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

M2oE: Modular Mixture of Experts for Multi-Morphology Reinforcement Learning of Modular Robots

Chang Liu[1,2], Qinchao Xu[1], Satoshi Yagi[1], Satoshi Yamamori[1,3],
Yaonan Zhu[2], Yusuke Iwasawa[2], Kazuya Yoshida[4], Jun Morimoto[1,3]

***Abstract*— Modular robots offer a promising solution for building versatile and adaptable robotic systems. For instance, space exploration robots can be designed to reconfigure to meet diverse task demands across varying environments. However, training such systems by Reinforcement Learning (RL) remains challenging due to the diversity of morphologies and the lack of simulation environments that support simultaneous multi-morphology learning. We present Modular Mixture of Experts (M2oE), a novel reinforcement learning backbone network that imitates the modular structure of robots to enable efficient and module-wise parallelizable policy learning for modular robots. In M2oE, the shared pool of experts, combined with an attention-based gating mechanism that dynamically selects experts based on inter-module correlations, enables both specialization and generalization. This structure supports training across multiple morphologies within a single framework, avoiding gradient conflicts and enhancing experience sharing across modules and morphologies. To support training, we also extend the Isaac Lab simulator with multi-morphology extensions that enable concurrent training across diverse robot configurations. Experiments on a space-exploration-inspired modular robot, Moonbot, demonstrate that M2oE significantly improves learning efficiency and achieves superior performance compared to both MLP and Transformer baselines. More information and the project video are available on the project website: `https://ryuuchou17.github.io/m2oe/`**

I. INTRODUCTION

Modular robots are robotic systems composed of multiple interconnected modules, each capable of performing specific functions and reconfiguring to adapt to varying tasks and environments [1]. Compared with traditional robots, their modular structure provides greater reconfigurability and scalability, enabling rearrangement of modules to suit different operational scenarios [2]. For instance, in space exploration, modular robots can dynamically adjust their configurations to traverse heterogeneous terrains and execute complex tasks [3], [4].

*This work was supported by JST Moonshot R&D, Grant Number: JPMJMS223B-3, and JSPS KAKENHI Grant Number: 22H04998 and 23K24925.

[1]Graduate School of Informatics, Kyoto University, Kyoto, Japan. {liu.chang, xu.qinchao, yamamori}@lm.sys.i.-kyoto-u.ac.jp, {yagi, morimoto}@i.kyoto-u.ac.-jp

[2]School of Engineering, The University of Tokyo, Tokyo, Japan. {chang.liu, yaonan.zhu, iwasawa}@weblab.t.-u-tokyo.ac.jp

[3]Dept. of Brain Robot Interface, Computational Neuroscience Labs, ATR, Kyoto, Japan. yamamori@atr.jp

[4]Dept. of Aerospace Engineering, Graduate School of Engineering, Tohoku University, Sendai 980-8579, Japan. yoshida.astro-@tohoku.ac.jp

Fig. 1. Three morphologies of the modular robot Moonbot: Minimal (left), Dragon (middle), and Tricycle (right). These diverse configurations pose challenges for reinforcement learning policies to generalize effectively across morphologies.

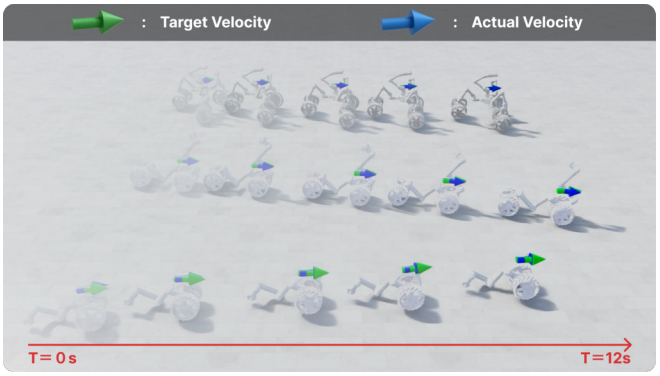

Fig. 2. Locomotion performance of multiple Moonbot morphologies trained with M2oE and the multi-morphology extension in Isaac Lab. All morphologies are controlled by the same M2oE policy and reliably track target velocity commands while maintaining stability and balance.

The complexity and diversity of morphologies in modular robots lead to significant challenges for control and policy learning. Nowadays, Deep Reinforcement Learning (DRL) has emerged as a powerful method for training diverse robots, ranging from quadrupeds [5]–[7] to humanoids [8]. Recent researches [9] have even achieved impressive human-motion-conditioned teleoperation and action generation using DRL. Researches have also explored DRL for modular robots [10], [11], demonstrating its potential in enabling robots to adapt to different configurations and tasks.

Especially for space robots, the lack of similar environments on Earth makes real-world training and testing impractical. Therefore, simulators play an important role in training space robots, providing a safe and cost-effective platform for policy learning and evaluation. GPU-accelerated simulators such as Isaac Lab [12] enable scalable parallel training, allowing efficient exploration of diverse tasks and

scenarios. These simulators are particularly valuable for space robots, where real-world deployment and testing on planets or moons are prohibitively costly and risky.

Recent advances in DRL and high-performance simulators enable scalable parallel training in simulation, providing a foundation for learning control policies across multiple morphologies. However, applying DRL to modular space robots remains challenging due to several factors:

- **Morphological Diversity:** Modular robots may differ in their module composition and structural configuration, which makes it challenging for a single policy to generalize across these morphologies because of the conflicting gradients [5].
- **Simulation Constraints:** Simulators play a crucial role in training space modular robots. Most existing simulators are designed for fixed robot configurations and do not explicitly support concurrent multi-morphology training, which limits experience sharing across configurations.

An example of space modular robots is **Moonbot** [4], as illustrated in Figure 1, consisting of multiple basic modules. The various morphologies of Moonbot can adapt to different terrains. Training a single locomotion policy that effectively controls all morphologies of Moonbot remains challenging due to the aforementioned issues.

