Integrating Controllable Motion Skills from Demonstrations

First Author
Institution1
Institution1 address
firstauthor@i1.org

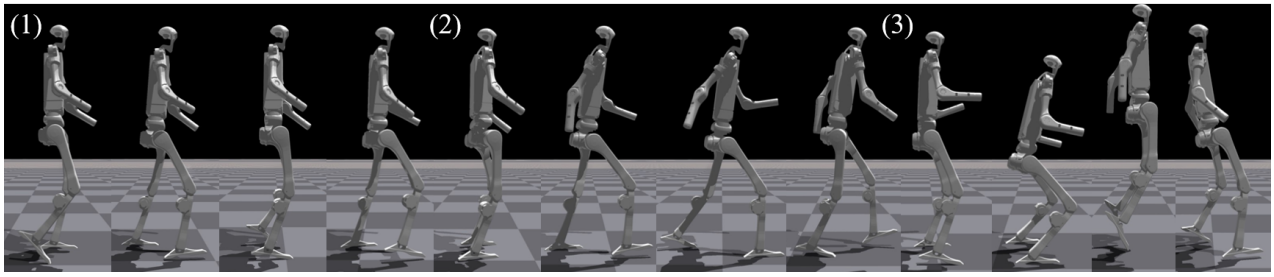


Fig. 1: Our method enables legged robots to flexibly integrate a range of different motion skills into a single controller, and can be further combined with a high-level NLI module to enable preliminary language-directed skill control. The language commands used here are (1) "Act as if you're a scary character", (2) "Return to normal walking style", (3) "Show me your jumping skills".

***Abstract*— The expanding applications of legged robots require their mastery of versatile motion skills. Correspondingly, researchers must address the challenge of integrating multiple diverse motion skills into controllers. While existing reinforcement learning (RL)-based approaches have achieved notable success in multi-skill integration for legged robots, these methods often require intricate reward engineering or are restricted to integrating a predefined set of motion skills constrained by specific task objectives, resulting in limited flexibility. In this work, we introduce a flexible multi-skill integration framework named Controllable Skills Integration (CSI). CSI enables the integration of a diverse set of motion skills with varying styles into a single policy without the need for complex reward tuning. Furthermore, in a hierarchical control manner, the trained low-level policy can be coupled with a high-level Natural Language Inference (NLI) module to enable preliminary language-directed skill control. Our experiments demonstrate that CSI can flexibly integrate a diverse array of motion skills more comprehensively and facilitate the transitions between different skills. Additionally, CSI exhibits good scalability as the number of motion skills to be integrated increases significantly.**

I. INTRODUCTION

With the increasing prevalence of legged robots, the demand for their enhanced capabilities is expected to grow continuously. One significant trend is the expectation for legged robots to manage a diverse array of motion skills to potentially cope with a wide range of tasks in real-world applications. Recently, RL-based approaches have reaped considerable success in the task of multi-skill integration for legged robots, e.g., impressive quadruped parkour [1], omnidirectional bipedal locomotion [2], or wonderful bipedal robots football match [3]. However, these RL-based approaches do not adequately fulfill this objective. One of the most hindering problems is the complex reward engineering required for skill learning using related methods. More frustratingly, the rewards designed for different skills are typically not generalizable, complicating the integration of multiple motion skills.

Integrating imitation learning (IL) with RL provides a feasible solution to these problems. Recently, a plethora of ongoing research in the field of character animation has demonstrated the effectiveness of this paradigm. By allowing agents to track the reference motion trajectories [4], [5], [6], or aligning with motion style from imitating reference motion [7], [8], [9], controllers trained via IL can perform motion skills naturally. As demonstration data serves as a reference for policy learning, RL approaches integrated with imitation learning significantly simplify reward engineering. Additionally, the design of rewards necessary for various motion skills becomes more standardized.

Recently, some methods in character animation have been adapted to the multi-skill integration for legged robots [9], [10], [11], [12], [13]. However, these approaches generally exhibit limited flexibility. Most of them rely on extra well-defined task objectives [9], [10], [11] to integrate a set of similar task-related skills, thereby constraining their applicability and the variety of motion skills they can integrate. Alternatively, other approaches [12], [13] attempt to integrate each different motion skill with the help of additional networks or training stages, which makes the corresponding training costs increase when the number of skills to be integrated increases. Overall, these approaches are limited in the range or number of motion skills that can be integrated.

In this work, we propose CSI, a flexible framework designed for legged robots to integrate multiple motion skills from reference motion clips into a single controller.

CSI is built upon Generative Adversarial Imitation Learning (GAIL) [14], an IL framework that obviates the need for skill-specific reward engineering. Furthermore, by incorporating key designs such as Conditional Imitation Learning and Condition-Aware Loss, CSI can use skill labels as a control interface for integrated motion skills, which makes it possible to access some external knowledge like natural language for skill control. Our experiments validate the effectiveness of CSI and demonstrate its notable ability to support language-directed skill control through the incorporation of high-level NLI modules.

In summary, the primary contributions of this paper are reflected in the following three aspects:

- We propose CSI, a flexible multi-skill integration framework. CSI enables legged robots to acquire versatile and controllable motion skills by effectively imitating reference motion capture data.
- Our approach provides a more controllable interface, enabling the flexible and easy leveraging of heuristic knowledge to improve the efficiency of skill execution.
- Detailed experiments and analyses of our approach are carried out on different datasets as well as on different robots, which validates the effectiveness and adaptability of our work.

Supplementary videos of this work are posted on https://vsislab.github.io/CSI_IL/.

II. RELATED WORK

A. Multi-Skill Integration for Legged Robots

In recent years, methods built upon the RL paradigm have achieved promising performance in integrating multiple motion skills for legged robots. Rodriguez *et al.* [2] introduced deep reinforcement learning (DRL) method to enable the bipedal robot NimbRo-OP2X [15] to learn agile omnidirectional locomotion, including walk forward, walk backward and steering. Zhuang *et al.* [1] proposed a two-stage RL training approach for the quadruped robots Go1 [16] and A1 [17] to acquire complex dexterous parkour maneuvers, such as creep forward and jumping. Tuomas *et al.* [3] employed DRL techniques to impart agile soccer skills on the small bipedal robot OP3 [18].

However, RL-based methods typically require meticulous reward engineering, which can be both labor-intensive and time-consuming, especially for multi-skill integration. To address this problem, the introduction of IL has proven effective in alleviating the need for explicit reward design, thereby offering considerable advantages for integrating multiple motion skills. This beneficial impact has been demonstrated by related works in the field of character animation. Won *et al.* [19] proposed a tracking-based method to integrate various motion skills from a large-scale open-source motion capture dataset [20] into a few controllers, and characters can switch between controllers to perform different motion skills. Peng *et al.* [21] and Zhu *et al.* [22] incorporated a shared embedding space within the learning process, enabling the integration of multiple skills into one single policy. Furthermore, this embedding space also serves as an interface to call integrated skills for subsequent training of downstream tasks.

These methods have also witnessed initial explorations on legged robots in recent studies. [9], [10], [11] incorporated velocity command tracking objectives during the training process to integrate a set of similar walking and running maneuvers into one controller. Vollenweider *et al.* proposed MultiAMP [12], which assigns an additional discriminator for each skill that needs to be integrated to aid learning. Han *et al.* [13] instead adopted a multi-stage training process, training specific Vector Quantized-Variational AutoEncoder [23] policy for each different skill, then integrating these policies into a single one by distillation.

