2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

GIMloco: Generic Internal Model-Based Locomotion for Quadruped Robots

Zhonghuai Yan[1], Quan Quan*[2]

***Abstract*— A central challenge in robust quadruped locomotion, which relies solely on proprioceptive information, is how to effectively encode the history of observations. While current methods, such as regression, struggle with high-dimensional multi-time-step histories, and Temporal Convolutional Networks (TCNs) incur computational overhead, we propose a more efficient and theoretically grounded alternative. Inspired by the Generic Internal Model (GIM) from control theory, we introduce GIMloco, which maps the history of proprioceptive observations into a compact and stable internal model space through a predesigned first-order integral system with stability and orthogonality guarantees. This encoded representation drives three downstream tasks: state estimation, latent variable learning, and control policy learning. Our experiments show that GIMloco outperforms strong baselines in velocity tracking, system overshoot, response speed. Furthermore, it can navigate more complex terrains while also demonstrating better training stability across random seeds. Crucially, our method reduces training time by two orders of magnitude compared to TCN-based approaches. Our work presents GIMloco as a robust and computationally efficient framework for locomotion based on proprioceptive information.**

I. INTRODUCTION

Currently, learning-based control methods have demonstrated remarkable effectiveness in addressing the challenge of legged robot locomotion[1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14]. Through learning, it is possible to obtain robust control policies that do not rely on precise physical models. These policies have been successfully transferred to real robots with zero-shot sim-to-real transfer, enabling agile locomotion over rugged terrain.

From the perspective of sensors used by quadruped robots, relying solely on proprioception without vision (camera) or Lidar[2], [4], [6], [9] to track velocity commands is a primary technical approach [3], [8], [11], and our work is based on this direction. Due to the limited sensors, the observation information only includes the gravity orientation from the IMU, base orientation, and motor angles and velocity from the motors. The robot cannot directly perceive terrain or other environmental information, which places higher demands on the motion control algorithm. Many researchers have made successful attempts by using privileged information during the training phase to decode environmental information from the history of proprioceptive observations. RMA[3] employs a two-stage training process: it first encodes extrinsic environmental factors using privileged information, and then, in the second stage, it regressively predicts these factors based on proprioceptive observations. HIMloco[15] employs a contrastive learning method to learn a latent variable that distinguishes between different trajectories, thereby implicitly obtaining a metric for assessing stability in various environments and for the robot itself.

When decoding environmental or hidden states from proprioceptive information, we identified drawbacks in mainstream approaches. Regression-based methods face a linear growth in input dimensionality with the length of the observation history, severely restricting the practical history window for high-dimensional states (e.g., 45 in HIMloco) and failing to model temporal relationships. In contrast, while TCNs[3] capture these relationships, they incur significant computational overhead, leading to an order-of-magnitude increase in training time due to operations such as sliding convolutions and data reshaping. This motivated our search for an observer architecture that could better balance the respective advantages of regression and TCNs.

Our method utilizes a GIM[16] to preprocess proprioceptive observation history first by mapping it into a high-dimensional internal model space with a linear integral system. Subsequently, we leverage data-driven techniques to learn the necessary decoding maps from this internal representation. The GIM system requires the a priori selection of its eigenvalues, which guarantee system stability while governing the decay and convergence rates of the temporal information. To ensure numerical stability, we assume state trajectories have finite energy in the L_2-norm sense and introduce orthogonal basis functions, determining all system parameters according to the construction algorithm by Linnemann[17].

Experiments show that GIM achieves enhanced performance, including higher scores in linear/angular tracking, terrain level, and velocity command, compared to regression-based approaches, while maintaining high training efficiency. Although GIM is structurally similar to conventional convolution, the proposed method demonstrates a two orders of magnitude improvement in both training and computational efficiency (as empirically demonstrated on an NVIDIA RTX 5080 GPU), while offering a priori guarantees regarding the stability of the convolutional system.

The structure of this paper is as follows: In Section II, we introduce the related work of other researchers. In Section III, we describe the key techniques used in this paper. Section IV presents our methodology. Section V presents our experiments and a comparison with baseline results. Finally, in Sections VI and VII, we provide a discussion

[1]Shen Yuan College, Beihang University, Beijing, 100191, P.R. China.
[2]School of Automation Science and Electrical Engineering, Beihang University, Beijing, 100191, P.R. China.
E-mail: {yanzhonghuai, qq_buaa}@buaa.edu.cn
*Corresponding author.