To tackle these challenges, we introduce the **Modular Mixture of Experts (M2oE)**, a novel reinforcement backbone framework designed for modular robots. M2oE serves as a DRL backbone network that enables policy learning across multiple morphologies within a shared modular embodiment. Its inherently module-wise parallelizable structure aligns with the nature of modular robots and facilitates efficient training through experience sharing.

In addition, we extend the Isaac Lab simulator to support multi-morphology training, allowing for concurrent policy learning across multiple robot configurations. As illustrated in Figure 2, we evaluate M2oE on a locomotion task involving three distinct Moonbot morphologies enabled by this extension. The results demonstrate that a single M2oE policy can reliably track target velocity commands while maintaining stability and balance across **all evaluated morphologies**. Extended discussions are provided in the following sections.

The main contributions of this work are summarized as follows:

- **M2oE Structure:** M2oE introduces an efficient learning backbone for modular robot design by employing a module-wise Mixture of Experts (MoE) structure. It employs an attention-based gating mechanism that dynamically selects experts based on inter-module correlations.
- **Multi-Morphology Training Extension:** We extend the Isaac Lab simulator with functionality for concurrent training across multiple morphologies. We design a group-based training pipeline and introduce tools such as a multi-layer terrain generator.
- **Performance Evaluation:** We evaluate M2oE on Moonbot, and demonstrate that it achieves higher learning efficiency and performance compared to baseline approaches.

The remainder of this paper is organized as follows. Section II reviews related work. Section III describes the proposed M2oE framework. Section IV details the experimental setup, and Section V presents the results. Finally, Section VI concludes the paper.

II. RELATED WORKS

A. Multi-morphology Training

Learning a single policy that operates across heterogeneous robot morphologies has achieved rapid progress. Previous works have explored various approaches to address the challenges of multi-morphology training. These include morphology-agnostic policies that normalize embodiment differences using padding/masking or encoder–decoder interfaces, enabling a single network to control multiple morphologies [6]. Attention-based architectures [13], [14] have also scaled cross-embodied imitation/reinforcement learning with a Transformer that casts variable-length observations and actions into tokens, supporting diverse morphologies. Morphology-aware policies [15], [16] leverage graph neural networks (GNNs) or Transformers to capture inter-joint correlations, enabling generalization across different robot structures.

B. Reinforcement Learning for Modular Robots

Modular robots have also garnered significant attention in recent years due to their adaptability and versatility. They have been applied in various domains, including space exploration [3], search and rescue [17], [18], and industrial automation [19]. However, traditional control methods often struggle to handle the complexity and variability of modular systems. Writing controllers for each possible configuration is impractical.

DRL has emerged as a promising approach for training robots, enabling them to learn complex behaviors through interaction with the environment [20], [21]. In scope of modular robots, DRL has also demonstrated potential in enabling robots to adapt to different configurations and tasks [22], [23]. Whitman et al. [10] proposed a graph neural network (GNN)-based policy architecture that represents each robot design as a graph structure. Combined with a model-based learning pipeline, their framework enables generalization across unseen robot designs. However, as a model-based approach, its performance may be sensitive to dynamics modeling errors and distribution shift. Sun et al. [11] introduced a hierarchical DRL framework that jointly optimizes morphology and control policies through co-evolution. While effective for exploring morphology–control co-design, training is typically performed on one morphology instance at a time, which limits concurrent experience sharing across multiple configurations within a unified policy.

C. Mixture of Experts (MoE)

Mixture of Experts (MoE) has become a workhorse in large-scale NLP/CV models [24]–[26], enabling efficient

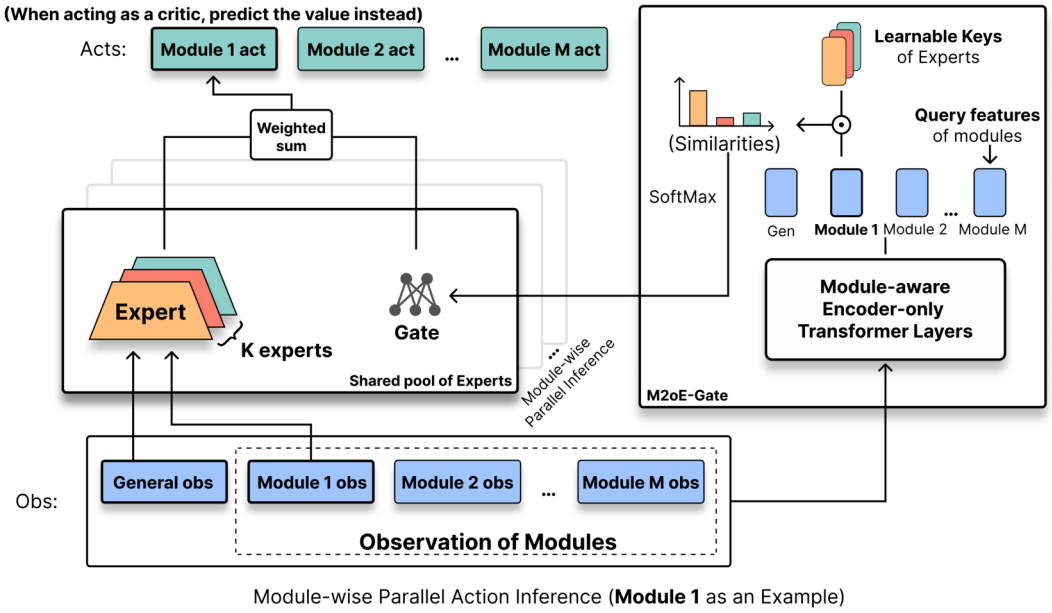

Fig. 3. **Structure of Modular Mixture of Experts (M2oE)**. There are two main components: **a module-wise shared pool of experts** and **an attention-based gating mechanism (M2oE-Gate)** that extract inter-module correlations to select experts for each module. The design of M2oE enables module-wise parallelism and automatic adaptation to morphologies with different numbers of modules. This figure illustrates the action inference process for the first module in a robot with M modules.

scaling of model capacity while maintaining computational efficiency. Its modular structure allows different experts to specialize in different aspects of the input data, promoting both specialization and generalization. Recent robotics work begins to exploit MoE to mitigate gradient conflicts and decouple skills in multi-task learning [5]. Broader MoE variant for embody AI (e.g., MoRE [27]) further highlights MoE's promise for scalable learning in robotics.

