

Hebbian Attractor Networks for Robot Locomotion

Alexander Dittrich, Fuda van Diggelen and Dario Floreano, *Fellow, IEEE*
 Laboratory of Intelligent Systems, EPFL, Switzerland
 {alexander.dittrich, fuda.vandiggelen, dario.floreano}@epfl.ch

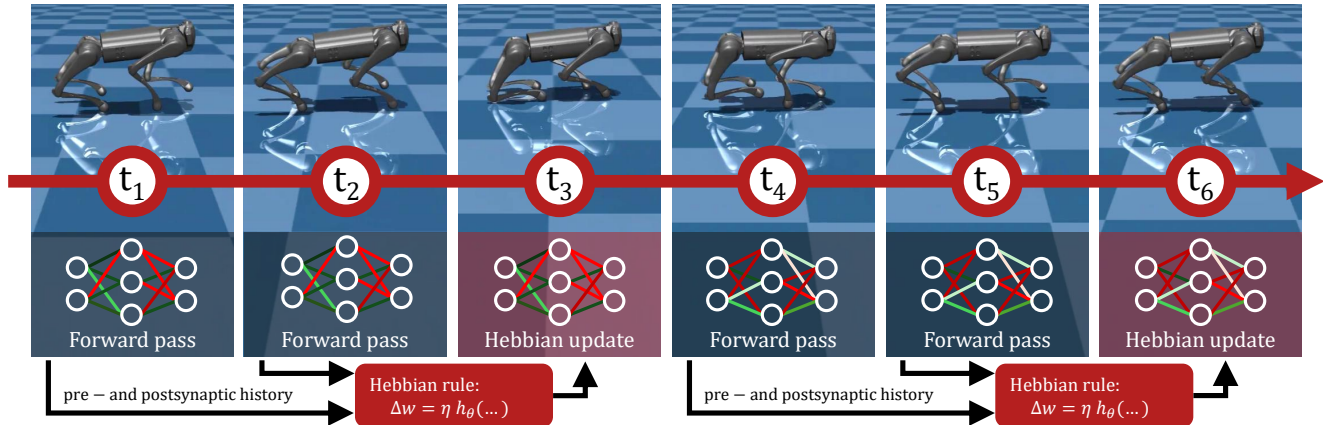


Fig. 1: Hebbian neural networks are updated during the lifetime based on correlations between pre- and postsynaptic activations and do not rely on gradient information. By introducing a slower rate of Hebbian updates than the action inference or averaging of the synaptic history, we can induce different attractor dynamics in weight space.

Abstract—Biological neural networks continuously adapt and modify themselves in response to experiences throughout their lifetime—a capability largely absent in artificial neural networks. Hebbian plasticity offers a promising path toward rapid adaptation in changing environments. Here, we introduce Hebbian Attractor Networks (HAN), a class of plastic neural networks in which local weight update normalization induces emergent attractor dynamics. Unlike prior approaches, HANs employ dual-timescale plasticity and temporal averaging of pre- and postsynaptic activations to induce either co-dynamic limit cycles or fixed-point weight attractors. Using simulated locomotion benchmarks, we gain insight into how Hebbian update frequency and activation averaging influence weight dynamics and control performance. Our results show that slower updates, combined with averaged pre- and postsynaptic activations, promote convergence to stable weight configurations, while faster updates yield oscillatory co-dynamic systems. We further demonstrate that these findings generalize to high-dimensional quadrupedal locomotion with a simulated Unitree Go1 robot. These results highlight how the timing of plasticity shapes neural dynamics in embodied systems, providing a principled characterization of the attractor regimes that emerge in self-modifying networks.

I. INTRODUCTION

Robot learning approaches follow a train-deploy paradigm: a neural network policy is trained in simulation to master a pre-defined task, by optimizing a static set of weight values. During deployment, these weights remain fixed, preventing

Code and supplementary material at <https://github.com/alexanderdittrich/hebbian-attractor-networks>

All authors are affiliated with the Laboratory of Intelligent Systems, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland.

adaptation to novel or changing conditions [1], [2], [3]. However, real-world environments are dynamic—sensors drift, terrain conditions vary, and actuator responses degrade. This brittleness contrasts sharply with biological nervous systems, which leverage continuous synaptic plasticity to self-modify throughout the organism’s lifetime. Inspired by the adaptivity of biological systems, we investigate continuously self-modifying networks in the form of Hebbian learning.

Hebbian neural networks (HNNs) display continuous adaptation of connection weights according to the co-activation of pre- and postsynaptic neurons [4]. When integrated into artificial neural networks, Hebbian plasticity introduces dynamic, self-modifying behaviors during deployment. Prior work has shown that Hebbian learning can produce effective controllers [5], [6], [7], [8], [9], [10]. However, how weight changes interact with control, stabilize behavior, and prevent runaway dynamics remain poorly understood.

In this work, we introduce Hebbian Attractor Networks (HANs)—a class of Hebbian neural networks where weight normalization induces emergent attractor dynamics. We show that max-normalized plasticity does more than prevent unbounded weight growth: it enables structured, recurring weight behaviors such as limit cycles and fixed-point attractors. These attractor dynamics are shaped by the temporal structure of the plasticity—specifically, the frequency of Hebbian updates and the averaging window of synaptic activations (Figure 1). Our method draws inspiration from the multi-timescale architecture of biological systems [11], [12]. We implement Hebbian updates that are decoupled from the control loop and based on temporally averaged

activation traces. These modifications result in qualitatively distinct weight dynamics. Fast updates promote oscillatory co-dynamic systems, where weight changes and motor control are tightly coupled. Slower updates with longer averaging windows induce stable fixed-points in weight space. Using standard locomotion benchmarks, we show that HANs support qualitatively distinct attractor regimes—fixed points and limit cycles—whereas prior Hebbian approaches with synchronized updates produce only co-dynamic limit cycles [5], [9]. By leveraging dual-timescale plasticity and activation averaging, HANs enable both co-dynamic and fixed-point attractor regimes—offering improved robustness and adaptability in robot locomotion. Here, plasticity is considered both in its role in meta-learning and as a mechanism that can give rise to characteristic network dynamics.

