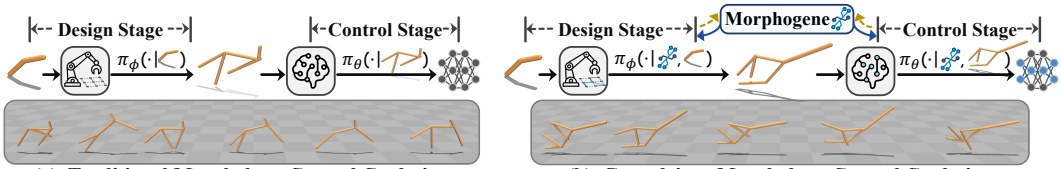
BRIDGING BODY AND BRAIN: GENE-DRIVEN MORPHOLOGY–CONTROL CO-DESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Morphology–control co-design jointly optimizes an agent's body structure and control policy as an integrated embodied system. However, existing methods typically model morphology design and control with separate networks coupled only indirectly through a shared task objective, limiting explicit high-level coordination. Inspired by natural genes that coordinate biological development, we introduce **Morphogene**, a compact latent blueprint that bridges an agent's body and brain. Through AdaConcat, Morphogene jointly conditions morphology and control generation at the limb level, allowing its variations to induce coordinated changes in both components. Building on this representation, we propose **GeCode**, which formulates co-design as exploration in the compact Morphogene space. Each Morphogene anchors a local design region in which nearby body–brain designs are explored, while performance-guided updates move these anchors toward promising regions for more efficient exploration of the broader design space. This process combines local refinement with global exploration while preserving body–brain compatibility. Extensive experiments across diverse 2D and 3D co-design tasks demonstrate that GeCode consistently outperforms existing state-of-the-art methods, achieving substantially faster convergence and higher final performance. The code is available at `https://anonymous.4open.science/w/GeCode-BC8B/`.



(a) Traditional Morphology-Control Co-design **(b) Gene-driven Morphology-Control Co-design**

Figure 1: (a) Traditional morphology–control co-design employs *separate* morphology-design and control networks coupled primarily through a shared task objective, resulting in weak body–brain coordination. (b) Inspired by biological genes, GeCode introduces Morphogene as a high-level latent blueprint that bridges body and brain. By propagating Morphogene variations to both morphology design and controller adaptation, GeCode drives their explicit coordination at the representation level while reformulating morphology search as exploration within a continuous latent space.

1 INTRODUCTION

Across species in nature, body morphology and neural control develop and evolve together, with the body shaping the physical conditions for behavior and the nervous system adapting to exploit them (Chiel & Beer, 1997; Bartolozzi et al., 2022). Nevertheless, embodied-agent design often treats morphology and control as largely independent components (Xiong et al., 2023; Gupta et al., 2021). Such one-sided optimization either confines control to the affordances of a fixed body (Haarnoja et al., 2024; Bolotnikova et al., 2025) or yields morphologies biased toward a fixed controller (Matthews et al., 2023; Zhao et al., 2020), motivating **morphology–control co-design** through the joint adaptation of agent's body and brain (Ha, 2019; Liu et al., 2025; Wang et al., 2025).

However, current embodied co-design methods typically employ separate morphology-generation and control-policy networks coupled primarily through the shared task objective (Eq. (1)) *rather than explicit coordination at the representation or decision level* (Yuan et al., 2022; Lu et al., 2025). Consequently, morphological changes do not directly guide controller adaptation, nor do controller

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

updates explicitly inform morphological design; this weak coupling obscures cross-component credit assignment and leaves body–brain coordination indirect and inadequately modeled (Dai et al., 2026).

In nature, genes address an analogous coordination problem by serving as **developmental blueprints** that encode shared high-level programs. Variations in these programs jointly shape morphology, neural circuitry, and behavior beyond adaptation driven solely by environmental feedback (Zador, 2019; Fleck et al., 2023). Inspired by this principle, we introduce **Morphogene** as a similar **developmental blueprint** for embodied agents. Rather than representing a single fixed morphology, each Morphogene abstracts high-level structural and geometric characteristics, *anchors* a local region of related body designs, and *jointly conditions* morphology generation and controller adaptation[1].

Specifically, unlike conventional approaches that model body and brain with separate networks (Yuan et al., 2022; Lu et al., 2025), a shared Morphogene explicitly couples morphology and control generation at the representation level. Morphogene is injected into both networks through AdaConcat, which transforms the agent-level representation into limb-specific conditions based on each limb's structural and functional role. Consequently, variations in Morphogene induce coherent changes in morphology design and corresponding controller adaptations, thereby bridging body and brain.

Building on Morphogene, we introduce **GeCode**, a gene-driven co-design framework that shifts co-design from a discrete morphology-design space (Gupta et al., 2021; Koike et al., 2024) to a compact, continuous Morphogene space. Specifically, GeCode maintains a small set of Morphogenes as design anchors, combining stochastic policy rollouts with a Morphogene-proximity reward to explore high-performing, locally coherent body–brain designs around each anchor. Performance-guided Morphogene updates subsequently move the anchors toward regions of higher-performing body–brain designs, thereby expanding exploration across the latent space. This local-to-global strategy enables broad yet targeted exploration without exhaustively evaluating discrete morphologies.

We evaluate GeCode across twelve 2D and 3D design spaces spanning diverse morphologies. GeCode consistently outperforms state-of-the-art methods, achieving on average 2.5× faster convergence and 69.48% higher task performance while incurring less than 4.0% additional wall-clock overhead.

Our contributions are as follows: (1) We introduce **Morphogene**, a high-level latent blueprint that bridges an agent's body and brain, together with AdaConcat for adaptive, limb-aware conditioning of morphology and control. (2) We propose **GeCode**, which reformulates morphology–control co-design as gene-driven exploration in the compact Morphogene space, allowing latent updates to induce coordinated body–brain adaptation. (3) Extensive experiments demonstrate that GeCode achieves state-of-the-art performance with faster convergence and higher final returns.

2 Related Work

Morphology–Control Co-design Joint morphology–control optimization has emerged as a promising direction in embodied intelligence (Strgar & Kriegman, 2026; Fay et al., 2026; Wang et al., 2025; Liu et al., 2025). Early evolutionary approaches relied on population-based morphology search and repeated controller adaptation, resulting in substantial sampling and computational costs (Bhatia et al., 2021; Gupta et al., 2021; Cheney et al., 2018). Recent RL-based methods jointly optimize sequential morphology construction and control (Yuan et al., 2022; Wang et al., 2023; Lu et al., 2025). However, their separate morphology and control networks remain weakly coupled, limiting explicit body–brain coordination. StackelbergPPO (Dai et al., 2026) strengthens optimization-level coordination by explicitly modeling morphology–control interdependence, but still leave their representation-level coupling implicit. In contrast, GeCode makes this coupling explicit through Morphogene, which jointly coordinates morphology and control generation throughout co-design.

Further discussion on **"Genes" in Intelligent Agents** and **Universal Control** is provided in App. A.

3 Preliminaries

Morphology Representation An agent's morphology is represented as an undirected graph $\mathcal{M} = (V, E, A)$, where each node $u \in V$ denotes a limb and each edge $e(u, v) \in E$ denotes a joint

[1]The anchoring role of a Morphogene resembles that of a ***prototype*** in representation learning. Here, however, we use the biologically inspired term ***developmental blueprint*** to emphasize its broader role: beyond serving as a representative embedding, a Morphogene can be adaptively moved through the latent space to guide morphology design while bridging morphology generation and controller adaptation in embodied agents.

connecting two limbs. The attribute set $A = \{\Lambda_u^V \mid u \in V\} \cup \{\Lambda_e^E \mid e \in E\}$ specifies limb properties (e.g., length and size) and joint properties (e.g., rotation ranges and motor torques).

Once the topology and attributes are specified, the simulator (MuJoCo in our implementation) instantiates the morphology. In our design spaces, each limb u is paired with *exactly one* associated joint, denoted by $e(u, \cdot)$, allowing their simulator-derived attributes to be combined as $\mathbf{m}_u = [\mathbf{m}_u^V \| \mathbf{m}_{e(u,\cdot)}^E] \in \mathbb{R}^{d_m}$. Here, $\mathbf{m}_u^V$ encodes fixed limb geometry, such as its reference position and orientation (e.g., `body_pos` and `body_iquat`), while $\mathbf{m}_{e(u,\cdot)}^E$ encodes the corresponding joint geometry (e.g., `jnt_pos`). The complete vector representation is $\mathcal{M}^{\text{vec}} = [\mathbf{m}_u]_{u \in V}$. Thus, $\mathcal{M}$ and $\mathcal{M}^{\text{vec}}$ describe the same morphology at the graph and vector levels, respectively, with the former used for sequential morphology design and the latter for constructing the Morphogene space.

Morphology-Control Co-Design Optimization Morphology–control co-design optimizes an agent's body structure and control policy through a ***Design Stage*** Φ_{des} and a ***Control Stage*** Φ_{ctrl}. During morphology design Φ_{des}, the morphology graph $\mathcal{M}_t = (V_t, E_t, A_t)$ is sequentially updated over T_{topo} topology design steps followed by T_{attr} attribute design steps. Each update follows

$$a_t^{\text{des}} \sim \pi_\phi^{\text{des}}(\cdot \mid \mathcal{M}_t), \qquad \mathcal{M}_{t+1} \sim P^{\text{des}}(\cdot \mid \mathcal{M}_t, a_t^{\text{des}}), \quad 0 < t \le T_{\text{des}},$$

where $T_{\text{des}} = T_{\text{topo}} + T_{\text{attr}}$. π_ϕ^{des} is the morphology-design policy, and P^{des} is the corresponding transition function. The morphology-editing action a_t^{des} specifies structural operations (e.g., adding or removing limbs) during topology design and physical-attribute updates (e.g., adjusting limb length) during attribute design. The resulting action sequence yields the final morphology $\mathcal{M}^\star = \mathcal{M}_{T_{\text{des}}}$.

During the control stage Φ_{ctrl}, the morphology $\mathcal{M}^\star$ remains fixed while the controller interacts with the environment to maximize the expected task return. At each control step, the interaction follows

$$a_t^{\text{ctrl}} \sim \pi_\theta^{\text{ctrl}}(\cdot \mid s_t^{\text{ctrl}}; \mathcal{M}^\star), \qquad s_{t+1}^{\text{ctrl}} \sim P^{\text{ctrl}}(\cdot \mid s_t^{\text{ctrl}}, a_t^{\text{ctrl}}; \mathcal{M}^\star), \qquad t > T_{\text{des}}.$$

Here, s_t^{ctrl} denotes the environment observation received by the controller, while π_θ^{ctrl} and P^{ctrl} denote the control policy and environment transition function under $\mathcal{M}^\star$, respectively.

The morphology–control co-design objective jointly optimizes the design and control policy parameters to maximize the expected discounted return:

$$\max_{\phi,\theta} \; J(\phi, \theta) = \mathbb{E}\left[\sum_{t=T_{\text{des}}+1}^{\infty} \gamma^{t-T_{\text{des}}-1} r_t^{\text{env}}\right]. \tag{1}$$

Here, γ is the discount factor, and r_t^{env} denotes the task reward provided by the environment (App. B).

Existing methods (Lu et al., 2025; Yuan et al., 2022) use separate morphology and control policies coupled mainly through the shared objective in Eq. (1), leaving no explicit mechanism to coordinate morphological changes with controller adaptation. To explicitly couple these two components, we formulate morphology generation and controller adaptation as Morphogene-conditioned processes (Sec. 4.2) and jointly optimize them under the augmented objective in Eq. (11).

Transformer-based Morphology Design and Control An agent is represented by an ordered sequence of N_L limb-wise states, where $\mathbf{s}_i \in \mathbb{R}^{d_s}$ is the proprioceptive observation of limb l_i. Each state is projected into an embedding space, and the resulting limb tokens are assembled as

$$\mathbf{h}_i = f_{\text{emb}}(\mathbf{s}_i), \quad i = 1, \ldots, N_L, \qquad \mathbf{H}^{(0)} = [\mathbf{h}_1; \cdots; \mathbf{h}_{N_L}] \tag{2}$$

where $f_{\text{emb}}(\cdot)$ is the shared embedding function (Gupta et al., 2022).

The Transformer encoder comprises L pre-normalized blocks, each applying layer normalization (LN) (Xiong et al., 2020) before multi-head self-attention (MSA) (Vaswani et al., 2017) and a feed-forward network (FFN), followed by residual updates:

$$\widetilde{\mathbf{H}}^{(\ell)} = \mathbf{H}^{(\ell-1)} + \text{MSA}_\ell\Big(\text{LN}_{\ell,1}\Big(\mathbf{H}^{(\ell-1)}\Big)\Big), \; \mathbf{H}^{(\ell)} = \widetilde{\mathbf{H}}^{(\ell)} + \text{FFN}_\ell\Big(\text{LN}_{\ell,2}\Big(\widetilde{\mathbf{H}}^{(\ell)}\Big)\Big), \; \ell = 1, \ldots, L. \tag{3}$$

The final token representation $\mathbf{H}^{(L)}$ is mapped by a linear action head to the distribution parameters $\mathbf{z}_{\text{act}} = f_{\text{act}}(\mathbf{H}^{(L)})$. During the design stage Φ_{des}, discrete actions follow a categorical distribution parameterized by $\text{softmax}(\mathbf{z}_{\text{act}})$. During the control stage Φ_{ctrl}, continuous actions follow a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{\text{act}}, \boldsymbol{\Sigma}_{\text{act}})$, whose mean and covariance are parameterized by $\mathbf{z}_{\text{act}}$.

4 METHOD

4.1 MORPHOGENE: A LATENT BLUEPRINT FOR BODY–BRAIN COORDINATION

Morphology Encoding and Morphogene Representation We define Morphogene as a compact latent blueprint that guides morphology generation and compatible control. To ground this blueprint in the underlying body structure, we learn a latent space that captures the correlated structural and geometric factors of morphology. As illustrated in Figure 2(a), the morphology encoder E_ψ maps an instantiated morphology $\mathcal{M}^{\text{vec}}$ to its morphology code $\mathbf{z}^{\text{morph}}$:

$$\mathbf{z}^{\text{morph}} = E_\psi(\mathcal{M}^{\text{vec}}), \qquad \mathbf{z}^{\text{morph}} \in \mathbb{R}^{d_g}, \tag{4}$$

Unlike **morphology code $\mathbf{z}^{\text{morph}}$**, which encodes **a specific realized morphology**, a **Morphogene** $\mathcal{G} \in \mathbb{R}^{d_g}$ serves as **a searchable anchor** in the same latent space. GeCode then iteratively updates these anchors to drive exploration of coordinated body–brain designs (Sec. 4.3).

RAE-based Morphogene Space Construction The Morphogene space is learned using a regularized autoencoder (RAE), whose decoder D_ψ reconstructs morphological attributes and predicts limb presence, thereby preserving structural and geometric information:

$$(\widehat{\mathcal{M}}^{\text{vec}}, \widehat{\boldsymbol{\chi}}) = D_\psi(\mathbf{z}^{\text{morph}}) = D_\psi(E_\psi(\mathcal{M}^{\text{vec}})), \tag{5}$$

Here, $\widehat{\boldsymbol{\chi}}$ denotes the predicted limb-presence mask, which determines the number of limbs in the reconstructed morphology. The RAE is trained using a masked, regularized reconstruction objective:

$$\mathcal{L}_{\text{rec}} = \mathcal{L}_{\text{attr}} + \lambda_{\text{mask}}\mathcal{L}_{\text{mask}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}}, \tag{6}$$

where $\mathcal{L}_{\text{attr}} = \frac{\|\boldsymbol{\chi}\odot(\widehat{\mathcal{M}}^{\text{vec}} - \mathcal{M}^{\text{vec}})\|_2^2}{\|\boldsymbol{\chi}\|_1}$ measures reconstruction error over valid morphology components, $\mathcal{L}_{\text{mask}} = \text{BCE}(\widehat{\boldsymbol{\chi}}, \boldsymbol{\chi})$ supervises limb-presence prediction, and $\mathcal{L}_{\text{reg}} = \|\mathbf{z}^{\text{morph}}\|_2^2$ regularizes the morphology codes to promote a compact Morphogene space.

Structural Constraints on the Morphogene Space While the reconstruction objective preserves the information required for morphology recovery, it does not explicitly impose meaningful geometric structure on the Morphogene space. We therefore regularize RAE with a topology-aware contrastive loss $\mathcal{L}_{\text{topo}}$ (Eq. (18)) and a tree-structured metric loss $\mathcal{L}_{\text{tree}}$ (Eq. (19)), detailed in App. C.2.

Specifically, $\mathcal{L}_{\text{topo}}$ clusters morphology codes with identical topologies while separating those with different topologies, whereas $\mathcal{L}_{\text{tree}}$ enforces relative-distance margins between structurally similar and dissimilar morphologies. Together, these losses align latent geometry with morphological relationships, thereby enabling smooth Morphogene exploration, as visualized in Figure 6(b).

4.2 MORPHOGENE-CONDITIONED MORPHOLOGY DESIGN AND CONTROL

GeCode jointly conditions the morphology-design policy $\pi_\phi^{\text{des}}(\cdot \mid \mathcal{G})$ and control policy $\pi_\theta^{\text{ctrl}}(\cdot \mid \mathcal{G})$ on a shared Morphogene $\mathcal{G}$, initially sampled at random from the learned Morphogene space, thereby explicitly coupling body and brain. Following Eq. (2), the state of limb l_i is first mapped to a stage-specific feature $\mathbf{h}_i = f_{\text{emb}}(\mathbf{s}_i; \Phi_\star)$, where $\Phi_\star \in \{\Phi_{\text{des}}, \Phi_{\text{ctrl}}\}$ denotes the active stage.

