Neural Circuit Architectural Priors for Rat Locomotion

Nikhil X. Bhattasali[1]*, **Jaron Cui**[1]*, **Lerrel Pinto**[1], **Grace W. Lindsay**[2,3]
[1]Department of Computer Science, [2]Department of Psychology, [3]Center for Data Science
New York University
nikhilxb@nyu.edu

Abstract

Animals generate behavior through continuous interactions between their neural circuits, bodies, and environment. Embodied simulations provide a means to study these interactions by testing neural circuit models in biologically realistic settings. Recent efforts have developed embodied simulations of diverse animals, and emerging work on biologically grounded neural architectures has begun to integrate circuit connectivity and single-unit dynamics from experimental data into artificial neural networks, introducing stronger inductive biases and closer mechanistic alignment with biology. How can we develop embodied simulations that enable the study of increasingly complex bodies and neural circuits? In this work, we address the challenge through coordinated advances in the simulated body, neural architecture, and training infrastructure. We introduce *Rat v2*, a biomechanical model that provides an efficient and flexible tool for simulating rat behavior. To control this body, we develop *Rat NCAP*, a biologically grounded neural architecture for locomotion based on mammalian spinal circuits that incorporates cell-type-specific connectivity and a novel flexor-extensor neuromechanical interface. To optimize circuit parameters, we develop GPU-accelerated simulation and training infrastructure. The integrated system learns to locomote at different speeds and transition between gaits without imitation learning, producing locomotion statistics, kinematics, and cell-type-specific neural activity consistent with experimental data. Systematic analyses reveal that the neuromechanical interface substantially affects trainability and gait quality, and *in silico* cell-type ablations reproduce known gait phenotypes and predict gait changes under graded perturbations. Together, these findings support the view that embodied simulations at an intermediate level of abstraction can simultaneously reproduce complex behavior and yield mechanistic insights into the neural circuits that generate it.

1 Introduction

Animals generate behavior through continuous interactions between their neural circuits, bodies, and environment. Embodied simulations provide a means to study these interactions by testing neural circuit models in biologically realistic settings, thereby complementing and extending experimental work [Ausborn et al., 2021]. Recent efforts have developed simulated body models of diverse animals, including the nematode [Sarma et al., 2018], fly [Lobato-Rios et al., 2022, Wang-Chen et al., 2024], salamander [Ijspeert et al., 2007], mouse [Ramalingasetty et al., 2021], and rat [Merel et al., 2019]. These models span a wide range of complexity and biorealism, raising a fundamental question about abstraction: How much biological detail should be incorporated into body models and neural circuit models if we aim to simultaneously reproduce complex behavior and gain mechanistic insights?

*Equal contribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

At the detailed end, we can build neuromusculoskeletal bodies with overactuated Hill-type muscular control, driven by conductance-based neurons arranged in biologically grounded circuits [Ramalingasetty et al., 2024]. This approach provides mechanistic insights and captures fine-grained phenomena that are crucial for certain questions, but the resulting models can be costly to simulate, difficult to optimize, and often underconstrained by available data, especially for complex behaviors. At the abstract end, we can use simplified bodies with joint position control, driven by standard artificial neural networks (ANNs) trained with reinforcement or imitation learning [Merel et al., 2019, Aldarondo et al., 2024]. This approach is more tractable to optimize and can scale to complex behaviors, but trained model mechanisms are difficult to relate to experimentally identified neural circuits, and producing naturalistic behavior often relies on imitation learning, which is biologically unrealistic and requires costly data collection. This motivates embodied simulations at an intermediate level of abstraction that balance scientific fidelity with computational tractability.

ANNs offer desirable features as neural models by abstracting away complex biophysical details like spikes and ion channels, and they can be flexibly trained to reproduce behavior and neural activity. Emerging work has begun to integrate biological structure from experimental data into ANNs, including circuit connectivity and single-unit dynamics. In the fly, where connectomic data is especially complete [Dorkenwald et al., 2024, Scheffer et al., 2020], this approach has yielded models that can reproduce and predict neural activity with cell-type specificity, while maintaining the trainability and abstraction of ANNs [Lappalainen et al., 2024, Duan et al., 2025]. In mammals, this approach is challenging because relevant neuronal and circuit information must be integrated from diverse sources, but a long history of anatomical, physiological, and genetic studies has characterized key circuits with sufficient resolution to motivate structured neural models.

In previous work on Neural Circuit Architectural Priors (NCAP), Bhattasali et al. [2024] showed that mammalian spinal cord circuits can constrain a biologically grounded neural architecture with cell-type-specific connectivity to control quadruped locomotion. The architecture introduced stronger inductive biases than standard ANNs, leading to efficient training and robust gaits in addition to closer mechanistic alignment with biology. However, that study used a simplified quadruped body with 12 actuated degrees of freedom, designed to isolate architectural principles rather than match biological reality. Scaling to biorealistic bodies matters for neuroscience, as richer bodies enable direct comparison to animal data and more convincing validation of circuit hypotheses. Yet achieving this scaling is far from straightforward. Biorealistic bodies are inherently harder to control, with higher-dimensional action spaces, more complex dynamics, and narrower stability margins. They also raise new questions about the neuromechanical interface—the way in which neural activity maps to body actuation, which can include joint torques, joint positions, or muscle activations. Finally, these bodies are costly to simulate and train, slowing hypothesis testing. How can we develop embodied simulations that enable the study of increasingly complex bodies and neural circuits?

In this work, we address the challenge through coordinated advances in the simulated body, neural architecture, and training infrastructure. We introduce *Rat v2*, a biomechanical rat body with 38 actuated degrees of freedom based on Merel et al. [2019] that improves anatomical fidelity while serving as a more efficient and flexible tool through GPU-accelerated simulation and a modular environment builder. To control this body, we develop *Rat NCAP*, a biologically grounded neural architecture for locomotion that scales mammalian spinal architectural priors from Bhattasali et al. [2024] to the rat by incorporating more complete neural connectivity and a novel flexor-extensor neuromechanical interface that enables more muscle-like control. To optimize circuit parameters, we develop GPU-accelerated simulation and training infrastructure that makes evolutionary optimization in this body efficient. Together, these components enable the study of increasingly complex bodies and neural circuits within embodied simulations at an intermediate level of abstraction.

The trained *Rat NCAP* produces locomotion at various speeds within a single model and transitions naturalistically between gaits. Importantly, the model produces naturalistic behavior without imitation learning, demonstrating the power of architectural priors to guide training. The trained model produces locomotion statistics, kinematics, and cell-type-specific neural activity consistent with experimental data. Systematic analyses reveal that the neuromechanical interface substantially affects trainability and gait quality. Finally, *in silico* cell-type ablations reproduce known gait phenotypes and predict gait changes under graded perturbations. Together, these findings support the view that embodied simulations at an intermediate level of abstraction can simultaneously reproduce complex behavior and yield mechanistic insights into the neural circuits that generate it.

The key contributions of this work are:

1. **Develop the biologically grounded *Rat NCAP* controller from locomotion circuits.**
2. **Validate trained behavior and neural activity against experimental data on rat locomotion.**
3. **Evaluate how neuromechanical interface affects trainability and gait quality.**
4. **Probe cell-type contributions and generate testable predictions via *in silico* ablations.**
5. **Introduce the *Rat v2* body and open-source simulation and training infrastructure.**

Our open-source code and videos are available at: https://rat-ncap.github.io/

2 Related Work

Biomechanical bodies Frameworks for embodied simulation now assemble diverse body models across organisms for comparative study [Arreguit et al., 2023, Zhang et al., 2025]. For rodent models specifically (i.e. mouse and rat), two prior bodies bracket the relevant abstraction space. At the detailed end, Ramalingasetty et al. [2024] developed a mouse neuromusculoskeletal body with Hill-type muscles anchored to measured attachment points, afferent feedback pathways (muscle spindle Ia and II, Golgi tendon Ib, and cutaneous afferents), and low-level reflex circuits. This level of detail enables fine-grained physiological comparisons while making optimization more demanding: their work produced a distribution of fixed-speed trotting solutions, while continuous speed modulation and gait transitions within a single model remained open. At the abstract end, Merel et al. [2019] developed a rat body with anatomically accurate proportions and mass distributions, actuated through joint positions without explicit muscles, and paired with an LSTM controller to support visually guided behaviors including gap crossing and maze foraging. Aldarondo et al. [2024] used this body with MLP and LSTM controllers trained through imitation learning on recorded rat kinematics, yielding neural representations predictive of brain activity. Both works demonstrated the expressive capacity of joint-level bodies with generic controllers rather than biologically grounded circuit structure, and naturalistic behavior was achieved by imitating recorded kinematics.

Our work builds on the Merel et al. [2019] rat body by improving its anatomical fidelity and extending it with a novel flexor-extensor neuromechanical interface that provides an intermediate level of abstraction between joint-level and muscle-level control, enabling systematic comparison of actuation abstractions. Meanwhile, we view detailed neuromusculoskeletal approaches and our approach as synergistic. An open question is whether the same locomotor phenomena can be modeled across different levels of abstraction, and progress at each level can inform the other.

Neural circuit models Locomotion is a useful testbed for biologically grounded control models because the underlying neural circuits are comparatively well characterized, and locomotion requires continuous interactions between neural circuits, the body, and the environment. There is a long history of biologically inspired locomotion controllers based on central pattern generators (CPGs) [Ijspeert, 2008, Yu et al., 2014]. This broad family includes stance-swing transition rules [Ekeberg and Pearson, 2005], parameterized trajectories and footfall templates [Iscen et al., 2019], learned oscillator policies [Bellegarda and Ijspeert, 2022, Shafiee et al., 2023], and reduced circuit motifs [Lodi et al., 2018]. Most are biologically inspired rather than tightly constrained in cell types and connectivity.

Both Ramalingasetty et al. [2024] and Bhattasali et al. [2024] built on cell-type-specific spinal circuit models of mammalian locomotion [Danner et al., 2016, 2017]. A key difference between them is neuronal abstraction: the former used biophysical neurons with bursting dynamics arising from ionic mechanisms of a persistent sodium current, whereas the latter used simplified neurons with bursting dynamics arising from an abstract relaxation oscillator [Bhattasali et al., 2026]. Also, the former specified more circuitry from biological data, while the latter learned certain modules.

Our work builds on Bhattasali et al. [2024] with substantial architectural and algorithmic changes that we found were crucial for scaling up to our biorealistic body. On the architectural side, major additions include a 2-layer afferent feedback module that learns sensory transformations and provides both resetting and non-resetting feedback, and a 3-layer pattern formation module that supports multiple stance/swing subphases, local reflexes, muscle synergies, and trunk actuation. On the algorithmic side, we rebuild the simulation and training infrastructure from the ground up to support GPU acceleration and modular components for biomechanical and neural circuit models.

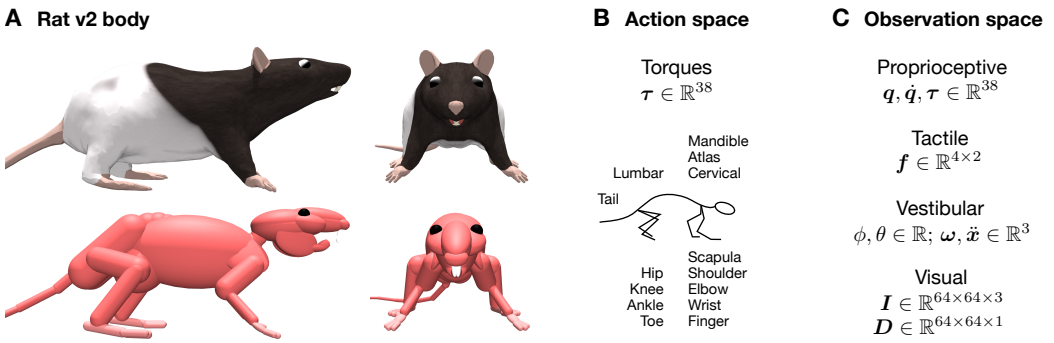


Figure 1 | **Biomechanical body.** (**A**) *Rat v2* has anatomically realistic kinematics, collision geometry, and rendering skins. (**B**) Actions are torque commands $\boldsymbol{\tau}$ for 38 actuated DOFs. (**C**) Observations are proprioceptive positions $\boldsymbol{q}$, velocities $\dot{\boldsymbol{q}}$, and applied torques $\boldsymbol{\tau}$; tactile contact forces $\boldsymbol{f}$; vestibular roll ϕ, pitch θ, angular velocity $\boldsymbol{\omega}$, and linear acceleration $\ddot{\boldsymbol{x}}$; visual RGB image $\boldsymbol{I}$ and depth map $\boldsymbol{D}$.

3 Methods

3.1 Biomechanical body

We build *Rat v2* (Figure 1A) from the biomechanical rat body of Merel et al. [2019], constructed through anatomical measurements of Long-Evans rats (*Rattus norvegicus*) and implemented in the MuJoCo physics engine [Todorov et al., 2012]. We refine the body to improve biological fidelity, extend it with biologically realistic skins to facilitate qualitative behavior evaluation, and integrate it into a new JAX-based environment framework to substantially increase efficiency and configurability.