The contribution of this paper is three-fold: (i) We provide a systematic analysis of HANs on continuous-control locomotion tasks, situating them relative to static evolutionary controllers and gradient-based RL baselines. (ii) We characterize the emergence of distinct attractor regimes in weight space, showing how update frequency and activation averaging shape the onset of oscillatory versus fixed-point dynamics. (iii) We demonstrate that the resulting attractor dynamics support adaptive control beyond toy benchmarks, scaling to quadrupedal locomotion on a simulated Unitree Go1 robot.

II. RELATED WORK

Self-modifying networks and meta-learning: From a wider perspective, Hebbian learning can be considered as a meta-learning approach. Other meta-learning methods like MAML [13] or fast weights [14], [15] enable rapid adaptation by performing gradient-based updates during deployment. These systems require explicit error signals and back-propagation to fine-tune parameters in real time. In contrast, Hebbian learning provides a bio-inspired alternative: adaptation occurs through local, activity-dependent plasticity without relying on global error signals or gradient computations.

Hebbian learning and weight dynamics: Hebbian plasticity utilizes local synaptic updates dependent solely on the activities of pre- and postsynaptic neurons, making it computationally simple and biologically plausible. Extensive research on Hebbian weight dynamics, such as Oja’s rule [16] and the BCM theory [17], has provided insights into how synaptic weights can self-modify toward stable configurations. However, these classical models primarily address static tasks or stationary input distributions, whereas in our work, we focus on dynamic, continuously evolving environments. While in its principal form (a product of pre- and postsynaptic activations), Hebbian learning can be optimized using backpropagation [18], [19], other work has explored the evolution of more expressive Hebbian rule formulations [20], [21], [22], [6]. Evolutionary algorithms are capable of exploring high-dimensional search spaces without imposing restrictive constraints on rule formulation [23], [24]. Several studies [8], [5], [9], [10] commonly use a generalized Hebbian formulation, known as the ABCD

rule, where synaptic weight updates depend linearly on combinations of pre- and postsynaptic neuron activations. To mitigate unstable weight growth, Max Normalization (MN) by rescaling the weights has been employed, improving performance and stability [9]. Najarro et al. [5] demonstrated that evolved Hebbian rules can enable networks to rapidly recover from physical damage, effectively self-organizing during deployment. Ferigo et al. [7] further showed in voxel-based robots the emergence of oscillatory weights with MN enabled, indicating a co-dynamic system in which the plastic weights act more as intrinsic pattern generators rather than mechanisms of learning. While prior work has demonstrated the utility of Hebbian learning in adaptive control, the role of update timing and temporal structure in shaping converging versus oscillatory behaviors in embodied systems remains largely unexplored — a gap we address through the framework of HANs.

III. HEBBIAN ATTRACTOR NETWORKS

A. Hebbian Neural Networks

The NNs used in this paper have a feedforward architecture consisting of L layers. Let $\mathbf{x}^{(k)}(t)$ denote the vector of activations at layer k and time $t \in \mathbb{N}$, and let $W^{(k)}(t)$ be the matrix of weights connecting layer $k-1$ to k . The network receives a state \mathbf{s}_t and produces a deterministic action \mathbf{a}_t as follows:

$$\mathbf{x}^{(0)}(t) = \mathbf{s}_t, \quad (1)$$

$$\mathbf{x}^{(k)}(t) = \varphi \left(W^{(k)}(t) \mathbf{x}^{(k-1)}(t) \right), \quad \text{for } k = 1, \dots, L, \quad (2)$$

$$\mathbf{a}_t = \mathbf{x}^{(L)}(t), \quad (3)$$

where φ is a nonlinear activation function (in our case, \tanh). The synaptic weights $W^{(k)}(t)$ are updated online during deployment via local Hebbian rules. Each connection (i, j) in layer k is associated with a parameterized Hebbian update function $h_{\theta_{ij}^{(k)}}$ and a local learning rate $\eta_{ij}^{(k)}$. The scalar Hebbian update at discrete time step t is computed as:

$$\Delta w_{ij}^{(k)}(t) = \eta_{ij}^{(k)} \cdot h_{\theta_{ij}^{(k)}} \left(\bar{x}_j^{(k-1)}(t), \bar{x}_i^{(k)}(t) \right), \quad (4)$$

where the presynaptic activation $\bar{x}_j^{(k-1)}(t)$ and postsynaptic activation $\bar{x}_i^{(k)}(t)$ are computed as a moving average (MA) of length M of the neuron’s activation values:

$$\bar{x}_i^{(k)}(t) = \frac{1}{M} \sum_{m=0}^{M-1} x_i^{(k)}(t-m) \quad (5)$$

The update function $h_{\theta_{ij}^{(k)}}$ follows the commonly used generalized ABCD rule and is defined as:

$$h_{\theta_{ij}^{(k)}}(x_j^{(k-1)}, x_i^{(k)}) = a_{ij}^{(k)} x_j^{(k-1)} x_i^{(k)} + b_{ij}^{(k)} x_j^{(k-1)} + c_{ij}^{(k)} x_i^{(k)} + d_{ij}^{(k)}, \quad (6)$$

where $\theta_{ij}^{(k)} = (a_{ij}^{(k)}, b_{ij}^{(k)}, c_{ij}^{(k)}, d_{ij}^{(k)})$ are the Hebbian coefficients of the update function. Both these coefficients and the associated local learning rates $\{\theta, \eta\}$ are optimized

using Evolutionary Strategies (ES) [25]. After each Hebbian update, we apply a layerwise MN to prevent unbounded growth and enable more complex weight dynamics [7], [9]. HNNs that utilize MN are denoted as HANs. Namely, for each layer k , we scale weights by their maximum absolute value:

$$w_{ij}^{(k)}(t+1) \leftarrow \frac{w_{ij}^{(k)}(t+1)}{\max_{i,j} |w_{ij}^{(k)}(t+1)|} \quad (7)$$

B. Dual-timescale in Hebbian Attractor Networks

Unlike previous works [5], [9], we do not apply Hebbian updates at every time step. Instead, we define a fast NN controller frequency f_{NN} and a slow Hebbian update frequency f_{hebb} and apply updates every $\tau_{\text{hebb}} = \lfloor f_{\text{NN}}/f_{\text{hebb}} \rfloor$ steps:

$$w_{ij}^{(k)}(t+1) = \begin{cases} w_{ij}^{(k)}(t) + \Delta w_{ij}^{(k)}(t), & \text{if } t \bmod \tau_{\text{hebb}} = 0, \\ w_{ij}^{(k)}(t), & \text{otherwise.} \end{cases} \quad (8)$$

This decoupling introduces a dual-timescale structure, as displayed in Figure 2. We aggregate pre- and postsynaptic activations over multiple forward passes and compute a moving average (Equation 5). This averaging enables the learning rule to operate on smoothed activation signals that reflect longer temporal dynamics rather than relying solely on instantaneous activations.