In contrast to prior MoE applications that primarily address task-level diversity or skill decoupling, our work explicitly aligns the MoE structure with the physical modularity of robots. M2oE performs module-wise expert routing based on inter-module correlations and enables concurrent training across multiple morphologies within a unified policy. This design allows expert specialization at the structural level rather than the task level, making it particularly suitable for multi-morphology reinforcement learning.

III. METHOD

In this section, we introduce the proposed **Modular Mixture of Experts (M2oE)** framework and its components: the task setup, the M2oE architecture, the attention-based gating mechanism, and our extension of the Isaac Lab simulator to support multi-morphology training.

A. Task Setup

We consider a modular robot consisting of M modules, each with its own set of sensors and actuators. The observation space of a modular robot is defined as the concatenation of general observations (whole-body states and task-specific states) and module-specific observations (e.g., joint positions, velocities, and torques):

$$\mathbf{o} = \left[\mathbf{o}_{\text{general}}, \mathbf{o}^{(1)}_{\text{module}}, \mathbf{o}^{(2)}_{\text{module}}, \ldots, \mathbf{o}^{(M)}_{\text{module}}\right], \quad (1)$$

where $\mathbf{o}^{(m)}_{\text{module}}$ represents the observations from the m-th module, and $\mathbf{o}_{\text{general}}$ represents the general observations.

Similarly, the action space is defined as the concatenation of module-specific actions:

$$\mathbf{a} = \left[\mathbf{a}^{(1)}_{\text{module}}, \mathbf{a}^{(2)}_{\text{module}}, \ldots, \mathbf{a}^{(M)}_{\text{module}}\right]. \quad (2)$$

TABLE I

OBSERVATION AND ACTION SPACE DIMENSIONS OF DIFFERENT MOONBOT MORPHOLOGIES.

	Observation Space		Action Space	
Type	Name	Dim.	Name	Dim.
Modular	Joint Position	9		
	Joint Velocity	9		
	Module Base Position	3	Arm Action	7
	End-Effector Position	3	Wheel Action	2
	Last Action	9		
	Total	**33**		**9**
General	Base Linear Velocity	3		
	Base Angular Velocity	3		
	Command Velocity	3	——	—
	Gravity Projection	3		
	Total	**12**		
Morphology Summary				
Morphology (# of Modules)	Observation Dim.		Action Dim.	
Minimal (1)	45		9	
Dragon (2)	78		18	
Tricycle (3)	111		27	

TABLE II

REWARD TERMS AND WEIGHTS USED IN THE LOCOMOTION TASK.

Term	Function	Weight
Linear Velocity Tracking (xy)	$\exp\left(-\frac{\|\mathbf{v}_{cmd}^{xy}-\mathbf{v}_{obs}^{xy}\|_2^2}{\sigma^2}\right)$	6.0
Angular Velocity Tracking (yaw)	$\exp\left(-\frac{(\omega_{cmd}^{z}-\omega_{obs}^{z})^2}{\sigma^2}\right)$	3.0
Difference from Initial Pose	$\exp\left(-\left(\frac{1}{0.25}\,\text{mean}\left(\lvert\mathbf{q}_{leg}-\mathbf{q}_{leg}^{0}\rvert\right)\right)^2\right)$	5.0
Wheel Rolling Consistency	$\frac{1}{N_{wheel}}\sum_{i=1}^{N_{wheel}}\left\lvert\lvert\omega_i r\rvert-\lVert\mathbf{v}_{cmd}^{xy}\rVert_2\right\rvert$	-1.0
Project of Gravity	$\lVert\mathbf{g}_b^{xy}\rVert_2^2$	-5.0
Undesired Contact	$\sum_c \mathbf{1}(\lVert\mathbf{f}_c\rVert_2>\tau)$	-0.5
Linear Velocity (z)	$\left(v_{obs}^{z}\right)^2$	-2.0
Angular Velocity (x, y)	$\lVert\boldsymbol{\omega}_{obs}^{xy}\rVert_2^2$	-0.1
Joint Acceleration	$\lVert\ddot{\mathbf{q}}_{leg}\rVert_2^2$	-2e-5
Joint Velocity	$\lVert\dot{\mathbf{q}}_{leg}\rVert_2^2$	-0.02
Joint Power	$\sum_{j\in\mathcal{J}_{leg}}\lvert\dot{q}_{leg,j}\,\tau_{leg,j}\rvert$	-5e-4
Action Rate	$\sum_{j\in\mathcal{J}_{leg}}(a_{j,t}-a_{j,t-1})^2$	-0.01

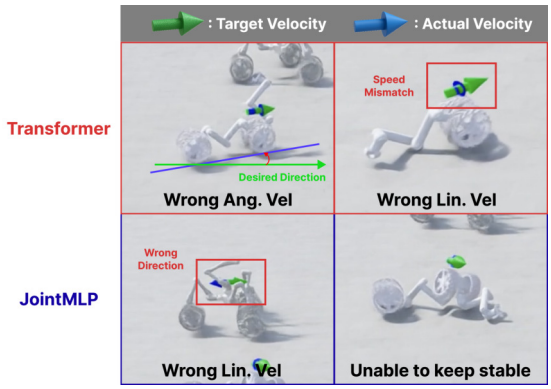

Fig. 4. **Failure cases of baseline methods** on the locomotion task across three Moonbot morphologies. Some morphologies fail to move forward or maintain balance, while others exhibit undesired yaw rotations.

