the integration of the body linear velocity $\mathbf{v}_t$ predicted by the state estimation network, which leverages the history of the robot's proprioception as detailed in Section III.D. This transformation process enables temporal extrapolation of the latest exteroceptive measurements by aligning previous points with the robot's current frame, forming the basis of a memory structure that is then fed into the exteroceptive encoder. The exteroceptive observation is formally defined as follows:

$$\mathbf{o}_t^{\text{e},K} = \mathbf{o}_t^{\text{e}} \oplus \hat{\mathbf{o}}_{t-1}^{\text{e}} \oplus \cdots \oplus \hat{\mathbf{o}}_{t-K}^{\text{e}}, \tag{2}$$

where $\mathbf{o}_t^{\text{e}}$ is the most recent exteroceptive observation at time t. $\hat{\mathbf{o}}_{t-K}^{\text{e}}$ is the previous exteroceptive observation at $t-K$, which has been transformed to the robot's body frame at time t, which is defined as

$$\hat{\mathbf{o}}_{t-k}^{\text{e}} = {}_{t-K}^{t-k}T^{-1} \cdot \mathbf{o}_{t-K}^{\text{e}}, \tag{3}$$

where $\mathbf{o}_{t-k}^{\text{e}}$ is the exteroceptive observation measured at time $t-k$ ($k \in [0, K]$), and ${}_{t-K}^{t-k}T$ is the $SE(3)$ transformation of the robot's pose from time t to $t-k$.

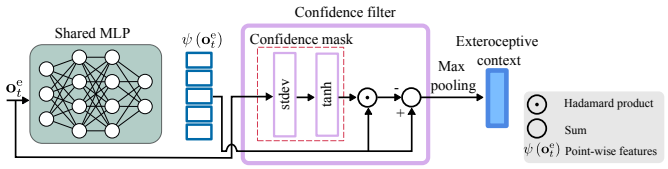


Fig. 3. The exteroceptive encoder uses a PointNet-based architecture as its backbone. We designed a confidence filter layer that statistically learns a masking layer that cancels out unreliable point features before aggregating them into the exteroceptive context $\mathbf{z}_t^{\mathrm{e}}$.

As the state estimation network only predicts the body linear velocity, we integrate this velocity over k steps using Euler integration to estimate the robot's position transformation. Although drift may accumulate in the pose estimate over time, we found that this is not significant because the integration is reset every K steps when a new exteroceptive measurement is received. Moreover, we also randomized the exteroceptive sensor pose with respect to the robot body frame during training and applied several levels of exteroception noise to enhance the policy's robustness against erroneous local pose estimation and exteroception data (Section IV.D).

In this work, we set the exteroception sampling rate to 10 Hz, resulting in $K = 5$ for constructing the exteroceptive memory. While higher sampling rates are theoretically possible depending on the choice of sensor, we selected 10 Hz due to practical constraints. In particular, higher rates might be slightly unreliable due to data processing latency on the limited onboard computing resources. Moreover, using 10 Hz sampling rate also allows us to change the sensor into LiDAR without the need to retrain the policy, as commercial 3D LiDAR sensors typically operate at 10 Hz.

C. Exteroceptive Encoder

We utilized 3D points as input to our framework to enable flexible compatibility with multiple sensor configurations, such as a 3D LiDAR sensor or depth camera. A PointNet-based architecture [49] is employed to effectively extract information from the input point cloud, accommodating an arbitrary number of points while maintaining robustness to noise.

Although the max-pooling layers in PointNet enforce invariance to the number and order of input points in the point cloud, they can become detrimental when outliers and heavy noise dominate the input. This issue arises because the max-pooling operation aggregates point features indiscriminately. To address this, we employ a confidence filter layer following the backbone PointNet architecture (Fig. 3). The confidence filter statistically rejects unreliable points in the latent space using a filtering operation, resulting in confidence-filtered points defined as:

$$\mathcal{C}\left(\mathbf{o}_t^{\mathrm{e},K}\right) = \psi^{\mathrm{e}}\left(\mathbf{o}_t^{\mathrm{e},K}\right) \cdot \left(1 - \tanh\left(\sigma\left(\mathbf{o}_t^{\mathrm{e},K}\right)\right)\right), \qquad (4)$$

where $\psi^{\mathrm{e}}(\cdot)$ is the backbone PointNet layer, and $\sigma(\cdot)$ is a standard deviation operator that statistically assesses the diversity of the input point cloud. A hyperbolic tangent operation, $\tanh(\cdot)$, is used to smoothly set an upper bound of $\sigma\left(\mathbf{o}_t^{\mathrm{e},K}\right)$ to one. Each point feature, $\psi\left(\mathbf{o}_t^{\mathrm{e},K}\right)$, is fed into a shared confidence mask layer that outputs confidence masks based on the statistics of the raw points. The confidence mask outputs a value close to 1 for high-variance features and a value close to 0 for low-variance features, due to the $\tanh$ layer. Following Eq. (4), $\mathcal{C}\left(\mathbf{o}_t^{\mathrm{e},K}\right)$ removes high-variance features, such as outliers, while preserving low-variance features. The filtered point features are then aggregated using max-pooling to obtain the exteroceptive context $\mathbf{z}_t^{\mathrm{e}}$.

D. Proprioceptive Encoder

The proprioceptive encoder is based on the context-aided estimator network (CENet) [13], modified by replacing standard fully connected layers with an MLP-mixer architecture [50]. This modification enables interactions across different proprioceptive modalities over multiple time frames, enhancing both the explicit estimation and latent representation of proprioception. To achieve this, we treat the proprioceptive features as tokens and the time frames as channels, enabling the MLP-mixer to learn interactions across different proprioceptive modalities and temporal dimensions. We employ MLPs with two hidden layers, each consisting of 256 hidden units, and use ELU activation functions for both the token-mixing and channel-mixing MLPs.

The encoder receives a stack of temporal observations at time t over the past H measurements as $\mathbf{o}_t^{\mathrm{p},H} = \left[\mathbf{o}_t^{\mathrm{p}}\ \mathbf{o}_{t-1}^{\mathrm{p}}\ \cdots\ \mathbf{o}_{t-H}\right]^T$, allowing the policy to infer context with short-term memory. Specifically, we set $H = 5$, with the policy running at 50 Hz, providing a memory window of 100 ms.

The proprioceptive encoder is trained to output a distribution over latent states using variational inference [51], supporting exploratory learning and serving as a denoising mechanism to aid domain adaptation. This stochastic latent representation significantly reduces the sim-to-real gap, resulting in smooth and robust real-world control [13].

The latent vector $\mathbf{z}_t^{\mathrm{p}}$ is used as input to the multi-modal mixer and for body velocity estimation via an additional estimation layer following the encoder. The body velocity estimation layer is a fully connected layer with 256 hidden units and ELU activation, which predicts the body linear velocity $\hat{\mathbf{v}}_t$ using the loss function in Section III.F1.

E. Multi-Modal Mixer

The multi-modal mixer network was implemented using an MLP-mixer architecture, where the input features are treated as tokens and the time frames are treated as channels. Both the token-mixing and channel-mixing MLPs consist of two hidden layers with 256 hidden units each and employ ELU activation functions.

To stabilize learning, we apply layer normalization separately to the proprioceptive and exteroceptive latent features before passing them into the multi-modal mixer. This normalization step ensures that the two modalities are on a comparable scale, thereby facilitating more effective fusion.

The multi-modal mixer was trained end-to-end alongside all other networks in DreamWaQ++, as shown in Fig. 2. However,

this setup presents a trade-off in numerical stability during the early stages of training.

A straightforward solution is to balance the weights of the reconstruction loss and KL divergence in $\mathcal{L}_{\text{VAE}}$. In variational inference, the prior distribution of the latent state is typically assumed to follow a normal distribution. Strong minimization of latent loss during training can help the encoder better approximate the latent space, enhancing training stability. However, this approach risks posterior collapse, wherein the encoder neglects critical input details, impairing the policy's ability to detect and respond to small environmental obstacles. To address this, we introduced a constrained reparameterization trick, defined as follows:

$$\mathbf{z} \sim N(g_\mu(\mathbf{x}), g_\sigma(\mathbf{x})), \tag{5}$$

where $\mathbf{z}$ is a stochastic latent vector and $\mathbf{x}$ is the input to the encoder network g. The subscripts μ and σ indicate the outputs of $g(x)$ that correspond to the mean and standard deviation of the latent distribution, respectively. $\mathbf{z}$ is sampled from a Gaussian distribution $N(\cdot, \cdot)$ with mean $g_\mu(\mathbf{x})$ and standard deviation $g_\sigma(\mathbf{x})$.

During the reparameterization step, we imposed hard constraints on the standard deviation of the distribution, ensuring $\sigma_{\min} \leq g_\sigma(\mathbf{x}) \leq \sigma_{\max}$. This constraint guarantees numerically stable samples that can be reliably propagated to subsequent network layers. By implementing this simple yet effective solution, we achieve greater training stability without compromising the policy's final performance. Empirically, we set $\sigma_{\min} = 0$ and $\sigma_{\max} = 5$ to facilitate stable training. This constrained reparameterization trick is applied across all encoder networks that use a stochastic layer.

F. Learning Objectives

We trained the multi-modal context encoder using three losses: an estimation loss, $\mathcal{L}_{\text{est}}$, a proprioceptive variational auto-encoder (VAE) loss, $\mathcal{L}^{\text{p}}_{\text{VAE}}$, and an exteroceptive VAE loss, $\mathcal{L}^{\text{e}}_{\text{VAE}}$. These losses are combined and included as an auxiliary term in the policy loss.

1) Estimation Loss: The estimation loss is used to train the proprioceptive encoder to explicitly estimate the body velocities of the robot, $\tilde{\mathbf{v}}_t$. The estimation objective was formulated using mean-squared-error (MSE) loss as:

$$\mathcal{L}_{\text{est}} = \text{MSE}(\tilde{\mathbf{v}}_t, \mathbf{v}_t), \tag{6}$$

where $\mathbf{v}_t$ as the ground-truth (GT) body velocity of the robot in the robot frame. We also adaptively bootstrap $\tilde{\mathbf{v}}_t$ during policy training to improve the robustness of the policy [13]. To avoid exploiting inaccurate estimation in the early stage of training, a bootstrapping probability, $p_{\text{boot}} \in [0, 1]$, is computed by measuring the coefficient of variation, $CV(\cdot)$ of the cumulative rewards $\mathbf{R} \in \mathbb{R}^{m \times 1}$. The probability is formulated as

$$p_{\text{boot}} = 1 - \tanh(CV(\mathbf{R})). \tag{7}$$

2) VAE Loss: The multi-modal context encoder is trained using an unsupervised method with two reconstruction tasks. First, the proprioceptive encoder is trained to reconstruct the future observation, $\tilde{\mathbf{o}}_{t+1}$, to encourage the predictive nature of the network. We employ β-VAE loss for the proprioceptive encoder, formulated as

$$\mathcal{L}^{\text{p}}_{\text{VAE}} = \text{MSE}(\tilde{\mathbf{o}}_{t+1}, \mathbf{o}_{t+1}) + \beta D_{\text{KL}}(q(\mathbf{z}^{\text{p}}_t | \mathbf{o}^{\text{p},H}_t) \parallel p(\mathbf{z}^{\text{p}}_t)), \tag{8}$$

where the first term is the reconstruction loss and the second term is the latent regularization loss expressed with a Kullback-Leibler (KL) divergence operation. The latent regularization is scaled with $\beta = 5.0$ to encourage disentanglement [13], [51]. The prior distribution of the proprioceptive context $p(\mathbf{z}^{\text{p}}_t)$ is parameterized using a Gaussian distribution and the posterior distribution $q(\mathbf{z}^{\text{p}}_t | \mathbf{o}^{H}_t)$ is approximated using a neural network, i.e. via the encoder network.