AdaConcat for Limb-wise Condition Injection As the agent-level Morphogene captures the morphology as a whole, uniformly injecting it across limbs cannot reflect their distinct structural and functional roles. We therefore introduce AdaConcat to adapt $\mathcal{G}$ for limb-specific conditioning:

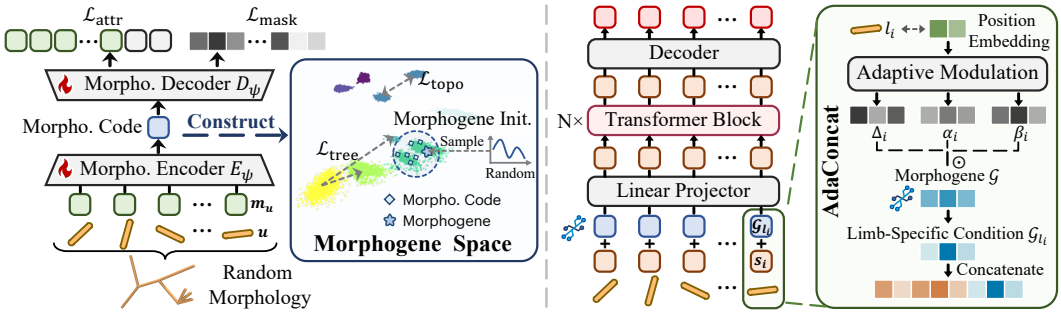
$$\mathbf{e}_i = \text{Emb}_{\text{pos}}(i) \in \mathbb{R}^{d_{\text{pos}}}, \qquad [\boldsymbol{\Delta}_i, \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i] = f_{\text{Ada}}(\mathbf{e}_i; \Phi_\star) \in \mathbb{R}^{3d_g}, \tag{7}$$

where $\mathbf{e}_i$ is the positional embedding of limb l_i, and $f_{\text{Ada}} : \mathbb{R}^{d_{\text{pos}}} \to \mathbb{R}^{3d_g}$ is the modulation network. For limb l_i, AdaConcat transforms $\mathcal{G}$ into the corresponding limb-specific condition $\mathcal{G}_{l_i}$ as

$$\mathcal{G}_{l_i} = \boldsymbol{\Delta}_i \odot [(\mathbf{1} + \boldsymbol{\alpha}_i) \odot \mathcal{G} + \boldsymbol{\beta}_i], \tag{8}$$

where $\odot$ denotes element-wise multiplication. See Figure 2(b) for an illustration.

The limb-specific condition $\mathcal{G}_{l_i}$ is concatenated with its corresponding limb feature to form the conditioned token $\mathbf{h}'_i = [\mathbf{h}_i \,\|\, \mathcal{G}_{l_i}]$. The conditioned tokens are assembled as $\mathbf{H}'^{(0)} = [\mathbf{h}'_1; \ldots; \mathbf{h}'_{N_L}]$



(a) RAE-based Morphogene Space **(b) Morphogene-conditioned Transformer**

Figure 2: **Overview of Morphogene space construction and conditioning.** (a) An RAE encodes randomly generated morphologies into morphology codes, forming a continuous latent space in which Morphogenes serve as searchable anchors. (b) AdaConcat adapts an agent-level Morphogene into limb-specific conditions for the morphology-design and control networks.

and processed by the Transformer blocks in Eq. (3), yielding a morphology-editing action a^{des} during Φ_{des} or a control action a^{ctrl} during Φ_{ctrl}:

$$a^{\text{des}} \sim \pi_\phi^{\text{des}}(\cdot \mid \mathcal{M}; \mathcal{G}),\ \Phi_\star = \Phi_{\text{des}}, \qquad a^{\text{ctrl}} \sim \pi_\theta^{\text{ctrl}}(\cdot \mid s^{\text{ctrl}}; \mathcal{M}^\star, \mathcal{G}),\ \Phi_\star = \Phi_{\text{ctrl}}. \tag{9}$$

In this way, the shared Morphogene aligns morphology generation with controller adaptation, allowing its variations to induce coherent body–brain changes for explicit morphology–control co-design.

Morphogene-Guided Local Exploration During co-design, each Morphogene anchors a local design region for exploring nearby high-performing morphology variants. GeCode promotes such exploration by augmenting the task objective in Eq. (1) with a Morphogene-proximity reward:

$$r_t^{\text{prox}} = \lambda_{\text{hit}}\, \mathbb{I}[t = T_{\text{des}}]\, \mathbb{I}[\text{sim}(\mathbf{z}_t^{\text{morph}}, \mathcal{G}) \geq \tau_{\text{sim}}], \qquad \mathbb{I}[\text{True}] = 1,\ \ \mathbb{I}[\text{False}] = 0. \tag{10}$$

At the end of Φ_{des}, the generated morphology $\mathcal{M}^\star$ is mapped to its morphology code $\mathbf{z}_{T_{\text{des}}}^{\text{morph}}$ using Eq. (4). A reward of λ_{hit} is assigned when the cosine similarity $\text{sim}(\cdot,\cdot)$ between the morphology code and its corresponding Morphogene $\mathcal{G}$ exceeds τ_{sim}. Thus, the joint training objective is

$$\max_{\phi,\theta}\ J(\phi,\theta) = \mathbb{E}\left[\sum_{t=1}^{T_{\text{des}}} r_t^{\text{prox}} + \sum_{t=T_{\text{des}}+1}^{\infty} \gamma^{\,t-T_{\text{des}}-1}\, r_t^{\text{env}}\right]. \tag{11}$$

Consequently, the morphology-design policy π_ϕ^{des} is optimized to maximize task return within the τ_{sim}-neighborhood of its conditioning Morphogene. This local search yields high-performing candidates for Morphogene updates, which subsequently drive global exploration of the latent space.

4.3 GeCode: Morphogene-Driven Co-design Optimization

GeCode performs morphology–control co-design in the compact Morphogene space, where each Morphogene serves as a design anchor for body–brain coordination (Figure 3; Algorithm 1). These anchors are updated using existing rollout statistics to efficiently explore diverse body–brain designs, *requiring neither additional environment interactions nor gradient-based latent optimization*.

Morphogene Initialization GeCode initializes a small set of Morphogenes, $\mathcal{P}_\mathcal{G} = \{\mathcal{G}_i\}_{i=1}^{N_G}$, by randomly and independently sampling from the learned RAE latent distribution. These Morphogenes are then iteratively refined throughout co-design, *with no new Morphogenes introduced thereafter*.

Morphogene-Conditioned Rollout At each co-design epoch, each rollout first samples a Morphogene $\mathcal{G}_i \in \mathcal{P}_\mathcal{G}$ according to $i \sim \mathbf{p}$, where $\mathbf{p} = (p_1, \ldots, p_{N_G})$ is the **performance-adaptive sampling distribution**. It is initialized only once at epoch 0 with $p_i = \frac{1}{N_G}$ and iteratively updated after every subsequent epoch according to Eq. (14). The sampled $\mathcal{G}_i$ remains fixed throughout the rollout and conditions both the morphology-design and control policies through AdaConcat (Figure 2(b)).

Because the morphology-design actions a^{des} in Eq. (9) are sampled stochastically, K rollouts conditioned on the same $\mathcal{G}_i$ may produce distinct morphology variants $\{\mathcal{M}_{i,k}\}_{k=1}^K$ within the local design region anchored by $\mathcal{G}_i$, thereby enabling ***local exploration*** around the anchor.

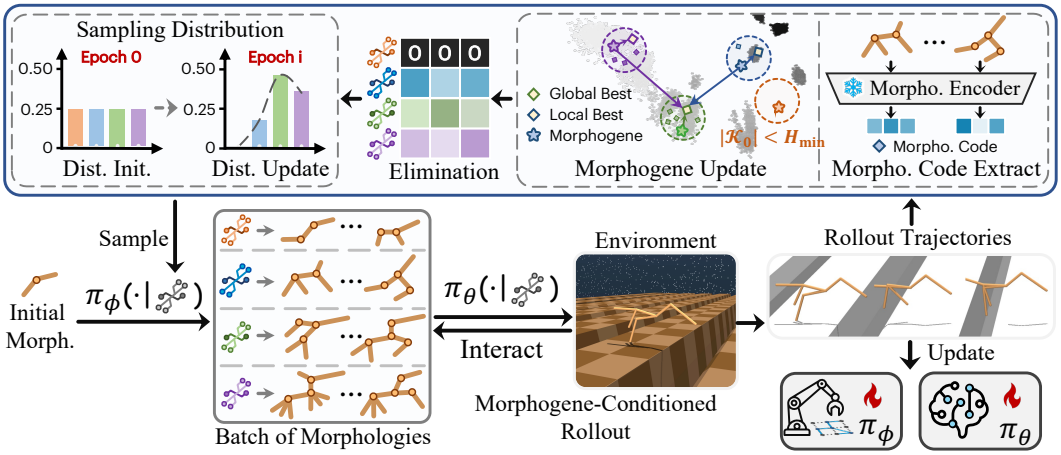


Figure 3: **Overview of GeCode.** GeCode reformulates morphology–control co-design as exploration over the compact Morphogene space, using a small set of Morphogenes as searchable design anchors. At each co-design epoch, sampled anchors jointly condition morphology generation and controller adaptation, while stochastic rollouts guided by a proximity objective explore their local neighborhoods. Rollout performance subsequently drives Morphogene updates and adaptive resampling, propagating high-performing design information across the set and enabling global latent-space exploration.

Morphogene Update After each epoch's rollouts, GeCode updates each Morphogene using the morphology codes of the best-performing design *within its local region* and *across the entire set*, referred to as the local and global best, respectively. The former preserves anchor-specific morphological characteristics, whereas the latter propagates the best-performing body–brain design across Morphogenes, thereby driving ***global exploration*** of the latent design space (see Figure 6(b)).

Specifically, each morphology $\mathcal{M}_{i,k}$ generated by rollout k conditioned on Morphogene $\mathcal{G}_i$ is mapped to its morphology code $\mathbf{z}_{i,k}^{\text{morph}}$ according to Eq. (4). To preserve locality around $\mathcal{G}_i$, we retain the rollout indices $\mathcal{K}_i = \{k \mid \text{sim}(\mathbf{z}_{i,k}^{\text{morph}}, \mathcal{G}_i) \geq \tau_{\text{sim}}\}$, where τ_{sim} filters out morphology codes that deviate excessively from their conditioning Morphogenes. The retained rollouts across all Morphogenes are collected in $\mathcal{K} = \{(i,k) \mid k \in \mathcal{K}_i,\ i = 1, \ldots, N_G\}$.

Let $\mathbf{z}_i^{\text{loc}}$ and $\mathbf{z}^{\text{glob}}$ denote the codes of the morphologies produced by the highest-return rollouts in $\mathcal{K}_i$ and $\mathcal{K}$, with corresponding returns R_i^{loc} and R^{glob}, respectively. Their weighted combination defines the update target for $\mathcal{G}_i$:

$$\overline{\mathcal{G}}_i = (1-\beta_i)\mathbf{z}_i^{\text{loc}} + \beta_i \mathbf{z}^{\text{glob}}, \qquad \mathcal{G}_i \leftarrow (1-\eta)\mathcal{G}_i + \eta\overline{\mathcal{G}}_i. \tag{12}$$

Here, $\beta_i = \min(1, \frac{R^{\text{glob}} - R_i^{\text{loc}}}{\max(|R^{\text{glob}}|, 1)})$ adaptively balances global guidance across Morphogenes, while η controls the Morphogene update rate.

Morphogene Elimination At the end of each epoch, a Morphogene $\mathcal{G}_i$ is retained only if at least $H_{\min}$ of its conditioned rollouts generate morphologies within its local region. Accordingly, the active Morphogene set is updated as

$$\mathcal{P}_{\mathcal{G}} = \{\mathcal{G}_i \mid \mathcal{G}_i \in \mathcal{P}_{\mathcal{G}},\ |\mathcal{K}_i| \geq H_{\min}\}, \tag{13}$$

This procedure progressively eliminates Morphogenes that cannot be reliably realized by the morphology-design policy until only one remains.

Adaptive Morphogene Sampling After eliminating Morphogenes, GeCode sets their next-epoch sampling probabilities $\mathbf{p}$ using the mean return of the top-N_{top} rollouts for each retained $\mathcal{G}_i$:

$$q_i = \text{AvgTop}\left(\{R_{i,k}\}_{k \in \mathcal{K}_i}, N_{\text{top}}\right), \qquad \mathbf{p} = \text{softmax}\left(\mathbf{q}\right). \tag{14}$$

Here, $R_{i,k}$ denotes the return of rollout k conditioned on $\mathcal{G}_i$, with $N_{\text{top}} \leq H_{\min}$.

Overall, GeCode coordinates global exploration through Morphogene updates with local exploration through stochastic Morphogene-conditioned rollouts. By operating on low-dimensional Morphogenes and reusing policy-training rollouts, it requires only lightweight vector operations, enabling efficient Morphogene-driven morphology–control co-design.

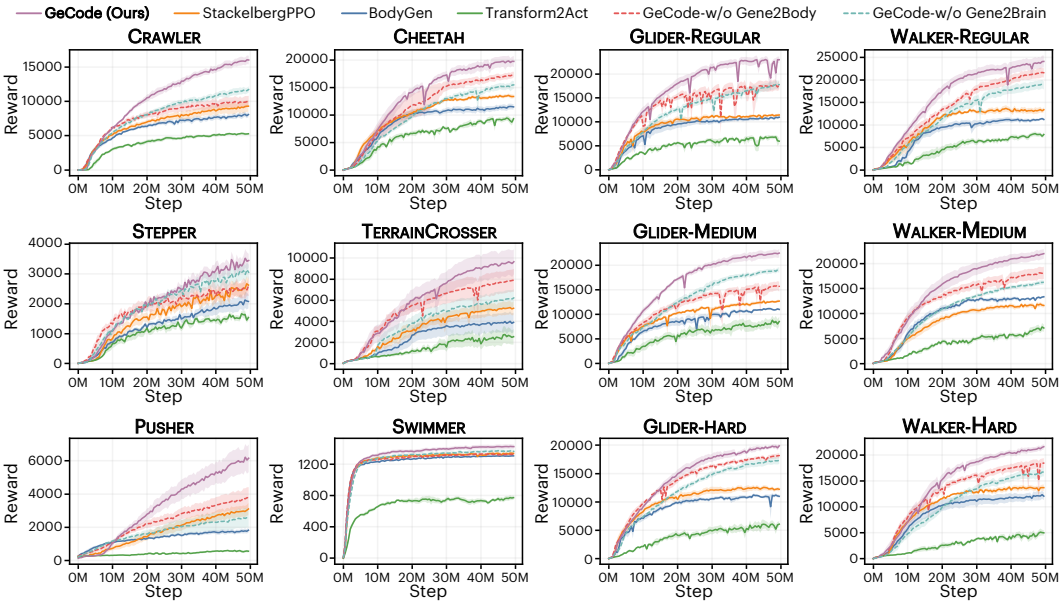


Figure 4: Performance comparison of GeCode with state-of-the-art baselines, including StackelbergPPO (Dai et al., 2026), BodyGen (Lu et al., 2025), and Transform2Act (Yuan et al., 2022), and its ablations, GeCode-w/o Gene2Body and GeCode-w/o Gene2Brain, across twelve co-design environments. Shaded regions indicate the standard error over five random seeds.

5 EXPERIMENTS

Our experiments evaluate whether Morphogene effectively bridges an agent's body and brain and whether GeCode leverages this coordination to achieve sample-efficient, high-performing morphology–control co-design. We further analyze the mechanisms underlying these gains.

Environments Following prior work (Lu et al., 2025; Dai et al., 2026), we evaluate GeCode across twelve MuJoCo environments (Todorov et al., 2012): CRAWLER, STEPPER, PUSHER, TERRAINCROSSER, CHEETAH, SWIMMER, GLIDER-(REGULAR, MEDIUM, HARD) and WALKER-(REGULAR, MEDIUM, HARD). These 2D and 3D benchmarks cover locomotion and manipulation across diverse design spaces and terrains. Further details are provided in App. B.

5.1 COMPARISON WITH BASELINES

Baselines We build GeCode on **(1) BodyGen** (Lu et al., 2025), a PPO-based Transformer framework for morphology–control co-design. GeCode augments this backbone with Morphogenes that bridge body generation and controller adaptation, while Morphogene updates drive efficient exploration of coordinated body–brain designs. BodyGen therefore serves as a direct baseline. We further compare GeCode with **(2) StackelbergPPO** (Dai et al., 2026), which formulates co-design as a Stackelberg game to explicitly model the optimization interdependence between morphology and control, and **(3) Transform2Act** (Yuan et al., 2022), which trains separate GNN-based policies within a unified PPO framework. Additional comparisons with evolutionary approaches such as **(4) NGE** (Wang et al., 2019) are reported in Table 3 of App. E.1. Implementation details are provided in App. D.

As shown in Figure 4, GeCode consistently outperforms all baselines by substantial margins, achieving an average final return approximately 69.48% higher than that of StackelbergPPO, a state-of-the-art co-design method. These gains extend beyond relatively simple tasks, such as GLIDER-REGULAR and WALKER-REGULAR, and become more pronounced in tasks with larger morphology spaces, including CRAWLER, GLIDER-HARD, and WALKER-HARD. This trend highlights how GeCode leverages Morphogene-guided exploration in a compact latent space to reduce effective search complexity as the morphology space expands. Performance gains on complex-terrain tasks such as STEPPER and TERRAINCROSSER further highlight Morphogene's role as a body–brain bridge, inte-

Table 1: Performance comparison of different Morphogene injection mechanisms. Results show mean episode rewards and standard errors over five random seeds.