Actions $\boldsymbol{a}_t$ are torque commands $\boldsymbol{\tau}$ for 38 actuated degrees of freedom (DOFs) spanning the head, cervical spine, forelimbs, lumbar spine, hindlimbs, and tail (Figure 1B). The cervical, lumbar, and tail DOFs use tendons to co-actuate multiple vertebral joints. Switching the original model from position to torque actuators enables modularly evaluating neuromechanical interfaces (Section 3.3).

Observations $\boldsymbol{o}_t$ have 4 modalities (Figure 1C). Proprioceptive observations provide positions $\boldsymbol{q}$, velocities $\dot{\boldsymbol{q}}$, and applied torques $\boldsymbol{\tau}$ for each actuated DOF. Tactile observations provide contact forces $\boldsymbol{f}$ at the forelimb finger and palm or hindlimb toe and heel. Vestibular observations provide head roll ϕ, pitch θ, angular velocity $\boldsymbol{\omega}$, and linear acceleration $\ddot{\boldsymbol{x}}$. Visual observations provide an RGB image $\boldsymbol{I}$ and depth map $\boldsymbol{D}$ from a head-mounted camera.

We refine the body specification to improve its ability to reproduce biological behavior, including changes in joint ranges/axes, actuator forces, collision geometry, sensor coverage, default poses, and naming. For a list of changes and their biological significance, see Appendix A.1.

We design biologically realistic rendering skins in 3 coat patterns: gray, black-hooded, and cinnamon. Skins do not affect physics simulation, but they facilitate qualitative visual comparison with experimental observations. For images and the creation pipeline, see Appendix A.2.

We build an environment framework in JAX [Bradbury et al., 2018] that yields orders-of-magnitude speedups in parallel simulation, while also enabling modular configuration of observation and action spaces, reward functions, and task logic. This framework enables large-scale embodied simulation for the neuroscientific community, and we are releasing a separate whitepaper along with the open-source code. For a software architecture overview, see Appendix A.3.

3.2 Neural circuit model

We develop *Rat NCAP* (Figure 2) by extending the neural circuit architectural prior from Bhattasali et al. [2024], which builds on spinal circuit models from Danner et al. [2017], Ausborn et al. [2019], Kim et al. [2022]. Scaling to the rat body requires substantial architectural advances as a rat's limb geometry is more complex than a quadruped robot's and requires producing phase-offset joint motions. Moreover, passive stiffness is not negligible, and trunk joints lie outside the previous 4-limb framework. We address these challenges by introducing richer RG half-center dynamics, a 2-layer AF module, and a 3-layer PF module with delays, reflexes, and synergies.

The architecture uses 2 neuron types: Basic and Bursting. The Basic neuron is a standard neuron with internal voltage v that evolves through leaky-integrator dynamics and produces rate-coded activity with a threshold activation function $f(v) = \text{clip}(v, 0, 1)$. The Bursting neuron adds an adaptation variable a that interacts with v to produce intrinsic bursting that drives half-center oscillations even without rhythmic input. For equations and hyperparameters, see Appendix B.1.

The architecture is composed of 4 modules: Rhythm Generation (RG) coordinates gaits between limbs, Brainstem Command (BC) provides speed-dependent control of step frequency and gait, Afferent Feedback (AF) transforms sensory observations into half-center and reflex inputs, and Pattern Formation (PF) transforms half-center outputs to motor actions.

$$\begin{aligned} c_t &= \text{BrainstemCommand}(v_t^{\text{target}}) \\ (\boldsymbol{o}_t^{\text{reset}}, \boldsymbol{o}_t^{\text{non-reset}}) &= \text{AfferentFeedback}(\boldsymbol{o}_t^{\text{interface}}) \\ (\boldsymbol{y}_t, \boldsymbol{h}_{t+1}) &= \text{RhythmGeneration}(\boldsymbol{o}_t^{\text{reset}}, \boldsymbol{h}_t, c_t) \\ \boldsymbol{a}_t^{\text{interface}} &= \text{PatternFormation}(\boldsymbol{y}_t, \boldsymbol{o}_t^{\text{non-reset}}, c_t) \end{aligned}$$

with neuromechanical interface-specific observations $\boldsymbol{o}_t^{\text{interface}}$ and actions $\boldsymbol{a}_t^{\text{interface}}$ (Section 3.3), task-specific target speed v_t^{target}, brainstem command c_t, resetting feedback $\boldsymbol{o}_t^{\text{reset}}$, non-resetting feedback $\boldsymbol{o}_t^{\text{non-reset}}$, central rhythm $\boldsymbol{y}_t$, and neuron hidden states $\boldsymbol{h}_t$. Rat NCAP controls 31 of 38 active DOFs for locomotion (the limbs and trunk), while passive DOFs are held near default poses. Parameters are indexed by body part $p \in \{\text{FL, FR, HL, HR, T}\}$ and shared to exploit symmetries.

Rhythm generation The RG module coordinates limb rhythms to produce speed-dependent gaits. Each limb is governed by a bursting flexor unit and basic extensor unit that are reciprocally inhibiting. These limb half-center oscillators are connected through cross, side, and diagonal commissural interneurons, with excitatory synapses promoting synchronization and inhibitory synapses promoting alternation [Bhattasali et al., 2024]. To generate larger and stronger steps at higher speeds, we introduce amplitude modulation of half-center activity by BC input in addition to existing frequency modulation, which reflects recruitment of larger neural populations [Shevtsova et al., 2015] that was not modeled in Danner et al. [2017]. We tune burst durations to match rat timescales and freeze synaptic weights during training. For the RG circuit diagram and tuning protocol, see Appendix B.2.

Brainstem command The BC module produces brainstem command $c_t \in [0, 1]$ to modulate RG half-center frequency/amplitude and interneuron activity. It uses a linear mapping with hyperparameters $w^{\text{BC}}, b^{\text{BC}}$ to align the command range with the target speed range:

$$c_t = f(w^{\text{BC}}\, v_t^{\text{target}} + b^{\text{BC}})$$

Afferent feedback The AF module produces sensory feedback for RG and PF through 2 layers. The sensor layer applies per-element gains and thresholds, modeling tunable sensory receptors:

$$\boldsymbol{o}_t^{\text{sensor},p} = f(\boldsymbol{w}^{\text{sensor},p} \circ \boldsymbol{o}_t^{\text{interface},p} + \boldsymbol{b}^{\text{sensor},p}), \qquad \boldsymbol{w}^{\text{sensor},p} \geq 0,\ \boldsymbol{b}^{\text{sensor},p} \leq 0$$

The reset layer produces signed feedback $\boldsymbol{o}_t^{\text{reset},p} \in \mathbb{R}^{\{\text{flx,ext}\}}$ that resets or entrains the RG rhythm:

$$\boldsymbol{o}_t^{\text{reset},p} = \boldsymbol{W}^{\text{reset},p}\, \boldsymbol{o}_t^{\text{sensor},p} + \boldsymbol{b}^{\text{reset},p}$$

The sensor outputs also pass directly to PF as non-resetting feedback $\boldsymbol{o}_t^{\text{non-reset},p} = \boldsymbol{o}_t^{\text{sensor},p}$, which triggers reflexes without affecting the rhythm. For AF details, see Appendix B.4.

Pattern formation The PF module produces interface-specific actions from RG through 3 layers. The delay layer produces copies $k \in \{\text{early, late}\}$ of each half-center output $h \in \{\text{flx, ext}\}$, which have distinct time constants $T_k \geq 0$ and biases $b_k \leq 0$ that set their delays:

$$z_t^{h,k,p} = \text{Basic}(y_t^{h,p} + b_k;\, T_k)$$

The synergy layer selects synergies by central rhythm, sensory feedback, and brainstem command:

$$\boldsymbol{s}_t^p = f(\boldsymbol{W}^{\text{syn},p}\, \boldsymbol{z}_t^p + \boldsymbol{W}^{\text{reflex},p}\, \boldsymbol{o}_t^{\text{non-reset},p} + \boldsymbol{w}^{\text{BC},p}\, c_t)$$

The motor layer maps synergy activations to interface-specific actions:

$$\boldsymbol{a}_t^{\text{interface},p} = f(\boldsymbol{W}^{\text{motor},p} \boldsymbol{s}_t^p + \boldsymbol{b}^{\text{motor},p})$$

The motor weights $\boldsymbol{W}^{\text{motor},p}$ are sign-constrained following biological priors, and biases $\boldsymbol{b}^{\text{motor},p}$ are initialized to produce the default standing pose. For PF details, see Appendix B.5.

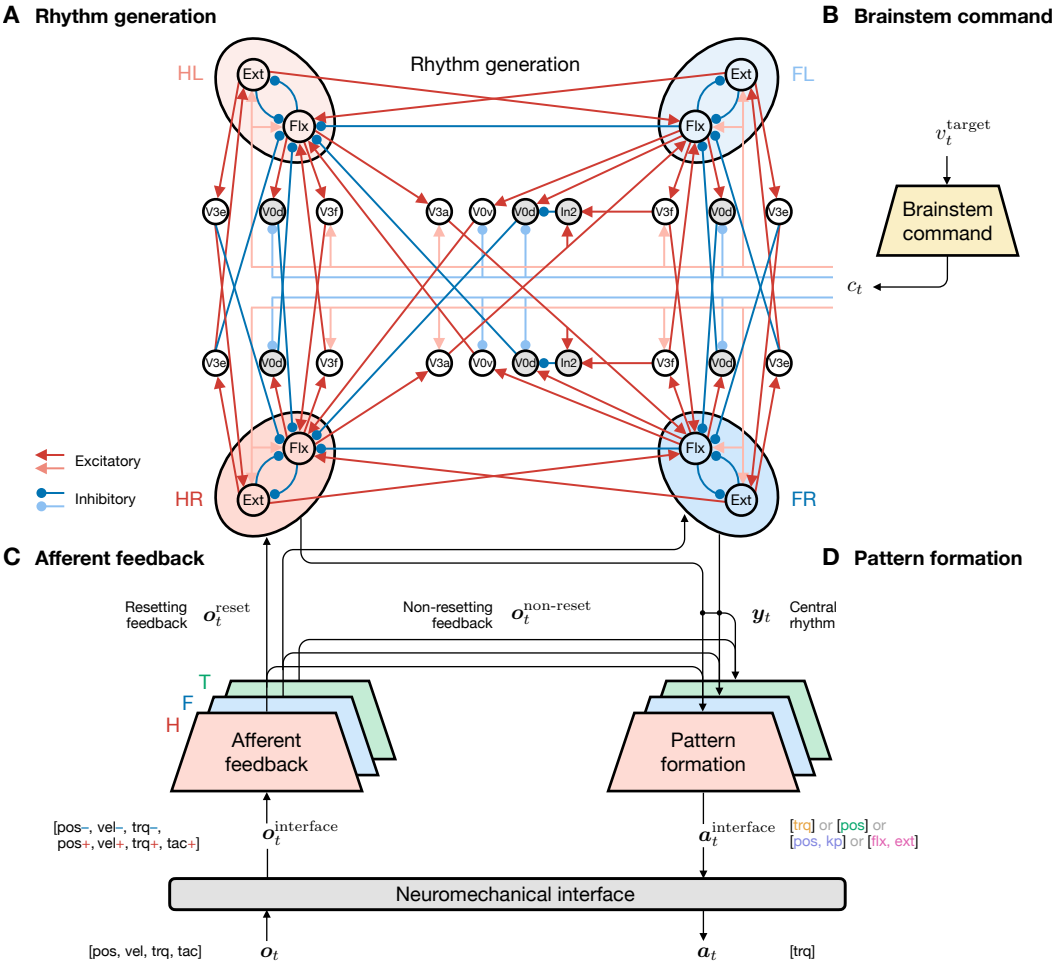


Figure 2 | **Neural circuit model.** (**A**) RG organizes limbs into reciprocally inhibiting flexor/extensor half-centers, connected through cross, side, and diagonal commissural interneurons. (**B**) BC converts target speed v_t^{target} into brainstem command c_t, modulating half-center frequency/amplitude and interneuron activity in RG to alter gait speed and pattern. (**C**) AF transforms interface observations $\boldsymbol{o}_t^{\text{interface}}$ into resetting feedback $\boldsymbol{o}_t^{\text{reset}}$ and non-resetting feedback $\boldsymbol{o}_t^{\text{non-reset}}$ passed to RG and PF, respectively. (**D**) PF combines central rhythm $\boldsymbol{y}_t$ and non-resetting feedback $\boldsymbol{o}_t^{\text{non-reset}}$ to produce interface actions $\boldsymbol{a}_t^{\text{interface}}$ through delay, synergy, and motor layers. Modules are indexed as forelimb (F), hindlimb (H), or trunk (T), with parameter sharing where possible to exploit body symmetries.