Second, the exteroceptive and multi-modal context encoders are trained with an exteroceptive VAE loss, formulated as

$$\mathcal{L}^{\text{e}}_{\text{VAE}} = \text{MSE}(\tilde{\mathbf{h}}_t, \mathbf{h}_t) + \beta D_{\text{KL}}(q(\mathbf{z}^{\text{pe}}_t | \mathbf{o}^{\text{pe}}_t) \parallel p(\mathbf{z}^{\text{pe}}_t)), \tag{9}$$

where $\mathbf{z}^{\text{pe}}_t = f_{\psi_{\text{mix}}}(\mathbf{z}^{\text{p}}_t \oplus \mathbf{z}^{\text{e}}_t)$ is the output of the multi-modal context encoder, as a result of feeding the concatenation of the proprioceptive and exteroceptive context vectors into the mixer network, $f_{\psi_{\text{mix}}}(\cdot)$. $\mathbf{o}^{\text{pe}}_t = \mathbf{o}^{\text{p},H}_t \oplus \mathbf{o}^{\text{e},K}_t$ is the observable proprioception and exteroception. The ground-truth robot-centric height scan, $\mathbf{h}_t$, is obtained from the simulator, and $\tilde{\mathbf{h}}_t$ is its reconstruction, which can be obtained via a decoder network that receives $\mathbf{z}^{\text{pe}}_t$ as its input.

A large value of β imposes a strong latent regularization, thus, limiting the reconstruction accuracy, and vice versa. Although it is only a single parameter, tuning β is non-trivial and lack of intuition. Therefore, we propose an adaptive β scheduling method to ease its tuning procedure by scaling it with a factor, k, computed as:

$$k = \exp\{(\delta \cdot (\tau - \mathcal{L}_{\text{recon}}))\}, \tag{10}$$

where $\delta > 0$ is the learning rate for k, τ is the allowed reconstruction error threshold, and $\mathcal{L}_{\text{recon}}$ is the reconstruction loss. Subsequently, β is updated using the following rule:

$$\beta \leftarrow \begin{cases} \beta_{\min} & \text{if } k\beta \leq \beta_{\min}, \\ k\beta & \text{if } \beta_{\min} \leq k\beta \leq \beta_{\max}, \\ \beta_{\max} & \text{if } k\beta > \beta_{\max}. \end{cases} \tag{11}$$

Intuitively, k is updated at every iteration depending on the reconstruction loss of the VAE network. When the reconstruction error exceeds a certain threshold, τ, then β is scaled down to allow learning of more accurate reconstruction. In contrast, when the reconstruction error is below the given threshold, β is scaled up to allow learning of more disentangled latent representation.

3) Contrastive Loss: Prior works trained an adaptation encoder using a regression loss to explicitly predict environment properties [10], [11]. However, this approach might suffer from *realizability gap* [52] caused by insufficient observations to reconstruct the environment properties. To circumvent this issue, we tighten the distribution gap between the learned latent representations of policy's observations and critic's

privileged observations, rather than requiring the policy to infer the privileged information via regression. We employ a contrastive learning framework by matching the distribution of the privileged latent features used for the critic with the latent features inferred from partial observations used for the actor within an asymmetric actor-critic setup. We define the contrastive loss as:

$$\begin{aligned}\mathcal{L}_{\text{contrastive}} =& \lambda \left\| \mathbf{z}_t^{\text{pe}} - g_{\theta_{\text{h}}}(\mathbf{h}_t) \right\|_2^2 \\ &+ (1-\lambda) \left\| \max(0, m - (\mathbf{z}_t^{\text{pe}} - \mathbf{z}_t^{\text{random}})) \right\|_2^2,\end{aligned} \tag{12}$$

where $g_{\theta_{\text{h}}}(\mathbf{h}_t)$ is the encoded ground-truth height scan, which is used as the positive anchor for the contrastive loss. Meanwhile, $\mathbf{z}_t^{\text{random}}$ is a random latent feature sampled from $\mathcal{U}[-1.0, 1.0]$, which is used as the negative anchor. The parameters $m \in \mathbb{R}^+$ and $\lambda \in [0, 1]$ are the margin for the negative pair separation and scaling factor, respectively. This contrastive loss forces the multi-modal latent feature to match the encoded ground-truth height scan, while also distancing the latent feature from an unstructured representation labeled by the uniformly random latent feature.

G. Skill Discovery

We incorporate an unsupervised RL objective through mutual information (MI) maximization for promoting skill discovery. This objective allows the emergence of novel behaviors while preserving stable behaviors induced by the handcrafted reward functions. Specifically, we maximize the MI between visited states and the latent variable inferred by the multi-modal context encoder.

The MI objective is introduced as a regularization term in the PPO loss function. We call this objective as *versatility gain*, which seeks to be maximized for inducing versatile locomotion behaviors. Thus, the versatility gain can balance exploration, exploitation, and reconstruction. The versatility gain is defined as:

$$\mathcal{G}_{\text{versatility}} = \mathcal{I}(\mathbf{o}_t^{\text{pe}}; \mathbf{z}_t^{\text{pe}}) = \mathcal{H}(\mathbf{z}_t^{\text{pe}}) - \mathcal{H}(\mathbf{z}_t^{\text{pe}} | \mathbf{o}_t^{\text{pe}}), \tag{13}$$

where $\mathcal{I}(\cdot;\cdot)$, $\mathcal{H}(\cdot)$, and $\mathcal{H}(\cdot|\cdot)$ are the mutual information, Shannon entropy, and conditional entropy operators, respectively. Eq. (13) comprises two terms that were essential for training. The first term maximizes the variation of the inferred latent variables, thus, promoting the variation of skills that can be obtained during policy learning. The second term minimizes the entropy of the latent states given an observation, thus, acting as a denoising operation to filter out noisy observations by clustering intrinsically similar observations into a similar latent representation.

Generally, the encoder is trained to minimize the KL divergence between $\mathbf{z}_t^{\text{pe}}$ and $\mathbf{o}_t^{\text{pe}}$, i.e. $\mathcal{L}_{\text{encoder}} \approx D_{\text{KL}}\left(\mathbf{z}_t^{\text{pe}} | \mathbf{o}_t^{\text{pe}}\right)$, effectively compressing raw observations while maintaining the original data distribution. Subsequently, jointly training the networks using $\mathcal{G}_{\text{versatility}}$ and $\mathcal{L}_{\text{encoder}}$ maximizes:

$$\begin{aligned}\mathcal{J} &\triangleq \mathcal{G}_{\text{versatility}} - \lambda_{\text{e}} \mathcal{L}_{\text{encoder}} \\ &= \mathcal{I}(\mathbf{o}_t^{\text{pe}}; \mathbf{z}_t^{\text{pe}}) - \lambda_{\text{e}} D_{\text{KL}}\left(\mathbf{z}_t^{\text{pe}} | \mathbf{o}_t^{\text{pe}}\right) \\ &= \mathcal{H}(\mathbf{z}_t^{\text{pe}}) - \mathcal{H}(\mathbf{z}_t^{\text{pe}} | \mathbf{o}_t^{\text{pe}}) + \lambda_{\text{e}} \left[\mathcal{H}(\mathbf{z}_t^{\text{pe}}, \mathbf{o}_t^{\text{pe}}) - \mathcal{H}(\mathbf{z}_t^{\text{pe}})\right] \\ &= \mathcal{H}(\mathbf{z}_t^{\text{pe}}) - \mathcal{H}(\mathbf{z}_t^{\text{pe}} | \mathbf{o}_t^{\text{pe}}) + \lambda_{\text{e}} \left[\mathcal{H}(\mathbf{z}_t^{\text{pe}} | \mathbf{o}_t^{\text{pe}}) + \mathcal{H}(\mathbf{o}_t^{\text{pe}}) - \mathcal{H}(\mathbf{z}_t^{\text{pe}})\right] \\ &= (1-\lambda_{\text{e}})\, \mathcal{H}(\mathbf{z}_t^{\text{pe}}) - (1-\lambda_{\text{e}})\, \mathcal{H}(\mathbf{z}_t^{\text{pe}} | \mathbf{o}_t^{\text{pe}}) + \lambda_{\text{e}} \mathcal{H}(\mathbf{o}_t^{\text{pe}}),\end{aligned} \tag{14}$$

where $\mathcal{H}(\cdot,\cdot)$ is a cross-entropy operation and $\lambda_{\text{e}} \in \mathbb{R}^+$ is the scaling factor for $\mathcal{L}_{\text{encoder}}$. Eq. (14) shows that choosing $\lambda_{\text{e}} = 1$ leads to entropy maximization on the state visitation that subsequently promotes policy exploration and skill discovery during training. Furthermore, choosing $\lambda_{\text{e}} < 1$ maximizes $\mathcal{H}(\mathbf{z}_t^{\text{pe}})$ and minimizes $\mathcal{H}(\mathbf{z}_t^{\text{pe}} | \mathbf{o}_t^{\text{pe}})$, effectively diversifying the distribution of $\mathbf{z}_t^{\text{pe}}$ while compressing $\mathbf{o}_t^{\text{pe}}$. In practice, we set $\lambda_{\text{e}} = 0.1$ for our experiments.

IV. Implementation Details

A. Simulation

We utilized NVIDIA Isaac Gym Preview 3 [53] as the simulator to train the controller and multi-modal context encoder networks, with training environments based on the Legged Gym library [54]. Domain randomization was applied across $3{,}500$ agents, completing training in approximately 11 hours on an NVIDIA A5000 GPU. The trained multi-modal context encoder and policy networks were then deployed without any fine-tuning on a physical robot or in the Gazebo simulator for the evaluations presented in this paper.

B. Low-Level Control

The policy and multi-modal context encoder networks run synchronously while processing asynchronous observations. Proprioceptive measurements are sampled at 200 Hz, and exteroceptive measurements at 10 Hz, with the controller integrating the latest measurements at 50 Hz. To enhance robustness against asynchronous observations, latency randomization was applied during training. Detailed information on randomized latency for all measurements is available in Table I.

The policy network generates target joint positions at 50 Hz, which are then sent to a low-level proportional-derivative (PD) controller operating at 200 Hz. Within the PD controller, target joint positions are converted into torque commands using proportional (K_p) and derivative (K_d) gains of 25 and 0.7, respectively. We made a custom interface using Pybind to access the Unitree SDK via our Python script that runs the RL policy and to send the target joint positions to the Unitree SDK using a ROS system. The Pybind interface then converts the target joint positions into torque commands, which are subsequently transmitted to the low-level motor controller.

C. Domain Randomization

We randomized multiple physical properties of the robot and environment to facilitate sim-to-real transfer. Additionally, we employed Roll-Drop [55] to encourage exploration and

TABLE I
DOMAIN RANDOMIZATION RANGES APPLIED IN THE SIMULATION.