B. Controllable Motion Skills

How to give controllability to the integrated skills is an important issue for the integration of multiple motion skills in legged robots. [9], [10], [11], [24] added additional task objectives such as tracking velocity command, to the training objective as a way to achieve controllability of the integrated motion skills through command input. This approach is constrained by the need for well-defined task objectives and is also limited by the specific task requirements that determine which motion skills can be integrated. For example, when the task involves velocity tracking, it becomes more difficult to integrate dance movements into the policy. Vollenweider *et al.* [12] applied one-hot skill code as input, each code corresponds a skill discriminator, through leanring to switch between these codes, the policy can adjust the output motion skill according to the input skill. Although this approach allows for a less restricted range of integrable motion skills, the training cost increases proportionally with the number of skills that need to be integrated.

In this paper, we propose CSI, a flexible framework that enables the integration of diverse motion skills into a single controller. CSI uses skill labels as control signals to enable controllability through integrated motion skills, which can be further combined with a pre-trained NLI module to achieve preliminary language-directed skill control. Unlike existing approaches, CSI eliminates the need for additional task objectives and multi-stage training to establish a control interface. We anticipate that our work will serve as a valuable reference for future multi-skill integration applications in legged robots.

III. METHOD

A. Preliminary

In this work, our goal is to enable legged robots to obtain versatile and controllable motion skills. To achieve this, we formulate the problem as a goal-conditioned [25] Markov Decision Process (MDP) $(S, A, \mathrm{c}, R, p_0, \gamma)$, where S is the state space, A denotes the action space, $c \in C$ signifies the input condition, $R = r(s_t, s_{t+1}, c)$ represents the reward for each time step, p_0 is initial state distribution, and $\gamma \sim (0, 1]$ denotes the discount factor. During the training process, we employ the RL algorithm to optimize the parameters of policy: $\pi_\theta : S \rightarrow A$, aiming to maximize the expected return

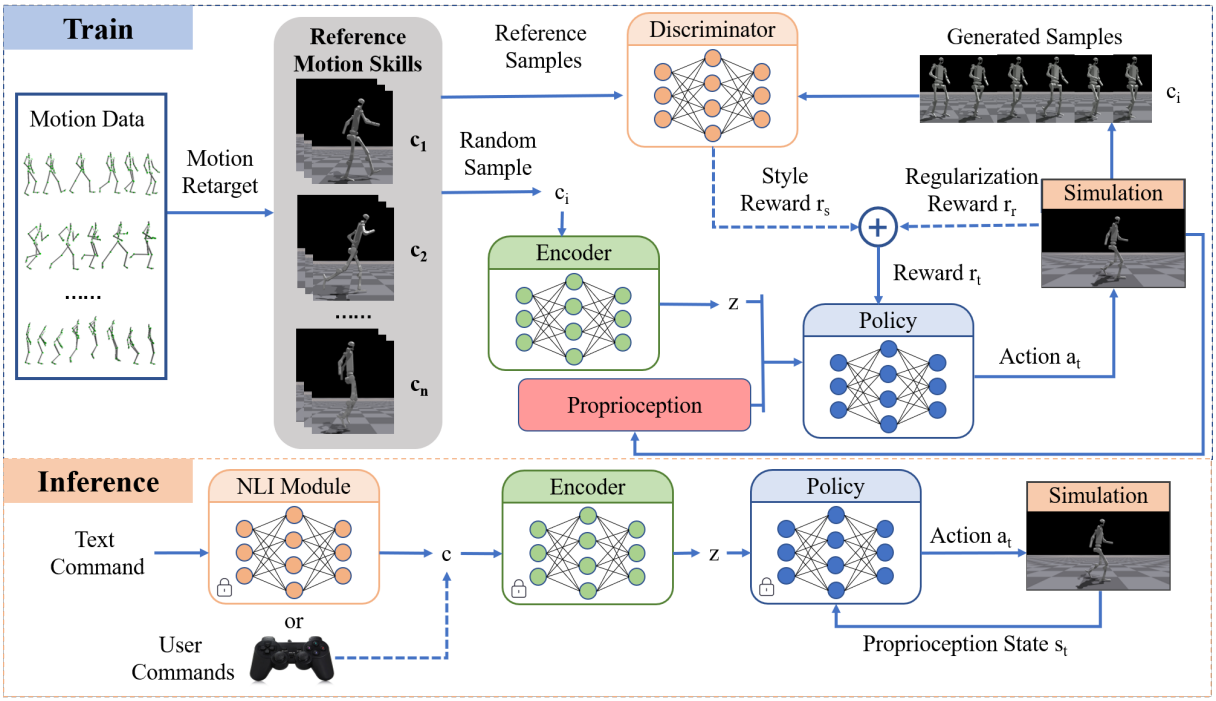


Fig. 2: Overview diagram of CSI. Through retargeting and skill labeling, a set of reference motion clips with corresponding labels can be obtained. During training, sampled motion skill labels c_i are mapped to latent vectors z through an encoder network, and the policy generates corresponding motion skills based on z. The discriminator is responsible for indirectly regulating the motions generated by the policy in a way that provides style rewards. After the training stage, a controller with integrated multiple motion skills is available. These integrated skills can be controlled directly through user commands or externally via a high-level pre-trained NLI module for language-directed skill control.

of the discounted episode reward $J(\pi) = E_{c\in C,\pi_\theta}[\sum_{t=0}^{T-1} \gamma^t r_t]$, where T represents the horizon length of an episode.

To enable legged robots to generally learn motion skills by imitating from demonstration motion data, our method is built upon GAIL [14]. In GAIL's framework, a generator is trained in an adversarial manner with a discriminator. The discriminator's role is to distinguish between *real* samples from expert demonstrations and *fake* samples generated by the generator. The feedback from the discriminator to the generator serves as a reward signal, guiding the generator to produce data that closely resembles the expert demonstrations. Typically, GAIL requires state-action pairs (s_t, a_t) as input, where expert demonstration data provides both the state and the corresponding action. In our approach, we utilize motion capture data as the source of expert demonstration. Motion capture data primarily records the skeletal states at each frame but does not include the explicit actions taken by the expert. To address this limitation, we adopt the paradigm of GAILfO [26] by utilizing state transitions (s_t, s_{t+1}) instead of state-action pairs (s_t, a_t). This adaptation enables the generator to learn from the evolution of states, even in the absence of explicit action conducted by the expert.

B. Conditional Imitation Learning

To achieve controllability over the integrated motion skills, we propose that all networks within the framework should be guided to operate according to some kind of instruction. Therefore, we introduce Conditional Imitation Learning (CIL) to guide the networks to the specified motion skills during the training process. For the discriminator, we add skill labels to the original samples input, which requires it to be able to judge the authenticity of given samples by taking into account the skill label information. For the policy and the value function, we introduce a simple encoder network to map the skill label to a latent vector z. z will be used as part of the input to the policy and the value function, which requires them to be able to respond according to the condition z. Based on the background of CIL, we design the basic training objective of the discriminator as follows:

Conditional Imitation Loss. In the framework of the vanilla GAN [27], a variational approximation of the Jensen-Shannon divergence [28] is commonly applied to achieve the adversarial training objective. Building on the concept of conditional probability, we introduce conditions based on the original objective:

$$\begin{aligned} L_I = &- E_{(s_t,s_{t+1})\in d^M}[log(D(s_t, s_{t+1}|c))] \\ &- E_{(s_t,s_{t+1})\in d^\pi}[log(1 - D(s_t, s_{t+1}|c))] \end{aligned} \tag{1}$$

where d^M and d^π represent the state transition distributions of the reference motion skills and those generated by the policy, respectively. c denotes the corresponding skill label. Given a motion state transition (s_t, s_{t+1}) combined with the

corresponding skill label c, the output of the discriminator is expressed as $D(s_t, s_{t+1}|c)$.

Gradient Penalty. We also introduce gradient penalty, which has been proven effective in reducing the destabilizing effects of adversarial training [7], [29], [30]:

$$L_{GP} = E_{(s_t,s_{t+1})\in d^M}[\|\nabla D(s_t, s_{t+1})\|_2^2] \tag{2}$$

Note that we calculate the gradient penalty with respect to all real samples, irrespective of the conditions corresponding to those samples.