To align with the decentralized nature of modular robots, during training, module ordering is randomized and modules are treated independently without fixed positional binding.

A DRL policy π is defined as a distribution over actions given the observations, parameterized by a neural network:

$$\pi(\mathbf{a}|\mathbf{o}) = \prod_{m=1}^{M} \pi(\mathbf{a}_{\text{module}}^{(m)} \mid \mathbf{o}_{\text{module}}^{(m)}, \mathbf{o}_{\text{general}}). \quad (3)$$

In our setup, M2oE serves as the backbone network for the DRL policy and is optimized using Proximal Policy Optimization (PPO) [28].

B. Modular Mixture of Experts (M2oE)

M2oE is a modular and scalable DRL backbone that mimics the modular structure of robots while leveraging the Mixture of Experts (MoE) paradigm [29], [30]. MoE has demonstrated effectiveness in various domains, including natural language processing [24], [25], computer vision [26], and robotics [5]. It has been proven effective in addressing gradient conflicts in multi-task learning. This framework promotes **experience sharing** across modules and morphologies, while also enabling **specialization** for specific configurations.

The whole framework is illustrated in Figure 3. M2oE consists of two main components:

TABLE III

AVERAGE STEP ERROR DURING TEST EPISODES ACROSS DIFFERENT MORPHOLOGIES. LOWER IS BETTER.

Model	Minimal		Dragon	
	Lin.	Ang.	Lin.	Ang.
JointMLP	0.1938	0.5897	0.4276	0.6457
Transformer	0.2547	0.6489	0.1087	0.6491
M2oE (Ours)	**0.1464**	**0.1977**	**0.0989**	**0.2100**
Model	**Tricycle**		**Average**	
	Lin.	Ang.	Lin.	Ang.
JointMLP	0.9454	0.6281	0.5223	0.6212
Transformer	0.1361	0.6516	0.1665	0.6499
M2oE (Ours)	**0.1257**	**0.1445**	**0.1237**	**0.1841**

- **Shared Pool of Experts:** Modules across different morphologies share a pool of K experts $[\pi_{\text{expert}}^{(1)}, \ldots, \pi_{\text{expert}}^{(K)}]$. This promotes knowledge transfer across morphologies and mitigates gradient conflicts through distributed specialization.
- **M2oE-Gate** M2oE-Gate is an attention-based gating mechanism that computes inter-module correlations. The output module features serve as queries, and their similarities with expert learnable key vectors determine the gating weights.

By leveraging a shared pool of experts and the gating mechanism that accommodates variable input sizes, M2oE can automatically adapt to morphologies with different numbers of modules.

For the same module across different morphologies, M2oE provides a shared pool of experts, which allows for experience sharing and cross-configuration learning. The experts can be formulated as a set of policies $[\pi_{\text{expert}}^{(1)}, \pi_{\text{expert}}^{(2)}, \ldots, \pi_{\text{expert}}^{(K)}]$, where K is the number of experts in the shared pool. Every expert is assigned with a learnable key vector $\mathbf{key}_k$, which is used in the gating mechanism. Experts take both general observation and module-specific observation as input. Each expert outputs a mean action vector with the same dimension as the module-specific action space: $\pi_{\text{expert}}^{(k)}(\mathbf{o}_{\text{module}}^{(m)}, \mathbf{o}_{\text{general}}) \in \mathbb{R}^{\dim(\mathbf{a}_{\text{module}}^{(m)})}$.

M2oE-Gate illustrated in Figure 3 adopts an Encoder-only Transformer architecture to extract inter-module dependencies. This use of Transformer has proven effective in processing graph-structured data [15], [31], [32] in domains like robotics learning and human motion prediction.

This Transformer produces a feature vector $\mathbf{h}_{\text{module}}^{(m)}$ for each module:

$$\left[\mathbf{h}_{\text{module}}^{(1)}, \mathbf{h}_{\text{module}}^{(2)}, \ldots, \mathbf{h}_{\text{module}}^{(M)}\right] = \text{Transformer}\left(\mathbf{o}\right). \quad (4)$$

These module embeddings are latent feature representations and serve directly as the query vectors in the subsequent attention-based gating mechanism. The gating distribution for the m-th module is computed using a Query-Key mechanism. Where the outputs of Encoder-only Transformer serve as queries, and the learnable expert key vectors $\mathbf{key}_k$ serve as keys.

$$\widehat{\mathbf{h}}^{(m)} = \frac{\mathbf{h}_{\text{module}}^{(m)}}{\left\|\mathbf{h}_{\text{module}}^{(m)}\right\|_2}, \qquad \widehat{\mathbf{key}}_k = \frac{\mathbf{key}_k}{\|\mathbf{key}_k\|_2}. \quad (5)$$