Parameter	Randomization range	Unit
Payload	$[-1, 2]$	kg
K_p factor	$[0.9, 1.1]$	$\mathrm{Nm\ rad^{-1}}$
K_d factor	$[0.9, 1.1]$	$\mathrm{Nms\ rad^{-1}}$
Motor strength factor	$[0.9, 1.1]$	Nm
Center of mass shift	$[-50, 50]$	mm
Friction coefficient	$[0.2, 1.25]$	-
System delay	$[0.0, 15.0]$	ms

TABLE II
NOISE AND SENSOR ALIGNMENT BIAS PARAMETERS INJECTED INTO THE OBSERVATION FOR THE POLICY NETWORK DURING TRAINING.

Observation	Noise range (μ)	Unit
Joint position	$[-0.01, 0.01]$	rad
Joint velocity	$[-1.5, 1.5]$	rad/s
Body linear velocity	$[-0.1, 0.1]$	m/s
Body angular velocity	$[-0.2, 0.2]$	rad/s
Gravity vector	$[-0.05, 0.05]$	$\mathrm{m/s^2}$
Exteroceptive measurement (low)	$[0.0, 0.03]$	m
Exteroceptive measurement (medium)	$[0.03, 0.1]$	m
Exteroceptive measurement (high)	$[0.1, 0.3]$	m
Exteroceptive bias roll	$[-0.2, 0.2]$	rad
Exteroceptive bias pitch	$[-0.15, 0.15]$	rad
Exteroceptive bias yaw	$[-0.1, 0.1]$	rad
Exteroceptive bias x	$[-0.1, 0.1]$	m
Exteroceptive bias y	$[-0.1, 0.1]$	m
Exteroceptive bias z	$[-0.1, 0.1]$	m

enhance robustness alongside physics randomization. Table I summarizes the details of the physical properties and their randomization ranges.

Additionally, we applied domain randomization to account for potential data asynchrony between proprioceptive and exteroceptive sensors caused by system delays. Such delays may arise from factors including data transmission latency, multithreaded execution on the robot, and hardware-induced timing jitter. These issues can violate the Markov property assumed in standard policy learning frameworks.

To mitigate this, we randomly delay proprioceptive observations within a range of $[0, 15]$ ms during training. This strategy encourages the policy to treat minor mismatches between proprioceptive and exteroceptive inputs as observation noise, thereby increasing its robustness to sensor delays during deployment.

While it is theoretically possible to extend the randomization range to further improve robustness, doing so can lead to overly conservative policies and degrade performance. Therefore, we set the maximum delay to 15 ms, which we found to be empirically appropriate given the computational constraints of our onboard hardware.

D. Adversarial Observations

During training, we injected noise into the proprioceptive and exteroceptive observations to make the policy be robust against noisy real-world observations. For the proprioceptive observations, a uniform noise was injected at each time step. For the exteroceptive data, we defined three different noise ranges, constituting low, medium, and high noise levels. The proportion of robots that operate with these exteroceptive noise scales during training was set to 30%, 50%, and 20%, for the low, medium, and high noise levels, respectively.

To handle erroneous extrinsic calibration between the exteroceptive sensor and the robot body frame, we also applied sensor alignment bias at the beginning of each episode to simulate extrinsic calibration error. This error encompasses biases in both the position and orientation of the sensor frame with respect to the robot body frame. The biases were sampled from a uniform distribution and consistently applied to the exteroceptive measurements throughout the episode. The ranges of the observation noises and sensor alignment biases are summarized in Table II.

E. Privileged States

We utilized a robot-centric local height map as the privileged exteroception, sampled around the robot in a 2.5D grid where each cell represents terrain height. The grid dimensions are $w = 1.1$ m by $h = 1.7$ m, with the first row positioned 0.9 m ahead of the robot's body frame, similar to the 3D voxel grid used for the multi-modal encoder input. The grid resolution is set to 5 cm.

For privileged proprioception, we utilized ground-truth data and simulation measurements, including body-centric linear and angular velocities, gravity vector, joint positions and angular velocities, external disturbance forces and torques, foot positions relative to the robot's body frame, and physical properties of the robot. The physical properties include friction, motor damping, motor stiffness, motor strength ratio, additional payload, and the robot's center of mass shift.

F. Reward Functions

We used the exact same rewards in [13]. On top of these rewards, we employed a reward curriculum to exponentially anneal some style rewards, i.e. joint torque, joint velocity, joint acceleration, action rate, and smoothness rewards. We observed that although these style rewards are important to ensure sim-to-real transfer, they can lead to suboptimal policies that make the robot only learn basic locomotion skills. This is because the agent may become overly focused on maximizing immediate rewards induced by these joint-level regularization rewards. Therefore, we gradually annealed the weight of the regularization rewards to allow the agent to explore a wider range of behaviors and styles after it has learned the basic locomotion task. The annealing follows the following rule:

$$w_{i+1} = \lambda w_i, \tag{15}$$

where w is the reward weight, i is the learning iteration, and λ is the annealing rate. We set $\lambda = 0.998$ and w_0 for the selected skill rewards are summarized in Table III.

V. EXPERIMENTAL RESULTS

A. Hardware Settings

All networks were trained to control a Unitree Go1 [34] robot. For the experiments, we used robots with different exteroception configurations, as shown in Fig. 4. Robots R1,

TABLE III
INITIAL WEIGHT w_0 FOR SELECTED STYLE REWARDS THAT WERE ANNEALED USING THE REWARD CURRICULUM.

Reward	Weight (w_0)
Joint torque	-5×10^{-6}
Joint velocity	-6×10^{-6}
Joint acceleration	-7.5×10^{-8}
Action rate	-1.5×10^{-5}
Smoothness	-1.5×10^{-5}

(a) (b) (c) (d)

Fig. 4. Hardware setup for robots (a) R1, (b) R2, (c) R3, and (d) R4. R1 was equipped with an Intel RealSense D435f camera, R2 was not equipped with any exteroceptive sensor and used only for blind locomotion, R3 was equipped with an Ouster OS-01 LiDAR, and R4 was equipped with two Livox Mid-360 LiDARs.

R3, and R4 are Unitree Go1 robots with different exteroception setups, while Robot R2 is a Unitree A1 robot without exteroception.

Robot R1 was equipped with an Intel RealSense D435f camera tilted $45°$ downward, streaming data at 15 Hz to the onboard Jetson Xavier NX. To protect cables from potential damage during falls, a canopy was added, increasing the payload by approximately 0.5 kg.

Robot R2, a Unitree A1 without exteroception, was used to compete against Robot R1. The blind locomotion policy, DreamWaQ [13] was deployed on the A1 robot because it has similar morphology and motor properties to the Go1, while offering higher torque. Consequently, the Unitree A1 is expected to perform comparably, if not better, than the Unitree Go1 with DreamWaQ, as it can apply additional torque to manage collisions during stair climbing.

Robot R3 was equipped with an Intel NUC PC and an Ouster OS-01 LiDAR. This configuration was used to evaluate the generalization and transferability of the learned controller on a robot with different exteroceptive sensors, resulting in an additional payload of approximately 3.0 kg. Finally, a fourth robot, R4, equipped with an Intel NUC PC and two Livox Mid-360 LiDARs, was utilized for further experiments in an asynchronous race setting shown in Figs. 5(c)-(e). The additional payloads on R3 and R4 are approximately 2.5 kg.

B. Fast Locomotion over Obstacles

1) Head-to-head Racing Across Stairs[2]*:* We benchmarked the proposed controller against DreamWaQ [13] (a baseline blind locomotion controller) and the robot's built-in perceptive controller [34] in a head-to-head stair-climbing race as shown in Fig. 5(a).

The experimental setup consisted of fifty stairs, and we used robots R1, R2, and R3 as described in Section V.A. Robots R1, R2, and R3 were controlled by DreamWaQ++, DreamWaQ,

[2]https://youtu.be/lAdPj3KTDo8

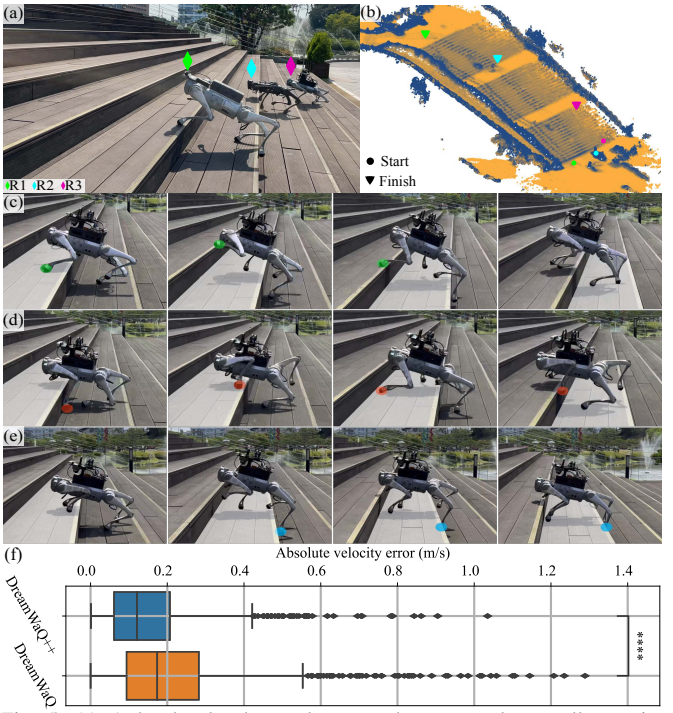


Fig. 5. (a) A head-to-head race between the proposed controller against baselines. (b) 3D map visualization of the race environment. Asynchronous stair-climbing experiments were conducted to ensure fairness, where the robot was controlled using (c) DreamWaQ++, (d) DreamWaQ, and (e) Unitree Go1's built-in controller. The white mask on the stair indicates the same stair plate that each robot interacted with over the four successive snapshots. (f) Velocity estimation error comparison in the stair-climbing task. The **** annotation indicates the significance level computed using a paired t-test.

and Unitree's built-in controller, respectively. All robots started from the same point before the stairs, except for R3, which was placed one stair ahead due to the built-in controller's difficulty with the height of the first step. For safety reasons, all robots were manually controlled by a human operator.

Robot R1 quickly outperformed the others, gaining a lead shortly after the starting point, as shown in Fig. 5(b). To mitigate frequent stumbles, robot R2 was commanded at a linear velocity of 1.2 m/s but continued to experience tracking errors due to frequent missteps on stair edges. In contrast, robot R1 adaptively adjusted its gait and foot placement, enabling faster traversal at a lower velocity command of approximately 1.0 m/s. Robot R3, however, struggled with stair climbing due to latency in its perceptive controller's reaction time, which depends on a local environmental map and limits both speed and agility.

The race concluded when one robot reached the final stair to assess overall reachability (Fig. 5(b)). Robot R1 completed the race in 35 s, covering a horizontal distance of approximately 30.03 m and a height of 7.38 m. By the same timestamp, R2 had covered about 20.05 m horizontally and 5.44 m in height, while R3 failed to finish after stumbling and covering only 6.38 m horizontally and 2.44 m in height. This experiment highlights the superior agility of the proposed controller in handling continuous obstacles.