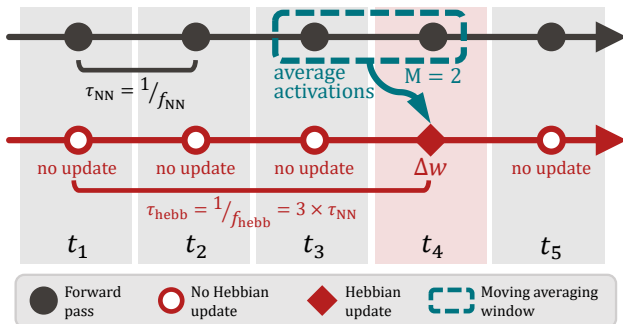


Fig. 2: Dual-timescale structure of HNN. Hebbian updates are executed at a lower frequency than the forward pass (here, $\tau_{\text{hebb}} = 3 \times \tau_{\text{NN}}$), and the moving average of pre- and postsynaptic activations is applied (here, $M = 2$).

In the following, we denote the variations of HANs in ($M = \dots$) superscript for the moving average and the timescale in ($f_{\text{NN}} = \dots \times f_{\text{hebb}}$) subscript, e.g. $\text{HAN}_{f_{\text{NN}}=4 \times f_{\text{hebb}}}^{M=10}$.

IV. EXPERIMENTAL RESULTS

We now investigate empirically how different structural components of Hebbian learning affect performance and weight dynamics.

Specifically, we perform an ablation on the effects of MN, dual-timescale plasticity ($f_{\text{NN}} > f_{\text{hebb}}$), and MA ($M > 1$) across a selected set of Hebbian learning conditions. Table I lists the ablation conditions of HANs. For instance, the work of Najarro et al. [5] uses the HAN condition (B) ($\text{HAN}_{f_{\text{NN}}=f_{\text{hebb}}}^{M=1}$) with MN, without MA ($M = 1$), and a single-timescale ($f_{\text{NN}} = f_{\text{hebb}}$).

TABLE I: Overview of tested HAN conditions.

Condition	MN	M	$f_{\text{NN}}/f_{\text{hebb}}$
(A) HNN	✗	1	1
(B) $\text{HAN}_{f_{\text{NN}}=f_{\text{hebb}}}^{M=1}$	✓	1	1
(C) $\text{HAN}_{f_{\text{NN}}=4 \times f_{\text{hebb}}}^{M=1}$	✓	1	4
(D) $\text{HAN}_{f_{\text{NN}}=f_{\text{hebb}}}^{M=10}$	✓	10	1
(E) $\text{HAN}_{f_{\text{NN}}=4 \times f_{\text{hebb}}}^{M=10}$	✓	10	4

A. Hebbian learning conditions influence performance

We evaluate HNNs and HANs on the standard continuous-control benchmarks from Gymnasium [26], namely Swimmer-v5, HalfCheetah-v5, Hopper-v5, and Walker2d-v5. These environments provide widely used reference tasks for assessing the utility of learning algorithms in locomotion and allow us to situate the behavior of Hebbian plasticity in relation to established methods. For all Hebbian architectures, we employ a compact network with a single hidden layer of 16 units, resulting in 800–1840 parameters depending on the task. Initial weights \mathbf{W} are sampled from $\mathcal{U}(-0.1, 0.1)$, while Hebbian coefficients and learning rates θ, η are drawn from $\mathcal{U}(-1.0, 1.0)$. Optimization is performed with ES using evosax [27]. The most important hyperparameters include a population size of 128, mutation rate of 0.5, and selection ratio of 10% over 1000 generations (500 for Swimmer). Each setting is evaluated four times per run, and results are averaged over 16 independent seeds (random initializations of network weights and environment seeds).

TABLE II: Average accumulated episodic return / fitness on Gymnasium locomotion benchmarks ($n = 16$ for HNN/HAN and MLP/GRU; $n = 10$ for PPO), where \pm captures a 95% confidence interval. MLP[†] omits bias terms to match the parameter count of HNNs/HANs. PPO[‡] matches the number of optimization parameters of HNNs/HANs.

Algorithm	Fitness (-)			
	Swimmer	Hopper	HalfCheetah	Walker2D
PPO [‡]	134.4 \pm 1.8	2641.2 \pm 1133.1	3722.9 \pm 1842.4	4022.8 \pm 2253.7
PPO	135.7 \pm 3.5	4068.8 \pm 179.5	9644.7 \pm 1084.4	7947.8 \pm 899.6
MLP [†]	361.4 \pm 0.4	3470.9 \pm 195.2	3496.7 \pm 382.6	3292.3 \pm 121.6
MLP	361.9 \pm 0.3	4186.0 \pm 129.8	3168.7 \pm 115.5	4310.9 \pm 339.1
GRU	363.7 \pm 0.4	3676.8 \pm 159.5	4454.6 \pm 413.5	4066.1 \pm 418.9
(A) HNN	359.5 \pm 0.6	2466.0 \pm 382.1	3370.3 \pm 601.7	3033.3 \pm 127.4
(B) HAN	363.6 \pm 0.2	3713.1 \pm 322.6	6385.3 \pm 601.9	4532.5 \pm 831.1
(C) HAN	363.7 \pm 0.2	3697.0 \pm 276.3	5055.5 \pm 762.6	4261.1 \pm 417.1
(D) HAN	363.9 \pm 0.2	3729.7 \pm 330.2	7239.6 \pm 430.2	4548.5 \pm 492.4
(E) HAN	363.9 \pm 0.3	4100.0 \pm 172.3	7394.0 \pm 301.5	4046.9 \pm 280.9

To isolate the contribution of plasticity, we compare HANs against several directly evolved static baselines using the same ES. These include (i) MLP controllers without plasticity but matched in parameter count (without bias terms), (ii) MLP and GRU controllers with bias parameters, where the GRU provides behavioral adaptation through recurrent hidden state rather than synaptic plasticity. This design