	CRAWLER	STEPPER	PUSHER	CHEETAH	TERRAINCROSSER	SWIMMER
w/ Direct Concat	14049.3±533.4	3296.0±560.1	4289.5±989.3	18734.4±523.1	7826.2±798.0	1353.5±27.4
w/ AdaConcat	**16143.3±583.0**	**3688.6±271.0**	**6250.2±757.9**	**19888.7±944.6**	**9670.6±1176.4**	**1422.5±29.5**
	GLIDER-REGULAR	**GLIDER-MEDIUM**	**GLIDER-HARD**	**WALKER-REGULAR**	**WALKER-MEDIUM**	**WALKER-HARD**
w/ Direct Concat	20998.1±1002.0	20333.9±884.1	18031.0±1571.1	19197.0±2119.3	19949.6±856.5	20647.6±851.3
w/ AdaConcat	**23317.5±627.9**	**22376.4±809.9**	**19972.8±332.1**	**24306.1±1293.0**	**21973.3±918.9**	**21614.6±521.8**

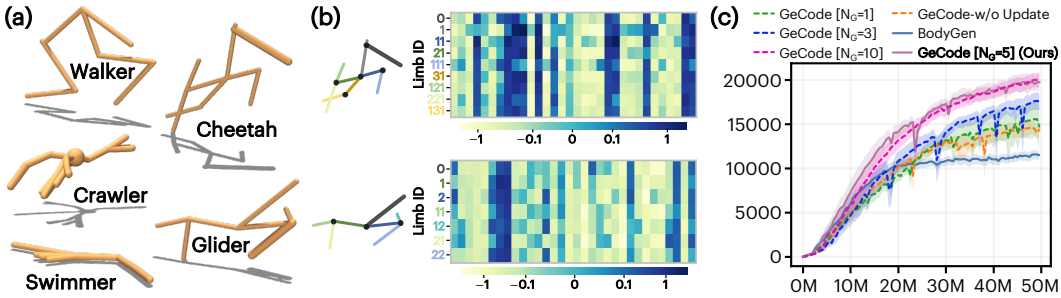


Figure 5: (a) Representative morphologies designed by GeCode. (b) Visualization of limb-specific conditioning produced by AdaConcat. (c) Ablations on Morphogene updates and set size N_G.

grating proprioception with morphology-dependent contact dynamics to improve control adaptation. Representative morphologies designed by GeCode are shown in Figure 5(a).

5.2 ABLATION AND ANALYSIS

In this section, we investigate five questions: (1) how Morphogene ***contributes separately*** to morphology design and control; (2) how its ***conditioning mechanism*** contributes to these benefits; (3) how Morphogene ***update*** and ***set size*** affect co-design; (4) whether ***structural regularization*** of the RAE latent space is necessary; and (5) what ***computational overhead*** the resulting procedure incurs.

Ablation of Morphogene Conditioning on Body and Brain Morphogene bridges body and brain by jointly conditioning morphology generation and controller adaptation. To isolate its role in each component, we remove Morphogene conditioning from either the morphology-design network (w/o Gene2Body) or the control network (w/o Gene2Brain). Figure 4 shows that both ablations consistently degrade performance, confirming their complementary contributions. Specifically, conditioning morphology generation on Morphogene is particularly important in morphology-intensive tasks with large, coupled design spaces, such as CRAWLER and STEPPER, where discovering a viable body is prerequisite to effective control. In simpler tasks with lower morphological complexity, such as CHEETAH and WALKER-REGULAR, conditioning the controller becomes more influential by enabling morphology-aware action adaptation.

Ablation on AdaConcat We evaluate the effectiveness of AdaConcat's adaptive conditioning for Morphogene injection by replacing it with direct concatenation in both the morphology-design and control networks, with results reported in Table 1. Direct concatenation treats Morphogene as a uniform global feature, disregarding limb-specific context and limiting its capacity to guide specialized design and control decisions. In contrast, AdaConcat adaptively modulates Morphogene for each limb (see Figure 5(b)), preserving the shared morphology blueprint while providing context-aware conditioning and thereby enabling finer-grained morphology–control coordination.

Ablation on Morphogene Updates and Set Size We isolate the contribution of Morphogene updates in Figure 5(c). Without updates, fixed Morphogenes cannot adapt alongside the evolving morphology-design and control policies, reducing them to static and potentially uninformative conditions. This result demonstrates that performance-guided updates are essential for guiding exploration toward promising latent regions and discovering higher-performing designs.

Figure 5(c) further examines the effect of Morphogene set size N_G. A single Morphogene ($N_G = 1$) enables limited exploration but provides insufficient coverage of the design space, whereas too many Morphogenes ($N_G = 10$) disperse early rollouts and slow initial convergence. The comparable final

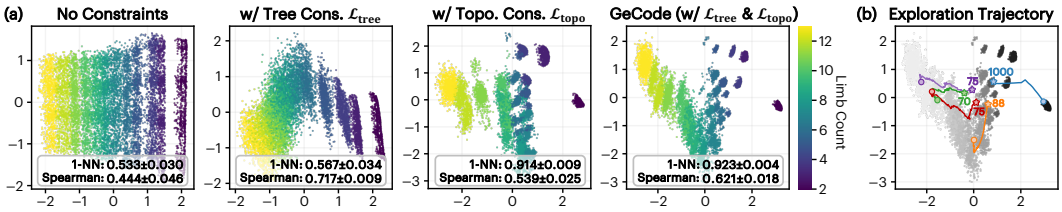
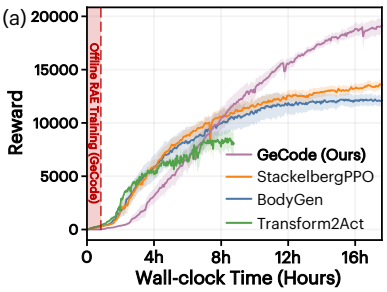


Figure 6: (a) Ablation study of structural constraints (i.e., $\mathcal{L}_{\text{topo}}$ and $\mathcal{L}_{\text{tree}}$) in the latent space. We report topology-aware 1-NN consistency and the Spearman rank correlation between latent and tree-edit distances, denoted "1-NN↑" and "Spearman↑", respectively. (b) Morphogene update trajectories over 1,000 co-design epochs. ○ and ☆ denote the initial and final positions, respectively, while numerical labels indicate elimination epochs. See Figure 14 in App. E.4 for additional trajectories.



(b) Training-Time Breakdown

	GeCode	StackelbergPPO	BodyGen	Transform2Act
Offline RAE Training	0.77h	N/A	N/A	N/A
Co-design Optimization	21.38h	19.96h	18.02h	8.94h
Total Time	22.15h	19.96h	18.02h	8.94h

(c) Cost to Match Baseline Performance

		StackelbergPPO	BodyGen	Transform2Act
Wall-clock Time	GeCode	10.30h ↓ **9.66h**	9.18h ↓ **8.84h**	7.06h ↓ **1.88h**
Environment Interactions (50M Total)	GeCode	21.2M ↓ **28.8M**	18.7M ↓ **31.3M**	13.7M ↓ **36.3M**

Figure 7: Computational efficiency of GeCode. (a) Performance comparison under a wall-clock time budget. (b) Detailed training costs, together with the resources required for GeCode to match each baseline's best performance. Further results are provided in Table 7 and Table 8 in App. E.3.

performance of $N_G = 5$ and $N_G = 10$ indicates that five Morphogenes provide sufficient coverage while balancing exploration diversity and learning efficiency.

Ablation on Structural Constraints in the Latent Space We further ablate the structural constraints used in RAE training. Figure 6(a) visualizes the learned Morphogene spaces and quantifies their local and global organization through topology 1-NN consistency and the Spearman correlation, respectively. Specifically, $\mathcal{L}_{\text{topo}}$ promotes local clustering among morphologies with shared topologies, whereas $\mathcal{L}_{\text{tree}}$ preserves global structural relationships. By jointly optimizing both objectives, the full model balances local topology consistency with global structural alignment, organizing the latent geometry according to morphological similarity and enabling Morphogene updates to induce smooth, structurally meaningful variations rather than arbitrary perturbations. Figure 6(b) shows that Morphogenes traverse broad regions of the latent space along structurally coherent trajectories.

Analysis of Computational Cost We evaluate computational efficiency against wall-clock time. As shown in Figure 7(a), multiple Morphogenes expose the controller to diverse morphologies early in training, imposing a modest initial adaptation cost. As performance-guided updates focus the anchors on promising morphologies and compatible controllers, GeCode converges rapidly and substantially outperforms the baselines under matched time budgets. Its significantly higher final returns further indicate superior body–brain designs. Specifically, GeCode reaches StackelbergPPO's peak performance with 48.4% less wall-clock time and 28.8M fewer environment interactions (Figure 7(c)). Meanwhile, one-time offline RAE training requires no environment interaction and accounts for only 3.5% of total training time, while Morphogene updates reuse existing rollouts and require only lightweight latent operations, without additional simulations or optimization loops.

6 Conclusion

We introduced GeCode, a Morphogene-driven framework for morphology–control co-design. Morphogene bridges body and brain by jointly conditioning morphology generation and controller adaptation through AdaConcat. GeCode further reformulates co-design as exploration in the compact Morphogene space, where Morphogenes serve as searchable design anchors. Stochastic rollouts enable local exploration around each anchor, while performance-guided updates move the anchors toward promising regions for global exploration. Extensive experiments show that GeCode achieves faster convergence and higher final performance. Limitations are discussed in App. F.2.

AI USE STATEMENT

The authors used generative AI tools to assist with writing, LaTeX and figure formatting, and code editing. All claims, citations, code, figures, and numerical results were independently reviewed and verified by the authors, who take full responsibility for the final submission.

ETHICS STATEMENT

This work studies morphology–control co-design in simulated environments using open-source simulators and benchmarks. It involves no human participants, animals, personal or sensitive data. We identify no specific ethical risks associated with the experiments presented in this work.

REPRODUCIBILITY STATEMENT

We describe GeCode and the experimental protocol in detail in the main paper and appendix. The appendix further provides environment and task specifications, algorithmic and implementation details, and analyses of sample and training efficiency. Additional results and ablation studies are included to facilitate reproducibility. Anonymous code is available at `https://anonymous.4open.science/w/GeCode-BC8B/`.

REFERENCES

Chiara Bartolozzi, Giacomo Indiveri, and Elisa Donati. Embodied neuromorphic intelligence. *Nature Communications*, 13(1):1024, 2022.

Jagdeep Bhatia, Holly Jackson, Yunsheng Tian, Jie Xu, and Wojciech Matusik. Evolution gym: A large-scale benchmark for evolving soft robots. In *Advances in Neural Information Processing Systems (NeurIPS'21)*, pp. 2201–2214, 2021.

Anastasia Bolotnikova, Kevin Holdcroft, Henry Cerbone, Christoph Belke, Auke Ijspeert, and Jamie Paik. Optimized user-guided motion control of modular robots. *Nature communications*, 16(1): 8675, 2025.

Longbing Cao. Humanoid robots and humanoid ai: Review, perspectives and directions. *ACM Computing Surveys*, 58(4):1–37, 2025.

Nick Cheney, Josh Bongard, Vytas SunSpiral, and Hod Lipson. Scalable co-optimization of morphology and control in embodied machines. *Journal of The Royal Society Interface*, 15(143):20170937, 2018.

Hillel J Chiel and Randall D Beer. The brain has a body: adaptive behavior emerges from interactions of nervous system, body and environment. *Trends in Neurosciences*, 20(12):553–557, 1997.

Yanning Dai, Yuhui Wang, Dylan Ashley, and Jürgen Schmidhuber. Efficient morphology-control co-design via stackelberg proximal policy optimization. In *Proceedings of the International Conference on Learning Representations (ICLR'26)*, pp. 144280–144316, 2026.

Kehlani Fay, Darin Djapri, Anya Zorin, James Clinton, Ali El Lahib, Hao Su, Michael Tolley, Sha Yi, and Xiaolong Wang. House of dextra: Cross-embodied co-design for dexterous hands. In *Proceedings of the International Conference on Learning Representations (ICLR'26)*, pp. 14889–14912, 2026.

Fu Feng, Jing Wang, Xu Yang, and Xin Geng. Learngene: Inheritable "genes" in intelligent agents. *Artificial Intelligence*, 348:104421, 2025.

Fu Feng, Ruixiao Shi, Yucheng Xie, Jianlu Shen, Jing Wang, and Xin Geng. Knowledge diversion for efficient morphology control and policy transfer. In *Proceedings of the International Conference on Machine Learning (ICML'26)*, pp. 1–17, 2026.

Gilbert Feng, Hongbo Zhang, Zhongyu Li, Xue Bin Peng, Bhuvan Basireddy, Linzhu Yue, Zhitao Song, Lizhi Yang, Yunhui Liu, Koushil Sreenath, et al. Genloco: Generalized locomotion controllers for quadrupedal robots. In *Proceedings of the Conference on Robot Learning (CoRL'22)*, pp. 1893–1903, 2023.

Jonas Simon Fleck, Sophie Martina Johanna Jansen, Damian Wollny, Fides Zenk, Makiko Seimiya, Akanksha Jain, Ryoko Okamoto, Malgorzata Santel, Zhisong He, J Gray Camp, and Barbara Treutlein. Inferring and perturbing cell fate regulomes in human brain organoids. *Nature*, 621: 365–372, 2023.

Ali Ghadirzadeh, Xi Chen, Petra Poklukar, Chelsea Finn, Mårten Björkman, and Danica Kragic. Bayesian meta-learning for few-shot policy adaptation across robotic platforms. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'21)*, pp. 1274–1280, 2021.

Agrim Gupta, Silvio Savarese, Surya Ganguli, and Li Fei-Fei. Embodied intelligence via learning and evolution. *Nature Communications*, 12:5721, 2021.

Agrim Gupta, Linxi Fan, Surya Ganguli, and Li Fei-Fei. Metamorph: Learning universal controllers with transformers. In *Proceedings of the International Conference on Learning Representations (ICLR'22)*, 2022.

David Ha. Reinforcement learning for improving agent design. *Artificial life*, 25(4):352–365, 2019.

Tuomas Haarnoja, Ben Moran, Guy Lever, Sandy H Huang, Dhruva Tirumala, Jan Humplik, Markus Wulfmeier, Saran Tunyasuvunakool, Noah Y Siegel, Roland Hafner, et al. Learning agile soccer skills for a bipedal robot with deep reinforcement learning. *Science Robotics*, 9(89):eadi8022, 2024.

Lei Han, Qingxu Zhu, Jiapeng Sheng, Chong Zhang, Tingguang Li, Yizheng Zhang, He Zhang, Yuzhen Liu, Cheng Zhou, Rui Zhao, et al. Lifelike agility and play in quadrupedal robots using reinforcement learning and generative pre-trained models. *Nature Machine Intelligence*, 6:787–798, 2024.

Wenlong Huang, Igor Mordatch, and Deepak Pathak. One policy to control them all: Shared modular policies for agent-agnostic control. In *Proceedings of the International Conference on Machine Learning (ICML'20)*, pp. 4455–4464, 2020.

Hyeonseong Jeon, Ainaz Eftekhar, Aaron Walsman, Kuo-Hao Zeng, Ali Farhadi, and Ranjay Krishna. Convergent functions, divergent forms. *Advances in Neural Information Processing Systems (NeurIPS'25)*, 38:46049–46076, 2025.

Ryosuke Koike, Ryo Ariizumi, and Fumitoshi Matsuno. Simultaneous optimization of discrete and continuous parameters defining a robot morphology and controller. *IEEE Transactions on Neural Networks and Learning Systems*, 35(10):13816–13829, 2024.

Hao Liu, Yanda Yang, Ting Wang, Jintao Yin, and Hongqiang Wang. Bioinspired growable humanoid robot with bone-mimetic linkages for versatile mobility. *Science Advances*, 12(4):eaea2831, 2026.

Huaping Liu, Di Guo, and Angelo Cangelosi. Embodied intelligence: A synergy of morphology, action, perception and learning. *ACM Computing Surveys*, 57(7):1–36, 2025.

Junfeng Long, ZiRui Wang, Quanyi Li, Liu Cao, Jiawei Gao, and Jiangmiao Pang. Hybrid internal model: Learning agile legged locomotion with simulated robot response. In *Proceedings of the International Conference on Learning Representation (ICLR'24)*, pp. 14084–14100, 2024.

Haofei Lu, Zhe Wu, Junliang Xing, Jianshu Li, Ruoyu Li, Zhe Li, and Yuanchun Shi. Bodygen: Advancing towards efficient embodiment co-design. In *Proceedings of the International Conference on Learning Representations (ICLR'25)*, pp. 41559–41580, 2025.

Enrico Marchesini, Davide Corsi, and Alessandro Farinelli. Genetic soft updates for policy evolution in deep reinforcement learning. In *Proceedings of the International Conference on Learning Representations (ICLR'21)*, 2021.

David Matthews, Andrew Spielberg, Daniela Rus, Sam Kriegman, and Josh Bongard. Efficient automatic design of robots. *Proceedings of the National Academy of Sciences*, 120(41):e2305180120, 2023.

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *Proceedings of the International Conference on Learning Representations (ICLR'19)*, 2019.

Austin Patel and Shuran Song. Get-zero: Graph embodiment transformer for zero-shot embodiment generalization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'25)*, pp. 14262–14269, 2025.