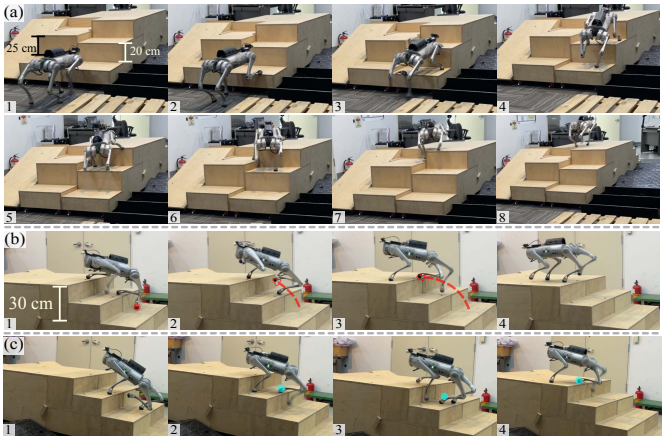


Fig. 6. (a) Affordance-aware locomotion when ascending stairs with rise of 25 cm on the left and 20 cm on the right side of the robot. (b) Emergent behavior to quickly and efficiently climb stairs with long foot swing motion, compared with a regular case (c) where the robot could not overcome two stair steps at once because the rear foot was located around the middle of the stair step.

2) Asynchronous Race[3]*:* We conducted additional asynchronous experiments using robot R4 (Fig. 4(d)) with autonomous navigation to ensure consistent path planning during stair climbing. This experiment aimed to validate the performance of DreamWaQ++ in a more controlled setting, where the same robot hardware was tested using both DreamWaQ and DreamWaQ++. Key snapshots highlighting the robot's movements are shown in Figs. 5(c)-(e).

A discernible behavior between DreamWaQ++ and DreamWaQ is in their gait adaptation. DreamWaQ++ raises the body and extends foot swing to step securely on the stairs (Fig. 5(c)), whereas DreamWaQ often collides and drags the foot along the stair edge before stepping (Fig. 5(d)), resulting in reduced efficiency and more frequent stumbles. The built-in controller uses a fixed gait, lacking spatial memory and adaptability, leading to rear leg stumbles (Fig. 5(e)).

We also measured state estimation errors for DreamWaQ++ and DreamWaQ during these experiments. This error, defined as the absolute difference between ground truth and estimated velocities, was computed using a LiDAR odometry algorithm [56]. As shown in Fig. 5(f), DreamWaQ++ demonstrated significantly lower estimation errors, indicating superior position estimation and gait adaptation—crucial for efficient stair climbing and enhanced stability and robustness.

C. Obstacle Awareness[4]

1) Obstacle Negotiation: In Fig. 6(a), the controller demonstrated affordance-awareness when navigating terrains of varying difficulty. The robot was given a forward velocity command of $0.6\ \mathrm{m/s}$ with a zero yaw rate command. Initially, when commanded to move toward the middle of the stairs (Fig. 6(a-2)), the robot moved toward the lower stair rise on the right side, opting for a less risky path. Subsequently, as the left and right stairs overlapped, a lower step appeared on the left side (Fig. 6(a-6)). The robot quickly adapted its path toward these easier steps, resisting the zero yaw rate command, thereby demonstrating the controller's ability to learn and perceive the affordances of different obstacles.

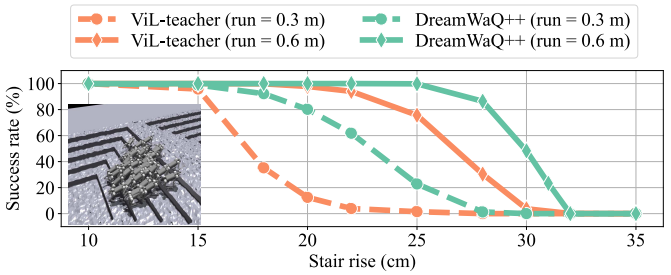


Fig. 7. Success rate (SR) comparison measured on each algorithm by simulating 1,000 robots to climb sequence of stairs. The SR is defined as the percentage of the number of robots that reached the last stair within 10 s over the total number of robots.

2) Foot Swing Adaptation: Fig. 6(b) illustrates the performance of the proposed controller on stairs with a 15 cm rise. Typically, the controller directs the robot to swing its feet, stepping on each stair sequentially. However, when a foot approaches the edge of a stair, the robot extends its swing phase (indicated by red arrows in Fig. 6(b)), allowing the rear foot to overcome a combined rise of 30 cm. In contrast, as shown in Fig. 6(c), the robot steps normally when the rear foot is positioned near the middle of the initial stair step. This adaptive behavior suggests that the controller retains a form of memory of the structure beneath the robot, leveraging fused information from the network architecture.

3) Quantitative Performance Assessment: We compared DreamWaQ++ with a visual locomotion controller based on ViNL [57], training the baseline with the same parameters as DreamWaQ++ but without the navigation pipeline from [57]. Additionally, we used only the teacher network to achieve upper-bound performance, noting that this network has access to the ground truth height map around the robot, which we refer to as *ViL-teacher*.

We simulated $1,000$ robots climbing stairs with increasing rises, as shown in Fig. 7. Using a stair run of $0.3\ \mathrm{m}$—a common real-world dimension—and an extended run of $0.6\ \mathrm{m}$ to represent low-slope, high-rise obstacles, the results in Fig. 7 indicate that DreamWaQ++ achieved success rates $20-40\%$ higher than the baseline with ground truth height map access. This finding suggests that accurate height map information alone is insufficient for high-performance locomotion, similar to animals' ability to climb without constant visual feedback.

This improvement is attributed to versatile skill learning promoted by the proposed versatility gain function $\mathcal{L}_{\mathrm{versatility}}$, which serves as an intrinsic reward encouraging exploration. Without this gain, the policy lacks exploratory behavior, often resulting in a conservative approach that fails to handle unseen terrains effectively.

D. Uncertainty Awareness and Adaptation[5]

Fig. 8 demonstrates an emergent locomotion behavior of the proposed controller when traversing terrains with signif-

[3]https://youtu.be/XVC7c5DlB4I
[4]https://youtu.be/mgNLLNxg52A
[5]https://youtu.be/G_lTWikijWk

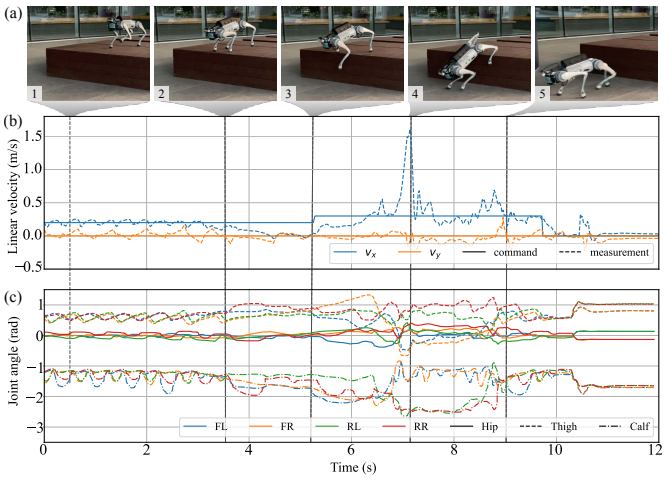


Fig. 8. An emergent probing skill enables the robot to check the upcoming terrain when it poses a high risk and uncertainty. (a) A sequence of the robot's movement to probe the upcoming terrain. (b) Corresponding velocity commands and estimation, showing how the controller resists the given command and allocates time for the robot to check for the terrain. (c) Significant knee flexion-extension (KFE) motions indicated by a sudden change in the calf joint angle, revealing the emergent adaptive behavior as a novel probing skill.

icant height differences. Figs. 8(a)-(c) visualize snapshots of the robot's motion, commanded and estimated base velocity, and joint angles, respectively. When confronted with a stage featuring a large elevation difference, the controller could not accurately gauge the terrain height near the robot's front feet. In response, the robot deliberately stops before the ridge (Fig. 8(a-2)) and orchestrates its feet to probe the terrain characteristics (Fig. 8(a-3)). Upon detecting solid ground, the robot continues moving forward and confidently descends from the edge of the stage (Fig. 8(a-4)). Subsequently, the robot spreads its rear legs and uses them as anchors, reducing the impact on the front legs upon landing.

E. Out-of-distribution Adaptation

1) Reacting to Sudden Changes of Foothold[6]*:* We evaluated the controller's adaptability to unexpected environmental changes, such as deformable and movable surfaces, which were not encountered during training. In Fig. 9(a), the robot initially moves toward a movable cart. Upon stepping onto the cart, an abrupt kick is applied, propelling the cart away from the robot.

The controller's swift response, shown in Fig. 9(a-4), involved manipulating the front hip joints (Fig. 9(b-4)) to create a support polygon approximately 20.12% larger than in normal locomotion (Fig. 9(c)), ensuring a safe landing after the platform was unexpectedly removed.

Fig. 9(d) visualizes the multi-modal contexts, forming a circular pattern corresponding to foot motion events in Fig. 9(a). When the cart is kicked at $t = 2.4$ s (Fig. 9(a-3)), the contexts form a new cluster (upper left of Fig. 9(d)). Subsequently, a distinct cluster appears (bottom left of Fig. 9(d)) at Fig. 9(a-4), with fewer embeddings as the policy quickly handles the situation. The embeddings then return to the original circular pattern after the robot lands safely, resuming normal locomotion at $t = 3.2$ s.

[6]https://youtu.be/tFI8xDwF4bU

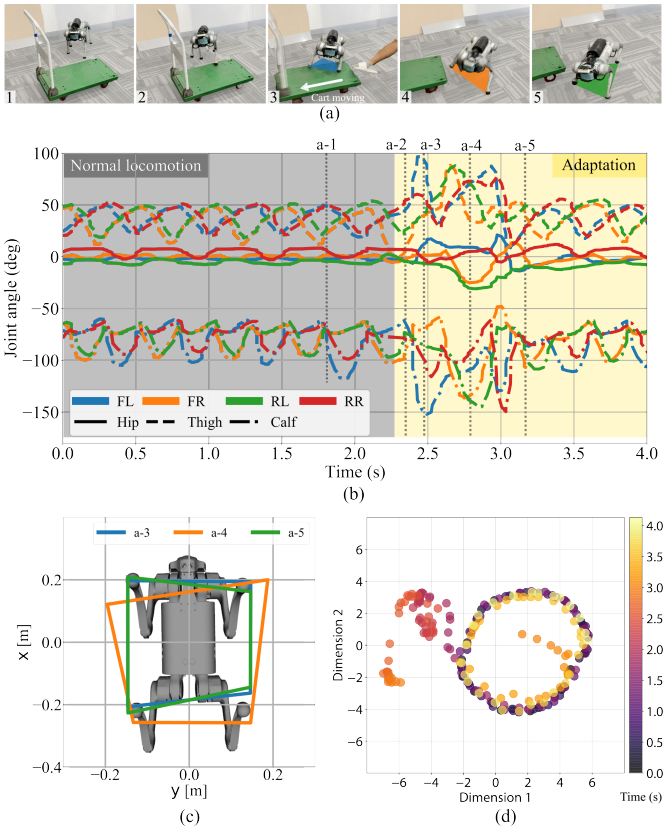


Fig. 9. Adaptation to sudden changes of foothold. (a) The robot is externally disturbed by quickly removing the platform it is stepping on. (b) An abrupt change in the robot's perception made the controller rapidly alter the robot's joints at around $t = 2.5$ s to (c) enlarge the robot's support polygon for ensuring a safe and stable landing. (d) A 2D embedding visualization using *pairwise controlled manifold approximation projection* (PaCMAP) [58] shows how the multi-modal context dynamically changes over time and capture changes in the environment, providing informative contexts to swiftly adapt the policy.