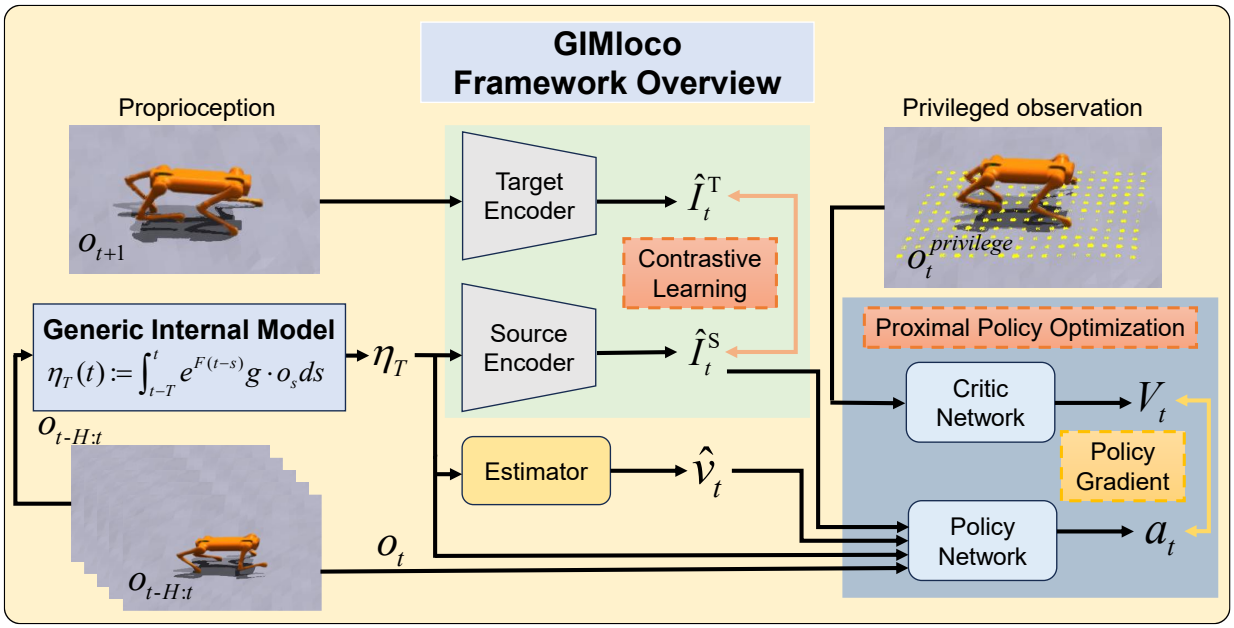


Fig. 1: Overview of Generic Internal Model(GIM) Locomotion framework. GIM maps the observation history vectors to internal model space as η_T. The Internal Model vector η_T drives the velocity estimation, contrastive learning block[15], and policy network.

and summary.

II. RELATED WORK

Reinforcement Learning (RL) has become an increasingly established and viable approach for legged locomotion[1], [2], [3], [18], [19] .

By leveraging techniques like high-fidelity simulators[20], Teacher-Student training frameworks[3], Curriculum Learning[7], [10], and Domain Randomization[15], algorithms can achieve zero-shot sim-to-real transfer. Within the domain of proprioception-based locomotion control, a prominent example of this approach is Rapid Motor Adaptation (RMA)[3], which uses an adaptation module to encode environmental extrinsics for rapid online adjustments using only onboard sensors.

Recently, L3P[21] utilizes diffusion models to train a latent-space-to-latent-space policy for efficient policy transfer across different robot types; SLR[22] learns a low-dimensional latent vector and its state transitions from proprioceptive observation history, achieving better performance without privileged information; LIT[23] designs a two-stage learning method for robust walking patterns driven by an ideal dynamics model to accelerate training; NeuralIMC[24] enhances performance on both quadrotors and quadrupedal robots by feeding the model prediction error into its policy network; and BAS[25] simultaneously achieves Adaptivity, Safety, and Agility through a framework comprising an agile policy for rapid obstacle avoidance, a recovery policy for collision prevention, a co-trained physical parameter estimator, and a learned Reach-Avoid (RA) value network.

The concept of inferring environmental properties from sensor history, or imagination, is a recurring theme. It is achieved through various mechanisms, including learning latent dynamics models for planning (Dreamer)[26] and using context estimators to infer privileged information (DreamWaQ)[11]. Our work follows this direction by pre-processing the history of proprioceptive observations using GIM, leading to improved observational performance.

III. PRELIMINARIES

A. System Modeling

For a general nonlinear control problem involving disturbances, the system is modeled as follows:

$$\dot{x} = f(x, u, w), \quad \dot{w} = s(w), \quad e = h(x, w) - q(w). \quad (1)$$

The first equation describes the plant, with its state $x \in \mathbb{R}^k$, input $u \in \mathbb{R}^q$, and external signal $w \in \mathbb{R}^p$. The system's output is denoted by $y = h(x, w)$, $y \in \mathbb{R}^p$. The second equation represents the exosystem dynamics; the external signal w includes disturbances and the reference signal to be tracked, and it is an autonomous system. The third equation defines the tracking error. The system's objective is to make the plant's output track the reference signal.