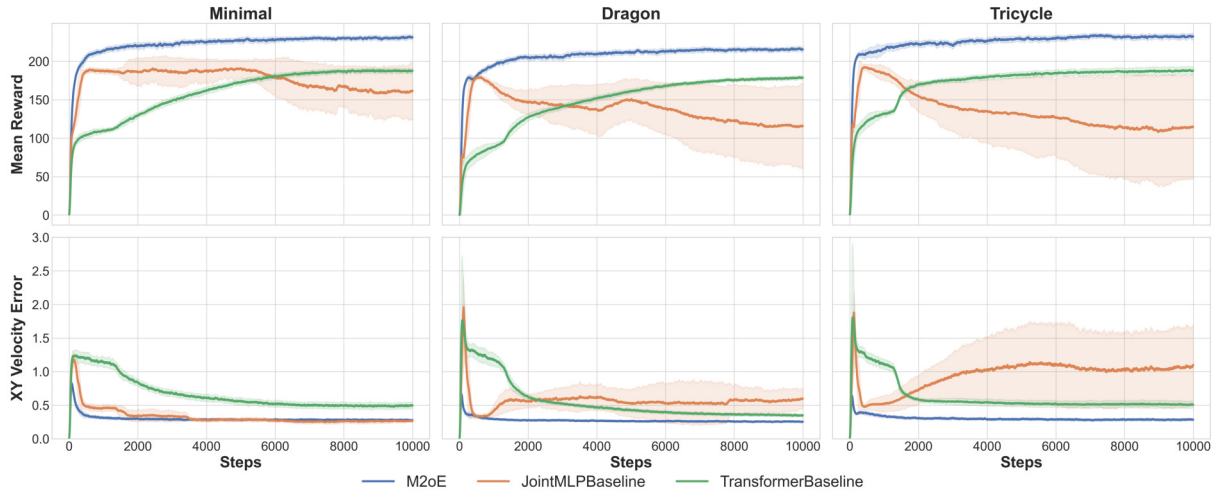

Fig. 5. **Learning Curves** of different methods on the locomotion task across three Moonbot morphologies. We report the average episodic return and the EMA of tracking error (MAE of target linear and angular velocities) during training. Solid lines show the EMA-smoothed (span = 50) mean performance across three seeds, while shaded bands represent the mean ±1 standard deviation. M2oE demonstrates stable and efficient convergence across all morphologies. JointMLP is able to exceed M2oE in linear velocity tracking error on Minimal morphology, but it fails to converge on other morphologies.

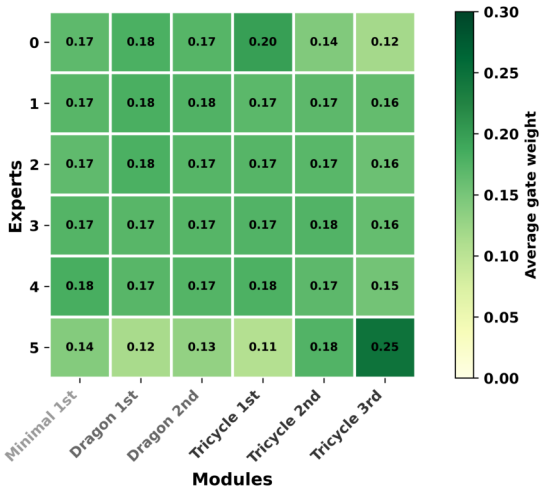

Fig. 6. **Heatmap of expert utilization** in M2oE across different morphologies. Each column represents a module, and each row represents an expert. The color intensity indicates the average gating value over test episodes.

$$g_{m,k} = \frac{\exp\left(\widehat{\mathbf{h}}^{(m)\top}\widehat{\mathbf{key}}_k\right)}{\sum_{j=1}^{K}\exp\left(\widehat{\mathbf{h}}^{(m)\top}\widehat{\mathbf{key}}_j\right)}. \tag{6}$$

Intuitively, each module first forms a context-aware representation of itself, and this representation determines the weights of experts based on inter-module relationships. The experts then produce candidate action means, which are aggregated according to the gating weights to form the final action mean. The final action mean of a module can be formulated as:

$$\boldsymbol{\mu}^{(m)}_{\text{module}} = \sum_{k=1}^{K} g_{m,k} \cdot \pi^{()}_{\text{expert}}(\mathbf{o}^{(m)}_{\text{module}}, \mathbf{o}_{\text{general}}). \tag{7}$$

The actual action $\mathbf{a}^{(m)}_{\text{module}}$ is then sampled from a distribution parameterized by this mean and a learned variance.

This structure provides several advantages:

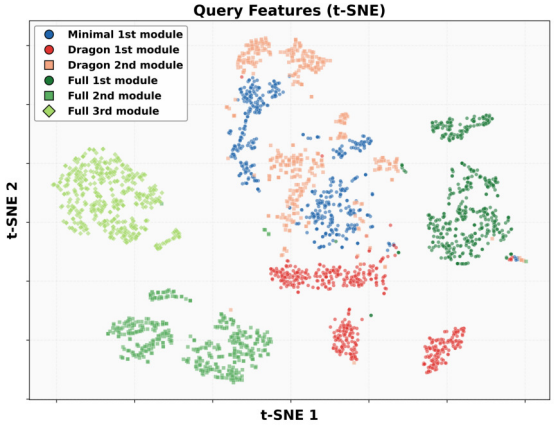

Fig. 7. **t-SNE visualization of gating query features across morphologies.** Each point represents a transformer output feature (query vector) produced by the gating network during forward locomotion at a fixed velocity command. Distinct clusters indicate that the gating network learns morphology- and module-specific representations in the shared embedding space, enabling expert selection across heterogeneous body structures.

- **Scalability:** M2oE can handle varying numbers of modules since the gating mechanism can process variable-length inputs and experts are shared across modules. This allows M2oE to adapt to varying module counts within the same embodiment family, also the experience sharing across morphologies improves training efficiency.
- **Module-wise Parallelism:** The shared pool experts and attention-based gate inherently support module-wise parallelization.
- **Specialization and Generalization:** The shared pool of experts allows for automatic specialization of experts for specific configurations while still promoting generalization across morphologies through experience sharing.

C. Multi-Morphology Training Extension in Isaac Lab

In addition to the M2oE framework, we extend the **Isaac Lab (v2.1.1)** to support **scalable multi-morphology training**.

The original simulator is primarily designed for single-morphology robots or multiple robots with identical structures [33], which limits its applicability to multi-morphology learning. To address this, we introduce several engineering extensions that enable concurrent training across diverse morphologies within a single simulation environment.

In each scene, multiple robot morphologies with different configurations are instantiated and controlled independently during training. To avoid interference between morphologies, we design a multi-layer terrain generator that assigns each morphology to a dedicated terrain layer along the z-axis, ensuring collision-free operation under a shared environment.

We also implement a group-based training pipeline to coordinate learning across morphologies.