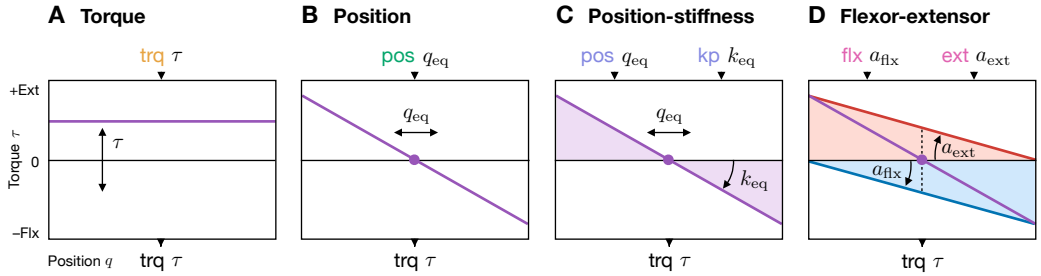


Figure 3 | **Neuromechanical interfaces.** Control laws show applied torque τ as a function of the current position q, with control variables reshaping the torque profile. (**A**) Torque control uses circuit outputs directly as torque τ and does not include spring-like negative feedback. (**B**) Position control only adjusts equilibrium position q_{eq} while using a fixed stiffness. (**C**) Position-stiffness control explicitly adjusts both equilibrium position q_{eq} and stiffness k_{eq}. (**D**) Flexor-extensor control uses antagonistic muscle activations $a_{\text{flx}}, a_{\text{ext}}$ to implicitly adjust both equilibrium position and stiffness.

3.3 Neuromechanical interfaces

We design 4 neuromechanical interfaces to explore how biorealistic circuits may benefit from biorealistic actuation. All interfaces share an observation transform from body observations $\boldsymbol{o}_t$ to interface observations $\boldsymbol{o}_t^{\text{interface}}$, rectifying proprioceptive and tactile signals into positive and negative components like afferents. The interfaces differ in the action transform from interface actions $\boldsymbol{a}_t^{\text{interface}}$ to body actions $\boldsymbol{a}_t$, applying a per-DOF control law with control variables output by the circuit.

Torque control (Figure 3A) sets torques directly: $\tau = \tau$

Position control (Figure 3B) adjusts only equilibrium position with a fixed stiffness: $\tau = k_{\text{eq}} \cdot (q_{\text{eq}} - q)$

Position-stiffness control (Figure 3C) explicitly adjusts both equilibrium position and stiffness, which biological limbs regulate to stabilize posture and movement [Shadmehr, 1993]: $\tau = k_{\text{eq}} \cdot (q_{\text{eq}} - q)$

Flexor-extensor control (Figure 3D) uses antagonistic muscles to implicitly adjust both equilibrium position and stiffness: differential activation shifts the equilibrium, and co-activation scales stiffness. For the derivation and biological motivation, see Appendix C. The circuit controls flexor and extensor muscle activations: $\tau = a_{\text{ext}} \cdot k_{\text{ext}} \cdot (q_{\text{ext}} - q) \; + \; a_{\text{flx}} \cdot k_{\text{flx}} \cdot (q_{\text{flx}} - q)$

3.4 Training setup

We jointly train a single model on flat-ground locomotion with 6 subtasks, each tracking a fixed target speed ($v^{\text{target}} \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$ m/s), using velocity-tracking rewards and a pose reward for standing. We use Augmented Random Search (ARS) [Mania et al., 2018] to optimize AF and PF parameters, which avoids backpropagation through neural dynamics and the simulator and is highly parallelizable on a GPU. For task, reward, and algorithm details, see Appendix D.

4 Results

4.1 Rat NCAP produces variable-speed locomotion and gait transitions

Rat NCAP learns to stand, walk, trot, and bound in a single model, with naturalistic, speed-dependent gaits (Figure 4A, Video 1). Although trained only at fixed speeds, Rat NCAP transitions from walk to trot to bound as brainstem command increases mid-episode (Figure 4B, Video 2). Prior work in this rat relied on imitation learning for naturalistic behavior [Aldarondo et al., 2024, Zhang et al., 2025] or produced less naturalistic movement without it [Merel et al., 2019]. Rat NCAP uses only simple rewards, suggesting that its biologically grounded architecture provides an effective inductive bias.

Comparisons with an MLP and Quadruped NCAP [Bhattasali et al., 2024] support this interpretation. Rat NCAP achieves return comparable to or higher than MLPs trained with deep reinforcement learning or the same ARS algorithm (Figure F.1), yet both MLPs produce unnatural poses and gaits (Figure F.2). Rat NCAP also generalizes at least as well to unseen bumpy terrain (Figure F.3). Finally, it achieves dramatically higher return than Quadruped NCAP, demonstrating that its architectural additions are critical for scaling to the more complex, biorealistic body (Figure F.4).

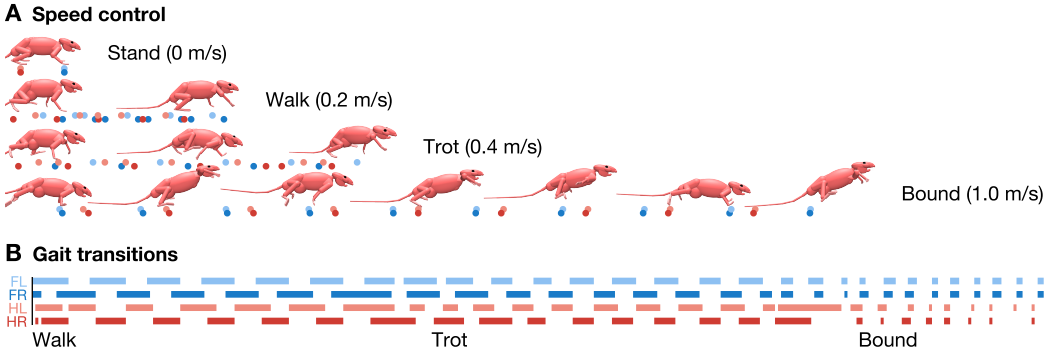


Figure 4 | **Locomotion behavior.** (**A**) Timelapse frames and footsteps over 2 s at each target speed. (**B**) Footfall pattern as brainstem command increases within an episode to produce gait transitions.

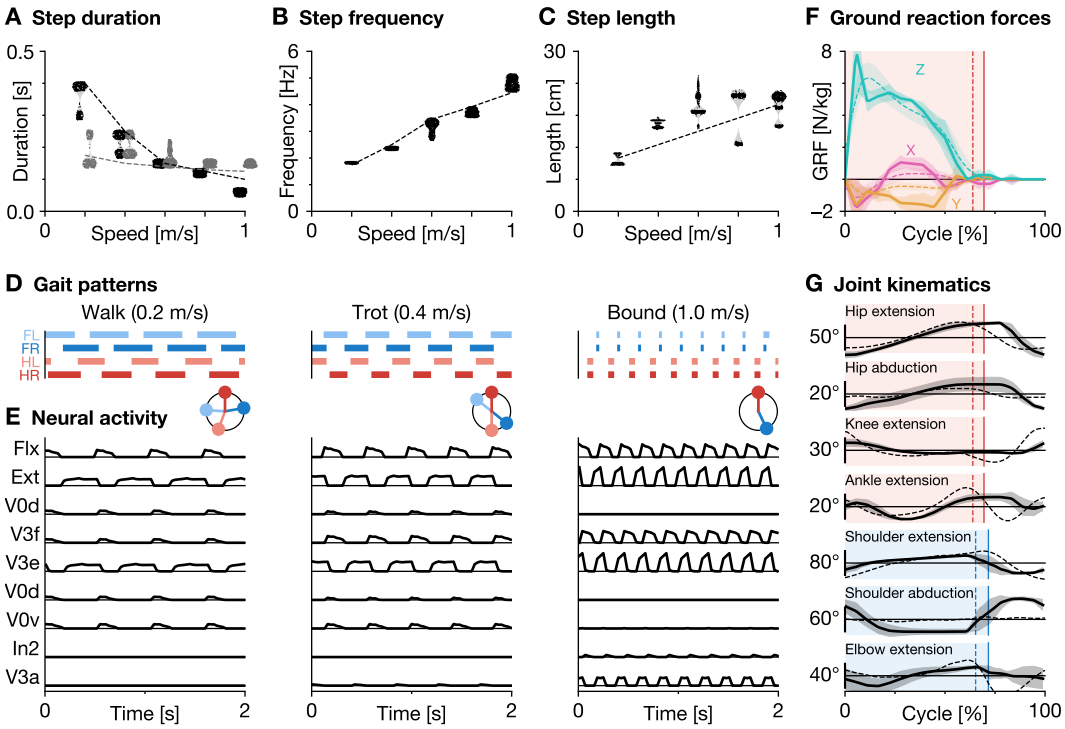


Figure 5 | **Experimental validation.** (**A–C**) Step statistics across target speeds: stance (black) and swing (gray) durations, step frequency, and step length. Speed increases mainly through shorter stance, as in rats. (**D**) Footfall patterns and interlimb phases for representative walk, trot, and bound. Gaits shift from left-right alternation to synchrony as speed increases. (**E**) RG cell-type activity for the same gaits. V0d and V0v activity decline and V3 activity rises with speed. (**F**) Ground reaction forces of the HR limb: forward X, lateral Y, and vertical Z, with stance shaded and liftoff marked. Forward force shows braking then propulsion. (**G**) Joint kinematics of the HR and FR limbs. Model data over 3 seeds, mean ± SD. Rat data (dashed) from (**A**) Gillis and Biewener [2001], (**B**, **C**) Clarke and Parker [1986], (**F**, **G** hind) Dienes et al. [2022], and (**G** fore) Bonnan et al. [2016].

4.2 Trained behavior matches experimental data on rat locomotion

Producing forward locomotion is necessary but not sufficient for a biologically grounded model, so we compare the trained behavior and neural activity with experimental data. Rat NCAP modulates speed as rats do: stance shortens while swing remains roughly constant (Figure 5A), raising step frequency within the experimentally reported range (Figure 5B). This asymmetry is consistent with flexor-dominated rhythm generation, in which flexor bursts driving swing remain brief while extensor bursts driving stance shorten with speed (Figure 5E). Step length matches reported values at the slowest and fastest speeds but exceeds them at intermediate speeds (Figure 5C).

Gait patterns change with speed, as interlimb phases shift from alternation in walk and trot to synchrony in bound (Figure 5D). RG activity shifts accordingly (Figure 5E): from walk to trot, V0d activity declines and V0v activity sustains diagonal alternation; in bound, both vanish as V3 activity promotes left-right synchrony, matching data-constrained biophysical models [Zhang et al., 2022].

Limb kinetics and kinematics are also consistent with experimental data. Vertical force peaks early in stance, forward force shows braking then propulsion (Figure 5F), and joint angles follow experimental profiles, including phase offsets between proximal and distal joints (Figure 5G). Agreement is closest during stance; during swing, the knee and elbow flex less than in rats, which may reflect limitations of the circuit or body model. Constraining non-planar limb DOFs substantially lowers return, whereas constraining lumbar DOFs has less effect, likely due to compensation (Figure G.1), though they are recruited in a speed-dependent manner when available (Figure G.2). Together, these results indicate that biologically grounded architecture shapes features of locomotion beyond the training rewards.

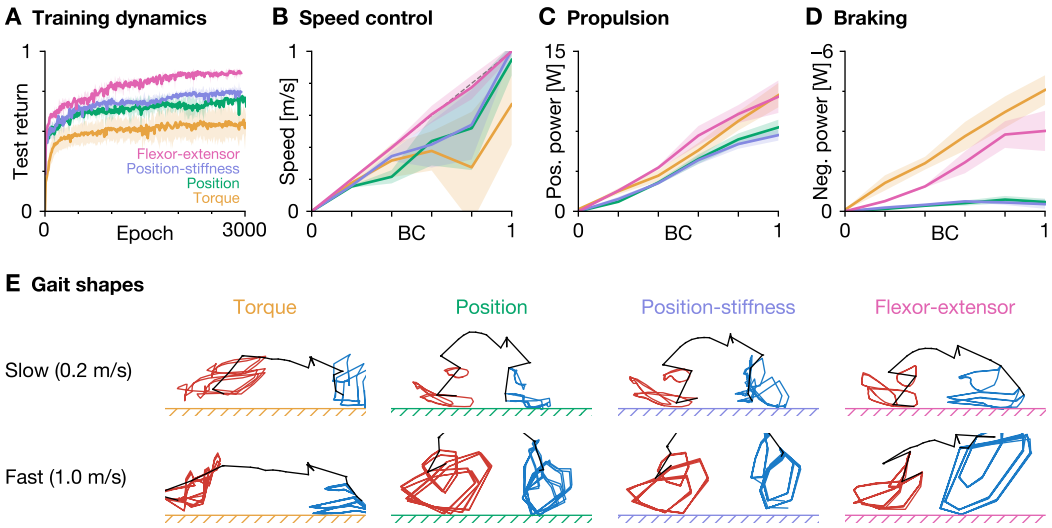


Figure 6 | **Neuromechanical interface comparison.** (**A**) Test return over training epochs for all 4 interfaces. Flexor-extensor achieves the highest return, and torque the lowest. (**B**) Speed as brainstem command increases from 0 to 1. Flexor-extensor tracks target speed most closely, while torque fails at high commands. (**C, D**) Positive (propulsion) and negative (braking) mechanical power, summed over actuators and averaged over timesteps. Flexor-extensor matches torque in propulsion and absorbs more energy than position-based interfaces, consistent with spring-like compliance. (**E**) Sagittal trajectories of the HR and FR limbs at slow and fast speeds. Flexor-extensor produces rounded trajectories, while position produces sharp arcs. Model data over 3 seeds, mean ± SD.