In this way, we can preliminarily guide the generator to mimic the specified motion skills during the training process. However, some practical considerations deserve to be noticed. For example, since training directly based on the above training objectives is still an unsupervised paradigm, input conditions are frequently disregarded by the networks during training, which results in controllers that exhibit only a limited set of uncontrollable motion skills. We add some additional designs to the training objectives of the discriminator to cope with these problems:

Condition Aware Loss. Due to the unsupervised learning nature of GAN, despite the conditional imitation loss described above, the discriminator still tends to ignore the input conditions during the training process, especially when the number of reference motion skills increases. To enhance the discriminator's sensitivity to motion skill labels, we construct mismatched samples that carry mismatched motion skill labels, alongside real and fake samples. In our setting, the mismatched samples should be judged as negative by the discriminator:

$$L_{CA} = -E_{(s_t,s_{t+1})\in d^M}[log(1 - D(s_t, s_{t+1}|\hat{c}))] \tag{3}$$

where $\hat{c}$ denotes the skill labels that do not match the input samples. Subsequent experiments demonstrate that L_{CA} significantly mitigates mode collapse and enables the controller to comprehensively master the integrated motion skills.

Weight Decay. For multi-skill integration, the reference sample capacity is typically limited, rendering the GAN network prone to overfitting and leading to a relative lack of diversity in the skills generated by the final trained controllers. To alleviate this issue, weight decay is introduced for the discriminator:

$$L_{WD} = \Sigma||\omega_D||^2 \tag{4}$$

where ω_D denotes the weight parameters of the discriminator network. This improvement enables the discriminator to focus more on the general features of each skill, thereby increasing the diversity of the generated skills.

Finally, our training objective for the discriminator is defined as:

$$L_D = \omega_i L_I + \omega_{ca} L_{CA} + \omega_{wd} L_{WD} + \omega_{gp} L_{GP} \tag{5}$$

where ω_i, ω_{ca}, ω_{wd} and ω_{gp} are hyperparameters to balance each item of the training objective.

C. Reward Setting

Similar to GAIL, the reward feedback in CSI is also derived from the discriminator. To encourage the generated motion skills to resemble the reference motion capture data, while maintaining alignment between the generated motions and the given conditions, we define conditional style reward [7] as follows:

$$r_s = -log[1 - D(s_t, s_{t+1}|c)] \tag{6}$$

Conditional style reward provides great regularization for learning motion skills under the distribution of reference motion skill dataset. However, transitions between different motion skills that are not represented in the reference dataset often result in unnatural phenomena, such as jittering. To alleviate this, we introduce some additional regularization terms:

$$\begin{aligned} r_v &= \sum \|\dot{q}_t - \dot{q}_{t+1}\|_2^2 \\ r_{ep} &= \sum |\tau_t \dot{q}_t| \\ r_a &= \|a_t - a_{t-1}\|_2^2 \\ r_t &= \sum |\tau_t| \end{aligned} \tag{7}$$

where q and τ represent joint rotation angles and torque, respectively. $\dot{q}_t$ denotes the angular velocity of each joint at time step t, a means action output by policy. All symbols $\sum$ denote the summation over every Degree of Freedom (DoF) of the robot. These regularization terms lead to a smoother overall performance of the motion skills generated by the policy.

Finally, the total reward is computed as the weighted sum of the style reward and all regularization terms:

$$r_r = w_s r_s + w_v r_v + w_{ep} r_{ep} + w_a r_a + w_t r_t \tag{8}$$

where w_s, w_v, w_{ep}, w_a, and w_t are weights used to balance each component of the rewards.

D. Language-Directed Skill Control

The low-level controller trained using CSI provides a skill control interface to leverage external knowledge. To implement language-directed skill control, we first manually bind an additional caption to each skill label as *skill caption label*, e.g. "Walk Forward", "Sprint", "Jump", etc. Then we align the output of the NLI module with those skill caption labels in a zero-shot manner:

Given a text input T_i as the premise, and a finite set of skill caption labels $L = \{L_1, L_2, L_3, ..., L_n\}$ as a set of hypotheses, where n corresponds to the number of motion skills integrated by the low-level controller. The high-level NLI module is acquired to determine the textual entailment relationship between the premise T_i and each skill caption label hypothesis $L_i \in L$: entailment (positive), contradiction (negative) or neutral.

After the above process, the NLI module will output the entailment scores corresponding to each skill caption label, where the skill label corresponding to the highest-scoring skill caption label will be taken as the condition input for the low-level controller. As demonstrated in the Section IV, this

zero-shot classification paradigm decouples the high-level and the low-level modules, enabling a flexible combination between these two modules.

E. Implementation Details

Model representation. The policy π, the value function $V(s_t, c)$ and the discriminator $D(s_t, s_{t+1}, c)$ are all parameterized as shallow MLP networks, each with hidden layers of size $[512, 256]$ and rectified linear unit (ReLU) activation functions. The encoder network is a MLP network of size $[128, 128]$, with the size of latent vector z being set to fixed 8-dimensional.

Observation space. An appropriate observation representation can guide policy training effectively. In our framework, the observation of the discriminator can be represented as $\{s_t, s_{t+1}, c\}$, where s_t denotes the motion state of the robot at time t and c denotes motion skill label. Specifically, motion state s is defined in terms of a relatively complete state representation:

$$s = \{h, R_r, v_r, \omega_r, q_j, \dot{q}_j, p_i\} \tag{9}$$

where h denotes the height of the root relative to the ground, and the root is roughly located near the center of the pelvis for humanoid robots and the geometric center of the torso for quadruped robots; R_r represents the orientation of the root; v_r and ω_r mean the linear velocity and angular velocity of the root, respectively; q_j and $\dot{q}_j$ are joint position and joint velocity of the $j-th$ joint respectively; p_i means the position of the $i-th$ end-effector expressed in the local coordinate frame of the root.

For the policy and the value function, the observation can be represented as $\{a_{t-1}, s_{propri}, z\}$, where a_{t-1} means the action of the last time step, z is the vector obtained by mapping the skill condition c by the encoder network, s_{propri} denotes proprioceptive states, which is defined as:

$$s_{propri} = \{v_r, g_{pro}, q_j, \dot{q}_j\} \tag{10}$$

where g_{pro} is the projected gravity vector, which contains information about the robot's orientation.

Action space. The action space of the policy is defined by the target joint rotation angles. A PD position controller translates the output of the policy into the motor torques, following the equation $\tau = k_p(a_t - \theta_t) - k_d\dot{\theta}_t$. In our settings, the policy is queried at a frequency of 50 Hz, and the PD position controller operates at a frequency of 200 Hz.

Motion retarget. For humanoid robots, we employ a retargeting method similar to the one provided in IsaacGymEnvs [31], and for quadruped robots, we utilize the processing flow described in [4]. The retargeted motion capture data are collected as reference dataset D^M for the subsequent training. State transitions sampled from D^M are treated as real samples for training the discriminator.

Training details. To speed up the training process, we employ a distributed implementation of PPO [32] across 4096 parallel simulated environments in Isaac Gym [31], [33]. The networks are trained for 2 billion environment steps, equivalent to approximately 2 years of simulation data, which can be collected in about 15 hours on a single RTX TITAN GPU. See TABLE II in the supplementary for detailed training settings.

TABLE I: Properties of different robots.