Deepak Pathak, Christopher Lu, Trevor Darrell, Phillip Isola, and Alexei A Efros. Learning to control self-assembling morphologies: a study of generalization via modularity. In *Advances in Neural Information Processing Systems (NeurIPS'19)*, 2019.

Kevin Qiu, Władysław Pałucki, Krzysztof Ciebiera, Paweł Fijałkowski, Marek Cygan, and Łukasz Kuciński. Robomorph: Evolving robot morphology using large language models. In *ICLR 2025 Workshop on Foundation Models in the Wild*, 2025.

Kenneth O Stanley, Jeff Clune, Joel Lehman, and Risto Miikkulainen. Designing neural networks through neuroevolution. *Nature Machine Intelligence*, 1:24–35, 2019.

Luke Strgar and Sam Kriegman. Accelerated co-design of robots through morphological pretraining. In *Proceedings of the International Conference on Learning Representations (ICLR'26)*, pp. 7450–7470, 2026.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'12)*, pp. 5026–5033, 2012.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS'17)*, 2017.

Tingwu Wang, Renjie Liao, Jimmy Ba, and Sanja Fidler. Nervenet: Learning structured policy with graph neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR'18)*, 2018.

Tingwu Wang, Yuhao Zhou, Sanja Fidler, and Jimmy Ba. Neural graph evolution: Towards efficient automatic robot design. In *Proceedings of the International Conference on Learning Representations (ICLR'19)*, 2019.

Yuxing Wang, Shuang Wu, Haobo Fu, Qiang Fu, Tiantian Zhang, Yongzhe Chang, and Xueqian Wang. Curriculum-based co-design of morphology and control of voxel-based soft robots. In *Proceedings of the International Conference on Learning Representations (ICLR'23)*, 2023.

Yuxing Wang, Zhiyu Chen, Tiantian Zhang, Qiyue Yin, Yongzhe Chang, Zhiheng Li, Liang Wang, and Xueqian Wang. Embodied co-design for rapidly evolving agents: Taxonomy, frontiers, and challenges. *arXiv preprint arXiv:2512.04770*, 2025.

Yucheng Xie, Fu Feng, Ruixiao Shi, Jing Wang, Yong Rui, and Xin Geng. KIND: Knowledge integration and diversion for training decomposable models. In *Proceedings of the International Conference on Machine Learning (ICML'25)*, pp. 68626–68645, 2025.

Yucheng Xie, Fu Feng, Ruixiao Shi, Jing Wang, Yong Rui, and Xin Geng. Divcontrol: Knowledge diversion for controllable image generation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI'26)*, pp. 27108–27116, 2026.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *Proceedings of International Conference on Machine Learning (ICML'20)*, pp. 10524–10533, 2020.

Zheng Xiong, Jacob Beck, and Shimon Whiteson. Universal morphology control via contextual modulation. In *Proceedings of the International Conference on Machine Learning (ICML'23)*, pp. 38286–38300, 2023.

Zheng Xiong, Risto Vuorio, Jacob Beck, Matthieu Zimmer, Kun Shao, and Shimon Whiteson. Distilling morphology-conditioned hypernetworks for efficient universal morphology control. In *Proceedings of the International Conference on Machine Learning (ICML'24)*, pp. 54777–54791, 2024.

Ye Yuan, Yuda Song, Zhengyi Luo, Wen Sun, and Kris M. Kitani. Transform2act: Learning a transform-and-control policy for efficient agent design. In *Proceedings of the International Conference on Learning Representations (ICLR'22)*, 2022.

Anthony M Zador. A critique of pure learning and what artificial neural networks can learn from animal brains. *Nature Communications*, 10:3770, 2019.

Allan Zhao, Jie Xu, Mina Konaković-Luković, Josephine Hughes, Andrew Spielberg, Daniela Rus, and Wojciech Matusik. Robogrammar: graph grammar for terrain-optimized robot design. *ACM Transactions on Graphics*, 39(6):1–16, 2020.

A Additional Related Work

"Genes" in Intelligent Agents Biological genes have inspired diverse machine learning methods. In evolutionary search for intelligent agents, candidate architectures and morphologies are encoded as gene-like genotypes, often binary strings, and evolved through selection and variation (Gupta et al., 2021; Qiu et al., 2025; Stanley et al., 2019; Marchesini et al., 2021). Such genotypes serve as search variables rather than semantically meaningful agent-level abstractions. Unlike search-oriented genotypes, learngenes (Feng et al., 2025) focus on knowledge inheritance by encoding task-agnostic knowledge from ancestral models into compact, inheritable network fragments for efficient downstream adaptation (Feng et al., 2026; Xie et al., 2025; 2026). Unlike both, Morphogene is an agent-level blueprint that jointly guides morphology and control generation, serving as a bridge for body–brain coordination rather than a search genotype or a carrier of transferable knowledge.

Universal Morphology Control Universal morphology control aims to learn a single policy capable of controlling agents with diverse body structures, thereby avoiding separate controller training for each morphology (Nagabandi et al., 2019; Pathak et al., 2019; Patel & Song, 2025). Early MLP-based approaches have limited flexibility when agents differ in limb count and action dimensionality (Ghadirzadeh et al., 2021; Feng et al., 2023). Graph neural networks address this limitation by representing limbs or actuators as nodes and propagating information according to morphological topology (Wang et al., 2018; Huang et al., 2020). More recently, MetaMorph employs a morphology-aware Transformer to model whole-body interactions (Gupta et al., 2022), whereas ModuMorph (Xiong et al., 2023) and HyperDistill (Xiong et al., 2024) improve cross-morphology generalization through morphology-conditioned modulation and policy distillation, respectively. DivMorph (Feng et al., 2026) further adopts a modular training paradigm that leverages knowledge diversion (Xie et al., 2025) to learn decomposable Transformer controllers, preserving agent-specific knowledge while supporting unified control and transfer. Collectively, these advances establish an important foundation for our Transformer-based approach to morphology–control co-design.

B Environment and Task Details

B.1 Benchmark Setup

Benchmark Environments Following prior work (Lu et al., 2025; Dai et al., 2026), we evaluate GeCode on twelve morphology–control co-design tasks: Crawler, Stepper, Pusher, TerrainCrosser, Cheetah, Swimmer, Glider-(Regular, Medium, Hard), and Walker-(Regular, Medium, Hard). Figure 9 provides a visual overview of the corresponding task environments. These tasks span planar and three-dimensional locomotion, uneven-terrain traversal, and object manipulation, providing a diverse benchmark for evaluating the effectiveness of morphology–control co-design across morphology spaces and task complexities.

Each agent is modeled as a morphology graph $\mathcal{M} = (V, E, A)$, where each node $u \in V$ represents a rigid limb and each edge $e(u, v) \in E$ represents a motor-actuated joint. The limb attributes Λ_u^V specify geometric properties, such as length and size, while the joint attributes $\Lambda_{e(u,v)}^E$ specify mechanical properties, such as rotation limits and maximum motor torque.

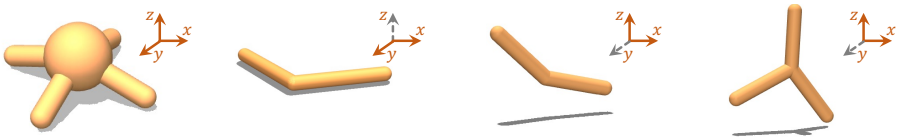


Figure 8: **Overview of Task-Specific Initial Morphologies.** Initial Type 1 is a three-dimensional morphology comprising four limbs. Initial Type 2 comprises two limbs connected by one joint in the x–y plane. Initial Type 3 comprises two limbs connected by one joint in the x–z plane. Initial Type 4 comprises three limbs connected by two joints in the x–z plane. These morphologies serve as the starting points for sequential morphology design.

Co-Design Procedure Each episode comprises a morphology-design stage Φ_{des} followed by a control stage Φ_{ctrl}. Morphology design begins from a task-specific initial morphology $\mathcal{M}_0$, selected from the four structures shown in Figure 8. Over T_{des} steps, the design policy π_ϕ^{des} sequentially updates $\mathcal{M}_t$ through morphology-editing actions a_t^{des}, first modifying the discrete topology and then updating continuous physical attributes. During topology design, a_t^{des} may add or remove a terminal limb or leave the structure unchanged. Each edit must satisfy task-specific upper bounds on tree depth, total limb count, and the number of children per limb, ensuring structurally valid morphologies. The subsequent attribute update specifies limb offsets, capsule radii, attachment positions, and actuator gear ratios. The resulting final morphology $\mathcal{M}^\star = \mathcal{M}_{T_{\text{des}}}$ remains fixed throughout Φ_{ctrl}.

Once the design stage Φ_{des} is complete, the simulator instantiates the final morphology $\mathcal{M}^\star$ as a physical agent and fixes its structural and physical parameters throughout the control stage Φ_{ctrl}. At each control step t, the policy π_θ^{ctrl} maps the proprioceptive state s_t^{ctrl}, which includes position, linear velocity, and angular velocity, to a continuous motor action a_t^{ctrl}. The resulting cumulative task return evaluates the body–brain pair $(\mathcal{M}^\star, \pi_\theta^{\text{ctrl}})$ and provides the shared objective for jointly optimizing the design policy π_ϕ^{des} and control policy π_θ^{ctrl}.

B.2 Task-Specific Specifications

We next describe the environment, objective, and design constraints of each task.

Crawler Crawler evaluates three-dimensional forward locomotion over level terrain. Its initial morphology is the Type 1 initial morphology in Figure 8, comprising a free-floating root body and four primary limbs. The morphology is limited to a tree depth of 4, a total of 29 bodies, and at most two child limbs per body. During attribute design, the root offset remains fixed, while non-root limb offsets are optimized in the x–y plane within $[-0.5, 0.5]^2$. Capsule radii, attachment positions, and actuator gear ratios are constrained to $[0.03, 0.10]$, $[0, 0.2]$, and $[20, 400]$, respectively.

The reward balances forward velocity and control effort:

$$r_t = \frac{x_{t+1} - x_t}{\Delta t} - \omega \cdot \frac{1}{N_L} \left\| a_t^{\text{ctrl}} \right\|_2^2, \tag{15}$$

where x_t denotes the root position along the forward axis, $\Delta t = 0.04$ is the control interval, $\omega = 10^{-4}$ weights the control penalty, N_L is the number of limbs, and a_t^{ctrl} is the continuous motor-action vector. An episode terminates if the root height falls outside $(0, 2.0)$ or its tilt from the upright orientation exceeds $60°$.

Stepper Stepper evaluates three-dimensional locomotion over a sequence of elevated steps. It uses the Type 1 initial morphology in Figure 8 and shares the topology constraints and continuous attribute ranges of Crawler: a maximum tree depth of 4, at most 29 bodies, and no more than two child limbs per body. The task adopts the forward-velocity and control-effort reward in Eq. (15), with control interval $\Delta t = 0.04$ and penalty weight $\omega = 10^{-4}$. Because the terrain elevation varies along the course, termination is determined by the root height relative to the local ground surface. An episode terminates if this relative height falls outside $(0, 2.0)$ or if the root tilt from the upright orientation exceeds $90°$. The task therefore favors morphologies with sufficient support, ground clearance, and reach to maintain forward progress across abrupt elevation changes.

Pusher Pusher evaluates three-dimensional object manipulation using the Type 1 initial morphology shown in Figure 8. The task requires the agent to push a movable cube along the positive x-axis while minimizing lateral drift along the y-axis. The morphology is constrained to a maximum tree depth of 4, at most 29 bodies, and no more than two child limbs per body. Its continuous attribute ranges are identical to those of Crawler and Stepper.

Let $(x_t^{\text{cube}}, y_t^{\text{cube}})$ denote the planar coordinates of the cube, and let $\mathbf{p}_t^{\text{cube}}$ and $\mathbf{p}_t^{\text{root}}$ denote the three-dimensional positions of the cube and robot root, respectively. The reward is

$$r_t = \frac{x_{t+1}^{\text{cube}} - x_t^{\text{cube}}}{\Delta t} - \kappa \frac{|y_{t+1}^{\text{cube}} - y_t^{\text{cube}}|}{\Delta t} + \frac{1}{1 + \|\mathbf{p}_{t+1}^{\text{cube}} - \mathbf{p}_{t+1}^{\text{root}}\|_2} - \omega \cdot \frac{1}{N_L} \left\| a_t^{\text{ctrl}} \right\|_2^2. \tag{16}$$

Here, $\kappa = 0.1$ weights lateral drift, $\Delta t = 0.04$ is the control interval, and $\omega = 10^{-4}$ weights control effort. The four terms reward forward cube motion, penalize lateral drift, encourage robot–cube

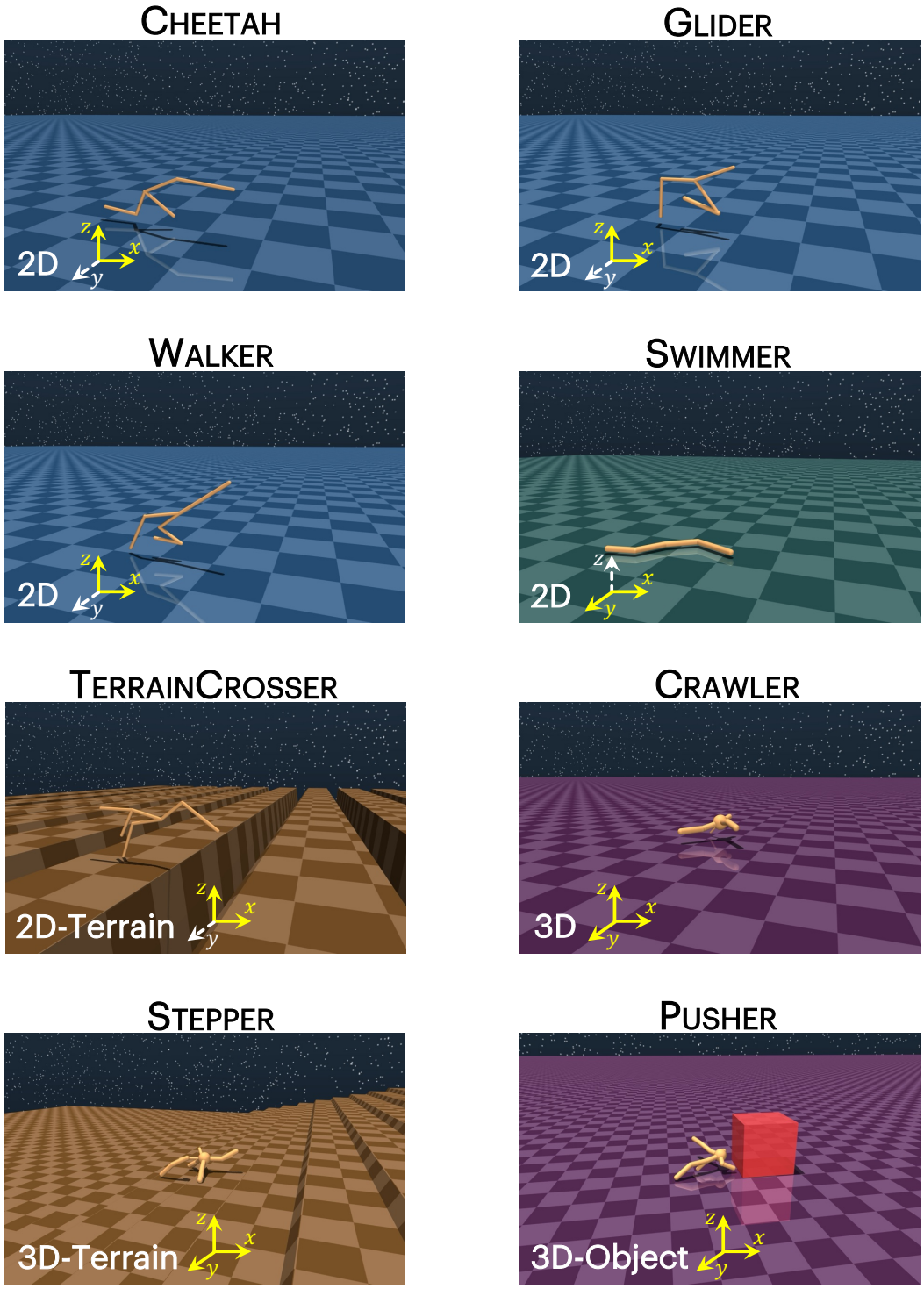


Figure 9: **Overview of the Benchmark Environments.** CRAWLER, STEPPER, and PUSHER are three-dimensional (3D) tasks, whereas the remaining environments are two-dimensional (2D), operating in either the x–z or x–y plane. The benchmarks span diverse morphology depths, symmetries, limb configurations, terrains, and task objectives, enabling a comprehensive evaluation of morphology–control co-design across locomotion and manipulation settings.

proximity, and penalize control effort, respectively. An episode terminates if the root height falls outside $(0, 2.0)$ or its tilt from the upright orientation exceeds $60°$.

Cheetah CHEETAH evaluates rapid planar locomotion along the positive x-axis. The agent uses the Type 3 initial morphology in Figure 8, comprising a root body and one primary limb arranged in the x–z plane. The morphology is constrained to a maximum tree depth of 4, at most 14 bodies, and no more than three child limbs per body. Non-root limb offsets are optimized within $[-0.5, 0.5]^2$ in the x–z plane, while capsule radii, attachment positions, and actuator gear ratios are restricted to $[0.03, 0.08]$, $[0, 0.5]$, and $[20, 400]$, respectively.