4.3 Neuromechanical interfaces strongly influence trainability and gait quality

We test whether a biologically grounded neural circuit benefits from biorealistic actuation by training Rat NCAP with each neuromechanical interface. Flexor-extensor control achieves the highest test return and tracks target speed most accurately, followed by position-stiffness and position control, while torque control achieves the lowest return with this architecture (Figure 6A, B).

Flexor-extensor control also produces qualitatively different gaits from the common choice of position control. It generates greater propulsion (Figure 6C) and absorbs substantially more energy (Figure 6D), as its antagonistic springs store and return energy analogous to tendon compliance. Correspondingly, its limb trajectories are rounded, with extended stance as the paw follows the ground, whereas position control produces sharp arcs with brief ground contact (Figure 6E, Video 3). Torque control absorbs the most energy, though its disorganized trajectories suggest uncoordinated braking rather than compliance. The benefits of flexor-extensor control are consistent with the view that animals act through muscles whose intrinsic mechanics provide stabilizing feedback rather than commanding joint torques [Dickinson et al., 2000]. Embodied models of motor circuits may thus benefit from muscle-like actuation even in this simple form, at little added complexity over position control.

4.4 Cell-type ablations reproduce known phenotypes and produce testable predictions

A central advantage of biologically grounded circuit models is the ability to ablate identified cell types *in silico*. Since training does not include RG ablations and RG remains fixed, these experiments test whether phenotypes of the non-embodied RG model persist when it drives a body through learned AF and PF modules. As in other biologically grounded and connectome-constrained models, RG synaptic strengths are not identifiable from experimental data, so we first test sensitivity to them. Under sign-preserving multiplicative noise, trot degrades only gradually and remains the dominant gait even at $\sigma = 0.5$ (Figure 7A), indicating that the gaits do not depend on finely tuned weights.

In mice, ablating V0v neurons eliminates trot, and additionally ablating V0d neurons leaves bound as the only coordinated gait [Bellardita and Kiehn, 2015]. Rat NCAP reproduces this combined phenotype: scaling V0v and V0d activity to zero replaces trot with bound (Figure 7B, Video 4). Removing V0-mediated alternation likely leaves V3-mediated synchrony unopposed, so the remaining

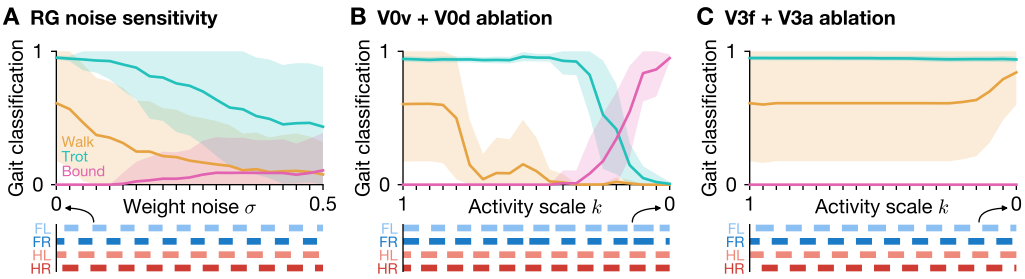


Figure 7 | **Cell-type ablations.** Fraction of step cycles classified as each gait by idealized interlimb phase ranges at 0.4 m/s, with footfalls for the indicated condition. (**A**) RG weights scaled by multiplicative Gaussian noise of standard deviation σ. Trot remains dominant at all noise levels. (**B**) V0v and V0d activity scaled by factor k. Trot switches abruptly to bound, reproducing the knockout phenotype at $k = 0$. (**C**) V3f and V3a activity scaled by factor k. Trot persists, so not all populations are equally critical. Model data over 3 seeds and 30 trials, mean $\pm$ SD.

commissural drive produces bound. Graded scaling further predicts a threshold-like transition, as trot persists until $k \approx 0.35$ and then gives way to bound by $k \approx 0.1$, consistent with abrupt gait changes in biophysical models as commissural drive varies [Danner et al., 2017]. Such graded reductions could be tested experimentally by partially reducing cell counts, intrinsic excitability, or synaptic efficacy in these populations. In contrast, scaling V3f and V3a activity to zero leaves trot intact (Figure 7C), so not all commissural populations are equally critical for trot. Together, these results show that RG cell-type phenotypes persist in an embodied setting and yield experimentally testable predictions.

5 Discussion

This work asked how to develop embodied simulations that enable the study of increasingly complex bodies and neural circuits. Our results support an intermediate level of abstraction: a biorealistic rat body controlled by a biologically grounded architecture and trained with simple rewards produces naturalistic locomotion and enables circuit-level analyses. A cell-type-specific spinal circuit model, previously studied in non-embodied settings, retains its gait-generating function when coupled to realistic body dynamics and closed-loop feedback. The resulting model supports validation against behavioral and neural data, reveals benefits of muscle-like actuation for circuit function, and predicts responses to perturbations not yet tested experimentally. This balance is practically useful for building embodied simulations and scientifically useful for evaluating circuit hypotheses in complex animals.

Our work has several limitations. First, behavior does not yet match rats in all respects, including step length at intermediate speeds and distal joint kinematics during swing. These discrepancies may arise from the circuit model, the neuromechanical interface, or the body, including the difficulty of representing compliance in rigid-body physics engines. Comparing the behavior of Rat NCAP with that of imitation-learned policies on the same body could separate these factors, since such policies approximate rat behavior within the kinematic limits of Rat v2. Second, circuit parameters are underconstrained: RG weights are not identifiable from available data, and the learned AF and PF modules find plausible rather than unique solutions. How much these connections vary across animals is unknown; if they vary little, future work could identify the tasks or rewards under which they reliably emerge, and if they vary widely, how developmental plasticity organizes them.

Our work can also be extended in several directions. Actuation could be refined by augmenting the flexor-extensor interface with force-velocity relationships and activation dynamics. Additional spinal and brainstem circuits could enable new behaviors, such as turning, postural adjustments, and non-periodic movements like rearing, jumping, and grooming. More complex environments with uneven terrain and perturbation forces could test robustness beyond flat-ground locomotion. A higher-level controller could sequence the resulting motor primitives to perform cognitive tasks such as navigation and foraging. Together, this work lays a foundation for embodied simulations that move from spinal locomotion toward naturalistic whole-body behavior in increasingly complex animals, while yielding mechanistic insights into the neural circuits that generate it.

Acknowledgments and Disclosure of Funding

Thanks to Charles Zhang and Josh Merel for assistance with the biomechanical body model. Thanks to Jonathan Arreguit, Elliot Abe, Eric Leonardis, Simon Danner, Shravan Ramalingasetty, Gizem Özdil, Pavel Tolmachev, and many others for insightful discussions and feedback.

This work was supported by the Fannie and John Hertz Foundation Fellowship (NXB); the Packard Fellowship for Science and Engineering (LP); the Institute of Information & Communications Technology Planning & Evaluation (IITP) with a grant by the Ministry of Science and ICT (MSIT) of the Republic of Korea in connection with the Global AI Frontier Lab International Collaborative Research Agreement (RS-2024-00469482, RS-2024-00509279) (LP); and the National Science Foundation AI Research Institute on Interaction for AI Assistants (ARIA) (No. 2433429) (JC, GWL).

Author Contributions

Conceptualization (NXB, JC, GWL). Investigation (NXB, JC). Software (NXB, JC). Supervision (GWL, LP). Writing: original draft (NXB, JC), review and editing (NXB, JC, GWL, LP).

References

D. Aldarondo, J. Merel, J. D. Marshall, L. Hasenclever, U. Klibaite, A. Gellis, Y. Tassa, G. Wayne, M. Botvinick, and B. P. Ölveczky. A virtual rodent predicts the structure of neural activity across behaviours. *Nature*, 632 (8025):594–602, Aug. 2024. doi: 10.1038/s41586-024-07633-4.

J. Arreguit, S. Tata Ramalingasetty, and A. Ijspeert. FARMS: Framework for animal and robot modeling and simulation, Sept. 2023.

J. Ausborn, N. A. Shevtsova, V. Caggiano, S. M. Danner, and I. A. Rybak. Computational modeling of brainstem circuits controlling locomotor frequency and gait. *eLife*, 8:e43587, Jan. 2019. doi: 10.7554/eLife.43587.

J. Ausborn, N. A. Shevtsova, and S. M. Danner. Computational modeling of spinal locomotor circuitry in the age of molecular genetics. *IJMS*, 22(13):6835, June 2021. doi: 10.3390/ijms22136835.

C. Bellardita and O. Kiehn. Phenotypic characterization of speed-associated gait changes in mice reveals modular organization of locomotor networks. *Current Biology*, 25(11):1426–1436, June 2015. doi: 10.1016/j.cub.2015.04.005.

G. Bellegarda and A. Ijspeert. CPG-RL: Learning central pattern generators for quadruped locomotion, Nov. 2022.

N. X. Bhattasali, V. Pattabiraman, L. Pinto, and G. W. Lindsay. Neural circuit architectural priors for quadruped locomotion, Oct. 2024.

N. X. Bhattasali, L. Pinto, and G. W. Lindsay. Simplified model of intrinsically bursting neurons, May 2026.

M. F. Bonnan, J. Shulman, R. Varadharajan, C. Gilbert, M. Wilkes, A. Horner, and E. Brainerd. Forelimb kinematics of rats using XROMM, with implications for small eutherians and their fossil relatives. *PLoS ONE*, 11(3):e0149377, Mar. 2016. doi: 10.1371/journal.pone.0149377.

J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: Composable transformations of Python+NumPy programs, 2018.

R. Cao and D. Yamins. Explanatory models in neuroscience, Part 1: Taking mechanistic abstraction seriously. *Cognitive Systems Research*, 87:101244, Sept. 2024. doi: 10.1016/j.cogsys.2024.101244.

K. A. Clarke and A. J. Parker. A quantitative study of normal locomotion in the rat. *Physiology & Behavior*, 38 (3):345–351, Jan. 1986. doi: 10.1016/0031-9384(86)90105-8.

S. M. Danner, S. D. Wilshin, N. A. Shevtsova, and I. A. Rybak. Central control of interlimb coordination and speed-dependent gait expression in quadrupeds. *The Journal of Physiology*, 594(23):6947–6967, Dec. 2016. doi: 10.1113/JP272787.

S. M. Danner, N. A. Shevtsova, A. Frigon, and I. A. Rybak. Computational modeling of spinal circuits controlling limb coordination and gaits in quadrupeds. *eLife*, 6:e31050, Nov. 2017. doi: 10.7554/eLife.31050.

M. H. Dickinson, C. T. Farley, R. J. Full, M. A. R. Koehl, R. Kram, and S. Lehman. How animals move: An integrative view. *Science*, 288(5463):100–106, Apr. 2000. doi: 10.1126/science.288.5463.100.

J. Dienes, B. Hicks, C. Slater, K. D. Janson, G. J. Christ, and S. D. Russell. Comprehensive dynamic and kinematic analysis of the rodent hindlimb during over ground walking. *Sci Rep*, 12(1):19725, Nov. 2022. doi: 10.1038/s41598-022-20288-3.