Property	**BRUCE (w/o arms)**	**AlienGo**	**H1 (w/o hands)**
Degrees of Freedom	10	12	19
Number of Links	11	18	22
Stand Heights (m)	0.7	0.6	1.8
Total Mass (kg)	4.8	21.43	48.58

IV. EXPERIMENT

In this section, we conduct detailed experiments on our CSI method. To demonstrate the adaptability of CSI on different robots, we select three different legged robots for our experiments including quadruped robot AlienGo [34], small humanoid robot BRUCE [35] and full-size humanoid robot Unitree H1 [36]. Factors such as morphology, mass, and varying numbers of DoF differ across these three robots, posing significant challenges for the controllable multi-skill integration. TABLE I shows some of the critical parameters of each robot. For humanoid robots, we define three different tasks: versatile locomotion *(H-Locomotion*), walking in different styles (*H-WalkStyle*), and simple interactions (*H-Interaction*). For quadruped robots, since there is little motion capture data available, we define only one locomotion task containing different gaits (*Q-Locomotion*). All of the motion capture data we use are retargeted from the CMU Mocap dataset [20] or specialized dataset provided by [37]. TABLE I in the supplementary shows detailed statistics on the content of the provided motion capture dataset.

The following four methods are used to comparatively validate our method:

- Conditional Adversarial Motion Prior (CAMP): Based on AMP [7], [9], one-hot skill labels are used as conditional inputs to the policy, the value function and discriminator, which enables the integration of several different motion skills into a single controller.
- Conditional Adversarial Latent Models (CALM) [38]: *state-of-the-art* multi-skill integration method in the field of character animation, which achieves integration and controllability of multiple motion skills by introducing additional motion encoder and latent space.
- Baseline-I: Our method without Condition Aware Loss.
- CSI (ours): The methodology presented in this paper.

See TABLE II and TABLE III in the supplementary for detailed settings for CAMP and CALM. Our comparative analysis primarily focuses on three key aspects: **skill coverage**, **skill controllability**, and **language-guided skill control**. Overall, through our experiments, we aim to answer the following questions:

- Whether the controllers trained using CSI are capable of generating versatile and controllable motion skills?
- How do the policies trained with our method perform in simulation quantitatively and qualitatively?

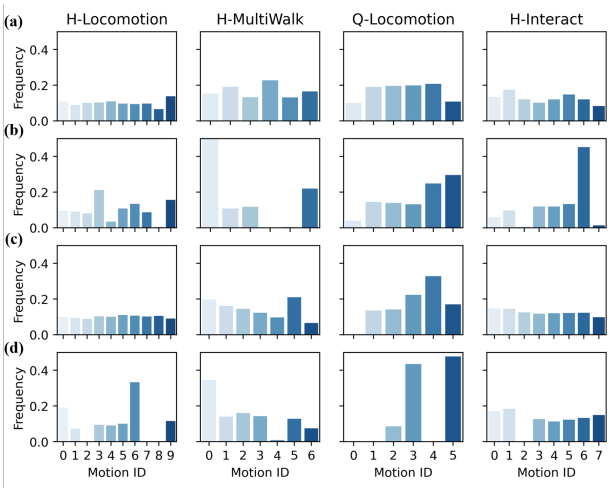


Fig. 3: Frequency distribution obtained by matching the controller-generated motion skills with the reference dataset under different tasks. Compared to the baselines **(b)** Baseline-I, **(c)** CAMP and **(d)** CALM, **(a)** CSI exhibits a **more even** distribution of motion skill coverage across different tasks.

A. Versatile Skills Integration

CSI can be applied to various legged robots to integrate different controllable motion skills. Fig. 2 in the supplementary qualitatively illustrates some of the motion skills integrated by the controllers for different tasks. It can be seen that CSI does not necessitate a high degree of stylistic similarity among the integrated motion skills. Both similar skills (e.g., pace and trot) and more distinct skills (e.g., dance and wave hello) can be seamlessly integrated by CSI. Furthermore, due to the incorporation of the IL paradigm, CSI does not require specific reward engineering for each motion skill. These characteristics enhance the generality of CSI for multi-skill integration tasks in legged robots.

The following experiments will quantitatively evaluate the performance of CSI's multi-skill integration. As an integrated skills library, it should encompass all reference motion skills. Therefore, we first validate the policy trained using CSI in terms of motion skill coverage. Specifically, a motion trajectory τ is obtained by conditioning the policy π with a randomly selected skill label $c \in C$. For each state transition pair (s_t, s_{t+1}) in trajectory τ, we apply motion matching to identify the motion clip m^* that contains the most similar motion transition pairs from the reference motion capture dataset D^M:

$$m^* = \underset{m \in D^M}{arg\ min} \underset{(\bar{s}_t, \bar{s}_{t+1}) \in m}{min} \|s_t - \bar{s}_t\|^2 + \|s_{t+1} - \bar{s}_{t+1}\|^2 \quad (11)$$

The process is repeated for each motion state transition pair in the motion trajectory τ. The motion category that receives the highest number of matches is considered as the category of the motion trajectory τ. In this experiment, a total of 2000 motion trajectories are collected and used to validate the motion skill coverage of different methods, with each skill label sampled with equal probability.

Fig. 3 shows the skill coverage of each method across different tasks. Compared to Baseline-I, CSI demonstrates more comprehensive and balanced mastery of all skills in each task, highlighting the effectiveness of the supervised learning paradigm for multi-skill integration tasks. CAMP also achieves extensive skill coverage across all tasks, attributed to the advantage of the Least Square GAN [39] in mitigating mode collapse compared to the vanilla GAN. Additionally, CALM exhibits significant degrees of mode collapse across all tasks, failing to integrate some skills into the controller. We attribute this primarily to its unsupervised learning paradigm.

In addition, the flexibility to switch between different motion skills is a crucial aspect of multi-skill integration. Specifically, a comprehensive and well-balanced probability distribution of skill transition is of significant interest for controllability. To evaluate this, we allow CSI and compared methods to generate motion trajectories by randomly sampling skill label c_1 at the first 200 time steps, followed by another skill label c_2 for the subsequent 200 time steps. For CALM, we instead switch between motion skills by randomly sampling reference motion clips and feeding them into CALM's motion encoder to obtain new latent codes.

Following this practice, a total of 2000 trajectories are collected. Eq. 11 is employed to determine the motion categories of the two motion trajectories that belonged before and after switching. Fig. 4 illustrates the skill transition probability distribution of the four methods on each task. It can be seen that CSI exhibits a more balanced capability to switch between different motion skills compared with its baselines.

B. Language-Directed Skill Control

In this experiment, we qualitatively demonstrate language-directed skill control task that CSI can accomplish when combined with the pre-trained NLI module. Based on the hierarchical combination approach described in Section III-D, we select the controller trained in the *H-Interaction* task as the low-level module. Text descriptions from the reference motion dataset used in this task are directly employed as skill caption labels. Further details can be found in TABLE I in the supplementary. For the high-level module, we adapt *bart-large-mnli*, a BART [40] model trained on MultiNLI dataset [41], specifically for NLI tasks.

In each experiment, we first issue textual commands (1) to *bart-large-mnli*, and then switch to textual commands (2) after 200 time steps. Fig. 5 presents three qualitative results on the language-directed skill control task. It can be seen that *bart-large-mnli* guides the low-level controller to generate semantically compliant skills based on the given textual commands, thereby facilitating the flexible switching between different motion skills.