Remark 1: This modeling framework is general and can be applied to robot locomotion control problems. Here, we assume that the command to be tracked and disturbances from the environment, electromechanical systems, etc., can all be modeled as an autonomous signal (exosystem) whose dynamics are unknown.

Fig. 2: The robot traversing a continuous segment of challenging mixed terrains, including descending stairs, discrete obstacles, gravel, and a damp, dense grassy area.

B. Regulation Based on the Generic Internal Model

In recent years, some researchers have theoretically developed the generic internal model technique for exosystem unknown output regulation[27], [28], [29]. By encoding the exosystem information, they have proven the existence of an injective relationship between the internal model state and a stable feedforward.

The form of the generic internal model is:

$$\begin{aligned} \eta^{\star}(t) &= \int_{-\infty}^{t} \exp(F(t-\tau))gu^{\star}(t)\mathrm{d}\tau \\ u^{\star}(t) &= \gamma(\eta^{\star}(t)). \end{aligned} \tag{2}$$

Here, (F, g) is a controllable pair, and F is a Hurwitz matrix. If the control input is a stable feedforward $u = \sigma(w)$, assume $\eta \in \mathbb{R}^m$:

The work by Marconi[16] showed that if the dimension of the internal model meets certain conditions, there exists some unknown nonlinear injective mapping γ that can losslessly decode the control feedforward from the internal model state.

Remark 2: The GIM differs from internal model construction methods represented by immersion[27], [28]. The GIM, on the other hand, primarily acts as an encoder for the external signal and is essentially an integral observer for a nonlinear system. In practice, for numerical computation, the controllable pair (F, g) can be chosen as a set of standard orthonormal bases in L_2 space, referencing the algorithm from [17].

C. Hybrid Internal Model Locomotion (HIMloco)[15]

Recently, researchers proposed Hybrid Internal Model Locomotion (HIMloco), which draws inspiration from internal model control and demonstrates strong sim-to-real transfer capability using only proprioceptive information. HIMloco takes the observation history as input, trains a linear velocity predictor through regression, and simultaneously trains the prediction of proprioceptive information using a contrastive learning method [30].

IV. METHODOLOGY

A. Framework Overview

We model robot locomotion control as a sequential decision problem. We only utilize proprioception from the robot's internal sensors and do not incorporate external sensors, such as cameras or GPS, into the decision-making process. The foundation of this work is a quadruped locomotion control method designed based on the Actor-Critic algorithm and optimized with PPO, shown in Fig. 1. We adopt the latent trajectory vector learning method from HIMloco[15], which uses a Source Encoder and a Target Encoder for feature extraction and is optimized with contrastive learning. However, in our framework, we modify the key components: the Source Encoder, the linear velocity Estimator, and the Policy Network are all now driven by the output of the GIM. The final inputs to our Policy Network consist of the current one-step observation, the estimated linear velocity, and the learned latent vector. During the training phase, the value network has access to more privileged information, such as the robot's linear velocity, terrain properties, and friction forces, which enables it to improve the accuracy of state value estimation.

Given that the GIM space and the proprioceptive observation space differ in their dimensionality and physical interpretation, we propose a dual-branch policy network. The two branches perform feature extraction in parallel, after which their outputs are concatenated and fed into a fully connected network to generate the final action.

B. State Space, Observation, and Action Space

At each time step, the planner issues a desired velocity command, $c_t = (v_{xc}, v_{yc}, \omega_{yawc})$, representing the robot's linear velocities in the forward and lateral directions, and its angular velocity around the vertical axis, respectively.

The robot's proprioceptive observation include data from joint encoders and the IMU. The joint encoders provide the current position $q \in \mathbb{R}^{12}$ and velocity $\dot{q} \in \mathbb{R}^{12}$ of each joint. The IMU outputs the robot's base angular velocity $\omega \in \mathbb{R}^3$ and the gravity direction vector $g \in \mathbb{R}^3$. These observations, along with the action a_{t-1} from the previous time step and the state of the GIM, form the input to the Policy Network.

The action output is the target angle $q_{target} \in \mathbb{R}^{12}$ that each joint of the robot should reach in the next time step.

C. Multi-dimensional Output Observer Design

Suppose we have a p-dimensional measurement output, i.e., $y \in \mathbb{R}^p$. We design independent observation maps for each output channel. We treat the p-dimensional output signal

TABLE I: Accuracy and Performance Comparison ↓ in simulation over 4096 trials.