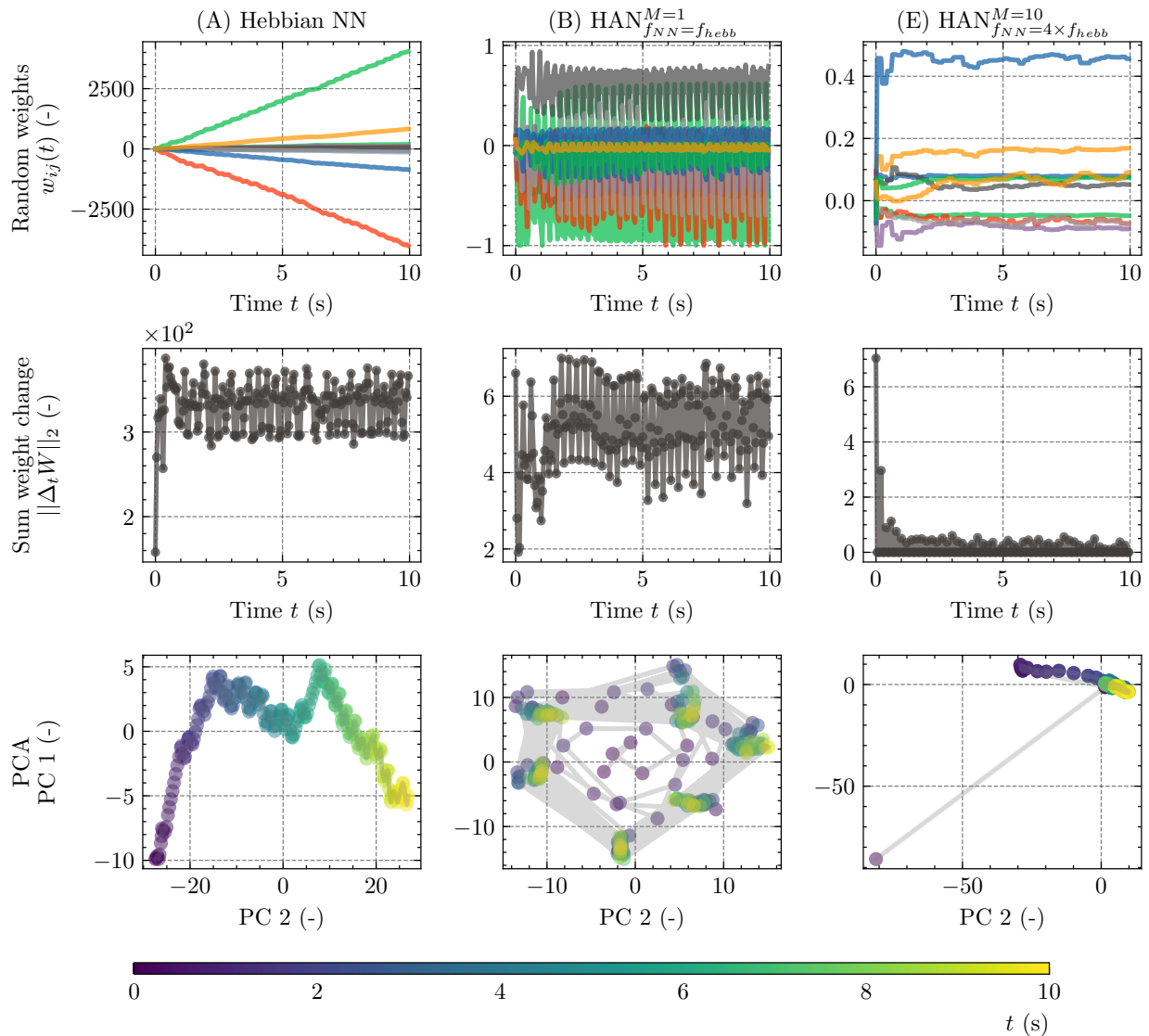


Fig. 3: Analysis of weight dynamics under three Hebbian learning conditions (A, B, E). The first row shows trajectories of 10 randomly selected plastic weights over time. The second row presents the total network plasticity, quantified as the summed ℓ_2 -norm of weight changes at each timestep. The third row depicts the evolution of plastic weights projected onto the first two principal components via Principal component analysis (PCA).

allows us to assess how much of the observed performance stems from within-lifetime Hebbian adaptation, beyond what can be captured by recurrence or direct optimization.

For additional context, we also report results with PPO [28], using Stable-Baselines3 [29]. Two architectures are considered: (a) a “tiny” variant matching the HANs (single hidden layer with 16 units for the actor), and (b) the default PPO architecture with two 64-unit hidden layers for the actor. In both cases, the critic uses the default two-layer 64-unit architecture. PPO is trained for 80 million environment steps with default hyperparameters, and results are averaged over 10 seeds. These PPO runs provide reference points, not direct benchmarks, as the optimization algorithms and inductive biases differ substantially.

Table II reports mean performance across conditions. The ablation study shows that adding max-normalization (MN)

substantially improves performance across all tasks. Incorporating either activation averaging, dual timescales, or both yields further gains. As a result, HANs outperform directly comparable static MLPs without bias, and in most cases also surpass MLPs and GRUs with bias, except for Hopper where static baselines remain competitive. When compared to PPO, HANs achieve comparable returns on simpler environments such as Swimmer, while gradient-based PPO with its larger default architecture attains the highest scores on Hopper, HalfCheetah, and Walker2D. These differences reflect the respective strengths of the methods: PPO leverages gradient updates and larger networks, whereas HANs exploit within-lifetime plasticity to enrich the expressivity of compact architectures. Importantly, HANs demonstrate that Hebbian rules can be harnessed to improve performance beyond what is achievable with static evolution or direct gradient