The reward is defined by the forward velocity of the root:

$$r_t = \frac{x_{t+1} - x_t}{\Delta t}. \tag{17}$$

Here, x_t denotes the root position along the forward axis and $\Delta t = 0.008$ is the control interval. An episode terminates if the root height falls outside $(0.7, 2.0)$ or if its absolute angular deviation from the upright orientation exceeds $20°$.

TerrainCrosser TERRAINCROSSER evaluates planar locomotion over discontinuous terrain with repeated gaps. It uses the Type 3 initial morphology in Figure 8 and shares the topology constraints and continuous attribute ranges of CHEETAH: a maximum tree depth of 4, at most 14 bodies, and no more than three child limbs per body. The task adopts the forward-velocity reward in Eq. (17), with control interval $\Delta t = 0.008$. Compared with CHEETAH, TERRAINCROSSER imposes a stricter minimum root height. An episode terminates if the root height falls outside $(1.0, 2.0)$ or if the absolute root pitch exceeds $20°$. The task therefore favors morphologies that combine rapid forward movement with sufficient reach, clearance, and stability to cross gaps.

Swimmer SWIMMER evaluates planar propulsion through a viscous fluid. It uses the Type 2 initial morphology in Figure 8, comprising a root segment connected to one actuated limb in the x–y plane. The morphology is constrained to a maximum tree depth of 4, at most 14 bodies, and no more than three child limbs per body. Non-root limb offsets are optimized within $[-1.5, 1.5]^2$, while capsule radii, attachment positions, and actuator gear ratios are restricted to $[0.06, 0.20]$, $[0, 0.2]$, and $[20, 300]$, respectively. The task adopts the forward-velocity and control-effort reward in Eq. (15), with control interval $\Delta t = 0.04$ and penalty weight $\omega = 10^{-4}$. No posture-based early termination is applied; each episode continues until the predefined time horizon.

Glider-Regular GLIDER-REGULAR evaluates planar locomotion within a compact, weakly branching morphology space. It uses the Type 4 initial morphology in Figure 8, comprising a root body and two primary limbs. The morphology is restricted to a maximum tree depth of 3, at most 5 bodies, and no more than one child limb per body, favoring chain-like structures. Limb offsets are optimized within $[-0.5, 0.5]^2$ in the task plane, while capsule radii, attachment positions, and actuator gear ratios are constrained to $[0.03, 0.08]$, $[0, 0.5]$, and $[20, 400]$, respectively. The task adopts the forward-velocity reward in Eq. (17), with control interval $\Delta t = 0.008$. An episode terminates if the root height falls outside $(0.7, 2.0)$ or if the absolute root pitch exceeds $60°$.

Glider-Medium GLIDER-MEDIUM uses the same Type 4 initial morphology, environment dynamics, continuous attribute ranges, and termination conditions as GLIDER-REGULAR. It also adopts the forward-velocity reward in Eq. (17), with $\Delta t = 0.008$. The maximum tree depth remains 3, while the body limit increases from 5 to 7 and the maximum number of child limbs per body increases from one to two. This larger branching factor introduces a moderately more complex topology space.

Glider-Hard GLIDER-HARD further expands the topology space by increasing the body limit from 7 to 9 and allowing up to three child limbs per body, while retaining a maximum tree depth of 3. All other design ranges, environment dynamics, reward, and termination conditions remain unchanged. Together, the three GLIDER settings provide a controlled evaluation of increasing morphology size and branching complexity under an otherwise identical locomotion objective.

Walker-Regular WALKER-REGULAR evaluates planar locomotion in a deeper morphology space than the GLIDER tasks, starting from the Type 4 morphology shown in Figure 8. The design space

permits a maximum tree depth of 4, at most 7 bodies, and no more than one child limb per body, thereby favoring deep, chain-like structures. Limb offsets are optimized within $[-0.5, 0.5]^2$ in the task plane, while capsule radii, attachment positions, and actuator gear ratios are restricted to $[0.03, 0.08]$, $[0, 0.5]$, and $[20, 400]$, respectively. The task uses the forward-velocity reward defined in Eq. (17), with a control interval of $\Delta t = 0.008$. An episode terminates if the root height falls outside $(0.7, 2.0)$ or if the absolute root pitch exceeds 60°.

Walker-Medium WALKER-MEDIUM retains the initial morphology, environment dynamics, continuous attribute ranges, reward, and termination conditions of WALKER-REGULAR. The maximum tree depth remains 4, while the body limit increases from 7 to 15 and the maximum number of child limbs per body increases from one to two. This expanded topology space accommodates both chain-like and moderately branched locomotor structures.

Walker-Hard WALKER-HARD further expands the topology space by increasing the body limit from 15 to 27 and allowing up to three child limbs per body, while retaining a maximum tree depth of 4. The initial morphology, environment dynamics, continuous attribute ranges, reward, and termination conditions remain unchanged. This setting defines the largest planar design space among the WALKER tasks and requires coordinated control of a potentially larger and more highly branched articulated body.

C METHOD DETAILS

C.1 ALGORITHM DETAILS

GeCode extends BodyGen (Lu et al., 2025) with a compact Morphogene space for explicit coordination between morphology design and control. The overall procedure comprises two phases. First, a structure-aware RAE is trained offline on randomly generated morphologies to construct the Morphogene space. GeCode then performs Morphogene-driven co-design using a small set of Morphogenes as latent design anchors, which jointly condition the morphology-design and control policies through AdaConcat. During co-design, policy rollouts provide both learning experience and performance statistics for iteratively updating, eliminating, and resampling these anchors. The complete procedure is summarized in Algorithm 1.

C.2 STRUCTURAL REGULARIZATION OF THE MORPHOGENE SPACE

As discussed in Sec. 4.1, reconstruction preserves the information required to recover a morphology but does not explicitly organize the latent geometry. We therefore impose two complementary structural regularizers on the morphology codes: a topology-aware contrastive loss $\mathcal{L}_{\text{topo}}$ for local topology clustering and a tree-structured metric loss $\mathcal{L}_{\text{tree}}$ for global structural alignment.

For a mini-batch $\mathcal{B}$, let $\mathbf{z}_i^{\text{morph}} = E_\psi(\mathcal{M}_i^{\text{vec}})$ denote the morphology code of sample i, and let $\widetilde{\mathbf{z}}_i^{\text{morph}} = \frac{\mathbf{z}_i^{\text{morph}}}{\|\mathbf{z}_i^{\text{morph}}\|_2}$ be its ℓ_2-normalized representation. The topology-aware contrastive loss treats morphologies with isomorphic topology graphs as positive pairs:

$$\mathcal{L}_{\text{topo}} = -\frac{1}{|\mathcal{B}_+|} \sum_{i \in \mathcal{B}_+} \frac{1}{|\mathcal{P}(i)|} \sum_{p \in \mathcal{P}(i)} \log \frac{\exp\Big((\widetilde{\mathbf{z}}_i^{\text{morph}})^\top \widetilde{\mathbf{z}}_p^{\text{morph}} / \tau_{\text{topo}}\Big)}{\displaystyle\sum_{k \in \mathcal{B} \setminus \{i\}} \exp\Big((\widetilde{\mathbf{z}}_i^{\text{morph}})^\top \widetilde{\mathbf{z}}_k^{\text{morph}} / \tau_{\text{topo}}\Big)}. \tag{18}$$

Here, $\mathcal{P}(i) = \{p \in \mathcal{B} \setminus \{i\} \mid \mathcal{T}_p \cong \mathcal{T}_i\}$ contains the samples whose topology graphs are isomorphic to that of sample i, $\mathcal{B}_+ = \{i \in \mathcal{B} \mid |\mathcal{P}(i)| > 0\}$ contains samples with at least one positive pair, and τ_{topo} is the temperature parameter. This objective draws morphology codes with the same topology closer while separating those with different topologies.

To preserve finer-grained structural relationships beyond topology classes, we construct a set of index triplets $(i, i_+, i_-) \in \mathcal{Q}$, corresponding to a reference morphology, a structurally similar morphology, and a structurally dissimilar morphology, respectively. The positive index i_+ is selected based on a small tree-edit distance from i, whereas the negative index i_- is selected based on a large tree-edit

Algorithm 1: GeCode: Morphogene-Driven Morphology–Control Co-Design

Input: Randomly initialized morphology-design and control policies π_ϕ^{des} and $\pi_\theta^{\mathrm{ctrl}}$; Morphogene set size N_G; Morphogene-space construction epochs E_{I}; Morphogene-driven co-design epochs E_{II}.

STAGE I: MORPHOGENE-SPACE CONSTRUCTION

Randomly generate and instantiate a collection of morphologies $\mathcal{D} = \{\mathcal{M}_j^{\mathrm{vec}}\}_{j=1}^{N_D}$
for $e = 1, \ldots, E_{\mathrm{I}}$ **do**
  Sample a morphology mini-batch $\mathcal{B} \subset \mathcal{D}$
  Encode $\mathcal{M}_j^{\mathrm{vec}} \in \mathcal{B}$ as $\mathbf{z}_j^{\mathrm{morph}} = E_\psi(\mathcal{M}_j^{\mathrm{vec}})$ using Eq. (4)
  Reconstruct $(\widehat{\mathcal{M}}_j^{\mathrm{vec}}, \widehat{\boldsymbol{\chi}}_j) = D_\psi(\mathbf{z}_j^{\mathrm{morph}})$ using Eq. (5)
  Update ψ with reconstruction loss in Eq. (6) and structural losses in Eqs. (18) and (19)
Retain and freeze E_ψ for co-design, and discard D_ψ
Independently sample $\mathcal{P}_\mathcal{G} = \{\mathcal{G}_i\}_{i=1}^{N_G}$ from the learned latent space
Initialize sampling probabilities $p_i \leftarrow \frac{1}{N_G}$

STAGE II: MORPHOGENE-DRIVEN CO-DESIGN

for $e = 1, \ldots, E_{\mathrm{II}}$ **do**
  for *each rollout* k **do**
    Sample a Morphogene $\mathcal{G}_i \in \mathcal{P}_\mathcal{G}$ according to $i \sim \mathbf{p}$
    Generate $\mathcal{M}_{i,k}$ using the Morphogene-conditioned design policy $\pi_\phi^{\mathrm{des}}(\cdot \mid \mathcal{G}_i)$ with AdaConcat, following Eqs. (8) and (9)
    Instantiate and encode the morphology as $\mathbf{z}_{i,k}^{\mathrm{morph}} = E_\psi(\mathcal{M}_{i,k}^{\mathrm{vec}})$ using Eq. (4)
    Compute the terminal proximity reward r_t^{prox} using Eq. (10)
    Execute the control policy $\pi_\theta^{\mathrm{ctrl}}(\cdot \mid \mathcal{G}_i)$ on $\mathcal{M}_{i,k}$, conditioned on the same Morphogene through AdaConcat according to Eqs. (8) and (9)
    Record the trajectory and task return $R_{i,k}$
  Optimize (ϕ, θ) using collected trajectories under Eq. (11)
  Construct local hit sets $\mathcal{K}_i = \{k \mid \mathrm{sim}(\mathbf{z}_{i,k}^{\mathrm{morph}}, \mathcal{G}_i) \geq \tau_{\mathrm{sim}}\}$ and their union $\mathcal{K} = \{(i,k) \mid k \in \mathcal{K}_i\}$
  if $\mathcal{K} \neq \varnothing$ **then**
    Select *global-best* code $\mathbf{z}^{\mathrm{glob}}$ by maximizing $R_{i,k}$ over $(i,k) \in \mathcal{K}$
    foreach *active* $\mathcal{G}_i$ *with* $\mathcal{K}_i \neq \varnothing$ **do**
      Select *local-best* code $\mathbf{z}_i^{\mathrm{loc}}$ by maximizing $R_{i,k}$ over $k \in \mathcal{K}_i$
      Update each Morphogene $\mathcal{G}_i \in \mathcal{P}_\mathcal{G}$ according to Eq. (12)
  Eliminate each Morphogene $\mathcal{G}_i \in \mathcal{P}_\mathcal{G}$ with $|\mathcal{K}_i| < H_{\min}$ (Eq. 13)
  Update the sampling distribution $\mathbf{p}$ over the retained Morphogenes using Eq. (14)

distance. The tree-structured metric loss enforces the corresponding reference code to be closer to the positive code than to the negative code by a margin:

$$\mathcal{L}_{\mathrm{tree}} = \frac{1}{|\mathcal{Q}|} \sum_{(i,i_+,i_-)\in\mathcal{Q}} \max\Big(0,\, d_{\cos}\Big(\widetilde{\mathbf{z}}_i^{\mathrm{morph}}, \widetilde{\mathbf{z}}_{i_+}^{\mathrm{morph}}\Big) - d_{\cos}\Big(\widetilde{\mathbf{z}}_i^{\mathrm{morph}}, \widetilde{\mathbf{z}}_{i_-}^{\mathrm{morph}}\Big) + \delta_{\mathrm{tree}}\Big). \quad (19)$$

Here, $d_{\cos}(\mathbf{u}, \mathbf{v}) = 1 - \mathbf{u}^\top \mathbf{v}$ denotes the cosine distance between normalized morphology codes, and δ_{tree} is the ranking margin.

The complete RAE training objective is

$$\mathcal{L}_{\mathrm{RAE}} = \mathcal{L}_{\mathrm{rec}} + \lambda_{\mathrm{topo}}\mathcal{L}_{\mathrm{topo}} + \lambda_{\mathrm{tree}}\mathcal{L}_{\mathrm{tree}}, \quad (20)$$

where λ_{topo} and λ_{tree} control the contributions of the two structural losses.

C.3 Offline and RL-Based Training of the Morphology Decoder

In this work, we learn the morphology latent space using an offline, reconstruction-based RAE decoder, as described in Sec. 4.1. As an alternative, the morphology-design policy can serve as an RL-trained decoder: rather than reconstructing morphology attributes directly, it maps a target morphology code to a morphology through sequential design actions. We describe this alternative below and discuss its trade-offs relative to offline reconstruction.

RL-based Policy Decoding The morphology-design policy π_ϕ^{des} can be trained to decode a target morphology code $\mathbf{z}^{\mathrm{morph}}$ into a valid design. Starting from an initial morphology $\mathcal{M}_0$, the policy generates topology and attribute design actions conditioned on the current morphology and target code, $a_t^{\mathrm{des}} \sim \pi_\phi^{\mathrm{des}}(\cdot \mid \mathcal{M}_t; \mathbf{z}^{\mathrm{morph}})$, yielding the final morphology $\mathcal{M}^\star = \mathcal{M}_{T_{\mathrm{des}}}$. We train the policy using the proximity reward in Eq. (10), with the target Morphogene $\mathcal{G}$ set to $\mathbf{z}^{\mathrm{morph}}$. This reward encourages the code of the generated morphology to meet the prescribed similarity threshold with respect to the target code.

RL-based decoding requires simulator rollouts and policy optimization, making pretraining more computationally demanding than direct supervised reconstruction. Moreover, the proximity reward alone does not explicitly impose the pairwise topological and tree-structural relationships encouraged by the regularizers in Eqs. (18) and (19); consequently, it provides less direct control over the organization of the latent space. Its principal advantage is that pretraining also optimizes the morphology-design policy. The pretrained policy can then be carried into co-design, where it continues to generate morphologies while being optimized jointly with the control policy. Accordingly, RL-based decoding incurs higher pretraining cost than offline reconstruction and, without additional structural regularizers, provides less direct control over latent-space organization; in return, it yields a morphology-design policy pretrained for Morphogene-conditioned generation that can be further optimized during co-design.

D Implementation Details

D.1 Training Details

Following standard reinforcement learning practice, we employ distributed trajectory collection across multiple CPU threads to improve training efficiency. All experiments are conducted with five random seeds on a machine equipped with two NVIDIA GeForce RTX 4090 GPUs. The neural networks are implemented in PyTorch 2.0.1, and the morphology–control environments are simulated using MuJoCo 2.1.0 (Todorov et al., 2012).

D.2 Hyperparameters

GeCode We implement GeCode in PyTorch on top of BodyGen, retaining its environment, morphology-design space, and policy-optimization settings. To construct the Morphogene space, we train the RAE for 100 epochs on randomly generated valid morphologies using Adam with a learning rate of 5e-4 and a batch size of 2,048. During Morphogene-driven co-design, the morphology encoder remains frozen, while Morphogenes are randomly initialized in the learned latent space and updated after each epoch using existing rollout statistics, without additional environment interactions. Detailed hyperparameters are provided in Table 2.

StackelbergPPO StackelbergPPO extends BodyGen with a Stackelberg optimization framework (Dai et al., 2026). We follow the official StackelbergPPO implementation: a Fisher regularization coefficient of $\lambda = 5.0$, up to 20 conjugate-gradient iterations with a relative-error tolerance of 10^{-3}, six follower-sampling steps per episode, and a gradient-normalization ratio of $\alpha = 1.0$.

BodyGen We follow the official BodyGen implementation (Lu et al., 2025), using Pre-LN MoSAT blocks with SiLU activations, a hidden dimension of 64, and learning rates of 5e-5 and 3e-4 for the policy and value function, respectively. Additional hyperparameters are provided in Table 2.

Table 2: Hyperparameters of GeCode adopted in all experiments.