Terrain Types	Velocity Types	Ranges	GIMloco(ours)	HIMloco	GIMloco w/o one step obs	GIMloco w/o gim input	PPO
Slopes	Linear	[-2, 2] m/s	**0.564**	0.588	0.584	0.584	0.638
	Angular	[-2, 2] rad/s	**0.067**	0.068	0.082	0.099	0.167
	Combined	[-1, 1] m/s, [-1, 1] rad/s	0.407 / **0.102**	**0.404** / 0.119	0.423 / 0.108	0.405 / 0.115	0.461 / 0.172
	Combined	[-2, 2] m/s, [-2, 2] rad/s	**1.075** / **0.221**	1.105 / 0.275	1.083 / 0.241	1.102 / 0.254	1.114 / 0.325
Rough Slopes	Linear	[-2, 2] m/s	**0.572**	0.588	0.588	0.591	0.643
	Angular	[-2, 2] rad/s	**0.067**	0.068	0.082	0.097	0.166
	Combined	[-1, 1] m/s, [-1, 1] rad/s	0.409 / **0.102**	**0.405** / 0.119	0.422 / 0.107	0.406 / 0.115	0.462 / 0.171
	Combined	[-2, 2] m/s, [-2, 2] rad/s	**1.070** / **0.219**	1.107 / 0.277	1.086 / 0.241	1.105 / 0.255	1.111 / 0.322
Stairs	Linear	[-2, 2] m/s	**0.568**	0.593	0.587	0.588	0.639
	Angular	[-2, 2] rad/s	**0.067**	0.068	0.083	0.100	0.167
	Combined	[-1, 1] m/s, [-1, 1] rad/s	0.408 / **0.103**	**0.405** / 0.120	0.422 / 0.107	**0.405** / 0.116	0.461 / 0.172
Discrete	Linear	[-2, 2] m/s	**0.563**	0.594	0.585	0.583	0.640
	Angular	[-2, 2] rad/s	**0.067**	0.068	0.083	0.097	0.168
	Combined	[-1, 1] m/s, [-1, 1] rad/s	0.408 / **0.102**	**0.404** / 0.119	0.424 / 0.109	0.406 / 0.116	0.462 / 0.172

* For all tests, the velocity command is randomly changed every 25 timesteps (0.5s) within a predefined range. This protocol primarily evaluates the **system's overshoot**, **response speed**, and **stability** under frequent command switching with the MSE metric. Velocity tracking ability can be partially presented by Fig. 4. The Value Network of the PPO baseline utilizes privileged information.

as p independent scalar signals. For each output channel $j \in \{1, \ldots, p\}$, we design an independent observation map.

Define the observation map for channel j:

$$\dot{\eta}_j = F_j \eta_j + g_j y_j. \tag{3}$$

This gives us p independent observation maps, where each map corresponds to a state $\eta_j \in \mathbb{R}^{m_j}$.

Concatenate into a total observation map and filter. Now, we combine these p independent modules.

The total GIM variable η: We concatenate the p independent state vectors to form a higher-dimensional total observation state vector η:

$$\eta = [\eta_1^\top \eta_2^\top \ldots \quad \eta_p^\top]^\top \tag{4}$$

where the total dimension is $m = m_1 + m_2 + \cdots + m_p$. Correspondingly, the total linear state filter is composed of p independent sub-filters. Its dynamics can be written in a block diagonal form:

$$\dot{\eta} = \text{diag}(F_1, \ldots, F_p)\eta + \text{diag}(g_1, \ldots, g_p)y. \tag{5}$$

Remark 3: In the actual algorithm deployment, the dimension of the observation y_t, same as o_t in Fig. 1, is 45, which includes the angular and angular velocity of each joint, as well as the gravity direction and base angular velocity from the IMU. Each dimension of the observation is fed into the GIM observer for independent encoding, using the same controllable pair (F, g). All eigenvalues of F are pre-defined as -0.3 and $\eta_j \in \mathbb{R}^{10}$.

D. Finite Time GIM Observers

In practical algorithm deployment, a finite-time version of the GIM could be considered. The GIM and integral observer techniques share the exact origin and are very similar in their algorithmic form. The finite-time observer was presented in the work of [31]. We briefly restate its derivation here:

Define a finite-time observation map η_T which only contains information from the most recent T time units:

$$\eta_T(t) := \int_{t-T}^{t} e^{F(t-\tau)} g(y(\tau)) \mathrm{d}\tau. \tag{6}$$

From the additivity of integrals, they satisfy the algebraic relationship:

$$\eta(t) = e^{FT}\eta(t-T) + \eta_T(t). \tag{7}$$

The finite-time information at time t, $\eta_T(t)$ is equal to the complete historical information at time t, $\eta(t)$, minus the complete historical information from time $t-T$, $\eta(t-T)$, attenuated by a factor of e^{FT} over T time units. Define a new state with a time-delay component.

V. EXPERIMENTS

In this chapter, we conduct simulation tests and ablation studies in various external environments using the Isaac Gym[20] high-fidelity simulation platform, and perform real-world experiments on the Unitree Go2. All hyperparameters are the same, especially observation history length T. Reward components and their corresponding weights align with [3], [32].