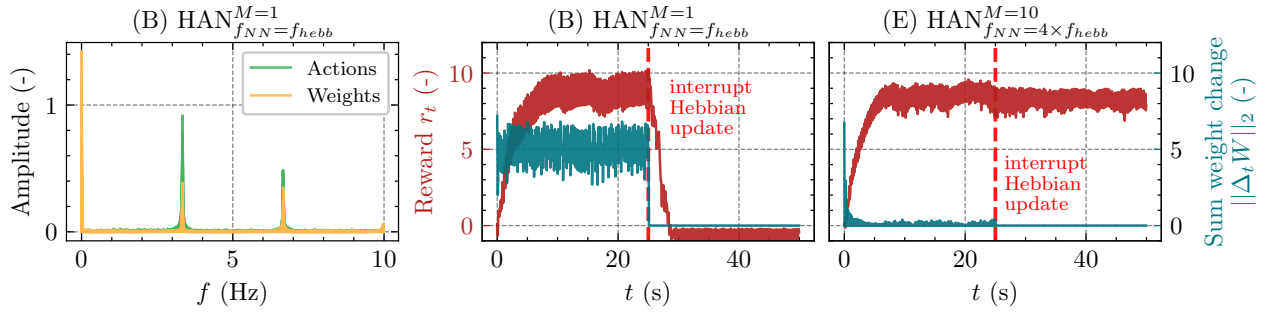


Fig. 4: The non-zero frequencies of action and weight signals exhibit peaks at approximately 4 Hz and 8 Hz. Interrupting the Hebbian update causes a pronounced drop in step rewards during deployment for the HAN condition (B). The HNN in condition (E) shows a minor effect in step rewards.

optimization on the same small architectures.

B. Fixed-point attractors emerge with decoupled Hebbian updates and averaging of activations

For further investigation of the attractor dynamics, we select the best-performing Hebbian coefficients in the HAN conditions (A), (B) and (E) of the HalfCheetah task. Figure 3 shows the weight dynamics during a single lifetime over 10s. The first row depicts 10 randomly selected plastic weights. For the second row, we compute the elementwise ℓ_2 distances of the weight matrices between the current t_n and the previous time step t_{n-1} , and we sum them over the network. This metric gives an indication of overall network plasticity, where a value of 0 signifies no change in network weights. In the third row, we use a PCA to reduce the plastic weights at each time step to two dimensions.

For reference, HNN in condition (A) shows the expected unbounded growth: individual weights diverge and no distinct attractor behavior emerges, as confirmed by the PCA embedding. Similarly, we observe sustained high plasticity in HANs under condition (B), with single weights oscillating periodically. The plastic weights remain within a $(-1, 1)$ -bound. In the PCA embedding, we observe that the plastic weights form an oscillatory pattern, where the principal components jump between several states. In condition (E), the total plastic weight change decreases after a short phase, remaining close to 0 thereafter. We observe that single weights remain constant after an initial jump. In the PCA embedding, the compressed weights remain within a region of the weight space after a strong initial change.

C. Attractor analysis

The results of Section IV-B show the onset of distinct weight dynamics conditioned on the different HAN conditions. We investigate the design of these different weight dynamics through additional ablation studies by altering Hebbian update frequency f_{hebb} and averaging window length M . We use HANs in condition (E) for the HalfCheetah task and perform 16 randomly initialized evolutions for each combination of window lengths $M = 1, 2, 4, 10, 20$ and Hebbian update frequencies $f_{\text{hebb}} = 1, 5, 10, 20$.

Figure 5 shows the fitness of the best performing individual at the final generation over different random seeds.

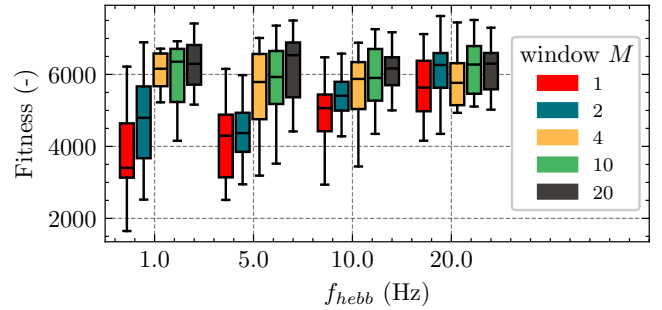


Fig. 5: Fitness at 1000 generations with different f_{hebb} and M ($n = 16$). A window length of $M = 1$ corresponds to no moving average, and $f_{\text{hebb}} = 20$ corresponds to no reduction in update frequency.

To assess whether a HAN converges to a fixed-point attractor, we track the weight change over time in the early and late phase of an episode. We compare the difference of the total weight change $\Delta_t W = \sum_k \|W_t^{(k)} - W_{t-1}^{(k)}\|_2$, where early denotes the first 5% of the rollout, and late the remaining rollout. We classify the HAN condition as converged if $\overline{\Delta W}_{\text{late}} < \rho \cdot \overline{\Delta W}_{\text{early}}$. For robustness, we added the threshold factor $\rho = 0.9$ and evaluate 3 population members at the last generation of 16 seeds for 30 independent rollouts. Each combination of f_{hebb} and M is evaluated 1440 times.

TABLE III: Ratio of HANs satisfying the convergence criterion for different f_{hebb} and M values over 1440 different evaluations.

M	Hebbian update frequency f_{hebb}			
	1 Hz	5 Hz	10 Hz	20 Hz
1	99.9 %	64.4 %	21.9 %	21.4 %
2	99.9 %	64.9 %	16.1 %	17.8 %
4	99.9 %	81.7 %	55.0 %	36.6 %
10	99.9 %	94.4 %	75.7 %	66.0 %
20	99.9 %	100.0 %	100.0 %	92.9 %

The results show that convergence to a fixed-point attractor depends on the choice of f_{hebb} and M . To achieve converging plastic weights, either low Hebbian update frequencies f_{hebb} , large average window lengths M or a combination of both is effective. Low f_{hebb} consistently results in converging weights; however, it has negative impacts on performance