C. Ablation Study

Weight Decay. As illustrated in Section III-B, the introduction of weight decay can increase the diversity of the generated motion skills. To quantitatively evaluate the

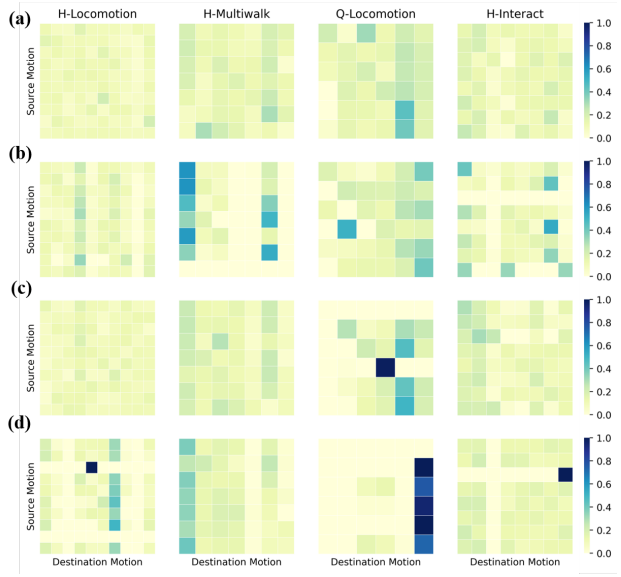


Fig. 4: Probability distributions of different motion skill transitions, where each row represents the probability of transferring from a source motion to each destination motion. Compared to the baselines **(b)** Baseline-I, **(c)** CAMP and **(d)** CALM, **(a)** CSI captures a **more balanced** distribution of motion skill transitions.

enhancement brought by weight decay, we adopt the Average Pairwise Distance (APD) [42], [43], which measures the diversity within a set of generated motion sequences. Specifically, given a set M containing N generated motion trajectories, each comprising a fixed length of L frames, the APD score of M is defined as:

$$APD(M) = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j \neq i}^{N} [\sum_{t=1}^{L} (\left\| s_t^i - s_t^j \right\|^2)]^{1/2} \qquad (12)$$

where s_t^i represents the $i-th$ state of trajectory m_i. A higher APD score indicates greater diversity among the motion states of the generated motion trajectories in M. For each task, we set the sampling probability of different skill labels to be equal and collect a total of 2000 motion trajectories, each with a fixed length of 200 steps. For each experiment, we perform 10 different samples and calculate the average APD as shown in TABLE II. Our method without weight decay is called Baseline-II. Evidently, the introduction of weight decay improves the diversity of generated motion skills on each task, which proves that weight decay effectively mitigates the overfitting of the discriminator.

V. CONCLUSION

In this work, we introduce CSI, a flexible framework that enables legged robots to acquire a wide range of controllable and diverse motion skills directly from demonstration motion data. This technology enables the rapid integration of multiple motion skills into a single controller. We believe this capability is advantageous for applications that require legged robots to possess a diverse set of skills. One future work is to deploy our work on real legged robots to validate its feasibility in real robotics applications. Another future work will focus on implementing more detailed motion skill control, such as controlling the direction or the velocity of motion skills. This will further improve the usefulness of our approach.

TABLE II: Skill diversity of different methods, where the scores for the best performance are bolded.

Task	H-Locomotion	H-MultiWalk	Q-Locomotion	H-Interaction
CAMP	1542.61	2023.37	1716.21	1629.36
CALM	1392.18	2334.03	1486.08	1790.76
Baseline-I	1634.51	1951.50	1397.09	1794.07
Baseline-II	1582.40	2249.88	1724.76	1775.34
CSI	**1743.63**	**2377.82**	**1952.63**	**1796.92**

REFERENCES

[1] Z. Zhuang, Z. Fu, J. Wang, C. G. Atkeson, S. Schwertfeger *et al.*, "Robot parkour learning," in *Proc. Conf. Robot Learn.*, 2023.

[2] D. Rodriguez and S. Behnke, "Deepwalk: Omnidirectional bipedal gait by deep reinforcement learning," *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 3033–3039, 2021.

[3] T. Haarnoja, B. Moran, G. Lever, S. H. Huang, D. Tirumala *et al.*, "Learning agile soccer skills for a bipedal robot with deep reinforcement learning," *Sci. Robot.*, vol. 9, no. 89, p. eadi8022, 2024.

[4] X. B. Peng, E. Coumans, T. Zhang, T.-W. E. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," *Robot.: Sci. Syst.*, 07 2020.

[5] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 143:1–143:14, Jul. 2018.

[6] K. Bergamin, S. Clavet, D. Holden, and J. R. Forbes, "Drecon: Data-driven responsive control of physics-based characters," *ACM Trans. Graph.*, vol. 38, no. 6, nov 2019.

[7] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "Amp: Adversarial motion priors for stylized physics-based character control," *ACM Trans. Graph.*, vol. 40, no. 4, Jul. 2021.

[8] Y. Wang, Z. Jiang, and J. Chen, "Learning robust, agile, natural legged locomotion skills in the wild," in *RoboLetics: Workshop on Robot Learning in Athletics @CoRL 2023*.

[9] A. Escontrela, X. B. Peng, W. Yu, T. Zhang, A. Iscen, K. Goldberg, and P. Abbeel, "Adversarial motion priors make good substitutes for complex reward functions," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, 2022, pp. 25–32.

[10] T. Li, Y. Zhang, C. Zhang, Q. Zhu, J. Sheng, W. Chi, C. Zhou, and L. Han, "Learning terrain-adaptive locomotion with agile behaviors by imitating animals," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 339–345.

[11] A. Tang, T. Hiraoka, N. Hiraoka, F. Shi, K. Kawaharazuka, K. Kojima, K. Okada, and M. Inaba, "Humanmimic: Learning natural locomotion and transitions for humanoid robot via wasserstein adversarial imitation," *arXiv preprint arXiv:2309.14225*, 2023.

[12] E. Vollenweider, M. Bjelonic, V. Klemm, N. Rudin, J. Lee, and M. Hutter, "Advanced skills through multiple adversarial motion priors in reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 5120–5126.

[13] L. Han, Q. Zhu, J. Sheng, C. Zhang, T. Li, Y. Zhang, H. Zhang *et al.*, "Lifelike agility and play in quadrupedal robots using reinforcement learning and generative pre-trained models," *Nat. Mach. Intell.*, Jul 2024.

[14] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Adv. neural inf. proces. syst.*, vol. 29, 2016.

[15] G. Ficht, H. Farazi, D. Rodriguez, D. Pavlichenko, P. Allgeuer, A. Brandenburger, and S. Behnke, "Nimbro-op2x: Affordable adult-sized 3d-printed open-source humanoid robot for research," *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, vol. 17, no. 05, p. 2050021, 2020.