This extension enables scalable multi-morphology training in Isaac Lab for both modular and non-modular robots, supporting more versatile and scalable reinforcement learning experiments. In the **Experiments** section, we demonstrate its effectiveness with M2oE on the experiment platform Moonbot. This extension is released as open source to facilitate future research in multi-morphology reinforcement learning.

IV. EXPERIMENTS

In this section, we introduce the experimental setup, including the robot morphologies, task definition, and baselines.

A. Experimental Setup

Moonbot [4] is a modular robot composed of multiple interconnected modules that can be reconfigured to adapt to diverse terrains and tasks. Each module is a 7-DoFs robotic arm with two wheel joints at the base, enabling both locomotion and manipulation capabilities. Based on this module, we introduce three morphologies (Figure 1): **Minimal**, consisting of a single module; **Dragon**, consisting of two modules; and **Tricycle**, consisting of three modules.

We evaluate M2oE with $K = 6$ experts in the shared pool on a **locomotion task** where Moonbots are required to track a random target velocity command while maintaining stability. This task challenges the policy to **generalize across morphologies** while leveraging shared experiences among modules. Both M2oE and baseline models are trained using the Proximal Policy Optimization (PPO) algorithm for 10k iterations using 2048 parallel environments, each containing three Moonbot morphologies. The reward encourages the robot to follow the target velocity while penalizing excessive energy consumption and instability. The PPO optimization objective is defined as:

$$\mathcal{L}_{\text{PPO}} = \mathcal{L}_{\text{surrogate}} + \lambda_v \mathcal{L}_{\text{value}} - \lambda_e \mathcal{L}_{\text{entropy}}, \tag{8}$$

where $\mathcal{L}_{\text{surrogate}}$ is the clipped surrogate objective, $\mathcal{L}_{\text{value}}$ is the value function loss, and $\mathcal{L}_{\text{entropy}}$ is the entropy bonus to encourage exploration. λ_v and λ_e are hyper-parameters that balance the contributions of each term.

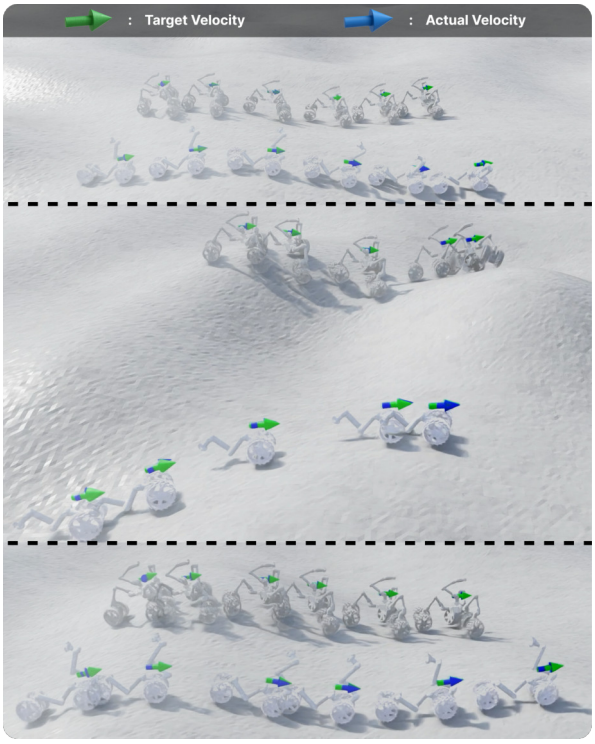

Fig. 8. **Zero-shot transfer to unseen wave terrain**. M2oE demonstrates a certain capability to generalize to unseen terrains.

To encourage balance expert usage, M2oE introduces a load balancing loss:

$$\mathcal{L}_{\text{M2oE}} = \mathcal{L}_{\text{PPO}} + \lambda_{lb} \mathcal{L}_{\text{lb}}, \tag{9}$$

$\mathcal{L}_{\text{lb}}$ is defined as:

$$\begin{aligned} \mathcal{L}_{\text{lb}} &= \mathrm{K}^2 \cdot \frac{1}{\mathrm{K}} \sum_{k=1}^{K} \left(\frac{1}{BM} \sum_{b=1}^{B} \sum_{m=1}^{M} g_{m,k}^{(b)} \right)^2 \\ &= \mathrm{K} \sum_{k=1}^{K} \left(\frac{1}{BM} \sum_{b=1}^{B} \sum_{m=1}^{M} g_{m,k}^{(b)} \right)^2, \end{aligned} \tag{10}$$

where B is the batch size, M is the number of modules, and $g_{m,k}^{(b)}$ is the gating value for the K-th expert in the shared pool for the m-th module in the b-th sample. λ_{lb} is a hyperparameter that controls the strength of the load balancing loss. The K^2 term normalizes $\mathcal{L}_{\text{lb}}$ within the range $[1, \mathrm{K}]$. This loss prevents experts collapsing to one single expert.

Table I summarizes the observation terms and action terms along with corresponding dimensions. We also provide the overall observation and action space dimensions for each morphology in the table. We also provide detailed functions of reward terms and the corresponding weights in Table II.

B. Baseline Methods

The multi-morphology challenges DRL policies in several aspects, including varying observation and action spaces, diverse dynamics, and potential gradient conflicts during training. Inspired by previous research [6], [15], we design two baselines based on two main approaches to address

the varying observation and action spaces: **padding-based methods** and **attention-based methods**:

Padding-based MLP: A standard MLP backbone that takes the jointed and padded observations as input. This approach is straightforward but may struggle to capture inter-module correlations effectively. To be fair, we implement the Layer Normalization [34] and the same activation function (ELU [35]) as in M2oE. We call this baseline **JointMLP**.