Simulation Environment Setup. We conduct experiments in an NVIDIA Isaac Gym simulator. The physics simulation runs at 200 Hz, and the control policy operates at 50 Hz across 4096 parallel environments. The policy outputs target joint angles for a PD controller.

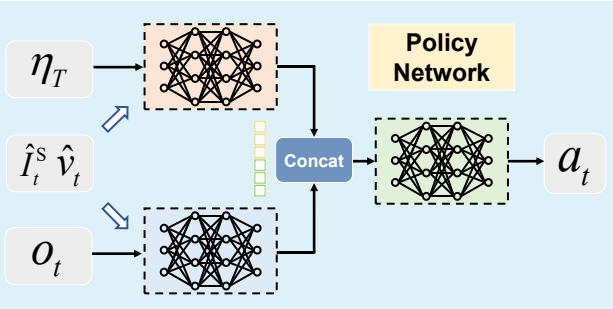


Fig. 3: Structure of Policy Network. Given that the internal model variable and the one-step observation occupy different spaces and have disparate scales, they are processed through separate channels before being concatenated and mapped to the final action.

Dynamics Randomization. Randomized parameters include the robot's physical properties (base payload: $[-1, 2]$ kg; center of mass offset: ± 0.05 m; link friction: $[0.2, 1.25]$) and its actuation system (motor output and PD gains scaled by $[0.9, 1.1]$).Velocity impulses up to 1.0 m/s and continuous forces of ± 30 N.

Training Curriculum Design. The terrain difficulty is dynamically adjusted based on the agent's travel distance: it is increased if the agent covers more than 50% of the terrain length, and decreased otherwise. In parallel, the maximum commanded velocity is increased only when two conditions are met: the velocity tracking reward exceeds 80% of its maximum, and at least 200 training iterations have passed since the last increase.

A. Simulation Tests

To systematically evaluate the performance of the control policy, we design a comprehensive simulation testing benchmark. The evaluation protocol is conducted on four different typical terrains: Slopes, Rough Slopes, Stairs, and Discrete Obstacles. In each terrain condition, the robot starts from a stationary state. It receives a constant velocity command for 25 simulation steps, after which the command is resampled randomly from a specified range to fully assess the policy's dynamic response and steady-state tracking capabilities.

- **GIMloco (Our full model)**: Contains all proposed modules and serves as the performance benchmark.
- **GIMloco w/o gim Input**: Removes the historical information embedding provided by the GIM module to the policy network, aiming to evaluate the direct contribution of GIM drived latent variable learning and linear velocity estimation.
- **GIMloco w/o one-step obs**: Removes the o_t input of the policy network, forcing it to rely entirely on the historical context provided by the GIM module for decision-making, to verify the importance of immediate state information.
- **HIMloco**: The HIMloco (Hybrid Internal Model) method serves as the baseline.
- **PPO**: The Value Network of the PPO utilizes privileged information to strengthen the baseline.

Our experimental results, presented in Table I, demonstrate that GIMloco consistently outperforms both HIMloco and an enhanced PPO baseline in terms of system overshoot, response speed, and stability. This indicates better performance, particularly during the critical transition phases of command switching.

The ablation studies clarify the source of this improvement. The GIMloco w/o gim Input variant, which performs on par with HIMloco, reveals that simply using the GIM for velocity estimation and latent learning offers no significant advantage in transition performance. In contrast, the GIMloco w/o one-step obs variant, which relies solely on the GIM variable η to drive the policy, was slightly weaker than HIMloco, suggesting this approach increases system response time.

Ultimately, the success of the complete GIMloco model stems from its dual-channel architecture. By combining direct one-step observation with processed GIM information, it effectively enhances both response speed and stability, leveraging the strengths of both inputs.

B. Ablation Study

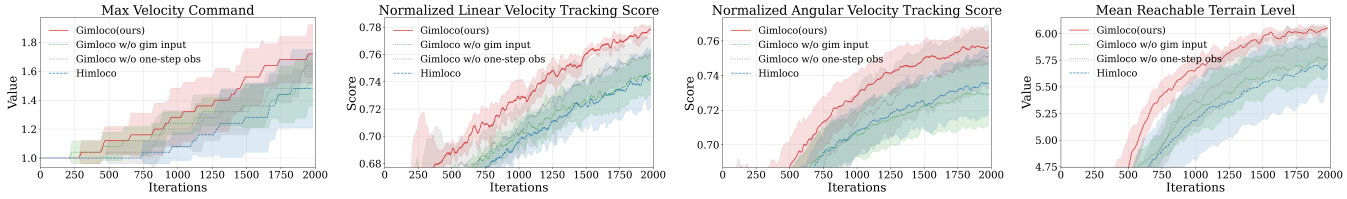
To verify the effectiveness of the key components in our proposed GIMloco model and to compare it with existing state-of-the-art methods, we design a series of rigorous ablation studies. This research aims to quantify the contribution of specific modules to the overall performance by removing them.