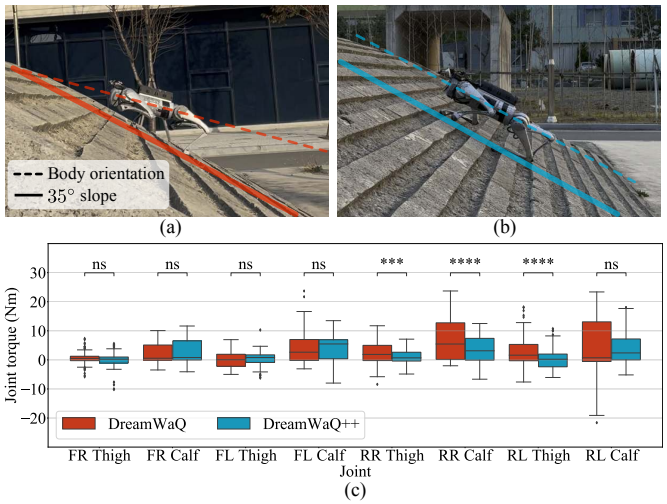


Fig. 10. Comparison of torque exertions when climbing a 35° slope using (a) DreamWaQ and (b) DreamWaQ++. The annotations on top of the boxplot in (c) indicate the significance level measured using a paired t-test method

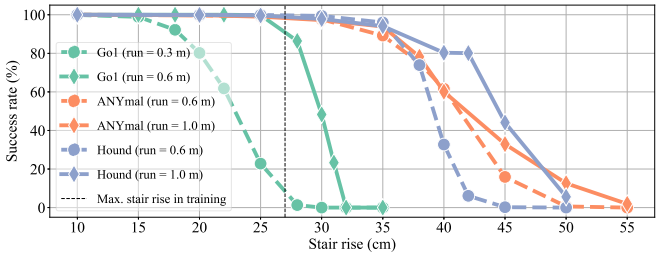


Fig. 11. Success rates for climbing different obstacles using various quadrupedal robots. We trained DreamWaQ++ for Unitree Go1, ANYmal-C, and Hound. The maximum stair rise imposed during training for all robots is 27 cm.

2) Climbing Over a Steep Slope[7]*:* Fig. 10(c) compares the torque exertion of controllers trained with DreamWaQ and DreamWaQ++ (see Figs. 10(a) and (b)). Both controllers were trained exclusively on rough slopes up to 10°, while in this experiment, the robot was tasked with climbing a 35° slope. DreamWaQ's policy attempts to maintain a flat body orientation (Fig. 10(b)), as it was trained with a blind locomotion controller to generalize across various terrains. Although broadly effective, this approach results in conservative behavior and greater torque exertion on the rear legs (Fig. 10(a)) to uphold the flat base pose.

In contrast, the DreamWaQ++ policy employs a crawling gait with a reduced body height relative to the slope surface, aligning the robot's base orientation with the slope inclination, which enhances stability and reduces torque on the rear legs, as shown in Fig. 10(b). DreamWaQ++'s terrain perception enables flexible gait adaptation rather than a conservative approach. Notably, rear leg torque in DreamWaQ++ is approximately 1.5 times lower than in DreamWaQ, demonstrating DreamWaQ++'s superior out-of-distribution adaptation.

F. Scalability to Other Platforms[8]

We evaluated the scalability of DreamWaQ++ by applying it to legged robots with different morphologies and sizes by training controllers for ANYmal-C [59] and Hound [60] robots. Note that we used the same reward functions and its corresponding weights. Particularly, we adjusted only the robot-specific parameters such as the robot models, the motor operation limit, motor stiffness and damping parameters, desired foot clearance, desired body height, and maximum foot contact force.

Fig. 11 presents the success rates of three different robots climbing stairs with varying runs and rises. For the Go1 robot, we used run values of 0.3 m and 0.6 m, while for ANYmal-C and Hound, we used 0.6 m and 1.0 m runs to accommodate their longer trunk sizes. Success rates were measured from 1,000 simulated robots, defined as the percentage reaching the last stair within 10 s.

As expected, Hound's extensive joint operation range enabled it to traverse more challenging terrains than the Go1 and ANYmal-C, achieving an 80% success rate on 42 cm-high stairs. Notably, the robots were only trained on obstacles

[7]https://youtu.be/lESpqB5LTnI
[8]https://youtu.be/x3RSJRUr7ro

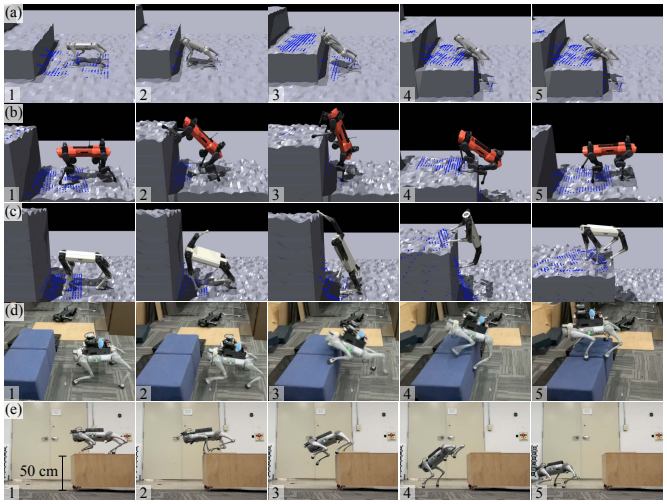


Fig. 12. We trained DreamWaQ++ for (a) Unitree Go1, (b) ANYmal-C, and (c) Hound to climb obstacles with a height of 0.6 m, 1.0 m, 1.5 m, respectively. (d) A real world experiment was conducted using a Go1 robot with a 2.5 kg payload on top of it.

up to 27 cm in height, underscoring the strong adaptability of DreamWaQ++. The controller's exploratory training maximizes hardware capability, with reward functions invariant to hardware differences.

This evaluation highlights DreamWaQ++'s versatility across various legged robots. Despite the sensitivity of deep RL algorithms to reward parameters, DreamWaQ++ demonstrated easy adaptability to different platforms without additional tuning.

G. Overcoming Large Obstacles[9]

Recently, there has been increasing interest in training quadrupedal robots for complex tasks such as jumping and leaping [38], [61]. These skills require robots to maximize actuator limits and perform highly agile motions. We further assessed the scalability of DreamWaQ++ for overcoming obstacles higher than the robot itself. We trained the Go1, ANYmal-C, and Hound robots in environments with extreme obstacle heights up to 0.6 m, 1.0 m, and 1.5 m, respectively. To enable learning of this agile skill, we made two modifications: (1) reducing the velocity tracking reward scale from 1.0 to 0.1, and (2) increasing the versatility gain scaling in the loss function from 0.1 to 0.2. Snapshots of the learned obstacle-climbing motions in simulation and real-world scenarios are shown in Fig. 12.

These modifications were complementary; without scaling up the versatility gain, reducing the velocity tracking reward alone led to a policy that resisted forward movement, while increasing versatility gain alone resulted in a conservative gait lacking skill discovery. Together, these adjustments allowed the policy to learn flexible gaits, yielding the agility required for complex tasks such as parkour.

Results in Fig. 12 illustrate how controllers on different robots produce distinct motions for overcoming large obstacles. In Fig. 12(a), Go1 uses a jumping motion due to its small

[9]https://youtu.be/pwwwmnd3Xnc

size. In contrast, ANYmal-C initially contacts the obstacle wall with its front legs (Fig. 12(b-2)), then uses them as anchors to climb (Fig. 12(b-3)). ANYmal-C's large joint range and torque limits enable it to climb obstacles up to 1.5 m. Hound swings its right front leg widely to anchor on top, then propels itself upward with the rear legs (Fig. 12(c-4)).

In a real-world experiment shown in Fig. 12(d), we validated the sim-to-real robustness of the controller. A Go1 robot with an additional 2.5 kg payload successfully climbed a 41 cm obstacle, a soft sofa block representing a deformable surface not encountered in training.

In Fig. 12(e), the robot was placed on a 50 cm stage, preventing simple terrain probing. Upon command, the robot leaped forward, avoiding rear leg collisions with the stage edges. This leap was facilitated by the velocity tracking relaxation, allowing the robot to pause at the edge (Fig. 12(e-2)) before initiating the leap. The robot kicks forward with its front legs, propels with the rear legs (Fig. 12(e-3)), and folds the rear legs to avoid collision (Fig. 12(e-4)). This demonstrates the versatility of the learned controller, serving as a prior adaptable to complex tasks.

VI. Ablation Study

A. Quantitative Analysis on Policy Performance

We evaluated the effectiveness of various multi-modal encoding strategies for learning a context representation that captures the dynamics of a quadrupedal robot. This ablation study examines the impact of various encoding methods on both policy performance and training dynamics. We trained several variants of the multi-modal context encoder, each using a different strategy as follows:

- **No memory.** This variant does not use any memory structure; the encoder processes only the current proprioceptive and exteroceptive measurements.
- **Implicit memory.** This variant employs an LSTM to implicitly encode the history of proprioceptive and exteroceptive inputs without explicitly storing past information.
- **No latent fusion.** In this variant, proprioceptive and exteroceptive features are simply concatenated, with no dedicated fusion mechanism—treating them as separate input modalities.
- **DreamWaQ++ w/o contrastive loss.** This variant trains DreamWaQ++ without the contrastive loss which is used to align the latent representations of proprioceptive and exteroceptive features.
- **DreamWaQ++ w/o versatility gain.** This variant trains DreamWaQ++ without the versatility gain which encourages the policy to explore a wider range of actions and behaviors.

We evaluated all variants in a simulated environment using 1,000 robots; each of the robots commanded to walk forward for 20 s. The evaluations were conducted in three test environments:

- **Stairs-easy**. Continuous stairs with a step height of 10 cm.
- **Stairs-hard**. Continuous stairs with a step height of 20 cm.

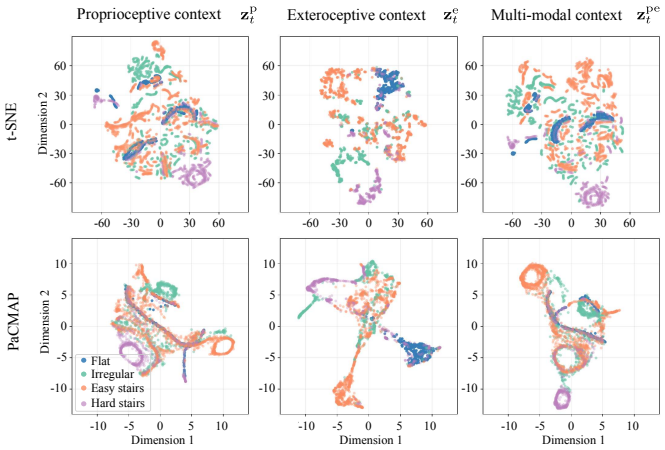


Fig. 13. Embedding visualization of the multi-modal context encoded by the proposed context encoder in different environments using PacMAP [58]. The highly disentangled multi-modal context serves as an informative prior for informing about the environment to the policy.

- **Discrete**. Discrete obstacle terrain with obstacle heights ranging from 5 cm to 25 cm.

For each variant, we report the average success rate, the average distance traveled, and the total number of calf collisions across the test environments. The results are summarized in Table IV.

The results presented in Table IV validate that DreamWaQ++ consistently outperforms all other variants across all test environments and evaluation metrics. Notably, we observed that the latent fusion mechanism plays a critical role: the variant without latent fusion exhibited the lowest success rates and shortest traveled distances despite not having the highest number of calf collisions. This suggests that latent fusion is crucial for maintaining robustness and stability, especially in situations where proprioceptive and exteroceptive inputs are inconsistent.