Hyperparameter	Value
MORPHOGENE-SPACE CONSTRUCTION	
Space construction epochs	100
RAE batch size	2,048
RAE Optimizer	Adam
RAE learning rate	5e-4
Limb-token dimension	128
Morphogene dimension d_g	32
Attribute reconstruction loss weight	1.0
Limb-presence mask loss weight λ_{mask}	1.0
Morphology-code regularization weight λ_{reg}	0.01
Topology-aware contrastive loss weight λ_{topo}	8e-3
Contrastive temperature τ_{topo}	0.1
Tree-structured metric loss weight λ_{tree}	0.01
Tree-structured metric margin δ_{tree}	0.2
Maximum tree triplets per batch	64
MORPHOGENE-DRIVEN CO-DESIGN OPTIMIZATION	
Initial Morphogene count N_G	5
Similarity threshold τ_{sim}	0.4
Proximity reward weight λ_{hit}	1,000
Morphogene update rate η	0.1
Minimum local hits for retention $H_{\min}$	5
Top-N_{top} returns for adaptive sampling (N_{top})	5
Policy Transformer depth	3
Policy Hidden dimension	64
Value Transformer depth	3
Value Hidden dimension	64
Transformer Layer Normalization	Pre-LN
Transformer Activation Function	SiLU
PPO optimizer	Adam
Policy learning rate	5e-5
Value learning rate	3e-4
Clip gradient norm	40.0
PPO clip ϵ	0.2
PPO batch size	50,000
PPO minibatch size	2,048
PPO Iterations Per Batch	10
Training Epochs	1000
Discount factor γ	0.995
GAE parameter λ_{GAE}	0.95

Transform2Act We follow the official Transform2Act implementation (Yuan et al., 2022), using three-layer GraphConv policy and value networks with hidden dimensions (64, 64, 64) and learning rates of 5e-5 and 3e-4, respectively. The policy employs a three-layer, 128-unit JSMLP with Tanh activations, while the value function uses a (512, 256) MLP.

E ADDITIONAL RESULTS

E.1 QUANTITATIVE RESULTS

Quantitative Performance Figure 4 presents the learning curves of GeCode and the RL-based baselines, including Transform2Act, BodyGen, and StackelbergPPO, while Table 3 reports the final episode returns as the mean and standard error over five random seeds. GeCode achieves the highest final return in all twelve environments. Relative to the strongest baseline in each task, its improvement ranges from 6.5% on SWIMMER to 104.6% on PUSHER.

The substantial gains on CRAWLER, GLIDER-HARD, and WALKER-HARD are particularly notable given their large, highly branched morphology spaces. Instead of directly exploring numerous discrete structures, GeCode performs local refinement around individual Morphogene anchors and updates these anchors toward promising regions of the latent space. This local-to-global strategy focuses rollouts on structurally coherent neighborhoods while sharing high-performing design information across anchors, thereby improving exploration efficiency as the morphology space expands.

The gains on PUSHER, STEPPER, and TERRAINCROSSER further highlight the importance of explicit body–brain coordination. By conditioning both morphology generation and control on the same Morphogene, GeCode provides the controller with a compact representation of the structural factors underlying the generated body. This shared conditioning enables the controller to adapt its actions to morphology-dependent object contacts, terrain interactions, and stability requirements, while improving credit assignment between body design and control behavior.

In contrast, the smaller gain on SWIMMER suggests that explicit body–brain coordination provides a more limited advantage when relatively simple morphologies are already sufficient for effective propulsion. Overall, these results show that GeCode derives its advantage from both structured latent-space exploration and explicit morphology–control coordination, with the benefits becoming more pronounced as structural search and control adaptation grow more tightly coupled.

Table 3: **Performance comparison across morphology–control co-design environments.** Returns from the final training episode are reported as means ± standard errors over five random seeds for GeCode, reinforcement-learning baselines, and evolutionary baselines.

Task	**NGE** (Wang et al., 2019)	**Transform2Act** (Yuan et al., 2022)	**BodyGen** (Lu et al., 2025)	**StackelbergPPO** (Dai et al., 2026)	**GeCode (Ours)**
CRAWLER	1482.5±525.0	5396.3±177.3	8208.6±374.3	9255.3±888.2	**16143.3±583.0**
STEPPER	870.6±215.5	1671.0±148.3	2087.2±243.3	2538.9±173.6	**3688.6±271.0**
PUSHER	551.6±120.7	571.8±78.6	1809.1±162.2	3054.4±354.9	**6250.2±757.9**
CHEETAH	2534.8±428.7	9542.1±635.0	11475.1±489.5	13246.3±299.9	**19888.7±944.6**
TERRAINCROSSER	827.2±427.2	2426.5±654.1	4016.4±911.6	5437.5±675.6	**9670.6±1176.4**
SWIMMER	384.5±112.0	782.4±23.6	1301.6±1.3	1335.9±17.2	**1422.5±29.5**
GLIDER-REGULAR	1567.8±756.7	6890.4±414.8	11091.0±211.8	11429.9±261.1	**23317.5±627.9**
GLIDER-MEDIUM	1649.6±763.6	8632.0±1185.1	11016.8±144.2	12652.3±222.3	**22376.4±809.9**
GLIDER-HARD	2081.3±348.2	6218.5±685.9	10973.7±268.3	12110.0±330.6	**19972.8±332.1**
WALKER-REGULAR	1402.9±595.5	7960.6±534.2	11555.3±220.1	13216.4±536.5	**24306.1±1293.0**
WALKER-MEDIUM	2600.4±481.7	7324.4±906.8	13244.4±342.1	11764.6±418.6	**21973.3±918.9**
WALKER-HARD	1504.6±553.2	4924.0±690.3	12126.0±825.2	13643.4±597.6	**21614.6±521.8**

Comparison with Evolutionary Search Table 3 further compares GeCode with NGE (Wang et al., 2019), an evolutionary method that optimizes populations of embodied designs through selection and variation. GeCode outperforms NGE across all twelve tasks, achieving approximately $3.7\times$ to $17.3\times$ the returns of NGE. Evolutionary co-design must distribute a finite interaction budget across numerous morphology candidates and evaluate the controller associated with each design. As the morphology space becomes larger and more highly branched, this population-level evaluation becomes increasingly expensive, while limited knowledge transfer across structurally distinct candidates hinders the joint identification of effective morphologies and compatible controllers.

In contrast, GeCode organizes co-design around a small set of searchable Morphogene anchors in a compact, structurally regularized latent space. Stochastic policy rollouts refine body–brain designs locally around each anchor, while performance-guided updates move the anchors toward promising regions for global exploration. Because all Morphogenes condition the same morphology-design and control policies, knowledge learned from different designs is shared through the policy parameters, while Morphogene updates reuse existing rollout trajectories without additional evaluations. GeCode therefore replaces costly population-level search with structured latent exploration that directly couples morphology refinement and controller adaptation.

E.2 ADDITIONAL ABLATION AND ANALYSIS

Analysis of RAE Training Table 4 reports the task-specific RAE training losses. The consistently small standard errors of the overall objective $\mathcal{L}_{\mathrm{RAE}}$ indicate stable optimization across all twelve

Table 4: **RAE training statistics across twelve co-design tasks.** Values are reported as means ± standard errors over five random seeds.

Task	$\mathcal{L}_{\text{RAE}}$	$\mathcal{L}_{\text{rec}}$			$\mathcal{L}_{\text{topo}}$	$\mathcal{L}_{\text{tree}}$
		$\mathcal{L}_{\text{attr}}$	$\mathcal{L}_{\text{mask}}$	$\mathcal{L}_{\text{reg}}$		
CRAWLER	0.3384 ± 0.0028	0.2928 ± 0.0025	0.0010 ± 0.0002	0.2558 ± 0.0031	5.2215 ± 0.0627	0.0267 ± 0.0031
STEPPER	0.3336 ± 0.0058	0.2889 ± 0.0060	0.0008 ± 0.0001	0.2509 ± 0.0085	5.1376 ± 0.0280	0.0267 ± 0.0048
PUSHER	0.3365 ± 0.0052	0.2911 ± 0.0053	0.0011 ± 0.0002	0.2585 ± 0.0098	5.1803 ± 0.0310	0.0237 ± 0.0031
CHEETAH	0.0908 ± 0.0025	0.0496 ± 0.0024	0.0001 ± 0.0000	0.2104 ± 0.0030	4.7858 ± 0.0152	0.0686 ± 0.0017
TERRAINCROSSER	0.0919 ± 0.0038	0.0496 ± 0.0033	0.0002 ± 0.0001	0.2410 ± 0.0172	4.8627 ± 0.0554	0.0720 ± 0.0041
SWIMMER	0.1259 ± 0.0053	0.0797 ± 0.0052	0.0003 ± 0.0000	0.2982 ± 0.0054	5.3088 ± 0.0431	0.0443 ± 0.0021
GLIDER-REGULAR	0.0736 ± 0.0017	0.0189 ± 0.0018	0.0000 ± 0.0000	0.1544 ± 0.0011	6.6390 ± 0.0056	0.0000 ± 0.0000
GLIDER-MEDIUM	0.0873 ± 0.0138	0.0361 ± 0.0128	0.0001 ± 0.0000	0.2098 ± 0.0174	6.1070 ± 0.0800	0.0229 ± 0.0139
GLIDER-HARD	0.0870 ± 0.0079	0.0387 ± 0.0072	0.0001 ± 0.0000	0.2459 ± 0.0252	5.6611 ± 0.0474	0.0357 ± 0.0030
WALKER-REGULAR	0.0783 ± 0.0029	0.0285 ± 0.0029	0.0001 ± 0.0000	0.1782 ± 0.0038	5.9831 ± 0.0055	0.0087 ± 0.0011
WALKER-MEDIUM	0.1166 ± 0.0030	0.0741 ± 0.0026	0.0003 ± 0.0000	0.2308 ± 0.0111	4.8997 ± 0.0504	0.0750 ± 0.0041
WALKER-HARD	0.1793 ± 0.0048	0.1380 ± 0.0045	0.0007 ± 0.0003	0.2334 ± 0.0053	4.7300 ± 0.0191	0.0423 ± 0.0035

Table 5: **Quantifying limb-specific Morphogene conditioning.** Limb Diversity, Relative Modulation, and Between-Limb Energy Ratio measure differences among limb-level conditions under AdaConcat and direct concatenation across twelve tasks. Values are reported as means ± standard errors over five random seeds.

	Limb Diversity		**Relative Modulation**		**Between-Limb Energy Ratio**	
Task	**AdaConcat**	Direct Concat	**AdaConcat**	Direct Concat	**AdaConcat**	Direct Concat
CRAWLER	$\mathbf{0.658 \pm 0.053}$	0.000	$\mathbf{0.482 \pm 0.041}$	0.000	$\mathbf{0.181 \pm 0.020}$	0.000
STEPPER	$\mathbf{0.746 \pm 0.126}$	0.000	$\mathbf{0.542 \pm 0.096}$	0.000	$\mathbf{0.203 \pm 0.049}$	0.000
PUSHER	$\mathbf{0.643 \pm 0.047}$	0.000	$\mathbf{0.480 \pm 0.036}$	0.000	$\mathbf{0.165 \pm 0.017}$	0.000
CHEETAH	$\mathbf{1.257 \pm 0.081}$	0.000	$\mathbf{0.841 \pm 0.050}$	0.000	$\mathbf{0.346 \pm 0.034}$	0.000
TERRAINCROSSER	$\mathbf{1.328 \pm 0.119}$	0.000	$\mathbf{0.894 \pm 0.076}$	0.000	$\mathbf{0.365 \pm 0.025}$	0.000
SWIMMER	$\mathbf{1.169 \pm 0.165}$	0.000	$\mathbf{0.772 \pm 0.110}$	0.000	$\mathbf{0.334 \pm 0.035}$	0.000
GLIDER-REGULAR	$\mathbf{2.288 \pm 0.537}$	0.000	$\mathbf{1.522 \pm 0.309}$	0.000	$\mathbf{0.410 \pm 0.068}$	0.000
GLIDER-MEDIUM	$\mathbf{1.328 \pm 0.191}$	0.000	$\mathbf{0.876 \pm 0.106}$	0.000	$\mathbf{0.350 \pm 0.061}$	0.000
GLIDER-HARD	$\mathbf{1.319 \pm 0.362}$	0.000	$\mathbf{0.862 \pm 0.229}$	0.000	$\mathbf{0.336 \pm 0.071}$	0.000
WALKER-REGULAR	$\mathbf{1.453 \pm 0.140}$	0.000	$\mathbf{0.968 \pm 0.070}$	0.000	$\mathbf{0.384 \pm 0.037}$	0.000
WALKER-MEDIUM	$\mathbf{1.031 \pm 0.136}$	0.000	$\mathbf{0.714 \pm 0.112}$	0.000	$\mathbf{0.290 \pm 0.039}$	0.000
WALKER-HARD	$\mathbf{1.275 \pm 0.090}$	0.000	$\mathbf{0.865 \pm 0.076}$	0.000	$\mathbf{0.369 \pm 0.016}$	0.000

environments. The near-zero limb-presence loss $\mathcal{L}_{\text{mask}}$ shows that the decoder reliably reconstructs variable limb configurations, which is essential for representing morphologies with different topologies. The attribute-reconstruction loss $\mathcal{L}_{\text{attr}}$ is generally higher for the three-dimensional tasks, including CRAWLER, STEPPER, and PUSHER, reflecting the greater complexity of their geometric representations. Across the GLIDER and WALKER task groups, $\mathcal{L}_{\text{attr}}$ generally increases as the branching constraints are relaxed, consistent with the greater morphological diversity admitted by the more challenging variants.

The topology-aware contrastive loss $\mathcal{L}_{\text{topo}}$ and tree-metric loss $\mathcal{L}_{\text{tree}}$ impose complementary structure on the latent space. Specifically, $\mathcal{L}_{\text{topo}}$ organizes morphology codes into coherent local neighborhoods according to topology, preventing structurally distinct morphologies from becoming arbitrarily entangled. In contrast, $\mathcal{L}_{\text{tree}}$ preserves graded structural relationships by ordering latent distances according to tree-edit similarity, such that nearby codes correspond to progressively related morphology structures. The consistently small standard errors indicate stable optimization of both objectives across environments. Moreover, the near-zero $\mathcal{L}_{\text{tree}}$ on weakly branching tasks such as GLIDER-REGULAR and its larger values in more structurally diverse tasks reflect the increasing difficulty of preserving global structural order as branching complexity grows. Together, these losses balance topology-aware local clustering with globally meaningful distance organization, enabling Morphogene updates to traverse the latent space through coherent structural variations.

Analysis of Limb-Specific Conditioning in AdaConcat Figure 10 shows that AdaConcat produces distinct limb-level conditions from the same Morphogene while preserving feature patterns shared across the body. By contrast, direct concatenation provides every limb with an identical Morphogene, requiring the policies to infer limb-specific distinctions from other inputs. Complementing the performance comparison in Table 1, the learning curves in Figure 11 show that GeCode generally

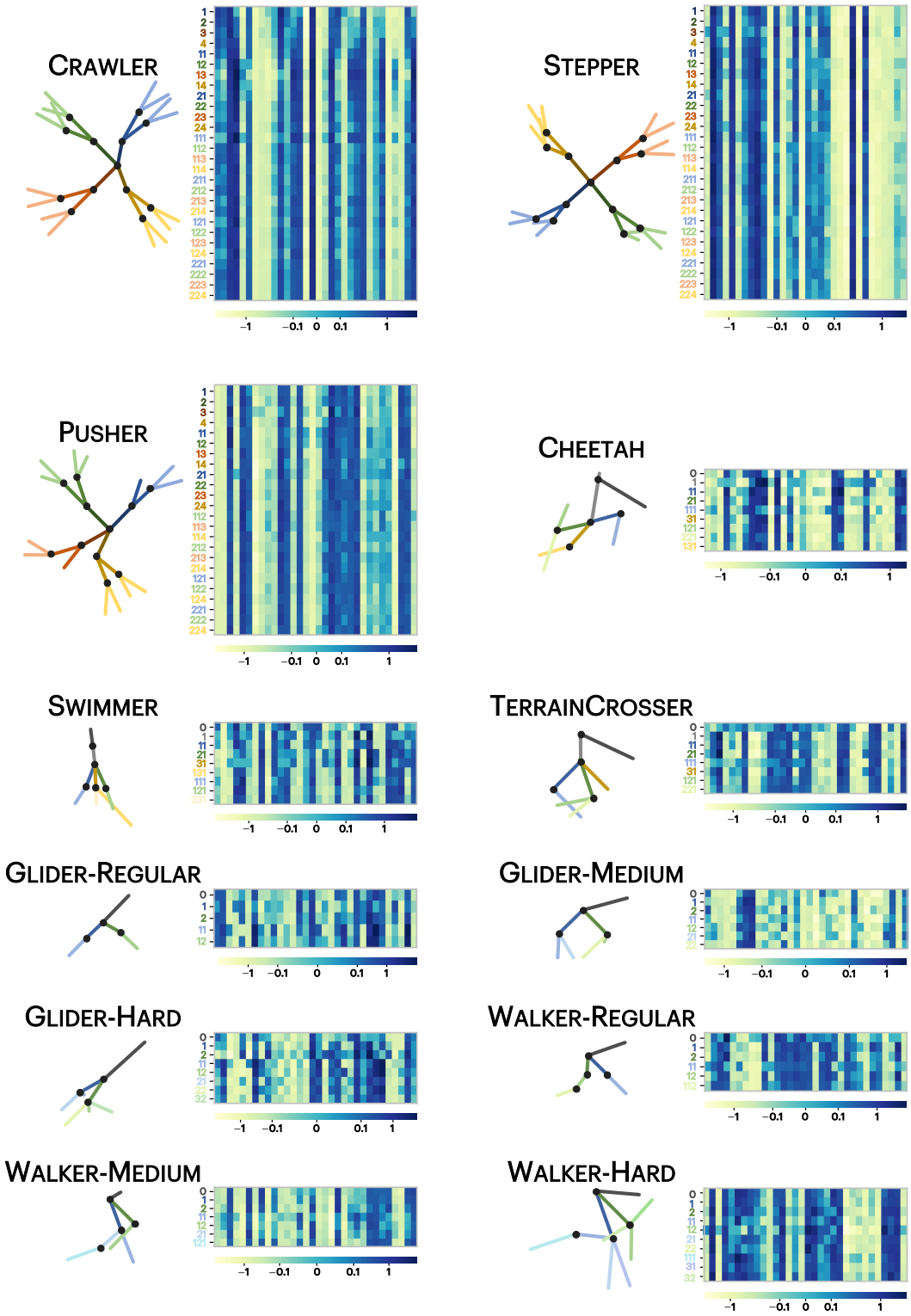


Figure 10: **Limb-specific Morphogene conditioning with AdaConcat.** Representative morphologies are paired with heatmaps of their limb-level conditions. Distinct patterns across rows show how AdaConcat adapts a shared Morphogene to individual limbs.