Transformer (Attention-based): Attention-based backbones are naturally suited for handling varying input and output dimensions. Previous works [31], [32] have shown that attention mechanisms can effectively capture inter-position correlations in graph-structured data, even in robot learning tasks [15]. So we implement an Encoder-only Transformer structure similar to Body-Transformer [15] as a baseline. This baseline is referred to as **Transformer**.

To ensure a fair comparison, the total parameter count of each baseline is matched (around 100k) by adjusting the number of layers and hidden dimensions.

V. RESULTS

In this section, we provide experimental results of M2oE, including both visualized and quantitative locomotion performance, training performance, insight studies of M2oE and a Zero-shot Generalization experiment.

A. Locomotion Performance Across Morphologies

We first present the locomotion performance to demonstrate the effectiveness of M2oE in training Moonbot across different morphologies. Robots are commanded to track a target velocity of (0.8, 0.0) m/s in local frame, the heading command along X-axis automatically generates a yaw rotation command. Figure 2 shows snapshots of different morphologies of Moonbot performing the locomotion task after training with M2oE. As illustrated, all evaluated morphologies controlled by M2oE reliably track the target velocity commands while maintaining stability and balance. In contrast, baseline methods struggle to learn effective policies and exhibit various failure modes, as shown in Figure 4.

B. Performance Comparison with Baselines

We quantitatively evaluate the performance of M2oE and baseline methods on the locomotion task. The average step error of tracking errors (the mean absolute error of target linear and angular velocities) during test episodes is used as the evaluation metric. It serves as a proxy for locomotion failure, where higher MAE typically corresponds to unstable behaviors such as drift or inability to track commands. The mean errors on 3000 test steps over 64 parallel environments is reported for each morphology in Table III.

As shown in the table, M2oE achieves the lowest error across all morphologies on both linear and angular velocity tracking, demonstrating its superior generalization and adaptability to different configurations. Other baselines can achieve comparable performance on some morphologies, but they struggle on others, indicating limited generalization capabilities and failure to resolve gradient conflicts effectively.

C. Training performance

Figure 5 compares the learning dynamics of M2oE and baselines on the locomotion task across three Moonbot morphologies. We report the mean and standard deviation of iteration rewards and mean linear velocity tracking error (MAE) over three seeds. JointMLP struggles to converge, with unstable returns and growing task errors. We attribute this degradation to gradient interference across morphologies with different coordination structures. Transformer achieves reasonable performance but but still underperforms M2oE.

M2oE demonstrates **faster convergence** and **higher final performance** compared to all baselines. It successfully converges within 2000 iterations and maintains a low task error throughout the training process. The smooth learning curves and stable task errors of **all** evaluated morphologies indicate that M2oE effectively mitigates gradient conflicts.

D. Insight Studies of M2oE

To analyze the expert mechanism in M2oE, we visualize the average gating utilization and the expert key vectors after training a M2oE model with $k = 6$ experts in the shared pool. Figure 6 shows the heatmap of average gating values for each expert across different modules and morphologies in M2oE. X-axis represents modules and Y-axis represents experts. This figure reveals several interesting patterns: The modules in Minimal and Dragon have similar functionalities, so they tend to utilize similar experts. While the modules in Tricycle have distinct roles in this morphology, leading to more diverse expert usage.

To further analyze the representation learned by the gating network, we visualize the query features (i.e., the Transformer output features used for expert routing) using t-SNE. As shown in Figure. 7, features of same module tend to form local clusters in the embedding space. Since these features serve as the query vectors for computing similarities with expert keys, the observed clustering indicates that the gating network organizes module features into structured representations that can support expert routing.

E. Zero-shot Generalization to Unseen Terrain

To further assess the generalization capabilities and robustness of M2oE, we conducted zero-shot transfer experiments on unseen wave terrain which is similar to lunar surface. As shown in Figure 8, M2oE demonstrates a certain capability to generalize to unseen terrains on all morphologies. The policy trained on flat terrain is able to generalize to wave terrains, with a little increase in failure rate.

VI. CONCLUSION

In this paper, we presented Modular Mixture of Experts (M2oE), a reinforcement learning backbone designed for modular robots with diverse morphologies. By aligning the MoE structure with the modular embodiment of robots, M2oE enables module-wise parallelism and efficient experience sharing across modules and morphologies.

We also extended the Isaac Lab simulator to support concurrent multi-morphology training, enabling scalable policy

learning across different robot configurations. Experiments on the Moonbot platform demonstrate that M2oE achieves higher learning efficiency and superior locomotion performance compared with baseline methods.

Future work includes extending M2oE to manipulation tasks, multi-agent coordination, and dynamic morphology reconfiguration, as well as investigating sim-to-real transfer to real-world modular robotic systems.

REFERENCES

[1] G. Liang, D. Wu, Y. Tu, and T. L. Lam, "Decoding modular reconfigurable robots: A survey on mechanisms and design," *The International Journal of Robotics Research*, vol. 44, no. 5, pp. 740–767, 2025.

[2] R. J. Alattas, S. Patel, and T. M. Sobh, "Evolutionary modular robotics: Survey and analysis," *Journal of Intelligent & Robotic Systems*, vol. 95, no. 3, pp. 815–828, 2019. [Online]. Available: https://doi.org/10.1007/s10846-018-0902-9

[3] M. A. Post, X.-T. Yan, and P. Letier, "Modularity for the future in space robotics: A review," *Acta Astronautica*, vol. 189, pp. 530–547, 2021.

[4] K. Uno, E. Neppel, G. H. Diaz, A. Mishra, S. Karimov, A. S. Jain, A. Habib, P. Pama, H. Gozbasi, S. Santra, and K. Yoshida, "Moonbot: Modular and on-demand reconfigurable robot toward moon base construction," *IEEE Transactions on Field Robotics*, vol. 2, pp. 847–874, 2025.

[5] R. Huang, S. Zhu, Y. Du, and H. Zhao, "Moe-loco: Mixture of experts for multitask locomotion," *arXiv preprint arXiv:2503.08564*, 2025.