Comparison Model Configurations Our experiments included the following four model configurations:

Training and Evaluation Protocol. To ensure a fair comparison and the statistical robustness of the results, each model configuration is trained independently under five different random seeds (seeds 1-5). The primary evaluation metrics included:

- **Normalized Linear Velocity Tracking Score**: Quantifies the policy's tracking accuracy for forward and lateral velocity commands.
- **Normalized Angular Velocity Tracking Score**: Quantifies the policy's tracking accuracy for turning commands.
- **Mean Reachable Terrain Level**: Measures the average level of the most complex terrain the policy can adapt to under the curriculum learning mechanism, serving as a core indicator of the model's robustness.
- **Max Velocity Command**: Monitors the upper limit of the velocity command in the curriculum learning process, reflecting the model's learning progress.

The ablation study, Fig. 4, Table II, Table III, reveals that, under identical hyperparameters, GIMloco outperforms the baseline HIMloco in both tracking score and training stability. Specifically, GIMloco achieves higher scores on the primary objectives, NLTS and NATS, indicating better linear and angular velocity tracking capabilities. Moreover,



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

Fig. 4: Ablation study across five random seeds with learning curves of (a) Maximum Velocity Command(MVC) (b) Normalized Linear Velocity Tracking Score, $\text{NLTS} = \exp(-\frac{\left\|v_{x,y}-v_{x,y}^{\text{target}}\right\|_2^2}{0.25})$ (c) Normalized Angular Velocity Tracking Score, $\text{NATS} = \exp(-\frac{\left\|\omega_{\text{yaw}}-\omega_{\text{yaw}}^{\text{target}}\right\|_2^2}{0.25})$ (d) Mean Reachable Terrain Level(MRTL).

TABLE II: Final Mean Scores ↑

Metric	Gimloco(ours)	Gimloco w/o gim input	Gimloco w/o one-step obs	Himloco
Norm. Linear Velocity Score	**0.7741**	0.7423	0.7562	0.7391
Norm. Angular Velocity Score	**0.7562**	0.7293	0.7483	0.7338
Mean Reachable Terrain Level	**6.0242**	5.7320	5.8777	5.6787
Max Velocity Command	**1.6878**	1.4736	1.5573	1.4719

TABLE III: Final Standard Deviations ↓

Metric	Gimloco(ours)	Gimloco w/o gim input	Gimloco w/o one-step obs	Himloco
Norm. Linear Velocity Score	**0.0042**	0.0147	0.0145	0.0196
Norm. Angular Velocity Score	0.0187	0.0193	**0.0179**	0.0181
Mean Reachable Terrain Level	**0.0419**	0.1836	0.1862	0.2854
Max Velocity Command	0.1683	0.1921	**0.0773**	0.2634

* The Final Mean Scores and Final Standard Deviations are calculated by averaging data from the final stage of training (the last 10%, corresponding to epochs 1800-2000) across five random seeds. This method is used to demonstrate the average performance on key metrics and the training stability of each model during the convergence phase.

these results are achieved on more complex terrains and under higher velocity commands, as evidenced by GIMloco's higher average scores for terrain and velocity command levels.

To dissect this performance gain, we analyze two variants. The Gimloco w/o gim input variant showed that training the latent variable with our internal model variable is marginally more effective than using the raw observation history as HIMloco does. More strikingly, the Gimloco w/o one-step obs variant, which relies solely on the GIM variable to drive the policy, outperformed HIMloco on all metrics. This result suggests that the processed GIM state is a richer input than the raw observation and partially validates the core GIM theory—that an ideal control feedforward can be constructed directly from the GIM variable via a nonlinear injection.

Despite the strong performance of using the GIM variable alone, our final architecture incorporates both the GIM variable and the raw one-step observation. This dual-channel design is a practical consideration for training stability, as the GIM's nonlinear mapping can require more time to converge at high command velocities. The one-step observation provides a direct, supplementary signal that stabilizes the learning process, and our results confirm that this fused approach yields the best overall performance.

C. Comparative Analysis of Learning Time

A comparison was conducted between the preprocessing model GIM and the HIMloco model, which utilizes a Temporal Convolutional Networks (TCNs) as its encoder, regarding their respective training durations. Experiments demonstrated that on a single NVIDIA RTX 5080 GPU, the GIM model completed one training epoch, learning 100 times faster than the TCNs-based module within 400 epochs.

TABLE IV: Average Learning Time Comparison

Method	Average Learning Time (s)
Gimloco(ours)	**1.89**
Himloco	**0.67**
Himloco+TCNs	183.10

D. Real World Experiments

To validate our approach, the GIM-based controller was deployed on a Unitree Go2 robot. We conducted a series of tests in diverse and complex environments to assess the algorithm's performance in obstacle negotiation, dynamic balance, gait stability, as well as its ability to climb and descend tall flights of stairs continuously.