S. Dorkenwald, A. Matsliah, A. R. Sterling, P. Schlegel, S.-c. Yu, C. E. McKellar, A. Lin, M. Costa, K. Eichler, Y. Yin, W. Silversmith, C. Schneider-Mizell, C. S. Jordan, D. Brittain, A. Halageri, K. Kuehner, O. Ogedengbe, R. Morey, J. Gager, K. Kruk, E. Perlman, R. Yang, D. Deutsch, D. Bland, M. Sorek, R. Lu, T. Macrina, K. Lee, J. A. Bae, S. Mu, B. Nehoran, E. Mitchell, S. Popovych, J. Wu, Z. Jia, M. A. Castro, N. Kemnitz, D. Ih, A. S. Bates, N. Eckstein, J. Funke, F. Collman, D. D. Bock, G. S. X. E. Jefferis, H. S. Seung, M. Murthy, The FlyWire Consortium, Z. Lenizo, A. T. Burke, K. P. Willie, N. Serafetinidis, N. Hadjerol, R. Willie, B. Silverman, J. A. Ocho, J. Bañez, R. A. Candilada, A. Kristiansen, N. Panes, A. Yadav, R. Tancontian, S. Serona, J. I. Dolorosa, K. J. Vinson, D. Garner, R. Salem, A. Dagohoy, J. Skelton, M. Lopez, L. S. Capdevila, G. Badalamente, T. Stocks, A. Pandey, D. J. Akiatan, J. Hebditch, C. David, D. Sapkal, S. M. Monungolh, V. Sane, M. L. Pielago, M. Albero, J. Laude, M. Dos Santos, Z. Vohra, K. Wang, A. M. Gogo, E. Kind, A. J. Mandahay, C. Martinez, J. D. Asis, C. Nair, D. Patel, M. Manaytay, I. F. M. Tamimi, C. A. Lim, P. L. Ampo, M. D. Pantujan, A. Javier, D. Bautista, R. Rana, J. Seguido, B. Parmar, J. C. Saguimpa, M. Moore, M. W. Pleijzier, M. Larson, J. Hsu, I. Joshi, D. Kakadiya, A. Braun, C. Pilapil, M. Gkantia, K. Parmar, Q. Vanderbeck, I. Salgarella, C. Dunne, E. Munnelly, C. H. Kang, L. Lörsch, J. Lee, L. Kmecova, G. Sancer, C. Baker, J. Joroff, S. Calle, Y. Patel, O. Sato, S. Fang, J. Salocot, F. Salman, S. Molina-Obando, P. Brooks, M. Bui, M. Lichtenberger, E. Tamboboy, K. Molloy, A. E. Santana-Cruz, A. Hernandez, S. Yu, A. Diwan, M. Patel, T. R. Aiken, S. Morejohn, S. Koskela, T. Yang, D. Lehmann, J. Chojetzki, S. Sisodiya, S. Koolman, P. K. Shiu, S. Cho, A. Bast, B. Reicher, M. Blanquart, L. Houghton, H. Choi, M. Ioannidou, M. Collie, J. Eckhardt, B. Gorko, L. Guo, Z. Zheng, A. Poh, M. Lin, I. Taisz, W. Murfin, Á. S. Díez, N. Reinhard, P. Gibb, N. Patel, S. Kumar, M. Yun, M. Wang, D. Jones, L. Encarnacion-Rivera, A. Oswald, A. Jadia, M. Erginkaya, N. Drummond, L. Walter, I. Tastekin, X. Zhong, Y. Mabuchi, F. J. Figueroa Santiago, U. Verma, N. Byrne, E. Kunze, T. Crahan, R. Margossian, H. Kim, I. Georgiev, F. Szorenyi, A. Adachi, B. Bargeron, T. Stürner, D. Demarest, B. Gür, A. N. Becker, R. Turnbull, A. Morren, A. Sandoval, A. Moreno-Sanchez, D. A. Pacheco, E. Samara, H. Croke, A. Thomson, C. Laughland, S. B. Dutta, P. G. A. De Antón, B. Huang, P. Pujols, I. Haber, A. González-Segarra, D. T. Choe, V. Lukyanova, N. Mancini, Z. Liu, T. Okubo, M. A. Flynn, G. Vitelli, M. Laturney, F. Li, S. Cao, C. Manyari-Diaz, H. Yim, A. Duc Le, K. Maier, S. Yu, Y. Nam, D. Bąba, A. Abusaif, A. Francis, J. Gayk, S. S. Huntress, R. Barajas, M. Kim, X. Cui, G. R. Sterne, A. Li, K. Park, G. Dempsey, A. Mathew, J. Kim, T. Kim, G.-t. Wu, S. Dhawan, M. Brotas, C.-h. Zhang, S. Bailey, A. Del Toro, R. Yang, S. Gerhard, A. Champion, D. J. Anderson, R. Behnia, S. S. Bidaye, A. Borst, E. Chiappe, K. J. Colodner, A. Dacks, B. Dickson, D. Garcia, S. Hampel, V. Hartenstein, B. Hassan, C. Helfrich-Forster, W. Huetteroth, J. Kim, S. S. Kim, Y.-J. Kim, J. Y. Kwon, W.-C. Lee, G. A. Linneweber, G. Maimon, R. Mann, S. Noselli, M. Pankratz, L. Prieto-Godino, J. Read, M. Reiser, K. Von Reyn, C. Ribeiro, K. Scott, A. M. Seeds, M. Selcho, M. Silies, J. Simpson, S. Waddell, M. F. Wernet, R. I. Wilson, F. W. Wolf, Z. Yao, N. Yapici, and M. Zandawala. Neuronal wiring diagram of an adult brain. *Nature*, 634(8032):124–138, Oct. 2024. doi: 10.1038/s41586-024-07558-y.

S. Duan, L. L. Dong, and I. Fiete. From synapses to dynamics: Obtaining function from structure in a connectome constrained model of the head direction circuit, May 2025.

Ö. Ekeberg. A combined neuronal and mechanical model of fish swimming. *Biol. Cybern.*, 69(5-6):363–374, Sept. 1993. doi: 10.1007/BF01185408.

O. Ekeberg and K. Pearson. Computer simulation of stepping in the hind legs of the cat: An examination of mechanisms regulating the stance-to-swing transition. *Journal of Neurophysiology*, 94(6):4256–4268, Dec. 2005. doi: 10.1152/jn.00065.2005.

G. B. Gillis and A. A. Biewener. Hindlimb muscle function in relation to speed and gait: *In Vivo* patterns of strain and activation in a hip and knee extensor of the rat (*Rattus norvegicus*). *Journal of Experimental Biology*, 204(15):2717–2731, Aug. 2001. doi: 10.1242/jeb.204.15.2717.

A. J. Ijspeert. Central pattern generators for locomotion control in animals and robots: A review. *Neural Networks*, 21(4):642–653, May 2008. doi: 10.1016/j.neunet.2008.03.014.

A. J. Ijspeert, A. Crespi, D. Ryczko, and J.-M. Cabelguen. From swimming to walking with a salamander robot driven by a spinal cord model. *Science*, 315(5817):1416–1420, Mar. 2007. doi: 10.1126/science.1138353.

A. Iscen, K. Caluwaerts, J. Tan, T. Zhang, E. Coumans, V. Sindhwani, and V. Vanhoucke. Policies modulating trajectory generators, Oct. 2019.

P. Kidger and C. Garcia. Equinox: Neural networks in JAX via callable PyTrees and filtered transformations, Oct. 2021.

Y. Kim, S. Aoi, S. Fujiki, S. M. Danner, S. N. Markin, J. Ausborn, I. A. Rybak, D. Yanagihara, K. Senda, and K. Tsuchiya. Contribution of afferent feedback to adaptive hindlimb walking in cats: A neuromusculoskeletal modeling study. *Front. Bioeng. Biotechnol.*, 10:825149, Apr. 2022. doi: 10.3389/fbioe.2022.825149.

A. N. Klishko, J. Harnie, C. E. Hanson, S. M. Rahmati, I. A. Rybak, A. Frigon, and B. I. Prilutsky. Effects of spinal transection and locomotor speed on muscle synergies of the cat hindlimb. *J Physiol*, 603(10): 3061–3088, May 2025. doi: 10.1113/JP288089.

J. K. Lappalainen, F. D. Tschopp, S. Prakhya, M. McGill, A. Nern, K. Shinomiya, S.-y. Takemura, E. Gruntman, J. H. Macke, and S. C. Turaga. Connectome-constrained networks predict neural activity across the fly visual system. *Nature*, Sept. 2024. doi: 10.1038/s41586-024-07939-3.

V. Lobato-Rios, S. T. Ramalingasetty, P. G. Özdil, J. Arreguit, A. J. Ijspeert, and P. Ramdya. NeuroMechFly, a neuromechanical model of adult Drosophila melanogaster. *Nat Methods*, 19(5):620–627, May 2022. doi: 10.1038/s41592-022-01466-7.

M. Lodi, A. Shilnikov, and M. Storace. Design of synthetic central pattern generators producing desired quadruped gaits. *IEEE Trans. Circuits Syst. I*, 65(3):1028–1039, Mar. 2018. doi: 10.1109/TCSI.2017.2759320.

H. Mania, A. Guy, and B. Recht. Simple random search provides a competitive approach to reinforcement learning. *arXiv:1803.07055 [cs, math, stat]*, Mar. 2018.

J. Merel, D. Aldarondo, J. Marshall, Y. Tassa, G. Wayne, and B. Ölveczky. Deep neuroethology of a virtual rodent, Nov. 2019.

S. T. Ramalingasetty, S. M. Danner, J. Arreguit, S. N. Markin, D. Rodarie, C. Kathe, G. Courtine, I. A. Rybak, and A. J. Ijspeert. A whole-body musculoskeletal model of the mouse. *IEEE Access*, 9:163861–163881, 2021. doi: 10.1109/ACCESS.2021.3133078.

S. T. Ramalingasetty, S. N. Markin, A. B. Lockhart, J. Arreguit, N. A. Shevtsova, A. J. Ijspeert, I. A. Rybak, and S. M. Danner. On all fours: A 3D framework to study closed-loop control of quadrupedal mouse locomotion. In *Artificial Life Conference Proceedings 36*, volume 2024, page 59. MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , 2024.

G. P. Sarma, C. W. Lee, T. Portegys, V. Ghayoomie, T. Jacobs, B. Alicea, M. Cantarelli, M. Currie, R. C. Gerkin, S. Gingell, P. Gleeson, R. Gordon, R. M. Hasani, G. Idili, S. Khayrulin, D. Lung, A. Palyanov, M. Watts, and S. D. Larson. OpenWorm: Overview and recent advances in integrative biological simulation of Caenorhabditis elegans. *Phil. Trans. R. Soc. B*, 373(1758):20170382, Oct. 2018. doi: 10.1098/rstb.2017.0382.

L. K. Scheffer, C. S. Xu, M. Januszewski, Z. Lu, S.-y. Takemura, K. J. Hayworth, G. B. Huang, K. Shinomiya, J. Maitlin-Shepard, S. Berg, J. Clements, P. M. Hubbard, W. T. Katz, L. Umayam, T. Zhao, D. Ackerman, T. Blakely, J. Bogovic, T. Dolafi, D. Kainmueller, T. Kawase, K. A. Khairy, L. Leavitt, P. H. Li, L. Lindsey, N. Neubarth, D. J. Olbris, H. Otsuna, E. T. Trautman, M. Ito, A. S. Bates, J. Goldammer, T. Wolff, R. Svirskas, P. Schlegel, E. Neace, C. J. Knecht, C. X. Alvarado, D. A. Bailey, S. Ballinger, J. A. Borycz, B. S. Canino, N. Cheatham, M. Cook, M. Dreher, O. Duclos, B. Eubanks, K. Fairbanks, S. Finley, N. Forknall, A. Francis, G. P. Hopkins, E. M. Joyce, S. Kim, N. A. Kirk, J. Kovalyak, S. A. Lauchie, A. Lohff, C. Maldonado, E. A. Manley, S. McLin, C. Mooney, M. Ndama, O. Ogundeyi, N. Okeoma, C. Ordish, N. Padilla, C. M. Patrick, T. Paterson, E. E. Phillips, E. M. Phillips, N. Rampally, C. Ribeiro, M. K. Robertson, J. T. Rymer, S. M. Ryan, M. Sammons, A. K. Scott, A. L. Scott, A. Shinomiya, C. Smith, K. Smith, N. L. Smith, M. A. Sobeski, A. Suleiman, J. Swift, S. Takemura, I. Talebi, D. Tarnogorska, E. Tenshaw, T. Tokhi, J. J. Walsh, T. Yang, J. A. Horne, F. Li, R. Parekh, P. K. Rivlin, V. Jayaraman, M. Costa, G. S. Jefferis, K. Ito, S. Saalfeld, R. George, I. A. Meinertzhagen, G. M. Rubin, H. F. Hess, V. Jain, and S. M. Plaza. A connectome and analysis of the adult Drosophila central brain. *eLife*, 9:e57443, Sept. 2020. doi: 10.7554/eLife.57443.

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

R. Shadmehr. Control of equilibrium position and stiffness through postural modules. *Journal of Motor Behavior*, 25(3):228–241, Sept. 1993. doi: 10.1080/00222895.1993.9942052.

M. Shafiee, G. Bellegarda, and A. Ijspeert. Puppeteer and marionette: Learning anticipatory quadrupedal locomotion based on interactions of a central pattern generator and supraspinal drive, Feb. 2023.

N. A. Shevtsova, A. E. Talpalar, S. N. Markin, R. M. Harris-Warrick, O. Kiehn, and I. A. Rybak. Organization of left–right coordination of neuronal activity in the mammalian spinal cord: Insights from computational modelling. *The Journal of Physiology*, 593(11):2403–2426, June 2015. doi: 10.1113/JP270121.

Y. Tassa, S. Tunyasuvunakool, A. Muldal, Y. Doron, P. Trochim, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, and N. Heess. Dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, Nov. 2020. doi: 10.1016/j.simpa.2020.100022.

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*, pages 5026–5033, Vilamoura-Algarve, Portugal, Oct. 2012. IEEE. doi: 10.1109/IROS.2012.6386109.

S. Wang-Chen, V. A. Stimpfling, T. K. C. Lam, P. G. Özdil, L. Genoud, F. Hurtak, and P. Ramdya. NeuroMechFly v2: Simulating embodied sensorimotor control in adult Drosophila. *Nat Methods*, 21(12):2353–2362, Dec. 2024. doi: 10.1038/s41592-024-02497-y.