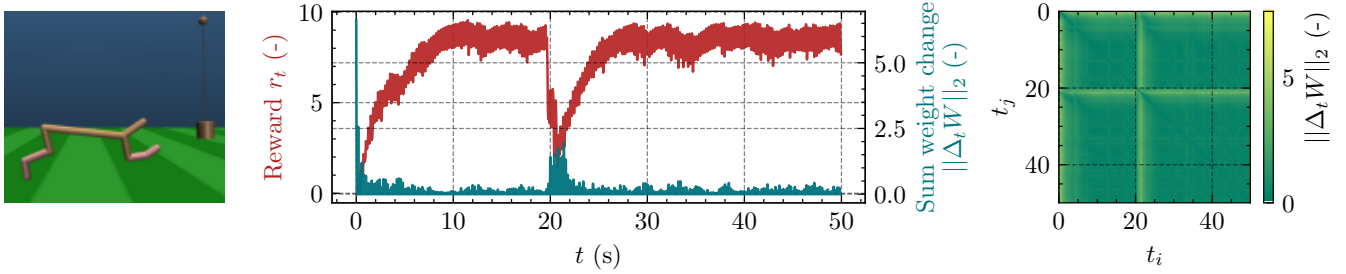


Fig. 6: Stepwise rewards summed weight changes during a 50s deployment. A collision around 19s causes a drop in reward, followed by recovery. Weight changes decrease over time, with a brief spike after the collision. The network returns to a similar weight configuration after the disturbance, indicating a stable attractor.

compared to higher f_{hebb} . We found that $M \geq 4$ mitigated the decrease in performance while maintaining the convergence properties.

We continue by analyzing the different properties of the weight attractors. A Fourier analysis on 30 randomly selected plastic weights $w_{ij}^{(k)}$ of the HANs in condition (B) shows that the dynamics of the weights are in tune with the locomotion pattern of the agent. The main non-zero frequencies of the plastic weights and the action signal are highly correlated (Figure 4, left). When we pause Hebbian updates during the lifetime, we see that the oscillatory attractor drives a vital part of locomotion as the agent is unable to move (Figure 4 middle, a substantial drop in reward after $t = 20$). This is not the case for the condition (E) which indicates a more robust attractor behavior.

The stability of the attractor is analyzed in a final perturbation experiment. Here, we randomly place a hanging pendulum with a mass of 50 kg in the locomotion path as a force perturbation (Figure 6, left). This condition has not been part of the training process; it thus requires online adaptation of the network. HANs in condition (E) show high plasticity at the start of periodic locomotion until a steady-state periodic gait is reached. Perturbations from this gait result in an increase in plasticity. Figure 6 (right) shows the summed ℓ_2 distance of weight snapshots between each timestep of the lifetime. This allows the comparison of all weight snapshots during the lifetime. The summed distance between plastic weights before and after the perturbation is close to zero, indicating the existence of a single fixed-point attractor and therefore a single weight configuration the network converges to.

D. Unitree Go1 locomotion

To assess whether our findings generalize to higher-dimensional systems, we extend to quadrupedal locomotion using a Unitree Go1 robot simulated in MuJoCo XLA (MJX) [30], modified from the Joystick task in MuJoCo Playground [31]. The reward consists of a Gaussian tracking term for target forward velocities of 1.0 m s^{-1} , 1.5 m s^{-1} and 2.0 m s^{-1} and a Gaussian term for maintaining zero yaw. We employ OpenAI-ES [23] with a population size of 512, exponential decay schedules for learning rate (0.1, decay 0.999) and mutation rate (0.2, decay 0.995) over 500

generations, with four repeats per evaluation and 1000-step episodes at $f_{\text{NN}} = 50 \text{ Hz}$.

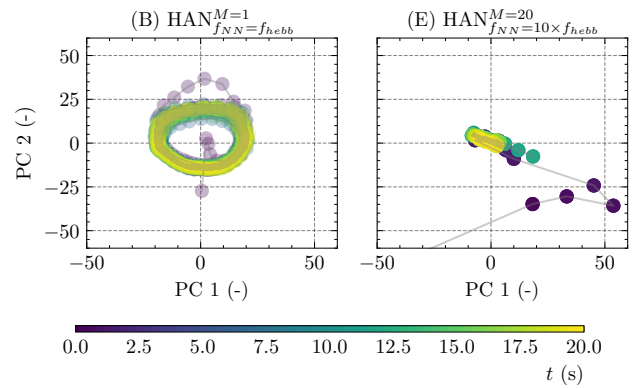


Fig. 7: PCA of HANs in conditions (B) and (E).

The observation space includes trunk angular and linear velocities, joint angles and velocities, and the gravity vector. The network outputs desired joint angles for each leg’s hip, thigh, and calf motors, tracked by a PD controller ($k_p = 50$, $k_d = 1$) at 1000 Hz. We train 10 randomly initialized HANs per target velocity under condition (B) ($\text{HAN}^{M=1}_{f_{\text{NN}}=f_{\text{hebb}}}$) and condition (E) ($\text{HAN}^{M=20}_{f_{\text{NN}}=10 \times f_{\text{hebb}}}$), as in Section IV-A.

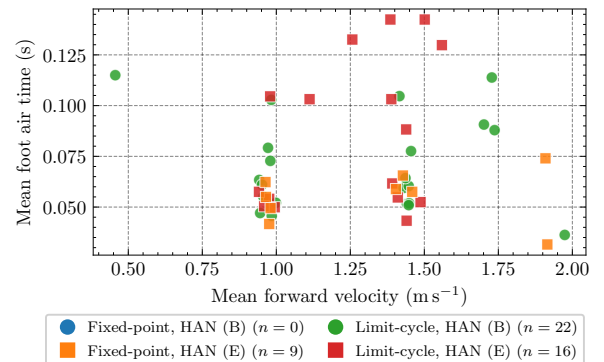


Fig. 8: Mean foot air time versus forward velocity for HANs in conditions (B) and (E) across 60 Unitree Go1 training runs, classified as fixed-point or limit-cycle attractors.

Consistent with the Gymnasium results (Section IV-B), condition (B) produces oscillatory weight trajectories cou-

pled to the gait cycle (Figure 7, left), while condition (E) converges to stable fixed-point attractors after an initial adaptation phase (Figure 7, right). Across 60 training runs, condition (B) produced sustained locomotion in 22 cases (36.7%), all exhibiting limit-cycle dynamics. Condition (E) yielded stable locomotion in 25 cases (41.7%), of which 9 (36.0%) converged to fixed-point attractors and 16 (64.0%) remained limit-cycle; the remaining 13 runs (21.7%) did not produce stable locomotion. The parsimonious reward design — rewarding only target speed and yaw — allows diverse gaits to emerge, summarized via mean forward velocity and foot air time in Figure 8, with attractors classified using the convergence criterion from Section IV-B ($\rho = 0.4$).