[6] N. Bohlinger, G. Czechmanowski, M. P. Krupka, P. Kicki, K. Walas, J. Peters, and D. Tateo, "One policy to run them all: an end-to-end learning approach to multi-embodiment locomotion," in *Conference on Robot Learning*. PMLR, 2025, pp. 3356–3378.

[7] 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.

[8] T. Lin, K. Sachdev, L. Fan, J. Malik, and Y. Zhu, "Sim-to-real reinforcement learning for vision-based dexterous manipulation on humanoids," *arXiv preprint arXiv:2502.20396*, 2025.

[9] Y. Ze, Z. Chen, J. P. AraÃšjo, Z.-a. Cao, X. B. Peng, J. Wu, and C. K. Liu, "Twist: Teleoperated whole-body imitation system," *arXiv preprint arXiv:2505.02833*, 2025.

[10] J. Whitman, M. Travers, and H. Choset, "Learning modular robot control policies," *IEEE Transactions on Robotics*, vol. 39, no. 5, pp. 4095–4113, 2023.

[11] J. Sun, M. Yao, X. Xiao, Z. Xie, and B. Zheng, "Co-optimization of morphology and behavior of modular robots via hierarchical deep reinforcement learning." in *Robotics: Science and Systems*, 2023.

[12] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar *et al.*, "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3740–3747, 2023.

[13] R. Doshi, H. R. Walke, O. Mees, S. Dasari, and S. Levine, "Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation," in *Conference on Robot Learning*. PMLR, 2025, pp. 496–512.

[14] J. Yang, C. Glossop, A. Bhorkar, D. Shah, Q. Vuong, C. Finn, D. Sadigh, and S. Levine, "Pushing the limits of cross-embodiment learning for manipulation and navigation," *arXiv preprint arXiv:2402.19432*, 2024.

[15] C. Sferrazza, D.-M. Huang, F. Liu, J. Lee, and P. Abbeel, "Body transformer: Leveraging robot embodiment for policy learning," *arXiv preprint arXiv:2408.06316*, 2024.

[16] H. Furuta, Y. Iwasawa, Y. Matsuo, and S. S. Gu, "A system for morphology-task generalization via unified representation and behavior distillation," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=HcUf-QwZeFh

[17] A. Lel, A. Miller, V. Sekin, S. Kriebisch, R. Bruder, C. Röhrig, and T. Straßmann, "A modularization concept for mobile robots in search and rescue applications," in *2023 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2023, pp. 25–31.

[18] L. Pfotzer, S. Ruehl, G. Heppner, A. Roennau, and R. Dillmann, "Kairo 3: A modular reconfigurable robot for search and rescue field missions," in *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*, 2014, pp. 205–210.

[19] J. Külz, M. Terzer, M. Magri, A. Giusti, and M. Althoff, "Holistic construction automation with modular robots: From high-level task specification to execution," *IEEE Transactions on Automation Science and Engineering*, 2025.

[20] S. Takeda, S. Yamamori, S. Yagi, and J. Morimoto, "An empirical evaluation of a hierarchical reinforcement learning method towards modular robot control," *Artificial Life and Robotics*, 2025.

[21] S. Takeda, S. Yamamori, S. Yagi, and J. Morimoto, "Hierarchically connecting modularly-learned policies to generate a controller for a combined robot system," in *IEEE 21st International Conference on Automation Science and Engineering (CASE)*, pp. 1722–1727.

[22] J. Whitman and H. Choset, "Learning modular robot visual-motor locomotion policies," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 11 908–11 914.

[23] A. Mishra, S. Santra, E. Neppel, E. M. R. Lombardi, S. Karimov, K. Uno, and K. Yoshida, "Multi-modal decentralized reinforcement learning for modular reconfigurable lunar robots," *arXiv preprint arXiv:2510.20347*, 2025.

[24] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," *Journal of Machine Learning Research*, vol. 23, no. 120, pp. 1–39, 2022.

[25] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann *et al.*, "Palm: Scaling language modeling with pathways," *Journal of Machine Learning Research*, vol. 24, no. 240, pp. 1–113, 2023.

[26] C. Riquelme, J. Puigcerver, B. Mustafa, M. Neumann, R. Jenatton, A. Susano Pinto, D. Keysers, and N. Houlsby, "Scaling vision with sparse mixture of experts," *Advances in Neural Information Processing Systems*, vol. 34, pp. 8583–8595, 2021.

[27] D. Wang, X. Wang, X. Liu, J. Shi, Y. Zhao, C. Bai, and X. Li, "More: Mixture of residual experts for humanoid lifelike gaits learning on complex terrains," *arXiv preprint arXiv:2506.08840*, 2025.

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

[29] W. Cai, J. Jiang, F. Wang, J. Tang, S. Kim, and J. Huang, "A survey on mixture of experts in large language models," *IEEE Transactions on Knowledge and Data Engineering*, 2025.

[30] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," in *International Conference on Learning Representations*, 2017. [Online]. Available: https://www.cs.toronto.edu/~hinton/absps/Outrageously.pdf

[31] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, and T.-Y. Liu, "Do transformers really perform badly for graph representation?" in *NeurIPS 2021*, December 2021. [Online]. Available: https://www.microsoft.com/en-us/research/publication/do-transformers-really-perform-badly-for-graph-representation/

[32] C. Liu, S. Yagi, S. Yamamori, and J. Morimoto, "Joint-aware transformer: An inter-joint correlation encoding transformer for short-term 3d human motion prediction," *IEEE Access*, vol. 12, pp. 156 683–156 693, 2024.

[33] I. L. P. Developers, "Spawning multiple assets," Online: https://isaac-sim.github.io/IsaacLab/v2.1.1/source/how-to/multi_asset_spawning.html, 2025, isaac Lab Documentation, Version 2.1.1, Last updated on Aug. 25, 2025.

[34] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[35] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv: Learning*, 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:5273326