J. Yu, M. Tan, J. Chen, and J. Zhang. A survey on CPG-inspired control models and system implementation. *IEEE Trans. Neural Netw. Learning Syst.*, 25(3):441–456, Mar. 2014. doi: 10.1109/TNNLS.2013.2280596.

C. Y. Zhang, Y. Yang, A. Sirbu, E. T. T. Abe, E. Wärnberg, E. J. Leonardis, D. E. Aldarondo, A. Lee, A. Prasad, J. Foat, K. Bian, J. Park, R. Bhatt, H. Saunders, A. Nagamori, A. R. Thanawalla, K. W. Huang, F. Plum, H. K. Beck, S. W. Flavell, D. Labonte, B. A. Richards, B. W. Brunton, E. Azim, B. P. Ölveczky, and T. D. Pereira. MIMIC-MJX: Neuromechanical emulation of animal behavior, Dec. 2025.

H. Zhang, N. A. Shevtsova, D. Deska-Gauthier, C. Mackay, K. J. Dougherty, S. M. Danner, Y. Zhang, and I. A. Rybak. The role of V3 neurons in speed-dependent interlimb coordination during locomotion in mice. *eLife*, 11:e73424, Apr. 2022. doi: 10.7554/eLife.73424.

A Biomechanical Body

A.1 MuJoCo specification

The MuJoCo XML file specifies the morphology, dynamics, actuators, and sensors of the body. We made the following changes to the body of Merel et al. [2019] to improve fidelity and usability:

- **Joint axes:** Shoulder supination, lumbar, cervical, and tail joints are flipped so that positive commands correspond to outward rotation, rightward bending, and upward extension. Toe and finger joints are flipped so that positive commands correspond to extension.
- **Joint ranges:** Asymmetric limits are applied to joints with asymmetric anatomical ranges, symmetric limits are recalibrated, and toe and finger ranges are expanded to permit flexion.
- **Default pose:** The default pose is a neutral standing pose with closed mouth.
- **Collision geometry:** Limb geometry is revised so that all 4 paws rest on the ground in the default pose, and the tail is simplified from 28 to 6 bodies to reduce computation.
- **Touch sensors:** Each paw has 2 sensors (digit and palm/heel) that report contact force rather than binary contact, providing graded information about paw loading.
- **Actuators:** Force limits are enabled, force ranges are recalibrated from stiffness and joint limits, and joint and tendon actuators are unified into a single interface.
- **Conventions:** Body parts use anatomical names (e.g., "caudal" $\rightarrow$ "tail"), units are degrees rather than radians, and actuators are ordered rostral-to-caudal, left-to-right, and proximal-to-distal.
- **Sensors and cameras:** An orientation sensor and 5 cameras support analysis and rendering.

Motion-capture keypoints are preserved, so that Rat v2 remains compatible with existing imitation datasets. The resulting action and observation spaces are listed in Table 1.

Table 1 | **Action and observation spaces.** Actions by body segment and joint (left) and observations by modality (right), with dimensions in brackets. Limb joints have left and right copies.

Segment	Action	Dim.
Head	Atlas: extend	[1]
	Mandible: open	[1]
Neck	Cervical: extend, bend, twist	[3]
Forelimb	Scapula: supinate, abduct, extend	[2, 3]
	Shoulder: supinate, extend	[2, 2]
	Elbow: extend	[2, 1]
	Wrist: extend	[2, 1]
	Finger: extend	[2, 1]
Trunk	Lumbar: extend, bend, twist	[3]
Hindlimb	Hip: supinate, abduct, extend	[2, 3]
	Knee: extend	[2, 1]
	Ankle: extend	[2, 1]
	Toe: extend	[2, 1]
Tail	Caudal: extend, bend	[2]
Total		**[38]**

Modality	Observation	Dim.
Proprioceptive	Positions $\boldsymbol{q}$	[38]
	Velocities $\dot{\boldsymbol{q}}$	[38]
	Torques $\boldsymbol{\tau}$	[38]
Tactile	Contact forces $\boldsymbol{f}$	[4, 2]
Vestibular	Roll ϕ	[1]
	Pitch θ	[1]
	Angular velocity $\boldsymbol{\omega}$	[3]
	Linear acceleration $\ddot{\boldsymbol{x}}$	[3]
Visual	RGB image $\boldsymbol{I}$	[64, 64, 3]
	Depth map $\boldsymbol{D}$	[64, 64, 1]

A.2 Rendering skins

We designed biologically realistic rendering skins in Blender to enable qualitative comparison between simulated and experimentally observed rat locomotion (Figure A.1). The MuJoCo body was imported into Blender, where each joint and body segment became a bone in an armature. A low-polygon mesh was built from reference images of Long-Evans rats, mirrored to enforce bilateral symmetry, and refined with hand-added asymmetries (e.g., tail curvature). The mesh was subdivided into a high-polygon surface, and bone weights were painted onto vertices to determine mesh deformation. Procedural textures model 3 Long-Evans coat patterns: gray, black-hooded, and cinnamon. A custom Blender plugin exports the rigged mesh and texture to MuJoCo's skin (`.skn`) and texture (`.png`) formats. Skins are purely cosmetic and do not affect the physics simulation.

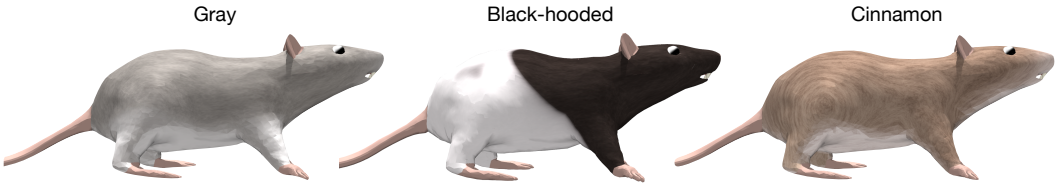


Figure A.1 | **Rendering skins.** Renderings of the 3 Long-Evans coat patterns. Skins facilitate qualitative comparison with experimental videos without affecting physics.

A.3 Environment framework

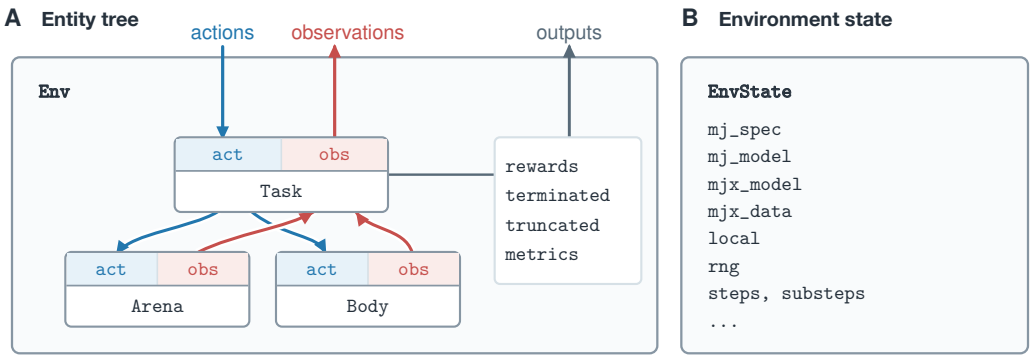
Training large populations of neural circuit models requires efficient, high-throughput simulation. We build an environment framework in JAX that adapts the `dm_control` composer [Tassa et al., 2020] to XLA compilation and GPU-accelerated physics (Figure A.2). The framework separates environment composition, simulation state, and execution, so that bodies, arenas, interfaces, and tasks can be reconfigured across experiments without changing the simulation code.

Entity tree An `Env` is composed of a tree of `Entity` objects with a `Task` at its root (Figure A.2A). Each `Entity` contributes its own model elements, such as bodies, sensors, actuators, or terrain, and the `Task` is a special `Entity` that additionally defines rewards, termination and truncation conditions, metrics, and subtasks. Each `Entity` has an action `Transform` that maps actions received from its parent to its own actions, and an observation `Transform` that maps its own observations to those exposed to its parent. `Transform` objects make components interchangeable: each neuromechanical interface is a pair of body `Transform` objects, swappable without modifying the `Task` or arena.

Environment state All simulation state is stored in an immutable `EnvState` (Figure A.2B), including the MuJoCo model and data, entity-local state, the random number generator key, and step counters. `Env` methods receive and return this state explicitly rather than modifying it in place, so that they can be compiled with XLA and vectorized across thousands of parallel environments. `Entity` and `EnvState` objects are defined with the Equinox library [Kidger and Garcia, 2021].

Lifecycle methods The `Env` exposes `reset` and `step` methods (Figure A.2C). `reset` initializes the physics data, resets each `Entity`, and computes initial observations. `step` applies action `Transform` objects and advances physics for each substep, then computes observations, rewards, termination, truncation, and metrics. `Task` hooks such as `on_reset`, `on_step_start`, and `on_step_end` implement task-specific behavior, such as sampling target speeds, without modifying the physics loop.

Physics backends Physics runs through MuJoCo XLA with either its XLA or Warp backend, so the same environment code runs on CPU, GPU, and TPU. The full simulation step, including contact detection, constraint solving, and sensor computation, compiles into GPU kernels, yielding large speedups over serial CPU simulation. Experiments in this paper use the Warp backend.



C Lifecycle methods

env.reset(state, rng, subtask=None)

```
mjx.make_data(...)
core.entity_reset(...)
task.on_reset(...)
mjx.forward(...)
observation, state = core.entity_observation(...)
reward, terminated, truncated = 0, False, False
metrics = task.get_metrics(...)
return Step(state, observation, reward, terminated, truncated, metrics)
```

env.step(state, action)

```
task.on_step_start(...)
for _ in range(sim_substeps):
    core.entity_action(...)
    mjx.step(...)
task.on_step_end(...)
observation, state = core.entity_observation(...)
terminated = task.get_terminated(...)
truncated = steps >= truncate if truncate is not None else False
rewards = task.get_rewards(...)
reward = task.get_reward(rewards)
metrics = task.get_metrics(...)
return Step(state, observation, reward, terminated, truncated, metrics)
```

Figure A.2 | **Environment framework.** (**A**) `Entity` tree, shown for a `Task` with arena and body entities. Actions pass down and observations pass up through each entity's `Transform` objects, which can be swapped without modifying other entities. (**B**) Environment state. All simulation state is held in an immutable, explicitly passed `EnvState`, so environments compile and vectorize in parallel. (**C**) Lifecycle methods. `reset` and `step` call `Task` and `Entity` hooks around physics updates.

B Neural Circuit Model

B.1 Neurons

The architecture uses 2 rate-coded neuron types with excitatory or inhibitory synaptic input.

Basic neuron The Basic neuron is a rate-coded leaky integrator with time constant T_v, bias B, weighted input x, internal voltage v, and output activity y:

$$x = \text{clip}(B + \textstyle\sum_i w_i y_i,\, -1, 1)\,, \qquad \frac{T_v}{4}\frac{dv}{dt} = x - v, \qquad y = f(v) = \text{clip}(v, 0, 1)$$

The neuron produces activity proportional to its voltage once the voltage exceeds zero.

Bursting neuron The Bursting neuron is a relaxation oscillator that produces intrinsic bursting dynamics through 2 coupled variables: voltage $v \in [-1, 1]$ and adaptation $a \in [0, 1]$. Adaptation depletes during the active phase and recovers during the quiet phase, driving alternating transitions between states. Input modulates the active and quiet durations, enabling phase resetting and entrainment. For equations and analysis, see Bhattasali et al. [2026].

B.2 Rhythm generation

RG follows the cell types, neurotransmitter identities, and connectivity of the spinal circuit model of Danner et al. [2017] (Figure B.1). Each limb has a Bursting flexor and a Basic extensor half-center. Commissural interneurons (V0d, V0v, V3) couple left and right limbs, and long propriospinal neurons couple forelimbs and hindlimbs through diagonal and homolateral pathways.

Tuning protocol RG weights and Bursting neuron hyperparameters (time constants, thresholds) were hand-tuned to produce the gait transition sequence stand $\rightarrow$ walk $\rightarrow$ trot $\rightarrow$ bound as brainstem command c_t increases from 0 to 1. Tuning followed the protocol of Bhattasali et al. [2024], using its weights as a starting point and adjusting time constants to match rat stepping, which is faster than that of a quadruped robot. RG weights are frozen throughout training.

Cell-type activity RG activity reproduces key features of speed-dependent cell-type activity reported for biophysical models validated against experimental data (Figure B.2; compare Figure 10 of Zhang et al., 2022). Inhibitory V0d activity is synchronous with flexor bursts and decreases toward high speeds. Excitatory V0v activity also decreases toward high speeds, but more slowly than V0d activity, which supports trot at intermediate speeds. Diagonal V3 activity increases toward high speeds as bound emerges. Half-center oscillations are flexor-dominated, with rhythm arising from the intrinsic bursting of the flexor half-center rather than from escape of the extensor half-center.