E. Morphological adaptation

Having established that conditions (B) and (E) give rise to distinct attractor dynamics, we now ask whether these differences affect adaptive capabilities under morphological damage. Following Najarro et al. [5], we evaluate adaptation capabilities on Ant-v5 [26] by significantly shortening the distal shin segment of one of the four legs. We compare PPO against HANs condition (B) ($\text{HAN}_{f_{\text{NN}}=f_{\text{hebb}}}^{M=1}$) and condition (E) ($\text{HAN}_{f_{\text{NN}}=10 \times f_{\text{hebb}}}^{M=20}$) across three training scenarios (no leg mutilation; front-right (FR) mutilated; front-left (FL) mutilated) and two unseen test scenarios (rear-left (RL) mutilated; rear-right (RR) mutilated) with an episode length of 500.

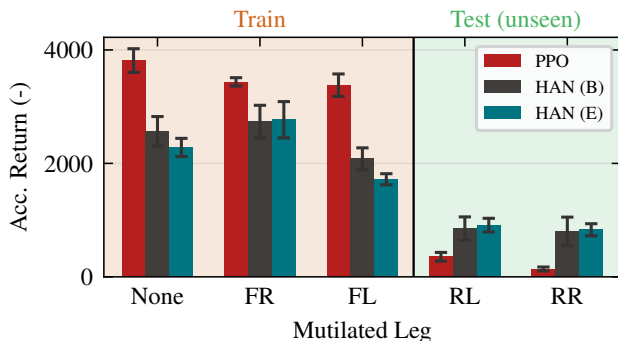


Fig. 9: Mean accumulated returns with 95% confidence intervals for each scenario. The left side shows training scenarios, while the right side shows unseen test scenarios. PPO significantly outperforms HANs within the training distribution, while HANs achieve higher returns on unseen test scenarios.

PPO is trained for 1B time steps with default hyperparameters, and both HAN conditions for 6B time steps with 15 random initializations using the same OpenAI-ES configuration as in Section IV-D. Mutilations are presented to PPO by dividing the 1536 training environments evenly, while HANs perform 4 repeated fitness evaluations per mutilation during meta-training. We evaluate the final policy or Hebbian rules of each seed 30 times per scenario (Figure 9). PPO significantly outperforms HANs within the training distribution. Despite dual-timescale plasticity and averaging, all HAN runs converge to limit-cycle attractors in this setting, with no significant differences between conditions (B) and

(E). Consistent with prior work [5], [9], the oscillatory weight dynamics may facilitate adaptation, as HANs maintain more consistent performance across seen and unseen morphologies.

V. DISCUSSION

In this work, we elucidate the effects of several Hebbian learning conditions on the dynamics of the weights. While plain HNN can solve simple benchmarks, we find that introducing MN yields HANs that achieve significantly higher average performance on all investigated benchmarks, confirming previous studies [7], [9] on different locomotion problems. It might be expected that a lower HAN update frequency would reduce responsiveness and therefore performance. Contrary to this intuition, decoupling HAN updates results in improved performance over the given benchmarks. Furthermore, averaging synaptic activations can achieve higher average performance (see HalfCheetah and Hopper). On common locomotion benchmarks, decoupled HANs allow identification of operating conditions with emerging fixed-point attractors that show comparable or higher performance than co-dynamic instances and static NNs. The timing of Hebbian updates strongly influences the evolution of Hebbian coefficients and gives rise to distinct plastic weight and attractor dynamics. A commonly used HAN configuration involving MN [5], [9], [8] introduces a limit cycle attractor with oscillating plastic weights in tandem with the periodic gait. This co-dynamic system drives the effective gait, as interrupting the Hebbian update during operational lifetime causes the agent to stumble (Figure 4, middle). These findings suggest that, in our setting, evolution tunes a co-dynamic system rather than creating a meta-learning mechanism. More importantly, the introduction of different timescales and moving average with reduced Hebbian update frequency f_{hebb} decouples the co-dynamics and results in the emergence of fixed-point attractors, which do not suffer from this weakness. In our perturbation test, neither freezing the weights nor applying a force perturbation disrupted the gait, suggesting robustness of the fixed-point attractor in this scenario. In the latter, the weights showed adaptation to regain stable locomotion before converging back to the same attractor (Figure 6). Under permanent morphological damage, both HAN conditions converge to limit-cycle attractors — even under the strong configuration of condition (E) — suggesting that evolution preferentially discovers oscillatory dynamics when adaptation across qualitatively different locomotion modes is required.

VI. CONCLUSION

This work provides a principled understanding of Hebbian learning and elucidates the role of dual-timescales and activation averaging. Specifically, weight analysis reveals the onset of different attractor dynamics, allowing us to design different adaptive properties in HANs. While the results highlight the potential of HANs for adaptive control, several limitations remain. ES, though effective for discovering plasticity rules, are sample-inefficient and have limited usability

for complex tasks; more sophisticated optimization methods could expand applicability. We use feedforward networks, yet introducing recurrence may yield richer dynamics through interactions between Hebbian updates and recurrent activity. Furthermore, this paper centers on periodic locomotion; generalization of HANs to more complex tasks—such as navigation, manipulation, or complex terrains—remains an open question. Future work will explore multi-attractor HANs, where autonomous transitions between attractors enable adaptation to sudden environmental changes. We hypothesize that such switching could be achieved by incorporating reward signals into the local Hebbian update, e.g. inspired by neuromodulation [32], [33], [34], helping bridge the sim-to-real gap. Finally, real-world experiments will be essential to assess HANs under noise, mechanical variation, and sensor drift.

ACKNOWLEDGEMENTS

This work was supported by the Swiss National Science Foundation (SNSF) and the Japan Society for the Promotion of Science (JSPS) under project number IZLJZ2.214053. Generative AI tools (OpenAI ChatGPT, Anthropic Claude) assisted with debugging, visualization scripts, and editorial improvements.