We also observed that the memory plays a significant role in collision avoidance. The variant without any memory structure resulted in the highest number of calf collisions across all environments. While the implicit memory variant performed worse than DreamWaQ++, it still achieved a reasonable level of performance, suggesting that even approximate temporal encoding can help improve safety.

Furthermore, in terms of regularization, the ablation study highlights the importance of both the contrastive loss and the versatility gain in DreamWaQ++. Removing the contrastive loss led to modest declines in all metrics, underscoring its value in aligning the latent spaces of proprioceptive and exteroceptive inputs, thereby improving policy learning. In contrast, removing the versatility gain resulted in more substantial performance degradation. This component encourages the policy to explore a broader range of behaviors. Without it, the policy tends to converge to a narrower action distribution, reducing adaptability and robustness in diverse terrains.

B. Embedding Analysis

The visualized embeddings in Fig. 13 reveal a distinctive ellipsoidal pattern in the proprioceptive context $\mathbf{z}_t^{\mathrm{p}}$, attributed

TABLE IV

ABLATION STUDY ON THE IMPACT OF DIFFERENT ARCHITECTURES AND REGULARIZATION LOSSES ON THE POLICY PERFORMANCE. THE BEST, SECOND BEST, AND WORST PERFORMING METHODS FOR EACH TEST ENVIRONMENT AND METRIC ARE HIGHLIGHTED ACCORDINGLY.

Method	Success rates [%] ↑			Traveled distance [m] ↑			Calf collisions ↓		
	Stairs-easy	Stairs-hard	Discrete	Stairs-easy	Stairs-hard	Discrete	Stairs-easy	Stairs-hard	Discrete
Architecture									
No memory	98.7	85.1	99.6	8.4 ± 1.2	5.8 ± 3.5	11.5 ± 0.9	30.1 ± 0.8	35.2 ± 10.2	27.4 ± 20.1
Implicit memory	99.1	92.5	99.5	9.1 ± 0.7	8.8 ± 1.3	9.8 ± 0.5	24.1 ± 5.1	37.8 ± 4.3	15.9 ± 2.7
No latent fusion	90.2	60.7	94.3	6.5 ± 1.3	3.4 ± 3.1	9.3 ± 0.8	22.3 ± 2.1	25.3 ± 2.3	13.5 ± 2.6
Regularization									
w/o contrastive loss	98.4	93.1	99.2	9.8 ± 0.5	8.9 ± 1.4	12.2 ± 0.4	12.3 ± 1.8	23.7 ± 0.7	8.4 ± 0.9
w/o versatility gain	98.2	89.4	99.3	9.7 ± 0.3	8.8 ± 1.5	12.4 ± 0.5	15.2 ± 0.5	31.5 ± 1.1	11.3 ± 1.4
DreamWaQ++	99.5	97.8	99.8	10.3 ± 0.8	9.5 ± 1.2	12.5 ± 0.1	10.5 ± 0.4	18.3 ± 2.5	5.8 ± 0.2

to the dynamic motion of the robot's feet, as $\mathbf{z}_t^{\mathrm{p}}$ integrates various proprioceptive measurements. Notably, the ellipsoid's size diminishes on more challenging terrains, likely reflecting the robot's foot swing period. The controller adjusts by orchestrating quicker foot swings over difficult terrains, ensuring frequent ground contact to enhance locomotion stability.

In contrast, the exteroceptive context $\mathbf{z}_t^{\mathrm{e}}$ displays clearer inter-class separation among environments compared to $\mathbf{z}_t^{\mathrm{p}}$. However, similarities among some embeddings likely result from simplified exteroceptive input, primarily 3D voxels ahead of the robot. The exteroceptive encoder effectively captures relevant geometric features, filtering out unnecessary details from raw 3D points.

For the multi-modal context $\mathbf{z}_t^{\mathrm{pe}}$, which fuses proprioceptive and exteroceptive inputs, Fig. 13 displays distinct clusters based on terrain difficulty. Embeddings from `flat`, `easy stairs`, and `irregular` terrains are partially clustered near the origin due to similarities in obstacle height, despite differences in obstacle placement and density. However, significant disentanglement is evident within the `easy stairs` and `irregular` terrain clusters, facilitated by proprioceptive information capturing terrain properties under the robot.

Simultaneously, the circular pattern from $\mathbf{z}_t^{\mathrm{p}}$ persists in $\mathbf{z}_t^{\mathrm{pe}}$, aiding in the correction of unreliable exteroceptive data. This feature contributes to the clear separation between `easy stairs` and `irregular` terrains, underscoring exteroception's auxiliary role in adapting the robot's gait to avoid upcoming obstacles.

C. Latent Modulation for Gait Control

Fig. 14 illustrates the distribution of each latent feature as the robot traversed irregular terrains. Embedding indices from `1` to `32` correspond to $\mathbf{z}_t^{\mathrm{p}}$, and indices from `33` to `64` correspond to $\mathbf{z}_t^{\mathrm{e}}$. While $\mathbf{z}_t^{\mathrm{p}}$ features display a uniform distribution, $\mathbf{z}_t^{\mathrm{e}}$ contains four embeddings with distinct scaling differences compared to other exteroceptive features. This finding raises the question: *Do these four exteroceptive embedding features correlate with the robot's foot swing motion?*

To address this question, we conducted an experiment by modulating these four $\mathbf{z}_t^{\mathrm{e}}$ features. The results in Fig. 14(b) indicate a clear trend: *scaling up the latent feature decreases gait frequency while increasing gait height, and vice versa*. The resulting gait pattern resembles that used in stair-climbing scenarios, suggesting that the multi-modal context encoder

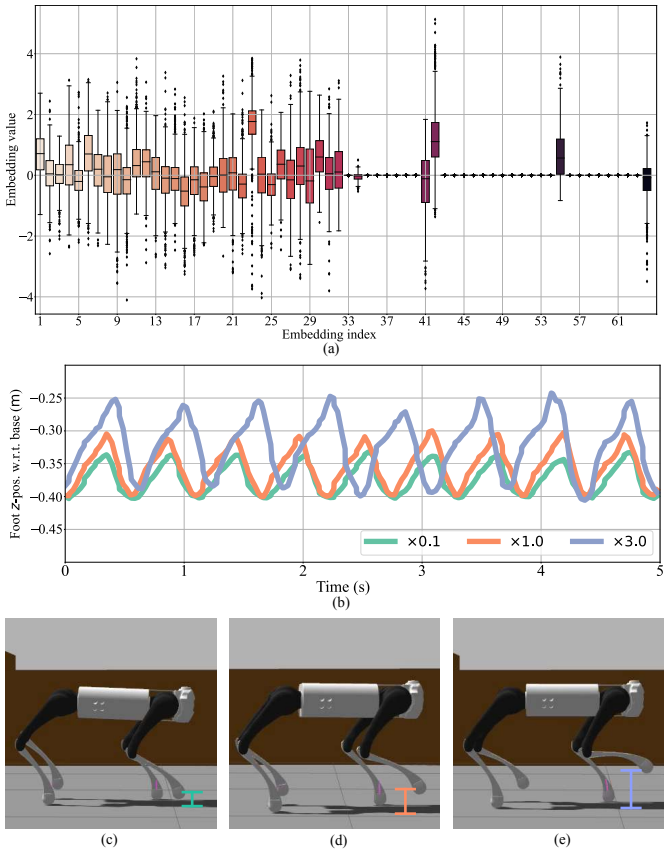


Fig. 14. (a) Boxplots of the multi-modal contexts in an irregular terrain, showing the distribution of embeddings activation from the multi-modal context and highlighting the contrast between activations in the exteroceptive context. Latent modulation on some exteroceptive embedding proportionally affects the gait height. (b) The robot exhibits different locomotion styles when particular latent variables (41, 42, 55, and 64th embeddings) were scaled with (c) 0.1, (d) 1.0, and (e) 3.0 times of its original value.

effectively activates key latent variables that directly influence gait. A limitation of this approach is that the activated latent variables may vary across training seeds, as they are learned in an unsupervised manner with multiple randomizations. However, upon convergence, these critical latent features reliably emerge within the latent space.

D. Cross-Modal Feature Correlation

Cross-modal correlations between context vectors are visualized as heatmap plots in Fig. 15, obtained by computing

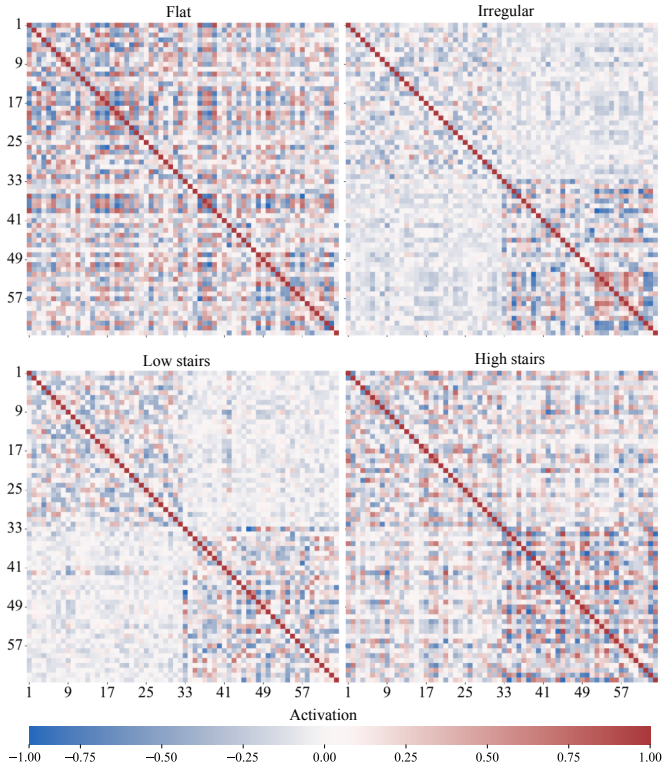


Fig. 15. Cross-modal feature correlations between proprioceptive and exteroceptive context vectors. The heatmap shows the correlation between the embedding features, where low correlation indicates substantial mismatches between proprioceptive and exteroceptive information.

cross-correlations between embedding features. Low cross-modal correlation on irregular terrains indicates substantial mismatches between proprioceptive and exteroceptive information. Additionally, stronger correlations are observed within the exteroceptive data due to its direct observability, as opposed to proprioception, which can only approximate the terrain beneath the robot. In contrast, higher cross-modal correlations on flat terrain result from the similar predictions of terrain structure provided by both proprioceptive and exteroceptive inputs.

Additionally, it can be observed that cross-modal correlation is higher on high stairs than on low stairs. This is because, on high stairs, the robot's feet are more likely to be in contact with the surface, providing more accurate proprioceptive information. In contrast, on low stairs, the robot's feet are more often in the air, leading to less accurate proprioceptive feedback. This discrepancy between proprioceptive and exteroceptive information results in lower cross-modal correlation on low stairs compared to high stairs.