B.3 Brainstem command

BC maps target speed to brainstem command through a fixed linear mapping, $c_t = f(v_t^{\text{target}})$, so that the command equals the target speed over the trained range. The command modulates RG half-center frequency and amplitude, interneuron activity, and PF synergies. BC has no trainable parameters.

B.4 Afferent feedback

Modules AF consists of forelimb (F), hindlimb (H), and trunk (T) modules. The F module is applied to FL and FR, and the H module to HL and HR, so that weights are shared between left and right limbs. Limb modules observe joint positions, velocities, and torques of their DOFs and the 2 tactile sensors of their paw. The T module observes joint positions, velocities, and torques of the lumbar DOFs, with bending and twisting observations mirrored into left and right copies. Each proprioceptive signal is rectified into positive and negative components (Appendix C.1).

Sensor layer The sensor layer applies a per-element gain and threshold followed by f, modeling the saturating response of sensory receptors. Sensor-layer outputs $\boldsymbol{o}_t^{\text{sensor},p}$ pass to the reset layer and to PF as non-resetting feedback, which triggers reflexes without affecting the rhythm.

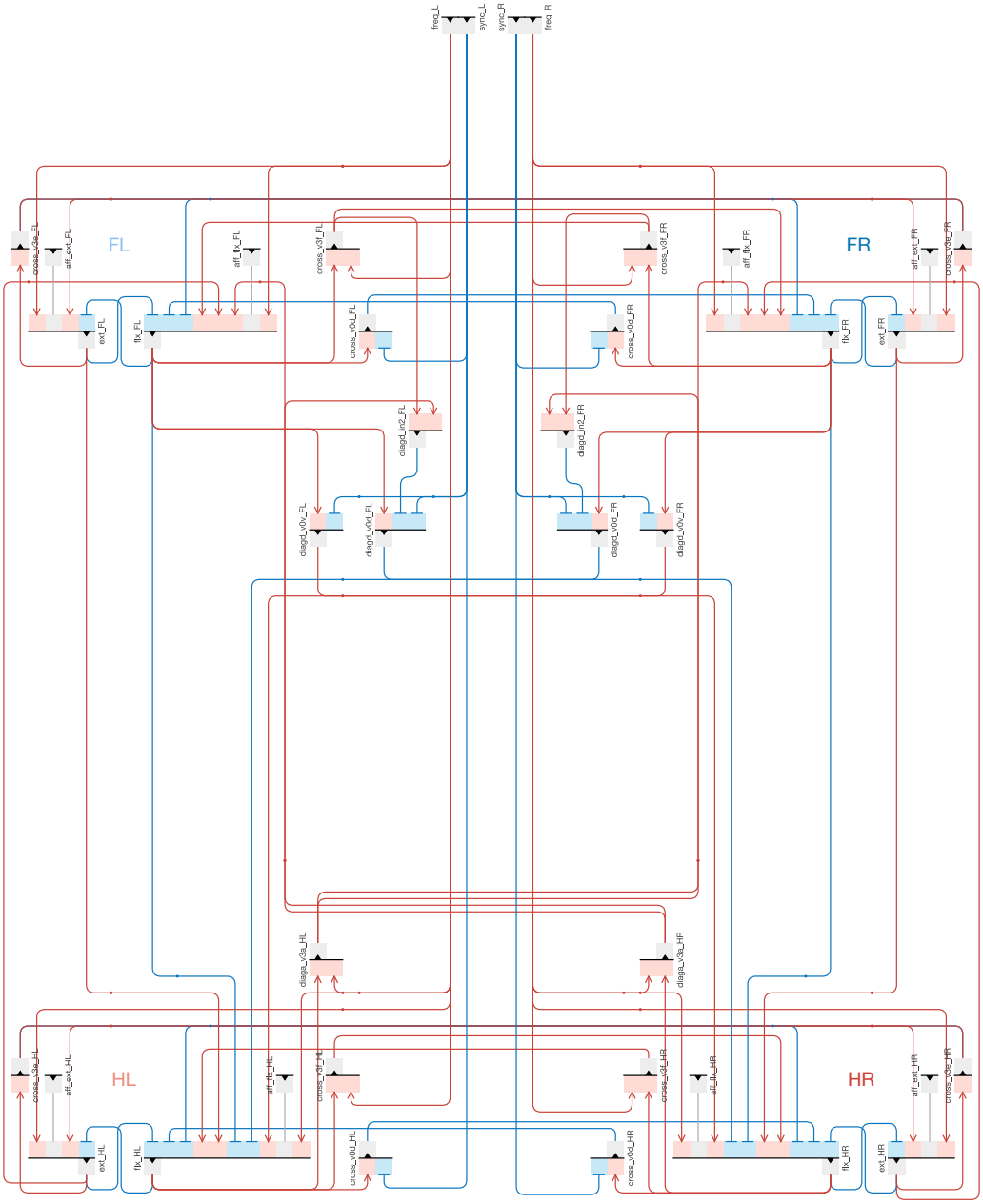


Figure B.1 | **RG circuit.** Full circuit schematic showing cell-type-specific connectivity between limb half-centers and interneuron populations. Excitatory synapses are shown in red, and inhibitory synapses in blue. Commissural interneurons (V0d, V0v, V3) mediate left-right coordination, and long propriospinal neurons mediate forelimb-hindlimb coordination.

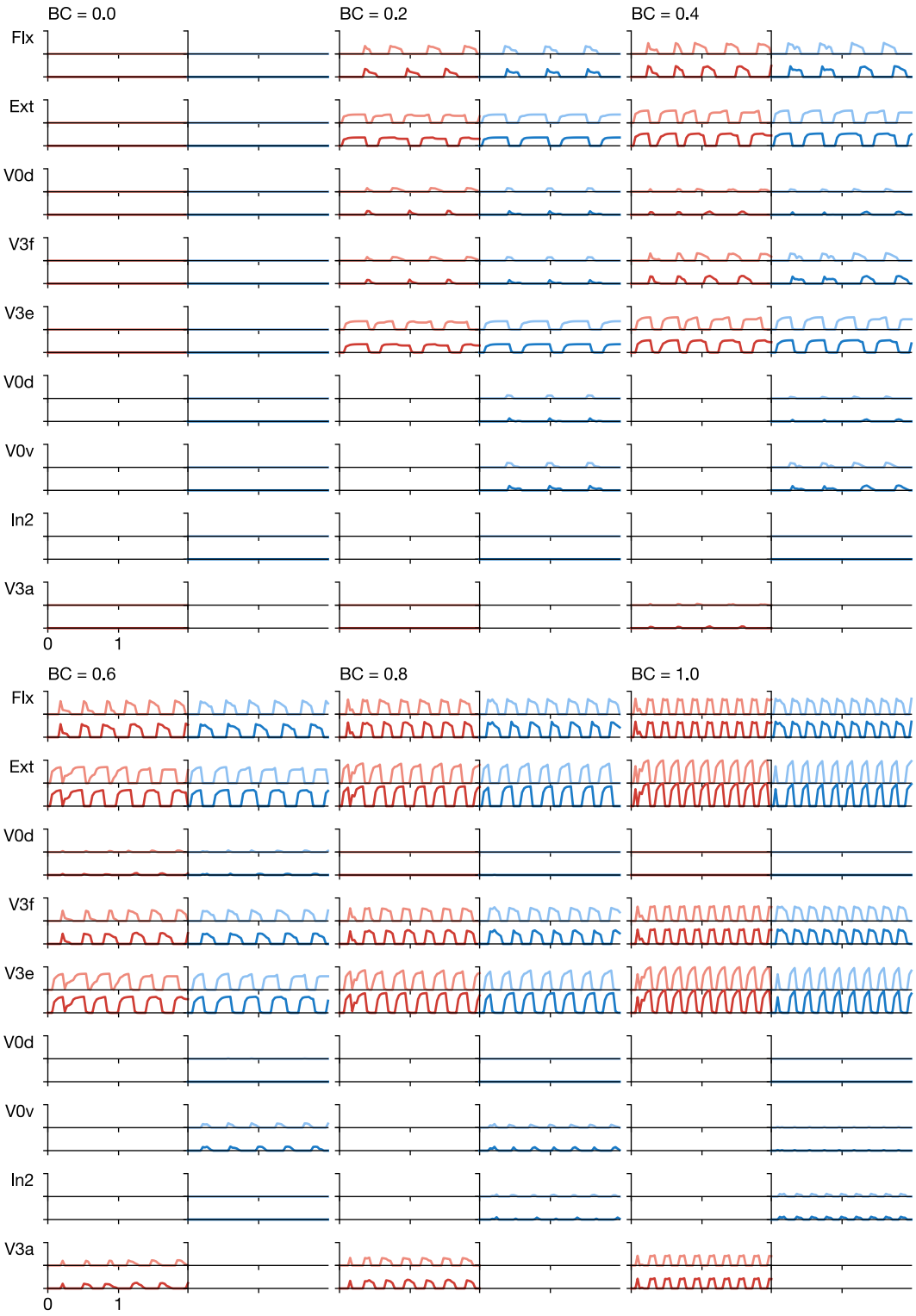


Figure B.2 | **RG activity.** Activity of RG half-centers and interneuron populations over 2 s at increasing brainstem command (BC), for hindlimbs and forelimbs, with left and right sides in light and dark traces. The first V0d row shows diagonal V0d neurons, and the second shows commissural V0d neurons. As BC increases, stepping frequency increases, V0d and V0v activity decline, and V3 activity increases, matching speed-dependent recruitment in biophysical models [Zhang et al., 2022].

Reset layer The reset layer maps each limb's sensor-layer outputs to resetting feedback for its own flexor and extensor half-centers, followed by clipping to $[-1, 1]$. Weights from other limbs and from the T module are fixed at 0, so the T module acts only through PF.

Initialization and constraints Sensor-layer and reset-layer biases are initialized and constrained to be nonpositive, so that each afferent responds only above a threshold. Reset-layer weights are initialized at zero without sign constraints, so that resetting feedback is learned rather than prescribed (Figure B.3). Local reset connections reflect the predominantly local action of limb afferents on spinal rhythm generators [Kim et al., 2022].

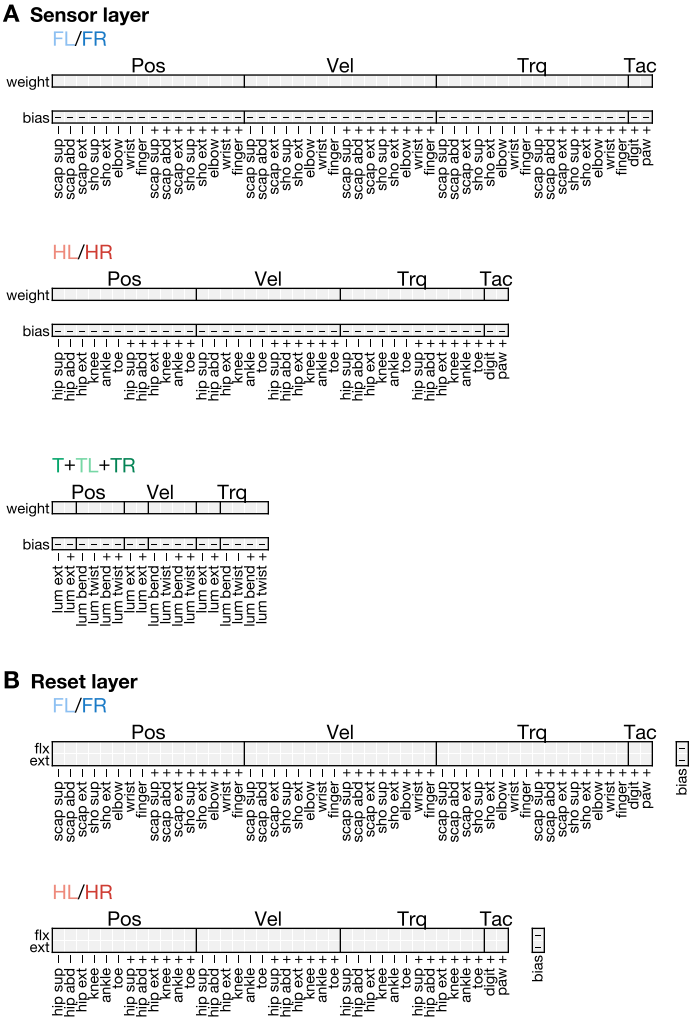


Figure B.3 | **AF initialization.** (**A**) Sensor-layer weights and biases of the F and H modules. Biases are constrained to be nonpositive, so that each afferent responds only above a threshold. (**B**) Reset-layer weights and biases from sensor-layer outputs to flexor and extensor half-centers. Weights start at zero, so that resetting feedback is learned. Columns show positive $(+)$ and negative $(-)$ components of each proprioceptive signal, and cell symbols show sign constraints.

B.5 Pattern formation

Modules PF consists of forelimb (F), hindlimb (H), and trunk (T) modules. The F and H modules are shared between left and right limbs, like their AF counterparts. The T module controls lumbar extension, bending, and twisting with 6 synergies per DOF group, receiving forelimb and hindlimb phases, brainstem command, and lumbar sensor outputs. Bending and twisting synergies have left and right copies with sign-inverted outputs, so that mirrored inputs produce mirrored actions.