REFERENCES

- [1] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, p. eabc5986, 2020.
- [2] Z. Li, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for versatile, dynamic, and robust bipedal locomotion control," *The International Journal of Robotics Research*, p. 02783649241285161, 2024.
- [3] S. Ha, J. Lee, M. van de Panne, Z. Xie, W. Yu, and M. Khadiv, "Learning-based legged locomotion: State of the art and future perspectives," *The International Journal of Robotics Research*, p. 02783649241312698, 2024.
- [4] D. O. Hebb, *The organization of behavior: A neuropsychological theory*. New York: Wiley and Sons, 1949.
- [5] E. Najarro and S. Risi, "Meta-learning through hebbian plasticity in random networks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 20 719–20 731, 2020.
- [6] A. Yaman, G. Iacca, D. C. Mocanu, M. Coler, G. Fletcher, and M. Pechenizkiy, "Evolving plasticity for autonomous learning under changing environmental conditions," *Evolutionary computation*, vol. 29, no. 3, pp. 391–414, 2021.
- [7] A. Ferigo, G. Iacca, E. Medvet, and F. Pigozzi, "Evolving hebbian learning rules in voxel-based soft robots," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 15, no. 3, pp. 1536–1546, 2022.
- [8] S. Schmidgall, "Adaptive reinforcement learning through evolving self-modifying neural networks," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*. Cancún Mexico: ACM, Jul. 2020, pp. 89–90.
- [9] B. Leung, W. Haomachai, J. W. Pedersen, S. Risi, and P. Manoonpong, "Bio-inspired plastic neural networks for zero-shot out-of-distribution generalization in complex animal-inspired robots," *arXiv preprint arXiv:2503.12406*, 2025.
- [10] F. van Diggelen, T. A. Karagüzel, A. G. Rincon, A. Eiben, D. Floreano, and E. Ferrante, "Emergent heterogeneous swarm control through hebbian learning," *arXiv preprint arXiv:2507.11566*, 2025.
- [11] K. C. Bittner, A. D. Milstein, C. Grienberger, S. Romani, and J. C. Magee, "Behavioral time scale synaptic plasticity underlies cal place fields," *Science*, vol. 357, no. 6355, pp. 1033–1036, 2017.
- [12] W. Gerstner, M. Lehmann, V. Liakoni, D. Corneil, and J. Brea, "Eligibility traces and plasticity on behavioral time scales: experimental support of neohebbian three-factor learning rules," *Frontiers in neural circuits*, vol. 12, p. 53, 2018.
- [13] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*. PMLR, 2017, pp. 1126–1135.
- [14] J. Schmidhuber, "Learning to control fast-weight memories: An alternative to dynamic recurrent networks," *Neural Computation*, vol. 4, no. 1, pp. 131–139, 1992.
- [15] J. Ba, G. E. Hinton, V. Mnih, J. Z. Leibo, and C. Ionescu, "Using fast weights to attend to the recent past," *Advances in neural information processing systems*, vol. 29, 2016.
- [16] E. Oja, "Simplified neuron model as a principal component analyzer," *Journal of mathematical biology*, vol. 15, pp. 267–273, 1982.
- [17] E. L. Bienenstock, L. N. Cooper, and P. W. Munro, "Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex," *Journal of Neuroscience*, vol. 2, no. 1, pp. 32–48, 1982.
- [18] T. Miconi, K. Stanley, and J. Clune, "Differentiable plasticity: training plastic neural networks with backpropagation," in *International Conference on Machine Learning*. PMLR, 2018, pp. 3559–3568.
- [19] T. Miconi, A. Rawal, J. Clune, and K. O. Stanley, "Backpropamine: training self-modifying neural networks with differentiable neuromodulated plasticity," in *International Conference on Learning Representations*, 2019.
- [20] D. Floreano and J. Urzelai, "Evolutionary robots with on-line self-organization and behavioral fitness," *Neural Networks*, vol. 13, no. 4-5, pp. 431–443, 2000.
- [21] A. Soltoggio, J. A. Bullinaria, C. Mattiussi, P. Dür, and D. Floreano, "Evolutionary advantages of neuromodulated plasticity in dynamic, reward-based scenarios," in *Proceedings of the 11th international conference on artificial life (Alife XI)*. MIT Press, 2008, pp. 569–576.
- [22] J. W. Pedersen and S. Risi, "Evolving and merging hebbian learning rules: increasing generalization by decreasing the number of rules," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2021, pp. 892–900.
- [23] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, "Evolution Strategies as a Scalable Alternative to Reinforcement Learning," Sep. 2017.
- [24] R. Miikkulainen, "Neuroevolution insights into biological neural computation," *Science*, vol. 387, no. 6735, p. eadp7478, 2025.
- [25] I. Rechenberg, "Evolutionstrategien," in *Simulationsmethoden in der Medizin und Biologie: Workshop, Hannover, 29. Sept.–1. Okt. 1977*. Springer, 1978, pp. 83–114.
- [26] M. Towers, A. Kwiatkowski, J. Terry, J. U. Balis, G. De Cola, T. Deleu, M. Goulao, A. Kallinteris, M. Krimmel, A. KG *et al.*, "Gymnasium: A standard interface for reinforcement learning environments," *arXiv preprint arXiv:2407.17032*, 2024.
- [27] R. T. Lange, "evosax: Jax-based evolution strategies," in *Proceedings of the Companion Conference on Genetic and Evolutionary Computation*, 2023, pp. 659–662.
- [28] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.
- [29] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of machine learning research*, vol. 22, no. 268, pp. 1–8, 2021.
- [30] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.
- [31] K. Zakka, B. Tabanpour, Q. Liao, M. Haiderbhai, S. Holt, J. Y. Luo, A. Allshire, E. Frey, K. Sreenath, L. A. Kahrs *et al.*, "Mujoco playground," *arXiv preprint arXiv:2502.08844*, 2025.
- [32] A. Soltoggio, P. Durr, C. Mattiussi, and D. Floreano, "Evolving neuromodulatory topologies for reinforcement learning-like problems," in *2007 IEEE Congress on evolutionary computation*. IEEE, 2007, pp. 2471–2478.
- [33] D. Kudithipudi, M. Aguilar-Simon, J. Babb, M. Bazhenov, D. Blackiston, J. Bongard, A. P. Brna, S. Chakravarthi Raja, N. Cheney, J. Clune *et al.*, "Biological underpinnings for lifelong learning machines," *Nature Machine Intelligence*, vol. 4, no. 3, pp. 196–210, 2022.
- [34] A. Soltoggio, K. O. Stanley, and S. Risi, "Born to learn: the inspiration, progress, and future of evolved plastic artificial neural networks," *Neural Networks*, vol. 108, pp. 48–67, 2018.