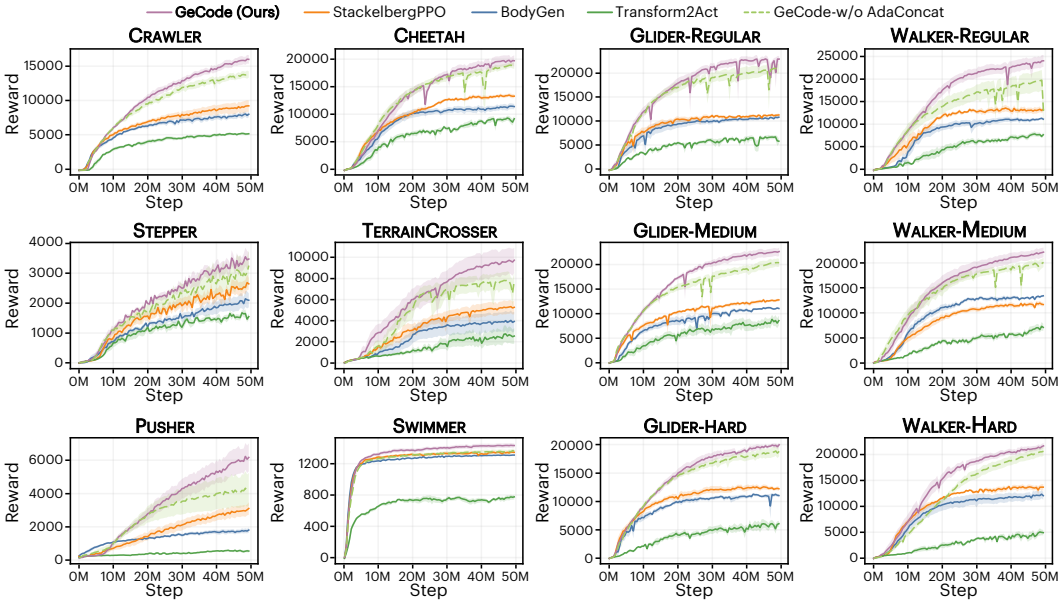


Figure 11: **Ablation of AdaConcat in morphology–control co-design.** Learning curves compare GeCode, its direct-concatenation variant (i.e., GeCode w/o AdaConcat), and reference baselines across twelve tasks. Shaded regions indicate the standard error over five random seeds.

outperforms this direct-concatenation variant, with pronounced gains on PUSHER and WALKER-REGULAR. These results suggest that adapting the shared Morphogene to each limb helps the policies account for differing roles, such as object contact and locomotor support, during morphology design and control.

To quantify the limb specificity of AdaConcat, we use three complementary measures. Let $\mathcal{G}_{l_i}$ denote the limb-specific condition produced by AdaConcat from $\mathcal{G}$ for limb l_i. Limb Diversity is the mean pairwise distance between limb conditions, normalized by the magnitude of $\mathcal{G}$:

$$D_{\text{limb}} = \frac{2}{N_L(N_L-1)} \sum_{i<j} \frac{\|\mathcal{G}_{l_i} - \mathcal{G}_{l_j}\|_2}{\|\mathcal{G}\|_2}.$$

Relative Modulation measures the average deviation of the limb-specific conditions from the shared Morphogene, normalized by its magnitude:

$$D_{\text{mod}} = \frac{1}{N_L} \sum_{i=1}^{N_L} \frac{\|\mathcal{G}_{l_i} - \mathcal{G}\|_2}{\|\mathcal{G}\|_2}.$$

Between-Limb Energy Ratio measures the proportion of total condition energy associated with variation across limbs:

$$R_{\text{between}} = \frac{\sum_{i=1}^{N_L} \|\mathcal{G}_{l_i} - \boldsymbol{\mu}_{\text{limb}}\|_2^2}{\sum_{i=1}^{N_L} \|\mathcal{G}_{l_i}\|_2^2}, \qquad \boldsymbol{\mu}_{\text{limb}} = \frac{1}{N_L} \sum_{i=1}^{N_L} \mathcal{G}_{l_i}.$$

Table 5 shows that AdaConcat consistently produces distinct limb-specific conditions from a shared Morphogene. Nonzero Relative Modulation indicates that these conditions differ from the shared Morphogene, while nonzero Limb Diversity and Between-Limb Energy Ratio show that the transformation varies across limbs. Between-limb variation accounts for 16.5%–41.0% of total condition energy, indicating substantial limb specificity while retaining a shared component. In contrast, all three measures are zero under direct concatenation, which provides every limb with the same unmodified Morphogene.

E.3 SAMPLE AND TRAINING EFFICIENCY

Sample Efficiency We measure sample efficiency by the number of environment interactions required to reach task-specific thresholds, defined by the highest returns attained during training

Table 6: **Environment interactions required to reach baseline performance thresholds.** For each task, the two thresholds are the highest returns attained during training by BodyGen (Lu et al., 2025) and StackelbergPPO (Dai et al., 2026), respectively. Entries report the required interactions in millions (M); ∞ indicates that a method did not reach the threshold within the 50M-interaction budget. The lowest interaction count in each row is shown in **bold**.

Task	Threshold (BodyGen)	GeCode	StackelbergPPO	BodyGen	Transform2Act	NGE	ESS
CRAWLER	8208.57	**14.35 M**	31.10 M	49.45 M	∞	∞	∞
STEPPER	2087.16	**20.90 M**	31.20 M	45.80 M	∞	∞	∞
PUSHER	1809.06	**14.20 M**	24.20 M	48.20 M	∞	∞	∞
CHEETAH	11475.06	**15.05 M**	25.15 M	45.40 M	∞	∞	∞
TERRAINCROSSER	4016.37	**14.05 M**	24.75 M	50.00 M	∞	∞	∞
SWIMMER	1301.57	**8.05 M**	14.70 M	36.75 M	∞	∞	∞
GLIDER-REGULAR	11091.03	**8.65 M**	28.00 M	49.85 M	∞	∞	∞
GLIDER-MEDIUM	11016.82	**10.55 M**	26.10 M	44.75 M	∞	∞	∞
GLIDER-HARD	10973.74	**12.65 M**	19.70 M	40.30 M	∞	∞	∞
WALKER-REGULAR	11555.34	**12.90 M**	18.80 M	50.00 M	∞	∞	∞
WALKER-MEDIUM	13244.43	**15.35 M**	∞	46.25 M	∞	∞	∞
WALKER-HARD	12125.98	**15.45 M**	22.25 M	46.55 M	∞	∞	∞
Task	**Threshold (StackelbergPPO)**	**GeCode**	**StackelbergPPO**	**BodyGen**	**Transform2Act**	**NGE**	**ESS**
CRAWLER	9255.29	**16.30 M**	46.30 M	∞	∞	∞	∞
STEPPER	2538.91	**26.65 M**	42.75 M	∞	∞	∞	∞
PUSHER	3054.39	**21.40 M**	48.85 M	∞	∞	∞	∞
CHEETAH	13246.30	**17.55 M**	37.25 M	∞	∞	∞	∞
TERRAINCROSSER	5437.52	**18.35 M**	50.00 M	∞	∞	∞	∞
SWIMMER	1335.90	**11.20 M**	37.05 M	∞	∞	∞	∞
GLIDER-REGULAR	11429.94	**9.20 M**	30.35 M	∞	∞	∞	∞
GLIDER-MEDIUM	12652.30	**12.80 M**	44.35 M	∞	∞	∞	∞
GLIDER-HARD	12110.01	**14.75 M**	29.60 M	∞	∞	∞	∞
WALKER-REGULAR	13216.44	**14.70 M**	27.85 M	∞	∞	∞	∞
WALKER-MEDIUM	11764.57	**13.65 M**	43.05 M	24.05 M	∞	∞	∞
WALKER-HARD	13643.36	**17.10 M**	39.30 M	∞	∞	∞	∞

by BodyGen and StackelbergPPO. As shown in Table 6, GeCode reaches both thresholds with the fewest interactions in all twelve tasks. Summed across tasks, GeCode requires 48.06% and 57.69% fewer interactions than the fastest task-specific baseline to reach the BodyGen and StackelbergPPO thresholds, respectively. For example, it reaches the BodyGen threshold on GLIDER-REGULAR in 8.65M interactions, compared with 28.00M for the fastest baseline. GeCode also reaches every StackelbergPPO threshold within the 50M-interaction budget, whereas BodyGen reaches only one.

This efficiency arises from the complementary roles of local rollouts and global Morphogene updates. Rollouts explore related body designs around each Morphogene, providing more consistent experience for the conditioned controller. Performance-guided updates and adaptive sampling subsequently direct interactions toward promising regions of the Morphogene space. Because these operations reuse policy-training trajectories, they improve the focus of exploration without additional environment interactions.

Training Efficiency GeCode maintains competitive training efficiency, with wall-clock costs comparable to—and often lower than—those of StackelbergPPO across the evaluated tasks. As shown in Table 7, offline RAE training requires only between 0.77 and 1.03 hours of wall-clock time, accounting for less than 4% of GeCode's overall training time and thus introducing minimal computational overhead. During co-design, Morphogene updates reuse trajectories collected for policy optimization and involve only morphology encoding and lightweight latent-space operations, requiring neither additional environment interactions nor nested policy-optimization procedures.

Table 7 shows that GeCode incurs moderately higher wall-clock costs on three-dimensional tasks such as CRAWLER, while demonstrating a clear efficiency advantage on two-dimensional planar tasks, particularly GLIDER and WALKER. The additional cost on three-dimensional tasks arises primarily from the greater complexity of the morphologies discovered by GeCode, rather than from the Morphogene update procedure itself. By exploring broader regions of the design space, GeCode identifies higher-performing and structurally richer agents with more limbs than StackelbergPPO.

Table 7: **Training-time computational cost across morphology design spaces.** Wall-clock time is reported for offline RAE training, morphology–control co-design, and their combined cost in GeCode, together with the training time of StackelbergPPO. The maximum number of limbs indicates the capacity of each task-specific design space. Values are reported as means ± standard errors over five random seeds.

Task	Space Size (max)	GeCode			StackelbergPPO
		Offline RAE Training	Co-design Optimization	Total Time	
CRAWLER	29	0.997 ± 0.017	30.755 ± 3.390	31.752 ± 3.389	**27.924 ± 1.617**
STEPPER	29	1.027 ± 0.020	29.560 ± 3.583	**30.587 ± 3.566**	30.969 ± 1.346
PUSHER	29	1.028 ± 0.022	22.754 ± 4.045	**23.782 ± 4.031**	25.447 ± 1.811
CHEETAH	14	0.767 ± 0.003	21.384 ± 0.676	22.151 ± 0.679	**19.959 ± 0.205**
TERRAINCROSSER	14	0.883 ± 0.023	24.279 ± 1.040	25.161 ± 1.033	**24.020 ± 1.441**
SWIMMER	14	0.839 ± 0.017	17.948 ± 1.214	**18.787 ± 1.220**	20.913 ± 0.748
GLIDER-REGULAR	5	0.877 ± 0.024	19.320 ± 0.809	20.197 ± 0.789	**19.319 ± 0.379**
GLIDER-MEDIUM	7	0.861 ± 0.025	19.316 ± 0.212	**20.178 ± 0.211**	20.494 ± 0.951
GLIDER-HARD	9	0.839 ± 0.023	19.728 ± 0.805	**20.567 ± 0.787**	21.072 ± 0.939
WALKER-REGULAR	7	0.878 ± 0.020	18.338 ± 0.692	**19.216 ± 0.685**	22.691 ± 0.417
WALKER-MEDIUM	15	0.823 ± 0.023	18.960 ± 0.424	**19.783 ± 0.405**	24.666 ± 0.899
WALKER-HARD	27	1.014 ± 0.016	22.248 ± 2.395	**23.263 ± 2.388**	23.574 ± 0.476
AVERAGE				**22.952 ± 0.760**	23.421 ± 0.514

Table 8: **Evaluation performance and inference-time computational cost of GeCode.** Evaluation returns, parameter counts, inference latency, total FLOPs, and per-limb FLOPs are reported across twelve tasks. For each random seed, the evaluation return is averaged over five episodes, while latency and FLOPs are measured per control decision. Results are reported as means ± standard errors over five random seeds.

Task	Eval. Return	Params. (M)	Latency (ms) ↓	FLOPs (M) ↓	FLOPs (Per Limb) (M) ↓
CRAWLER	15899.313 ± 619.609	0.5522	16.734 ± 0.759	9.354	0.323
STEPPER	3336.782 ± 335.130	0.5522	31.431 ± 8.901	7.617	0.319
PUSHER	6019.497 ± 688.621	0.5524	13.951 ± 0.970	8.682	0.321
CHEETAH	20130.416 ± 1012.861	0.5517	1.511 ± 0.766	2.195	0.305
TERRAINCROSSER	9760.178 ± 1195.632	0.5518	12.976 ± 11.144	2.385	0.305
SWIMMER	1438.432 ± 25.512	0.5517	1.743 ± 1.113	2.319	0.305
GLIDER-REGULAR	23770.548 ± 668.018	0.5515	0.635 ± 0.049	1.514	0.303
GLIDER-MEDIUM	22664.985 ± 821.074	0.5515	0.607 ± 0.032	1.945	0.304
GLIDER-HARD	20096.701 ± 302.635	0.5515	0.616 ± 0.002	2.441	0.305
WALKER-REGULAR	24500.695 ± 1315.028	0.5517	0.614 ± 0.039	2.070	0.304
WALKER-MEDIUM	22144.726 ± 950.681	0.5517	14.692 ± 12.166	2.824	0.306
WALKER-HARD	21732.894 ± 543.298	0.5517	45.465 ± 12.229	3.072	0.307

These larger morphologies increase simulation cost and produce longer limb-token sequences for Transformer processing, thereby extending policy-optimization time. **The increased wall-clock cost therefore reflects GeCode's ability to explore more complex morphology regions, rather than substantial overhead from its latent-space search mechanism.**

On two-dimensional planar tasks such as WALKER, where the generated morphologies are comparable in scale to those of StackelbergPPO, GeCode incurs lower computational cost. This advantage reflects the computational efficiency of Morphogene-space optimization. Unlike StackelbergPPO, which incorporates follower updates and second-order optimization, GeCode performs lightweight, performance-guided latent updates using existing rollout statistics. Its optimization overhead therefore remains small, with the overall computational cost determined primarily by morphology complexity rather than the Morphogene update procedure.

Inference Efficiency Table 8 demonstrates that GeCode remains computationally lightweight during evaluation. Its parameter count is nearly constant across tasks at approximately 0.552M, reflecting the use of shared morphology-design and control networks rather than morphology-specific models. Total computation ranges from 1.514M to 9.354M FLOPs per control decision, primarily reflecting differences in the number of limbs processed. Notably, the per-limb cost remains within a narrow range of 0.303M–0.323M FLOPs, indicating that **inference scales predictably with morphology size**. GeCode therefore supports structurally diverse agents without introducing design-specific networks or substantial inference overhead.

E.4 VISUALIZATION

Morphologies Discovered by GeCode Figure 12 presents representative morphologies discovered by GeCode across the evaluated environments. The generated agents exhibit substantial diversity in topology, limb arrangement, and body proportions, ranging from compact locomotor structures to highly branched designs for complex-terrain traversal and object manipulation. Rather than remaining confined to minor variations of the initial morphology, Morphogene-driven exploration enables GeCode to reach structurally distinct yet task-appropriate regions of the design space. The diversity and validity of these agents demonstrate that GeCode provides broad morphology-space coverage while preserving the realizability and functional relevance of the discovered designs.

Visualization of Structural Organization in the Morphogene Space Figure 13 examines whether distances between morphology codes reflect meaningful structural relationships. The pairwise distance matrices exhibit clear block and graded patterns aligned with morphology complexity, topology, and limb configuration. Morphologies with similar limb counts and branching structures form compact latent neighborhoods, whereas structurally distinct designs are separated by larger distances. This organization is particularly evident in the GLIDER variants, where morphologies with the same limb count form low-distance blocks. In broader design spaces, such as CRAWLER, STEPPER, and WALKER-HARD, latent distances generally increase as structural complexity and branching patterns diverge. Variations within the same limb-count group further show that the representation captures finer geometric and connectivity differences beyond body count. These results demonstrate that the learned latent space places structurally similar morphologies close together while separating increasingly different designs. This organization supports Morphogene-driven co-design by enabling coherent local exploration around each Morphogene and structured global exploration through performance-guided updates.