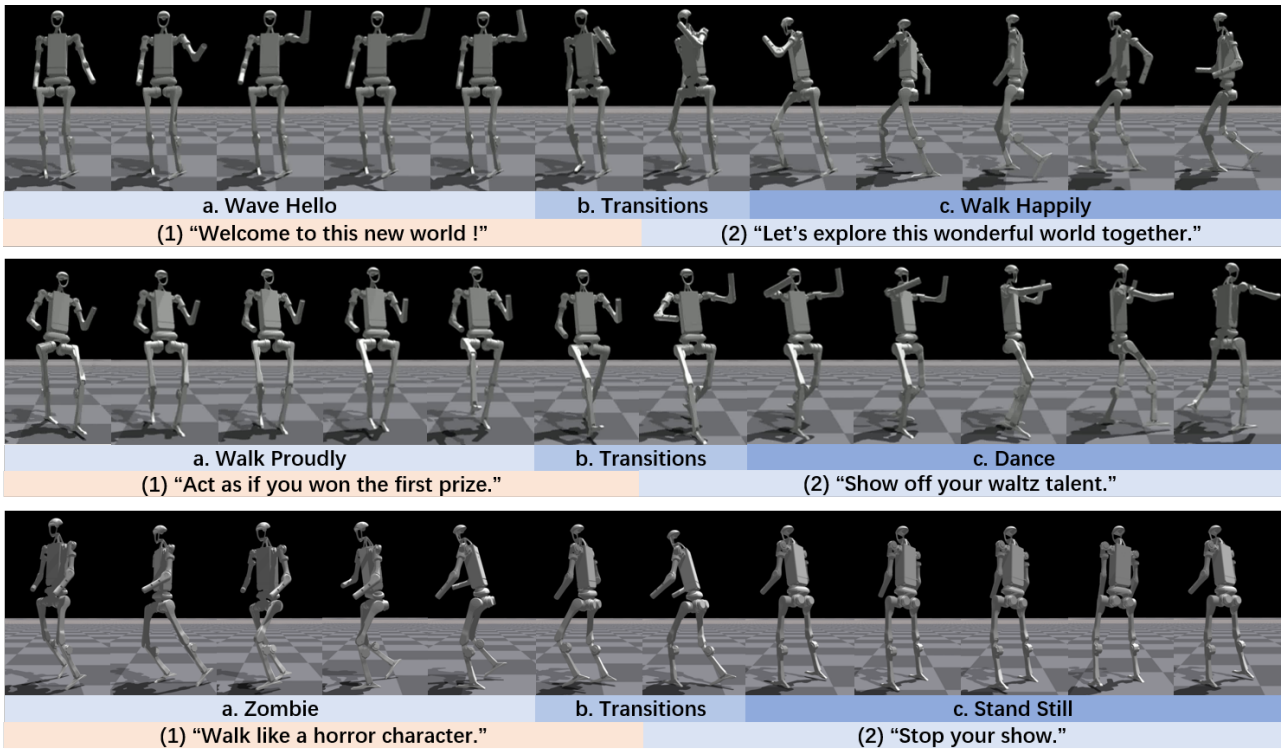


Fig. 5: Snapshots of the motion sequences generated by CSI, where the sub-skills (first row) and corresponding text commands (second row) are also provided. Our CSI enables language-directed skill control and transition by combining with a high-level NLI module.

[16] "Unitree go1," https://www.unitree.com/go1/.

[17] "Unitree a1," https://www.unitree.com/a1/.

[18] "robot op3 source code," https://github.com/ROBOTIS-GIT/ROBOTIS-OP3.

[19] J. Won, D. Gopinath, and J. Hodgins, "A scalable approach to control diverse behaviors for physically simulated characters," *ACM Trans. Graph.*, vol. 39, no. 4, aug 2020.

[20] "Cmu mocap dataset," http://mocap.cs.cmu.edu/.

[21] X. B. Peng, Y. Guo, L. Halper, S. Levine, and S. Fidler, "Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters," *ACM Trans. Graph.*, vol. 41, no. 4, Jul. 2022.

[22] Q. Zhu, H. Zhang, M. Lan, and L. Han, "Neural categorical priors for physics-based character control," *ACM Trans. Graph.*, vol. 42, no. 6, dec 2023.

[23] A. Van Den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," *Adv. neural inf. proces. syst.*, vol. 30, 2017.

[24] C. Li, M. Vlastelica, S. Blaes, J. Frey, F. Grimminger, and G. Martius, "Learning agile skills via adversarial imitation of rough partial demonstrations," in *Proc. Conf. Robot Learn.*, 2023, pp. 342–352.

[25] E. Chane-Sane, C. Schmid, and I. Laptev, "Goal-conditioned reinforcement learning with imagined subgoals," in *Int. Conf. Mach. Learn., ICML*, 2021, pp. 1430–1440.

[26] F. Torabi, G. Warnell, and P. Stone, "Adversarial imitation learning from state-only demonstrations," in *Proc. Int. Joint Conf. Auton. Agents Multiagent Syst., AAMAS*, 2019, pp. 2229–2231.

[27] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Adv. neural inf. proces. syst.*, vol. 27, 2014.

[28] M. Menéndez, J. Pardo, L. Pardo, and M. Pardo, "The jensen-shannon divergence," *J. Franklin Inst.*, vol. 334, no. 2, pp. 307–318, 1997.

[29] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein gans," in *Adv. neural inf. proces. syst.*, 2017, p. 5769–5779.

[30] L. Mescheder, A. Geiger, and S. Nowozin, "Which training methods for gans do actually converge?" in *Int. Conf. Mach. Learn., ICML*, 2018, pp. 3481–3490.

[31] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin *et al.*, "Isaac gym: High performance gpu based physics simulation for robot learning," 2021.

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

[33] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Proc. Conf. Robot Learn.*, 2022, pp. 91–100.

[34] "Unitree aliengo," https://www.unitree.com/aliengo/.

[35] "Westwood robotics," https://www.westwoodrobotics.io/bruce/.

[36] "Unitree h1," https://www.unitree.com/h1/.

[37] H. Zhang, S. Starke, T. Komura, and J. Saito, "Mode-adaptive neural networks for quadruped motion control," *ACM Trans. Graph.*, vol. 37, no. 4, jul 2018.

[38] C. Tessler, Y. Kasten, Y. Guo, S. Mannor, G. Chechik, and X. B. Peng, "Calm: Conditional adversarial latent models for directable virtual characters," in *Proc. - SIGGRAPH Conf. Pap.*, 2023.

[39] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vision*, 2017, pp. 2794–2802.

[40] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proc. Annu. Meet. Assoc. Comput. Linguist.*, 2020, pp. 7871–7880.

[41] A. Williams, N. Nangia, and S. Bowman, "A broad-coverage challenge corpus for sentence understanding through inference," in *NAACL HLT - Conf. N. Am. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol. - Proc. Conf.*, 2018, pp. 1112–1122.

[42] M. Hassan, D. Ceylan, R. Villegas, J. Saito, J. Yang, Y. Zhou, and M. Black, "Stochastic scene-aware motion prediction," in *Proc. IEEE Int. Conf. Comput. Vision*, 2021, pp. 11 354–11 364.

[43] Z. Dou, X. Chen, Q. Fan, T. Komura, and W. Wang, "C·ase: Learning conditional adversarial skill embeddings for physics-based characters," in *Proc. - SIGGRAPH Asia Conf. Pap., SA*, 2023.

Supplementary Material for 'Integrating Controllable Motion Skills from Demonstrations'

TABLE I: Detailed statistics of motion capture dataset used in different robots and tasks.

Robot	Task	Motion	Duration
BRUCE	H-Locomotion	Walk Forward	2.55s
		Walk Backward	2.32s
		Walk Rightward	2.32s
		Walk Leftward	2.98s
		Turn Right	1.48s
		Turn Left	1.15s
		Turn Right in Place	1.15s
		Turn Left in Place	1.65s
		Run	0.83s
		Jump	1.65s
	H-WalkStyle	Walk Briskly	2.10s
		Walk Hobble	3.65s
		Walk Stealthily	4.15s
		Zombie	4.98s
		Walk Happily	4.98s
		Marching	4.98s
		Walk Slowly	3.65s
AlienGo	Q-Locomotion	Pace	0.64s
		Trot	0.54s
		Pace Backward	0.64s
		Trot Backward	0.54s
		Turn Left	0.64s
		Turn Right	0.76s
H1	H-Interact	Dance	5.02s
		Hand Shake	4.82s
		Stand Still	4.72s
		Walk Forward	2.55s
		Wave Hello	5.38s
		Walk Proudly	5.22s
		Walk Happily	3.12s
		Zombie	4.93s

A. Hyperparameters

Adam is used as optimizer for the policy, the value function, and the discriminator in this work, with a fixed learning rate during training. Detailed hyperparameter settings for CSI are shown in TABLE II. We find that these hyperparameter combinations are suitable for training all tasks in our experiment.

B. Baseline Settings

Due to the substantial increase in training costs associated with MultiAMP [1] and [2] as the number of integrated motion skills grows, we have chosen not to use them as baselines. For a fair comparison, all baselines share the same observation space and action space, and for the same robot, the PD controller parameters used are also identical. In addition, the policies, the value functions, and discriminators used in all baselines are set to the same parameter sizes. Some specific training settings for CAMP and CALM are briefly described below.