E. Locomotion under Exteroception Failure

Fig. 16 shows the emergent behaviors of DreamWaQ++ when climbing stairs. This experiment ablates foot swing adaptation by providing the policy with a white noise input. The robot is commanded to climb the stairs under both conditions. The red arrows in Fig. 16(a) illustrate the foot swing motion of the robot when exteroception functions normally, allowing the robot to adapt its foot swing trajectory to climb two stairs

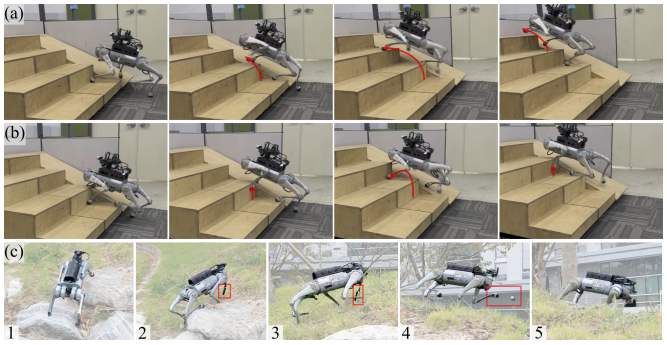


Fig. 16. Locomotion under exteroception failures. (a) The robot swiftly adapts its gait when climbing the stairs under normal exteroception condition. Meanwhile, (b) the robot's foot collides with the stairs and yield a foot-trapping reflex when white noise is provided as exteroception input. (c) In an extreme failure case, the robot adapts to the detached camera by making contact with the ground using its feet and knees to ensure a stable pose.

at once. In contrast, when the robot receives white noise input, as in Fig. 16(b), the robot's foot tends to collide with the stairs. However, the robot adapts with a foot-trapping reflex, dragging its foot along the stair's vertical surface before placing it on the next step.

Additionally, the experiment in Fig. 16(c) shows the robot climbing large rocks by swinging its feet extensively, resulting in strong vibrations. These vibrations eventually caused the camera to fall, yielding depth point cloud measurements with significant calibration error. The problem worsens when the camera becomes completely detached from the robot (see Fig. 16(c-4)). Under these conditions, the controller no longer receives new data streams. Interestingly, the robot adapts its gait to move by making contact with the ground using its feet and knees (Fig. 16(c-5)). This motion results in a more stable pose, with the robot adapting additional ground contacts to handle high-risk locomotion when exteroception becomes extremely unreliable.

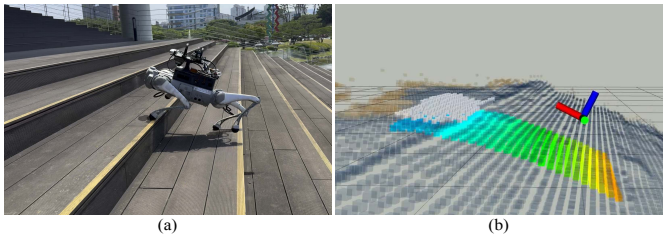
F. Terrain Reconstruction

We decoded the latent features from the context encoder recorded during the asynchronous stair race experiment to reconstruct the terrain map, as shown in Fig. 17. The reconstructed terrain map resembles the ground truth terrain map constructed with [62].

However, the reconstructed map is less accurate than the ground truth due to three factors:

1) The robot's exteroception is limited to front-facing 3D points, and the encoder's memory is not long-term.
2) The encoder-decoder structure lacks residual connections, as in [38], focusing the latent representation on relevant features.
3) Variational autoencoder regularization in the latent space prioritizes feature disentanglement over precise reconstruction.

Nonetheless, this slight reduction in reconstruction accuracy is acceptable, as the primary goal of the context encoder is to learn a structured representation for control rather than to fully reconstruct the terrain map.



(a) (b)

Fig. 17. (a) An snapshot of the scene during the asynchronous stair race using DreamWaQ++. (b) The terrain map is reconstructed from the recorded latent features and the ground truth terrain map is constructed using [62]. The white points are the forward 3D scan input, while the reconstruction points surrounding the robots are colored based on the height relative to the robot's base.

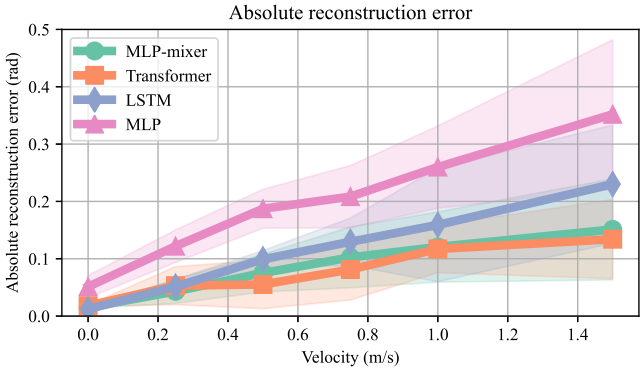


Fig. 18. Future state prediction error using different models and varying robot velocities. The MLP-mixer model used in DreamWaQ++ shows comparable performance to the Transformer model while being more lightweight and computationally efficient.

We further evaluated the impact of the backbone sequence model on the controller's performance by comparing the controller's performance across different sequence models. Performance was assessed by measuring the accuracy of future joint position predictions, as shown in Fig. 18.

The results in Fig. 18 show that the proposed MLP-mixer architecture performs comparably to the Transformer model. The MLP-mixer offers the advantage of being more lightweight and computationally efficient than the Transformer. As indicated by the curve, the prediction error increases with the robot's velocity due to the more dynamic nature of the motion, which requires higher accuracy in future state predictions.

G. State Estimation Accuracy

We evaluate the proprioceptive-based state estimation accuracy of DreamWaQ++ which utilizes an MLP-mixer architecture, and compare it with other common approaches such as fully connected networks [13] and RNN-based architectures [12]. Since accurate state estimation is crucial for constructing the hierarchical exteroceptive memory, we also evaluate how different state estimation architectures impact the reconstruction quality for the exteroceptive input.

We conducted a controlled experiment in a simulated stair environment, where the robot traverses a staircase with step heights uniformly sampled from the range $[0.1,\ 0.3]$ m. To isolate perception performance, the robot was commanded to move forward at a constant speed of 0.5 m/s. Subsequently, we measured the absolute reconstruction error across 187 height points in the robot's surrounding area. The results are summarized in Table V. All measurements represent the mean over 5 s of continuous walking using $1{,}000$ simulated robots.

TABLE V
STATE ESTIMATION AND TERRAIN RECONSTRUCTION ACCURACY.

Architecture	**Estimation error** ↓			**Reconstruction error** [m] ↓
	v_x [m/s]	v_y [m/s]	v_z [m/s]	
MLP	0.0571	0.0739	0.0715	0.1978
RNN	0.0497	0.0613	0.0514	0.1253
MLP-mixer	0.0363	0.0331	0.0385	0.0674

The results indicate that the MLP-mixer architecture outperforms both the fully connected MLP and RNN-based architectures in terms of state estimation accuracy, achieving the lowest error across all velocity components. This superior performance is attributed to the MLP-mixer's ability to effectively capture temporal dependencies and interactions among different proprioceptive modalities, resulting in a more accurate representation of the robot's state. Furthermore, the improved state estimation contributes to a more precise reconstruction of the surrounding terrain, as shown by the lowest reconstruction error among the evaluated architectures.

VII. CONCLUSION

We have proposed DreamWaQ++, a resilient yet lightweight obstacle-aware locomotion controller that enhances the resilience of its precursor, DreamWaQ [13]. DreamWaQ++ reduces the need for expensive onboard computation and increases versatility by incorporating 3D point cloud data as an exteroceptive modality. Our experiments demonstrate that the proposed controller exhibits enhanced explainability, opening possibilities for integration with model-based counterparts or higher-level planning modules to enable greater autonomy. A promising avenue for future work involves integrating an active tilting mechanism into the camera mount using an additional servo motor. By simultaneously learning both locomotion and camera tilting, this approach could yield a controller that actively maximizes its observability, resembling the active sensing behavior of animals.

REFERENCES

[1] C. Gehring, P. Fankhauser, L. Isler, R. Diethelm, S. Bachmann, M. Potz, L. Gerstenberg, and M. Hutter, "ANYmal in the field: Solving industrial inspection of an offshore HVDC platform with a quadrupedal robot," in *Field and Serv. Robot.*, G. Ishigami and K. Yoshida, Eds. Singapore: Springer, 2021, pp. 247–260.

[2] M. Tranzatto, T. Miki, M. Dharmadhikari, L. Bernreiter, M. Kulkarni, F. Mascarich, O. Andersson, S. Khattak, M. Hutter, R. Siegwart *et al.*, "CERBERUS in the DARPA subterranean challenge," *Sci. Robot.*, vol. 7, no. 66, p. eabp9742, 2022.

[3] S. Hong, Y. Um, J. Park, and H.-W. Park, "Agile and versatile climbing on ferromagnetic surfaces with a quadrupedal robot," *Sci. Robot.*, vol. 7, no. 73, p. eadd1017, 2022.

[4] P. Arm, G. Waibel, J. Preisig, T. Tuna, R. Zhou, V. Bickel, G. Ligeza, T. Miki, F. Kehl, H. Kolvenbach *et al.*, "Scientific exploration of challenging planetary analog environments with a team of legged robots," *Sci. Robot.*, vol. 8, no. 80, p. eade9548, 2023.

[5] F. Jenelten, R. Grandia, F. Farshidian, and M. Hutter, "TAMOLS: Terrain-aware motion optimization for legged systems," *IEEE Trans. Robot.*, vol. 38, no. 6, pp. 3395–3413, 2022.

[6] C. D. Bellicoso, F. Jenelten, P. Fankhauser, C. Gehring, J. Hwangbo, and M. Hutter, "Dynamic locomotion and whole-body control for quadrupedal robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2017, pp. 3359–3365.

[7] C. Gehring, C. D. Bellicoso, P. Fankhauser, S. Coros, and M. Hutter, "Quadrupedal locomotion using trajectory optimization and hierarchical whole body control," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 4788–4794.

[8] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, "MIT Cheetah 3: Design and control of a robust, dynamic quadruped robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2018, pp. 2245–2252.

[9] S. Hong, J.-H. Kim, and H.-W. Park, "Real-time constrained nonlinear model predictive control on SO(3) for dynamic legged locomotion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2020, pp. 3982–3989.

[10] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Sci. Robot.*, vol. 5, no. 47, p. eabc5986, 2020.

[11] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid motor adaptation for legged robots," in *Robot. Sci. Syst.*, 2021.

[12] G. Ji, J. Mun, H. Kim, and J. Hwangbo, "Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4630–4637, 2022.

[13] I. M. A. Nahrendra, B. Yu, and H. Myung, "DreamWaQ: Learning robust quadrupedal locomotion with implicit terrain imagination via deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 5078–5084.

[14] S. Choi, G. Ji, J. Park, H. Kim, J. Mun, J. H. Lee, and J. Hwangbo, "Learning quadrupedal locomotion on deformable terrain," *Sci. Robot.*, vol. 8, no. 74, p. eade2256, 2023.

[15] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Sci. Robot.*, vol. 7, no. 62, p. eabk2822, 2022.

[16] S. Gangapurwala, M. Geisert, R. Orsolino, M. Fallon, and I. Havoutis, "RLOC: Terrain-aware legged locomotion using reinforcement learning and optimal control," *IEEE Trans. Robot.*, vol. 38, no. 5, pp. 2908–2927, 2022.

[17] R. Yang, M. Zhang, N. Hansen, H. Xu, and X. Wang, "Learning vision-guided quadrupedal locomotion end-to-end with cross-modal transformers," in *Proc. Int. Conf. Learn. Represent.*, 2022.

[18] C. S. Imai, M. Zhang, Y. Zhang, M. Kierebiński, R. Yang, Y. Qin, and X. Wang, "Vision-guided quadrupedal locomotion in the wild with multi-modal delay randomization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2022, pp. 5556–5563.