Fig. 2 illustrates an experiment in a challenging, fused-terrain environment designed to test the robot's adaptability. This scene includes intersecting sections of fine gravel, discrete obstacles, steps, and deep, damp grass, which the Go2 must navigate continuously. This requires the controller to respond immediately to changes in terrain. Throughout these transitions, our GIM-based algorithm exhibited high adaptability and robustness.

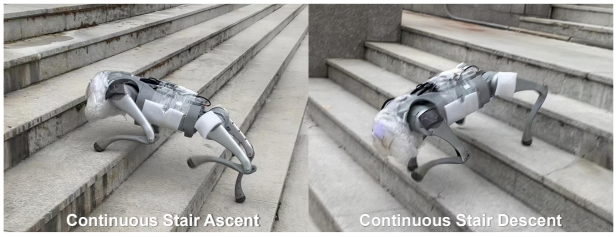


Fig. 5: Real-world experiments: continuously ascending and descending multiple high steps. For more scenarios, gait details, and the complete experimental process, please see the supplementary video.

Fig. 5 demonstrates the GIM-based Go2's ability to continuously ascend and descend multiple high steps (with a step height approaching the robot's standing height), as well as its ability to jump down from a high platform. Trained with our GIM-based algorithm, the Go2 exhibits a regular gait and excellent environmental adaptability.

VI. Discussion

The GIM can be analyzed from several perspectives. From a frequency-domain viewpoint, it functions as a low-pass filter with a sliding window, structurally designed to remove high-frequency noise from the state observation sequence. From a geometric perspective, the GIM projects the state observation trajectory onto a set of a priori designed orthonormal basis functions in an L_2 space. The internal model vector, η_T, represents the coefficients of this projection. This approach treats the observation sequence as a holistic trajectory, giving it a structural advantage over regression-based methods that assume observations are independent. Finally, from a computational viewpoint, leveraging the time-shifting property of convolution enables a highly efficient implementation, requiring only a single matrix multiplication and addition per time step, making it far more efficient than TCNs.

In practice, we find that both the information used and the way it is organized are critically important. The first consideration is the length of the observation history. The eigenvalue, $-\lambda$, of the GIM's dynamic matrix F determines the decay rate of the historical signal. For an observation sequence of length T, the decay of the most distant signal is estimated by $e^{-\lambda T}$. In this work, we design the eigenvalue of F to be -0.3, which for a history length of $T = 6$ yields a decay factor of $e^{-\lambda T} \approx 0.165$. The second choice relates to the Policy Network's architecture. Proprioceptive sensing imposes partial observability, making it essential to incorporate history, unlike in a fully observable system, where a one-step Markov assumption would hold. Our experiments validate this, showing that a dual-channel architecture, which independently processes and then fuses the current one-step observation with the historical context, yields superior results.

VII. Conclusions

This work demonstrates that a Generic Internal Model (GIM) observer, which maps the history of proprioceptive observations into an internal model space, can effectively drive state estimation, latent variable learning, and the control policy.

Our approach yields better performance in both training and testing, enhancing training stability, the accuracy of velocity command tracking, and enabling rapid and stable responses during command transitions.

However, the practical application of this method to quadrupedal robots is still limited, as the GIM's eigenvalues and dimensionality must be set a priori. Despite this limitation, we are confident that the GIM-based observer holds significant potential and will prove its value in more complex embodiments, such as humanoid robots, and across various other motion control algorithms.

Acknowledgment

This work is supported in part by the National Natural Science Foundation of China under Grant 62573021 and the National Key Research and Development Program of China under Grant 2025YFE0216900. Zhonghuai Yan thanks the support of the Beijing Natural Science Foundation(QY24113).

We are grateful to Jiarong Lin for the technical support with the Unitree Go2 equipment and thank Zifeng Zhang for his help in conducting the experiments. The authors would like to thank Chenyu Wang, Zhaolong Shen, Haipeng Cao, Yuechen Li for valuable discussions and insightful suggestions.

References

[1] 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 *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2245–2252.

[2] W. Yu, D. Jain, A. Escontrela, A. Iscen, P. Xu, E. Coumans, S. Ha, J. Tan, and T. Zhang, "Visual-locomotion: Learning to walk on complex terrains with vision," in *Conference on Robot Learning*, 2021.

[3] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid motor adaptation for legged robots," in *Robotics: Science and Systems*, 2021.

[4] R. Yang, M. Zhang, N. Hansen, H. Xu, and X. Wang, "Learning vision-guided quadrupedal locomotion end-to-end with cross-modal transformers," in *International Conference on Learning Representations*, 2022.