TABLE II: Hyperparameters of CSI

Parameters	Value
Style-Reward Weight	1.0
Conditional Imitation Loss Weight	1.0
Condition Aware Loss Weight	1.0
Weight Decay Loss Weight	0.0001
Gradient Penalty Weight	5.0
DoF Velocity Penalty Weight	-1e-4
Action Rate Penalty Weight	-1e-2
Energy Penalty Weight	-2e-5
Torque Penalty Weight	-1e-4
Adjust Ratio	0.5
Discriminator Batch Size	512
MiniBatch Size	32768
Learning Rate(for all network)	5e-5
Discount Factor	0.95
Discriminator Replay Buffer Size	1e6
PPO Clip Threshold	0.2
GAE	0.95

TABLE III: Hyperparameters of CALM

Parameters	Value
Motion Encoder MLP Size	[1024, 512]
Policy and Value Function MLP Size	[256, 256]
Conditional Discriminator MLP Size	[256, 256]
Encoder Obs Steps	60
Learning Rate	2e-5
Condition Style Reward Weight	1.0
Encoder Regularization Coeff	0.1
Latent Dimension	64
Discriminator Batch Size	512
MiniBatch Size	16384
Gradient Penalty Weight	5.0
Discount Factor	0.99
GAE	0.95
PPO Clip Threshold	0.2

CAMP The overall architecture of CAMP is similar to that of AMP [3], but one-hot coding of motion skill labels is added to the observation inputs, and the training objective of the discriminator is modified to a conditional loss:

$$\begin{aligned} L = & -E_{(s_t,s_{t+1})\in d^{\mathcal{M}}}[D(s_t,s_{t+1}|c)-1]^2 \\ & -E_{(s_t,s_{t+1})\in d^{\pi}}[D(s_t,s_{t+1}|c)+1]^2 \\ & +\omega_{gp}E_{(s_t,s_{t+1})\in d^{\mathcal{M}}}[\|\nabla D(s_t,s_{t+1})\|_2^2] \end{aligned} \tag{1}$$

Accordingly, conditional style reward in CAMP is defined as:

$$r_s = max[0, 1-0.25(D(s_t,s_{t+1}|c)-1)^2] \tag{2}$$

Finally, the hyperparameter settings used for CAMP are the same as that of CSI.

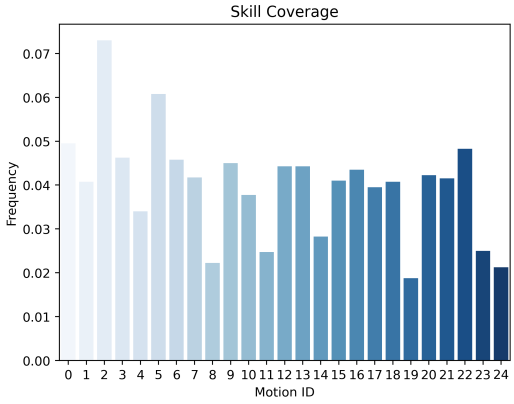


Fig. 1: CSI demonstrates great scalability even if the number of skills to be integrated increases significantly.

CALM With motion encoder and latent space, after the pre-training stage, the controller trained by CALM [4] can control a specific integrated skill by providing a short clip of corresponding reference motion data as motion encoder's input. Therefore, in our experimental setup, baseline CALM is only subjected to the pre-training stage. Except for some hyperparameter modifications, we have preserved as much as possible the default settings in the open-source code of CALM. Some key hyperparameter settings used for CALM are shown in TABLE III. It is also worth noting that the motion state representation used for the CALM's motion encoder is the same as that used for its discriminator. Finally, for a fair comparison, we adjust all the observations of CALM's policy, value function, and discriminator to be the same as other baselines (except condition input).

C. Scalability Analysis

This section demonstrates the scalability of CSI when the number of reference motion skills increases. We significantly increase the number of motion skills required to be integrated into H1 by constructing a dataset containing 25 different reference motion skills. This dataset contains all the reference datasets used by the humanoid robots in this paper, as well as a set of similar mirror motion skills (*wave left hand* and *wave right hand*, *left leg kicking* and *right leg kicking*, *left hand shake* and *right hand shake*). Additionally, we keep the size of each network unchanged and only extend the training duration to 2.5 times of the original. Fig. 1 illustrates the coverage of motion skills. CSI demonstrates excellent scalability by mastering all motion skills despite a significant increase in the number of skills to be integrated and the inclusion of a set of similar motion skills.

D. Generality Analysis

This section briefly analyzes the generality of CSI. As shown in Fig. 2, CSI can be applied to different humanoid robots BRUCE and H1 as well as quadruped robot AlienGo, and they share the same training process. For other humanoid robots, CSI is also theoretically applicable, but the difficulty lies in how to kinematically retarget the motion capture data for the robot to obtain the corresponding reference motion dataset. Additionally, the motion capture data of quadrupedal creatures is more difficult to obtain than that of human beings, so the source of the reference motion skill is a major obstacle to the application of CSI on quadruped robots. In this regard, we believe that future work is necessary to further expand more sources of reference motions, such as extracting reference motions from videos [5], or obtaining reference motions from generative models [6], [7]. These expanded reference motion data sources will further enhance the generality of CSI.

Another point is that CSI requires less correlation between motion skills that need to be integrated. The reference motion skills that can be integrated by previous work [8], [9], [10] are limited by the task objective. For example, in locomotion tasks, policies can usually only integrate task-relevant motion skills such as walking, standing, steering, and running. Integrating additional jumping skills would require further adjustments to the task objective. CSI does not have any obvious relevance requirements for this, as shown in Fig. 2, where similar motion skills, such as *trot* and *pace*, as well as very different motion skills, such as *dance* and *zombie*, can be integrated, highlighting the generality of CSI for multi-skill integration.

E. Initialization

We adopt a mixed initialization strategy to accelerate motion skill learning. At the beginning of each episode, a set of initialization state and skill label pairs s, c are selected to initialize each agent, where 70% of s are sampled from d^M, the remaining 30% is set to default state, and all skill labels c are randomly sampled from skill label space C. Note that c and s may derive from different reference motion skills, which would be beneficial for learning to switch between different motion skills.

F. Skill labelling

In this work, we crop all the reference motion clips used so that they contain only one motion skill. We then artificially assign a unique skill label to each clip. Actually, for unstructured reference motion data, such as motion capture data clips with a mixture of multiple motion skills, skill labels can be considered to be acquired by pre-trained skeleton-based action recognition networks like [11], [12], and similar practice has been reported in [13].

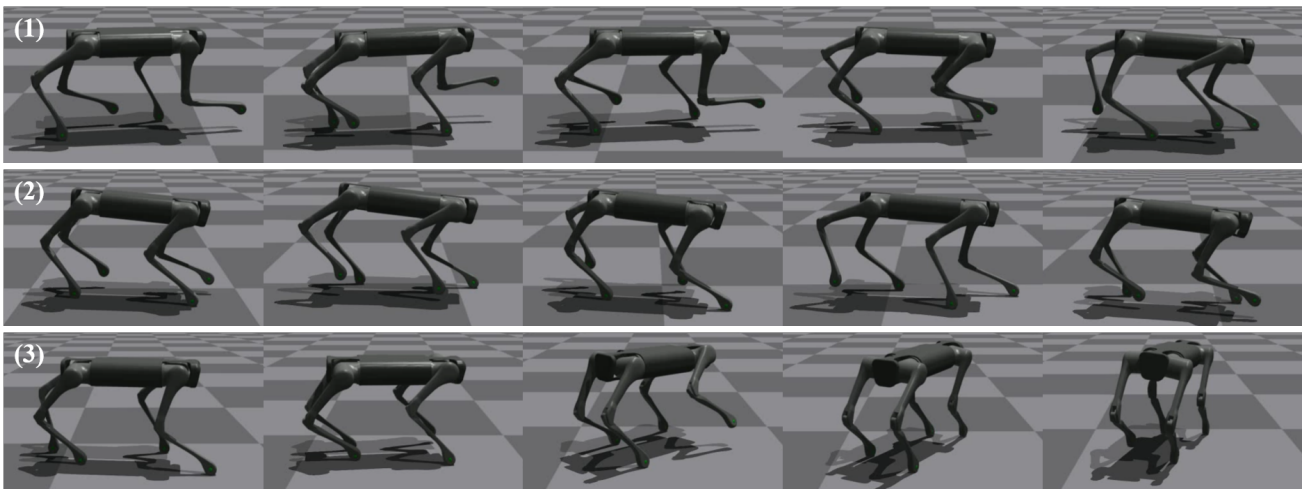
REFERENCES

[1] E. Vollenweider, M. Bjelonic, V. Klemm, N. Rudin, J. Lee, and M. Hutter, "Advanced skills through multiple adversarial motion priors in reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 5120–5126.