Visualization of Morphogene Update Trajectories Figure 14 shows that valid morphology codes form distinct clusters associated with structural complexity, indicating that the learned latent spaces preserve meaningful morphological relationships. During co-design, Morphogenes follow task-dependent trajectories through this structured support rather than moving as unconstrained latent perturbations. In broad design spaces, such as CRAWLER, CHEETAH, and WALKER-HARD, different Morphogenes traverse complementary regions and reach morphology families with varying structural complexity. Progressive elimination of underperforming Morphogenes then concentrates the rollout budget on promising regions while retaining diversity among the surviving trajectories.

These trajectories reveal how GeCode integrates local and global exploration. Stochastic rollouts refine morphology variations around each Morphogene, whereas performance-guided updates move the Morphogenes across broader regions of the latent space. Consequently, a small Morphogene set can efficiently cover diverse yet structurally coherent morphology families without exhaustively evaluating independent designs.

F DISCUSSION

F.1 COMPARISON WITH UNIMAL

Morphology Design Spaces UNIMAL (Gupta et al., 2021) provides a well-structured design space for articulated agents, with constrained limb addition, deletion, and continuous attribute mutations that support diverse yet physically valid morphologies. Its explicit mutation operators offer a systematic basis for studying how body structure affects locomotion performance. The BodyGen-based benchmark adopted in this work supports comparable design operations but evaluates co-design across twelve task-specific environments spanning 2D and 3D locomotion, uneven-terrain traversal, and object manipulation. Each environment specifies its own initial morphology, task objective, and structural constraints. Together, these settings admit a wider variety of viable designs—from compact planar agents to larger, highly branched bodies—and impose different demands on the controllers that operate them.

The BodyGen-based benchmark permits larger and more diverse morphologies than UNIMAL. Although both benchmarks initialize CRAWLER with an ant-like structure, the design space used here allows up to 29 limbs, compared with UNIMAL's limit of 10. Within the GLIDER and WALKER

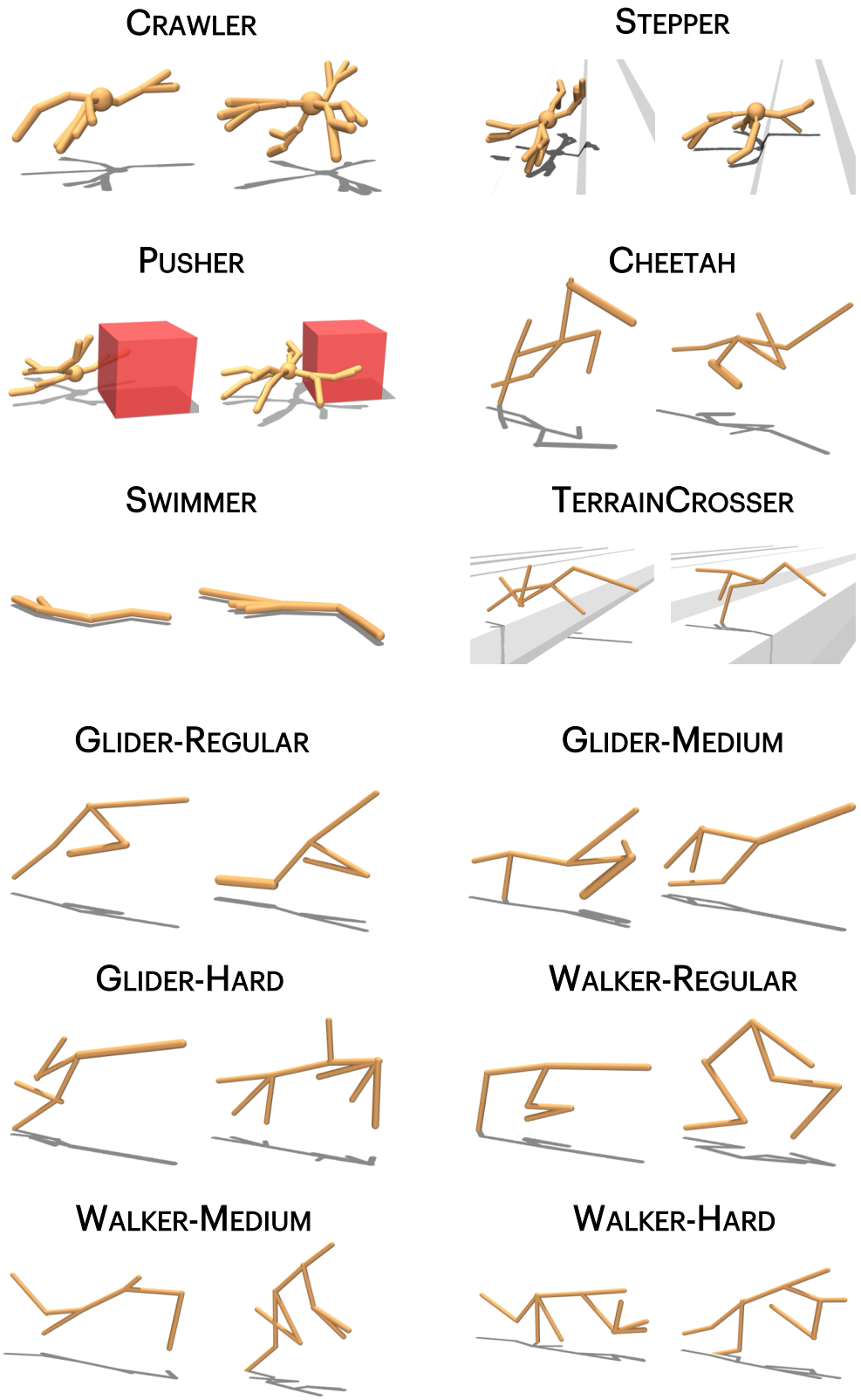


Figure 12: **Representative morphologies discovered by GeCode across the co-design environments.** The generated agents exhibit task-specific variations in topology, limb configuration, and body proportions, demonstrating the structural diversity enabled by Morphogene-driven co-design.

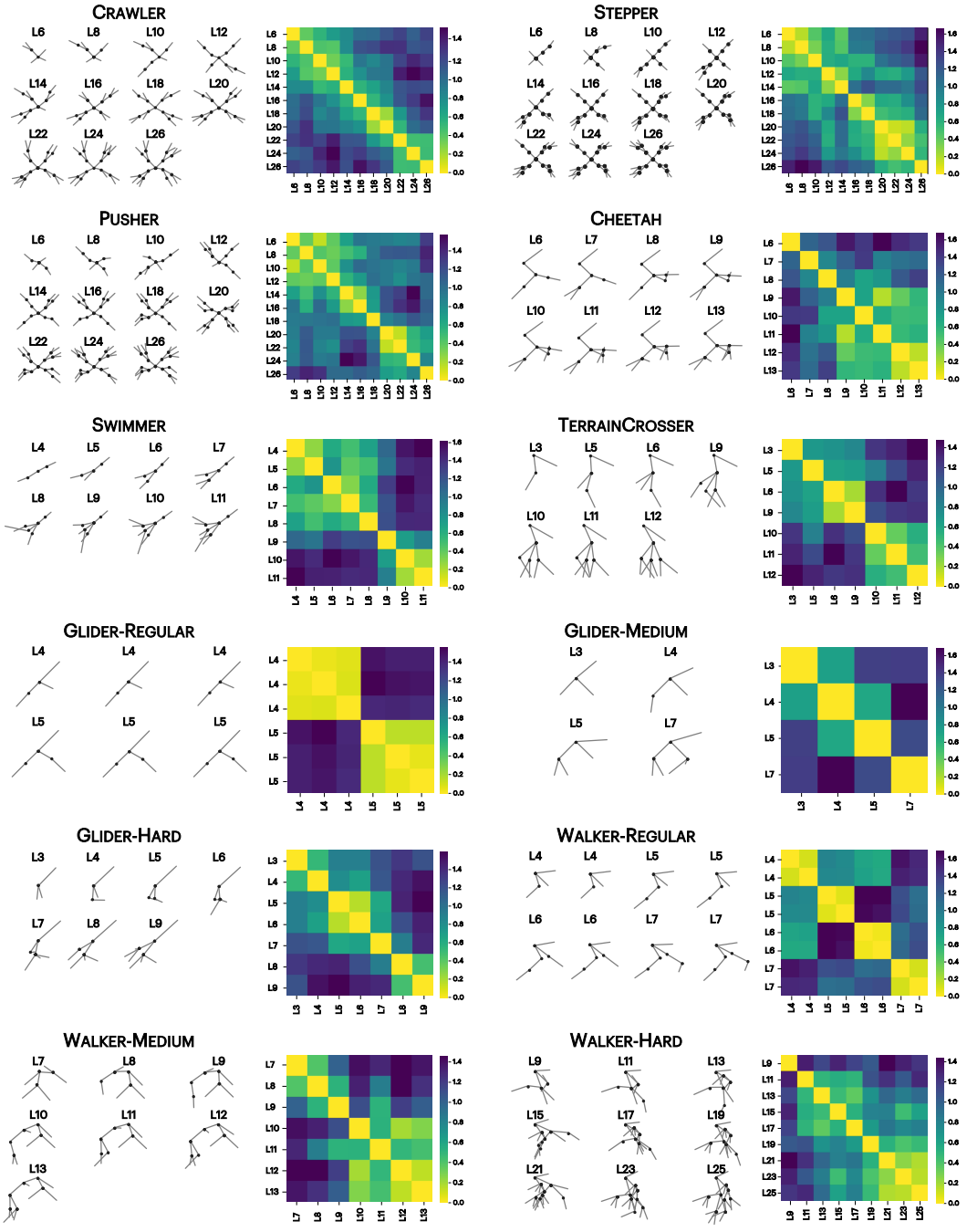


Figure 13: **Structural organization of morphology codes in the Morphogene space.** Representative morphologies are shown together with their pairwise cosine-distance matrices, where L_n denotes the number of limbs. Lighter colors indicate greater latent similarity, whereas darker colors indicate larger distances. The block and gradual patterns demonstrate that morphologies with similar complexity, topology, and limb configurations are organized into nearby regions of the latent space.

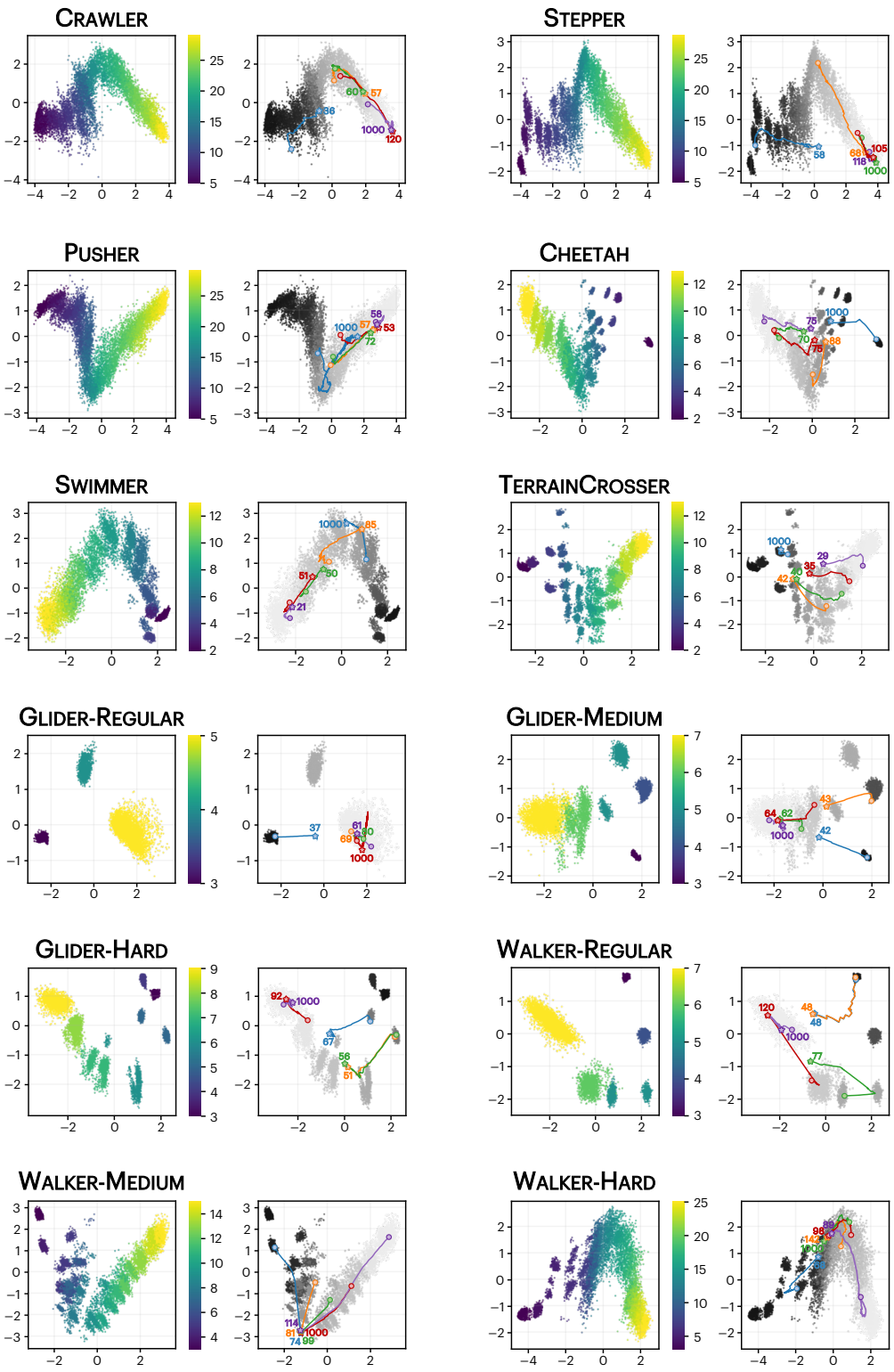


Figure 14: **Morphogene update trajectories in task-specific latent spaces.** For each environment, the left panel shows a PCA projection of valid morphology codes, while the right panel overlays Morphogene trajectories during co-design. These trajectories illustrate how performance-guided updates explore complementary regions of the structured latent space. ○ and ☆ mark the initial and final positions, respectively. Numerical labels indicate the epoch at which each Morphogene is eliminated or, for retained Morphogenes, the final epoch of the 1,000-epoch co-design process.

families, progressively relaxed branching limits further expand the range of possible topologies while preserving each task objective. Together, the BodyGen-based environments test whether a co-design method can explore structurally diverse morphologies and learn compatible controllers under distinct task demands. By expressing topology edits and attribute updates as sequential actions evaluated through policy rollouts, the benchmark also provides a natural setting for RL-based morphology design and for assessing how effectively task feedback guides body–brain co-adaptation.

Search Methods and Computational Cost Methods using the UNIMAL design space address morphology-evaluation costs differently. DERL (Gupta et al., 2021) evolves individual morphologies and trains a controller for each candidate, making fitness evaluation expensive. LOKI (Jeon et al., 2025) reduces this burden by sharing controllers among clusters of similar morphologies and exploring new designs through dynamic local search. Both approaches nevertheless search over explicit morphology candidates, whose performance must be evaluated as exploration proceeds.

GeCode instead organizes exploration around a small set of anchors in the compact Morphogene space. Conditioned rollouts explore body–controller designs locally around each anchor, while performance-guided updates move the anchors toward promising latent regions. Because these rollouts also train the shared design and control policies and provide the returns used for anchor updates, GeCode requires neither candidate-specific controller training nor a separate evaluation loop for global latent-space exploration, thereby balancing effective co-design with negligible additional computational cost (see App. E.3 for a detailed efficiency analysis).

F.2 Limitations and Future Work

In this section, we discuss the limitations of the current framework and outline directions for extending Morphogene-driven co-design.

(1) Scaling to highly diverse morphologies. GeCode shares a control policy across Morphogenes, allowing experience to transfer between related designs. Early in training, however, the explored morphologies may differ substantially in topology and dynamics. A shared controller with limited capacity may learn some bodies more readily than others, biasing their observed returns and, in turn, the performance-guided Morphogene updates. GeCode partly mitigates premature concentration by incorporating the best locally generated morphology code into each anchor update, thereby preserving anchor-specific exploration. This does not fully remove biases arising from unequal controller adaptation. Future work could use Morphogenes to guide knowledge routing or modular policy selection (Feng et al., 2026; Xie et al., 2026; 2025), enabling specialization across body structures while retaining the benefits of shared learning.

(2) Shared Morphogene Spaces Across Tasks. The Morphogene space is constructed from morphological structure and geometry rather than task rewards, suggesting that it may capture body-level factors shared across tasks. In this study, we construct the space and train the policies separately for each environment. Although offline Morphogene-space construction requires no environment interaction and incurs modest computational cost, whether Morphogenes can transfer to new objectives or terrains remains an open question. Future work could investigate a shared Morphogene space across compatible design domains, together with task-adaptive conditioning that reuses body-level knowledge while adapting morphology generation and control.

(3) Physical deployment. Like much of morphology–control co-design research, GeCode is evaluated in simulation using MuJoCo (Todorov et al., 2012). Extending such methods to physical robots remains challenging: exploring diverse body structures requires fabrication or reconfigurable hardware, while most available robot platforms have fixed morphologies, such as humanoid (Cao, 2025; Liu et al., 2026) and quadruped designs (Han et al., 2024; Long et al., 2024). Fabrication constraints, actuator limits, and differences between simulated and real contact dynamics further complicate deployment. Our results demonstrate Morphogene-driven co-design in simulation rather than on hardware. Nevertheless, Morphogene's compact representation of coordinated body–brain designs could help identify promising structural features and control requirements, providing guidance for the design and fabrication of physical robots.