[19] A. Agarwal, A. Kumar, J. Malik, and D. Pathak, "Legged locomotion in challenging terrains using egocentric vision," in *Proc. PMLR Conf. Robot Learning*, 2022, pp. 403–415.

[20] R. Yang, G. Yang, and X. Wang, "Neural volumetric memory for visual locomotion control," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 1430–1440.

[21] M. Bloesch, C. Gehring, P. Fankhauser, M. Hutter, M. A. Hoepflinger, and R. Siegwart, "State estimation for legged robots on unstable and slippery terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2013, pp. 6058–6064.

[22] M. Camurri, M. Ramezani, S. Nobili, and M. Fallon, "Pronto: A multi-sensor state estimator for legged robots in real-world scenarios," *Frontiers in Robotics and AI*, vol. 7, p. 68, 2020.

[23] J.-H. Kim, S. Hong, G. Ji, S. Jeon, J. Hwangbo, J.-H. Oh, and H.-W. Park, "Legged robot state estimation with dynamic contact event information," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 6733–6740, 2021.

[24] Y. Kim, B. Yu, E. M. Lee, J.-H. Kim, H.-W. Park, and H. Myung, "STEP: State estimator for legged robots using a preintegrated foot velocity factor," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4456–4463, 2022.

[25] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 3019–3026, 2018.

[26] H. Lim, M. Oh, and H. Myung, "Patchwork: Concentric zone-based region-wise ground segmentation with ground likelihood estimation using a 3D LiDAR sensor," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 6458–6465, 2021.

[27] S. Lee, H. Lim, and H. Myung, "Patchwork++: Fast and robust ground segmentation solving partial under-segmentation using 3D point cloud," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2022, pp. 13 276–13 283.

[28] M. Oh, E. Jung, H. Lim, W. Song, S. Hu, E. M. Lee, J. Park, J. Kim, J. Lee, and H. Myung, "TRAVEL: Traversable ground and above-ground object segmentation using graph representation of 3D LiDAR scans," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 7255–7262, 2022.

[29] T. Miki, L. Wellhausen, R. Grandia, F. Jenelten, T. Homberger, and M. Hutter, "Elevation mapping for locomotion and navigation using GPU," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2022, pp. 2273–2280.

[30] M. K. Ho, D. Abel, C. G. Correa, M. L. Littman, J. D. Cohen, and T. L. Griffiths, "People construct simplified mental representations to plan," *Nature*, vol. 606, no. 7912, pp. 129–136, 2022.

[31] S. Di Marco, A. Tosoni, E. C. Altomare, G. Ferretti, M. G. Perrucci, and G. Committeri, "Walking-related locomotion is facilitated by the perception of distant targets in the extrapersonal space," *Sci. Rep.*, vol. 9, no. 1, p. 9884, 2019.

[32] P. Chopra, D. M. Castelli, and J. B. Dingwell, "Cognitively demanding object negotiation while walking and texting," *Sci. Rep.*, vol. 8, no. 1, pp. 1–13, 2018.

[33] T. Killeen, C. S. Easthope, L. Demkó, L. Filli, L. Lőrincz, M. Linnebank, A. Curt, B. Zörner, and M. Bolliger, "Minimum toe clearance: Probing the neural control of locomotion," *Sci. Rep.*, vol. 7, no. 1, p. 1922, 2017.

[34] "Unitree Go1," accessed on 2022.08.24. [Online]. Available: https://m.unitree.com/products/go1

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

[36] A. Loquercio, A. Kumar, and J. Malik, "Learning visual locomotion with cross-modal supervision," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 7295–7302.

[37] X. Cheng, K. Shi, A. Agarwal, and D. Pathak, "Extreme parkour with legged robots," *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 11 443–11 450, 2023.

[38] D. Hoeller, N. Rudin, D. Sako, and M. Hutter, "ANYmal parkour: Learning agile navigation for quadrupedal robots," *Sci. Robot.*, vol. 9, no. 88, p. eadi7566, 2024.

[39] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," in *Robot. Sci. Syst.*, 2022.

[40] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[41] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Adv. Neural Inf. Process. Syst.*, vol. 30, pp. 5998–6008, 2017.

[42] L. C. Melo, "Transformers are meta-reinforcement learners," in *Proc. PMLR Int. Conf. Mach. Learn.*, 2022, pp. 15 340–15 359.

[43] C. Li, S. Blaes, P. Kolev, M. Vlastelica, J. Frey, and G. Martius, "Versatile skill control via self-supervised adversarial imitation of unlabeled mixed motions," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 2944–2950.

[44] K. Rakelly, A. Gupta, C. Florensa, and S. Levine, "Which mutual-information representation learning objectives are sufficient for control?" *Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 26 345–26 357, 2021.

[45] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. PMLR Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.

[46] K. Rakelly, A. Zhou, C. Finn, S. Levine, and D. Quillen, "Efficient off-policy meta-reinforcement learning via probabilistic context variables," in *Proc. PMLR Int. Conf. Mach. Learn.*, 2019, pp. 5331–5340.

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

[48] D. Hoeller, N. Rudin, C. Choy, A. Anandkumar, and M. Hutter, "Neural scene representation for locomotion on structured terrain," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 8667–8674, 2022.

[49] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 652–660.

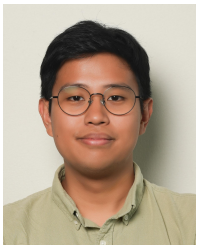
[50] I. O. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit *et al.*, "MLP-mixer: An all-MLP architecture for vision," *Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 24 261–24 272, 2021.

[51] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "β – VAE: Learning basic visual concepts with a constrained variational framework," in *Proc. Int. Conf. Learn. Represent.*, 2017.

[52] Z. Fu, X. Cheng, and D. Pathak, "Deep whole-body control: Learning a unified policy for manipulation and locomotion," in *Proc. PMLR Conf. Robot Learning*, 2022, pp. 138–149.

[53] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa *et al.*, "Isaac Gym: High

performance GPU-based physics simulation for robot learning," *Adv. Neural Inf. Process. Syst. Track on Data. and Bench.*, 2021.
[54] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Proc. PMLR Conf. Robot Learning*, 2021, pp. 91–100.
[55] L. Campanaro, D. De Martini, S. Gangapurwala, W. Merkt, and I. Havoutis, "Roll-Drop: Accounting for observation noise with a single parameter," in *Prof. of Learn. Dynam. and Control Conf.*, 2023, pp. 718–730.
[56] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct LiDAR-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053–2073, 2022.
[57] S. Kareer, N. Yokoyama, D. Batra, S. Ha, and J. Truong, "ViNL: Visual navigation and locomotion over obstacles," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 2018–2024.
[58] Y. Wang, H. Huang, C. Rudin, and Y. Shaposhnik, "Understanding how dimension reduction tools work: an empirical approach to deciphering t-SNE, UMAP, TriMAP, and PaCMAP for data visualization," *The Journal of Machine Learning Research*, vol. 22, no. 1, pp. 9129–9201, 2021.
[59] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch *et al.*, "ANYmal – A highly mobile and dynamic quadrupedal robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2016, pp. 38–44.
[60] Y.-H. Shin, S. Hong, S. Woo, J. Choe, H. Son, G. Kim, J.-H. Kim, K. Lee, J. Hwangbo, and H.-W. Park, "Design of KAIST HOUND, a quadruped robot platform for fast and efficient locomotion with mixed-integer nonlinear optimization of a gear train," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 6614–6620.
[61] Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao, "Robot parkour learning," in *Proc. PMLR Conf. Robot Learning*, 2023, pp. 73–92.
[62] M. Oh, B. Yu, I. M. A. Nahrendra, S. Jang, H. Lee, D. Lee, S. Lee, Y. Kim, K. C. Marsim, H. Lim, and H. Myung, "TRIP: Terrain traversability mapping with risk-aware prediction for enhanced online quadrupedal robot navigation," *arXiv:2411.17134*, 2024.



I Made Aswin Nahrendra received the B.S. and M.S. degrees in electrical engineering from Bandung Institute of Technology, Bandung, Indonesia, in 2018 and 2019, respectively. He received the Ph.D. degree in robotics program and electrical engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2024. He was a postdoctoral fellow in the Information & Electronics Research Institute, KAIST, Daejeon, Republic of Korea from 2024 to 2025. He is currently a research scientist in the Physical AI Team, AI Research Center at KRAFTON. His research interests include reinforcement learning, control, and robot learning for legged robots.

Byeongho Yu received the M.S. and Ph.D. degrees in Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2021 and 2025, respectively. He is currently the CEO of URobotics. His research interests are primarily in the field robotics, including visual-inertial-leg odometry and state estimation for legged robots, with a particular focus on robust autonomy in unstructured environments.

Minho Oh received the B.S. degree in convergence engineering from Daegu Gyeongbuk Institute of Science and Technology (DGIST), Daegu, Korea, in 2020, and the M.S. degree in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2021. He is currently pursuing the Ph.D. degree in electrical engineering, KAIST, and leads the spatial intelligence team as the CTO of URobotics. His research interests include SLAM and terrain traversability mapping for enhanced autonomous navigation.

Dongkyu Lee received the B.S. degree in Electrical and Computer Engineering from the University of Seoul, Seoul, Korea, in 2021, and the M.S. degree in the Robotics Program from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2023. He is currently pursuing the Ph.D. degree in Electrical Engineering at KAIST and is currently the CTO of URobotics Corp. His research interests include robot navigation, path planning, decision-making, and robot learning.

Seunghyun Lee received the B.S. and M.S. degrees in the school of electrical engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Republic of Korea, in 2022 and 2024, respectively. He is currently pursuing the Ph.D. degree at Urban Robotics Lab in the school of electrical engineering, KAIST. His research interests include legged robot locomotion and deep reinforcement learning.

Hyeonwoo Lee received the B.S. and M.S. degrees in mechanical engineering from Yonsei University, Seoul, Republic of Korea, in 2020 and 2022, respectively. He is currently pursuing the Ph.D. degree in electrical engineering from the Korea Advanced Institute of Science and Technology with the Urban Robotics Lab. His research interests include reinforcement learning, motion and path planning for quadruped robots, and visual navigation.

Hyungtae Lim received the B.S. degree in mechanical engineering, and M.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Republic of Korea, in 2018, 2020, and 2023, respectively. He was a postdoctoral fellow in the Information & Electronics Research Institute, KAIST, Daejeon, Republic of Korea from 2023 to 2024. He is currently a postdoctoral associate in the Laboratory for Information & Decision Systems (LIDS), Massachusetts Institute of Technology (MIT), Cambridge, MA, USA. His research interests include SLAM (simultaneous localization and mapping), 3D registration, 3D perception, long-term map management, deep learning, and spatial AI.

Hyun Myung received the B.S., M.S., and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1992, 1994, and 1998, respectively. He was a Senior Researcher with the Electronics and Telecommunications Research Institute, Daejeon, from 1998 to 2002, a CTO and the Director with the Digital Contents Research Laboratory, Emersys Corporation, Daejeon, from 2002 to 2003, and a Principle Researcher with the Samsung Advanced Institute of Technology, Yongin, Korea, from 2003 to 2008. Since 2008, he has been a Professor with the Department of Civil and Environmental Engineering, KAIST, and he was the Chief of the KAIST Robotics Program. From 2019, he is a Professor with the School of Electrical Engineering. His current research interests include autonomous robot navigation, SLAM (simultaneous localization and mapping), spatial AI/ML, and swarm robots.