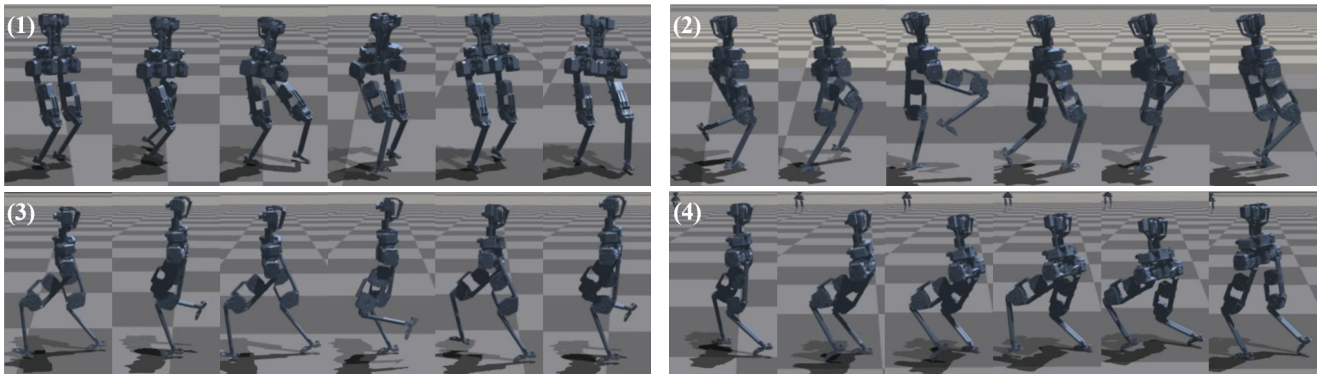
[2] L. Han, Q. Zhu, J. Sheng, C. Zhang, T. Li, Y. Zhang, H. Zhang *et al.*, "Lifelike agility and play in quadrupedal robots using reinforcement learning and generative pre-trained models," *Nat. Mach. Intell.*, Jul 2024.

[3] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "Amp: Adversarial motion priors for stylized physics-based character control," *ACM Trans. Graph.*, vol. 40, no. 4, Jul. 2021.

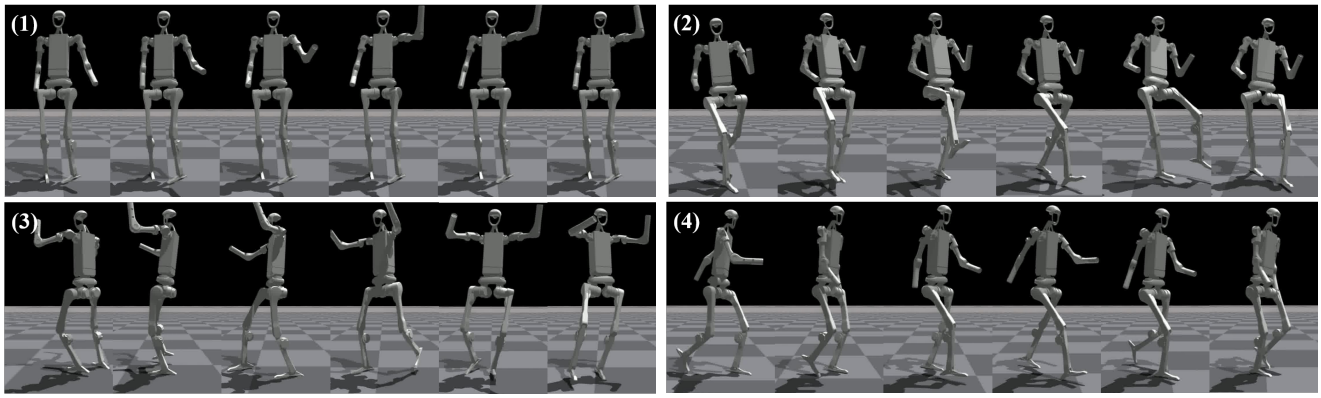
[4] C. Tessler, Y. Kasten, Y. Guo, S. Mannor, G. Chechik, and X. B. Peng, "Calm: Conditional adversarial latent models for directable virtual characters," in *Proc. - SIGGRAPH Conf. Pap.*, 2023.



(a) AlienGo: **(1)Trot, (2)Pace, (3)Turn**



(b) BRUCE: **(1)Zombie, (2)Marching, (3)Walk Happily, (4)Walk Stealthily**



(c) H1: **(1)Wave Hello, (2)Walk Proudly, (3)Dance, (4)Walk Forward**

Fig. 2: CSI can be applied to different legged robots for versatile and flexible multi-skill integration.

[5] Z. Cai, W. Yin, A. Zeng, C. Wei, Q. Sun, W. Yanjun, H. E. Pang *et al.*, "Smpler-x: Scaling up expressive human pose and shape estimation," *Adv. neural inf. proces. syst.*, vol. 36, 2024.

[6] B. Jiang, X. Chen, W. Liu, J. Yu, G. Yu, and T. Chen, "Motiongpt: Human motion as a foreign language," *Adv. neural inf. proces. syst.*, vol. 36, 2024.

[7] G. Tevet, S. Raab, B. Gordon, Y. Shafir, D. Cohen-or, and A. H. Bermano, "Human motion diffusion model," in *ICLR - Int. Conf. Learn. Represent.*, 2023.

[8] A. Escontrela, X. B. Peng, W. Yu, T. Zhang, A. Iscen, K. Goldberg, and P. Abbeel, "Adversarial motion priors make good substitutes for complex reward functions," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, 2022, pp. 25–32.

[9] T. Li, Y. Zhang, C. Zhang, Q. Zhu, J. Sheng, W. Chi, C. Zhou, and L. Han, "Learning terrain-adaptive locomotion with agile behaviors by imitating animals," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 339–345.

[10] A. Tang, T. Hiraoka, N. Hiraoka, F. Shi, K. Kawaharazuka, K. Kojima, K. Okada, and M. Inaba, "Humanmimic: Learning natural locomotion and transitions for humanoid robot via wasserstein adversarial imitation," *arXiv preprint arXiv:2309.14225*, 2023.

[11] H. Duan, Y. Zhao, K. Chen, D. Lin, and B. Dai, "Revisiting skeleton-based action recognition," in *Proc IEEE Comput Soc Conf Comput Vision Pattern Recognit*, 2022, pp. 2959–2968.

[12] H. Duan, J. Wang, K. Chen, and D. Lin, "Pyskl: Towards good practices for skeleton action recognition," in *MM - Proc. ACM Int. Conf. Multimed.*, 2022, p. 7351–7354.

[13] Z. Dou, X. Chen, Q. Fan, T. Komura, and W. Wang, "C·ase: Learning conditional adversarial skill embeddings for physics-based characters," in *Proc. - SIGGRAPH Asia Conf. Pap., SA*, 2023.