[5] F. Deng, I. Jang, and S. Ahn, "Dreamerpro: Reconstruction-free model-based reinforcement learning with prototypical representations," in *International Conference on Machine Learning*. PMLR, 2022, pp. 4956–4975.

[6] A. Agrawal, S. Chen, A. Rai, and K. Sreenath, "Vision-aided dynamic quadrupedal locomotion on discrete terrain using motion libraries," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 4708–4714.

[7] 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 Robotics and Automation Letters*, vol. 7, no. 2, pp. 4630–4637, 2022.

[8] J. Wu, G. Xin, C. Qi, and Y. Xue, "Learning robust and agile legged locomotion using adversarial motion priors," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4975–4982, 2023.

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

[10] R. Yang, G. Yang, and X. Wang, "Neural volumetric memory for visual locomotion control," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 1430–1440.

[11] I. M. Aswin Nahrendra, B. Yu, and H. Myung, "Dreamwaq: Learning robust quadrupedal locomotion with implicit terrain imagination via deep reinforcement learning," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 5078–5084.

[12] L. M. Smith, J. C. Kew, T. Li, L. Luu, X. B. Peng, S. Ha, J. Tan, and S. Levine, "Learning and adapting agile locomotion skills by transferring experience," in *Robotics: Science and Systems*, 2023.

[13] Z. Zhuang, Z. Fu, J. Wang, C. G. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao, "Robot parkour learning," in *Conference on Robot Learning*, 2023.

[14] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *The International Journal of Robotics Research*, vol. 43, no. 4, pp. 572–587, 2024.

[15] J. Long, Z. Wang, Q. Li, L. Cao, J. Gao, and J. Pang, "Hybrid internal model: Learning agile legged locomotion with simulated robot response," in *International Conference on Learning Representations*, 2024.

[16] L. Marconi and L. Praly, "Uniform practical nonlinear output regulation," *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1184–1202, 2008.

[17] A. Linnemann, "Convergent ritz approximations of the set of stabilizing controllers," *Systems & Control Letters*, vol. 36, no. 2, pp. 151–156, 1999.

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

[19] Z. Xie, X. Da, M. Van de Panne, B. Babich, and A. Garg, "Dynamics randomization revisited: A case study for quadrupedal locomotion," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 4955–4961.

[20] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu based physics simulation for robot learning," in *Advances in Neural Information Processing Systems*, vol. 35, 2021.

[21] Z. Zheng, G. Zhan, B. Shuai, S. Qin, J. Li, T. Zhang, and S. E. Li, "Transferable latent-to-latent locomotion policy for efficient and versatile motion control of diverse legged robots," *arXiv preprint arXiv:2503.17626*, 2025.

[22] S. Chen, Z. Wan, S. Yan, C. Zhang, W. Zhang, Q. Li, D. Zhang, and F. U. D. Farrukh, "SLR: Learning quadruped locomotion without privileged information," in *Conference on Robot Learning*, 2024.

[23] W. Xiao, S. Lyu, Z. Gong, R. Wang, and D. Wang, "Learning robotic policy with imagined transition: Mitigating the trade-off between robustness and optimality," *arXiv preprint arXiv:2503.10484*, 2025.

[24] F. Gao, C. Yu, Y. Wang, and Y. Wu, "Neural internal model control: Learning a robust control policy via predictive error feedback," *IEEE Robotics and Automation Letters*, 2025.

[25] Y. Zhong, C. Zhang, T. He, and G. Shi, "Bridging adaptivity and safety: Learning agile collision-free locomotion across varied physics," *arXiv preprint arXiv:2501.04276*, 2025.

[26] P. Wu, A. Escontrela, D. Hafner, P. Abbeel, and K. Goldberg, "Daydreamer: World models for physical robot learning," in *Conference on robot learning*. PMLR, 2023, pp. 2226–2240.

[27] B. A. Francis and W. M. Wonham, "The internal model principle for linear multivariable regulators," *Applied mathematics and optimization*, vol. 2, no. 2, pp. 170–194, 1975.

[28] A. Isidori and C. I. Byrnes, "Output regulation of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 35, no. 2, pp. 131–140, 2002.

[29] S. Wang, M. Guay, Z. Chen, and R. D. Braatz, "A nonparametric learning framework for nonlinear robust output regulation," *IEEE Transactions on Automatic Control*, 2024.

[30] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto, "Reinforcement learning with prototypical representations," in *International Conference on Machine Learning*. PMLR, 2021, pp. 11 920–11 931.

[31] G. Kreisselmeier and R. Engel, "Nonlinear observers for autonomous lipschitz continuous systems," *IEEE Transactions on Automatic Control*, vol. 48, no. 3, pp. 451–464, 2003.

[32] G. B. Margolis and P. Agrawal, "Walk these ways: Tuning robot control for generalization with multiplicity of behavior," in *Conference on Robot Learning*. PMLR, 2023, pp. 22–31.