Delay layer The delay layer produces early and late copies of each limb's flexor and extensor half-center outputs with distinct delays, yielding 4 phase signals. Delayed copies permit desynchronized joint motions within a step cycle. Early copies use time constants T_v of 0 ms (flexor) and 30 ms (extensor) with bias $B = -0.05$, and late copies use 50 ms (flexor) and 80 ms (extensor) with biases of 0 and -0.3, respectively.

Synergy layer The synergy layer recruits 5 distinct synergies: liftoff and advance during swing, and touchdown, support, and propulsion during stance. This decomposition follows the phase-specific organization of muscle synergies in cat locomotion [Klishko et al., 2025]. The propulsion synergy has 1 copy, and the rest have 3 copies each, yielding 13 synergies per limb. Inputs combine phase signals, local non-resetting feedback (reflexes), and brainstem command (speed-dependent recruitment), inspired by mutually inhibiting populations of neurons in biological pattern formation.

Motor layer The motor layer outputs a linear combination of recruited synergies to produce interface actions. The output format depends on the neuromechanical interface (Section 3.3). For flexor-extensor control, each synergy specifies flexor and extensor activations for every joint of the limb. For position control, each synergy specifies equilibrium positions for every joint of the limb.

Initialization and constraints Synergy-layer weights are initialized so that flexor phase signals excite swing synergies and inhibit stance synergies, and extensor phase signals do the opposite (± 1). Motor-layer weights are sign-constrained and initialized with small values on joints consistent with each synergy's role (Figure B.4). For example, hindlimb liftoff synergies flex the knee and ankle, advance synergies flex the hip, and support and propulsion synergies extend the hip, knee, and ankle. The copies of each synergy share an initialization and differentiate during training. In the T module, motor-layer weights are initialized at zero and synergy-layer weights from limb phases are initialized small (0.01). Motor-layer biases are initialized at values that produce the interface equilibrium pose: moderate co-activation (0.8) of flexors and extensors. For flexor-extensor control, biases remain trainable, so that co-activation can decrease to make joints more compliant. For position control, biases are frozen at 0.5, so that the equilibrium pose remains the default standing pose.

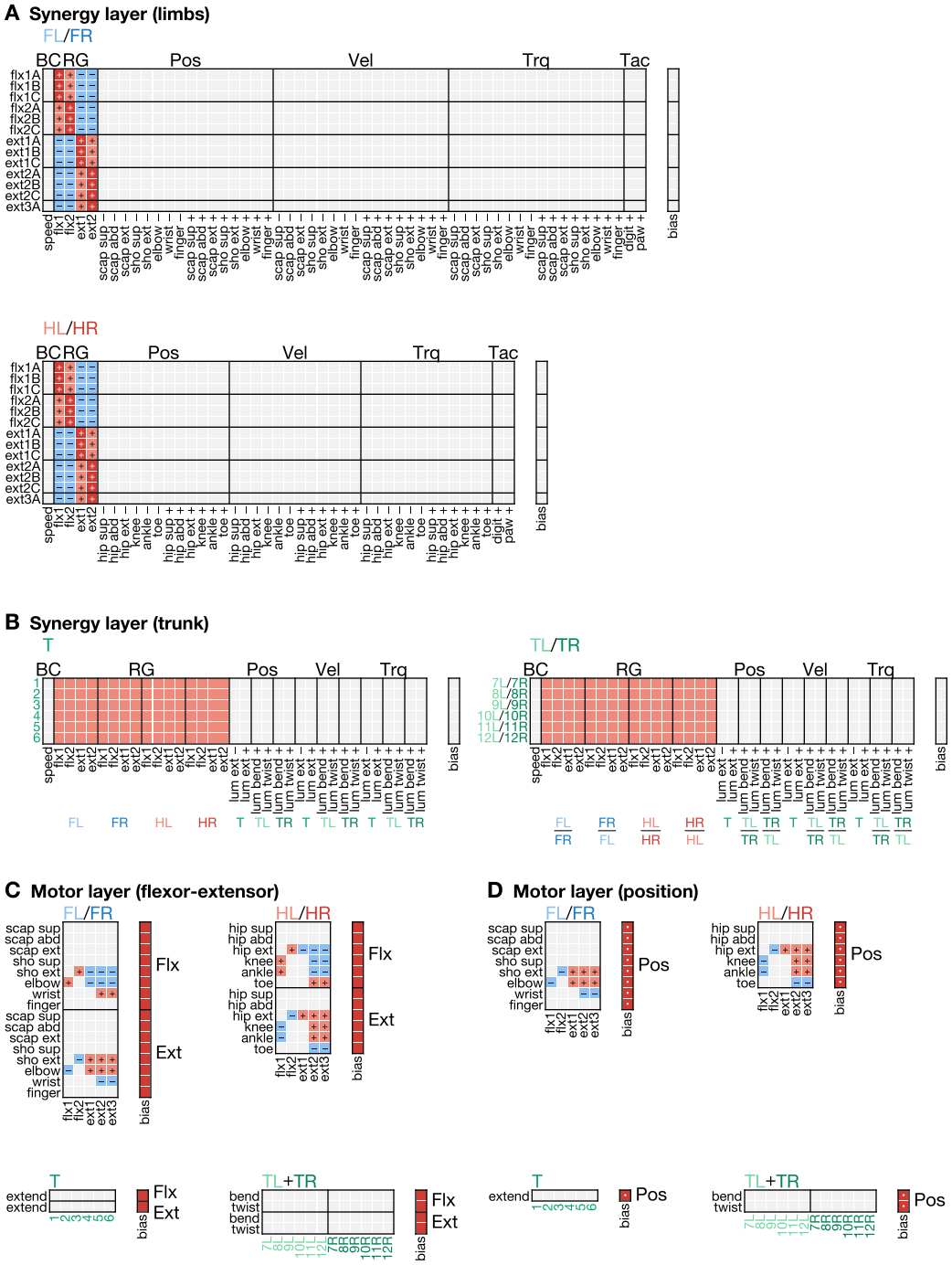


Figure B.4 | **PF initialization.** (**A**) Synergy-layer parameters of the F and H modules. Flexor phase signals excite swing synergies and extensor phase signals excite stance synergies, each inhibiting the other group. (**B**) Synergy-layer parameters of the T module for lumbar extension and lumbar bending and twisting. Trunk synergies receive weak excitation from forelimb and hindlimb phases. (**C, D**) Motor-layer parameters for flexor-extensor and position control. The 2 interfaces share the synergy layer and differ only in the motor layer. Swing synergies flex and stance synergies extend the relevant joints, while biases set the equilibrium pose. Trunk motor-layer weights start at zero and are learned. Cell color shows initial values (positive, negative), and cell symbols show sign constraints.

B.6 Parameter counts

Rat NCAP has 35 fixed parameters in RG and BC and 2,133–2,238 trained parameters in AF and PF (Table 2). The interfaces differ only in the motor layer, whose size depends on the number of control variables per DOF. The MLPs have about 50 times more parameters than Rat NCAP.

Table 2 | **Parameter counts.** Parameters of Rat NCAP by module (top) and of the MLP baselines by layer (bottom) for each neuromechanical interface.

	Torque	Position	Position-stiffness	Flexor-extensor
Rat NCAP				
RG (fixed)	33	33	33	33
BC (fixed)	2	2	2	2
AF sensor layer	200	200	200	200
AF reset layer	180	180	180	180
PF synergy layer: RG, BC inputs	324	324	324	324
PF synergy layer: AF inputs	1,324	1,324	1,324	1,324
PF motor layer	105	105	210	193
Trained	2,133	2,133	2,238	2,221
Total	**2,168**	**2,168**	**2,273**	**2,256**
MLP				
Hidden layer 1 [102, 256]	–	26,368	–	26,368
Hidden layer 2 [256, 256]	–	65,792	–	65,792
Output layer [256, 31 or 62]	–	7,967	–	15,934
Total	–	**100,127**	–	**108,094**

C Neuromechanical Interfaces

C.1 Observation transform

All interfaces share an observation transform that rectifies proprioceptive and tactile observations into nonnegative components, analogous to afferents that signal opposite directions of joint motion and loading. Each joint has angle q with flexion and extension limits $[q_{\text{min}}, q_{\text{max}}]$ and range $R = q_{\text{max}} - q_{\text{min}}$. Position is normalized to $\hat{q} = (q - q_{\text{min}})/R \in [0, 1]$, and its negative and positive components measure closeness to the flexion and extension limits, quantifying the amount of flexion and extension. Velocity and torque are sign-rectified. Tactile contact force is nonnegative and has only a positive component:

$$o^{\text{pos}-} = 1 - \hat{q}, \qquad o^{\text{vel}-} = \max(0, -\dot{q}), \qquad o^{\text{trq}-} = \max(0, -\tau),$$
$$o^{\text{pos}+} = \hat{q}, \qquad o^{\text{vel}+} = \max(0, \dot{q}), \qquad o^{\text{trq}+} = \max(0, \tau), \qquad o^{\text{tac}+} = f$$

C.2 Flexor-extensor control

Flexor-extensor control actuates each joint with an antagonistic pair of linear springs, a flexor and an extensor, scaled by activations $a_{\text{flx}}, a_{\text{ext}} \in [0, 1]$ output by the neural circuit.

Geometric interpretation The springs can be pictured as a pulley system acting antagonistically on the joint (Figure C.1A). Each spring wraps around a pulley of constant radius at the joint, so joint rotation changes its length linearly. The flexor has stiffness k_{flx} and rests at its attachment point $q_{\text{flx}} \leq q_{\text{min}}$, and the extensor has stiffness k_{ext} and rests at $q_{\text{ext}} \geq q_{\text{max}}$. Since the attachment points lie beyond the joint limits, both springs remain stretched throughout the range of motion and can only pull, as muscles do: the flexor always produces flexion torque, and the extensor always produces extension torque. The muscles play a tug-of-war, with activations setting how strongly each pulls. The joint settles where the pulls balance, or at a joint limit if the balance point lies beyond it.

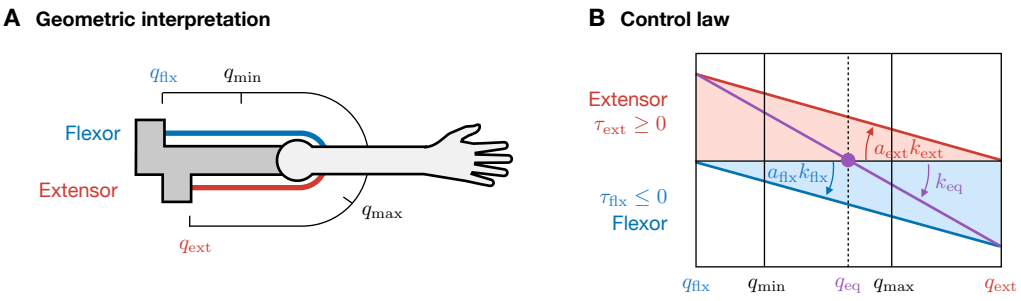


Figure C.1 | **Flexor-extensor control.** (**A**) Geometric interpretation. Flexor and extensor springs act on the joint like a pulley system, pulling against each other from attachment points beyond the joint limits. (**B**) Control law. Flexor and extensor torques versus joint angle, shaded over activations from 0 to 1. Net torque acts as a single spring with activation-dependent stiffness and equilibrium position.

Control law The joint torque is the sum of extensor and flexor torques (Figure C.1B):

$$\tau = \tau_{\text{ext}} + \tau_{\text{flx}}, \qquad \tau_{\text{ext}} = a_{\text{ext}} \cdot k_{\text{ext}} \cdot (q_{\text{ext}} - q), \qquad \tau_{\text{flx}} = a_{\text{flx}} \cdot k_{\text{flx}} \cdot (q_{\text{flx}} - q)$$

with $\tau_{\text{ext}} \geq 0$ and $\tau_{\text{flx}} \leq 0$ within the joint limits. Grouping terms yields position-stiffness control with activation-dependent stiffness k_{eq} and equilibrium position q_{eq}:

$$\tau = k_{\text{eq}} \cdot (q_{\text{eq}} - q), \qquad k_{\text{eq}} = a_{\text{ext}} \cdot k_{\text{ext}} + a_{\text{flx}} \cdot k_{\text{flx}}, \qquad q_{\text{eq}} = \frac{a_{\text{ext}} \cdot k_{\text{ext}} \cdot q_{\text{ext}} + a_{\text{flx}} \cdot k_{\text{flx}} \cdot q_{\text{flx}}}{k_{\text{eq}}}$$

The equilibrium position is a stiffness-weighted average of the 2 attachment points. Differential activation shifts the equilibrium position toward the more active muscle's attachment point, while co-activation scales stiffness without moving the equilibrium position. The neural circuit can therefore control both equilibrium position and stiffness through 2 activations with fixed flexor and extensor identities, a principle reflected in biological motor control [Shadmehr, 1993]. Unlike in position-stiffness control, the 2 quantities are coupled: stiffness is greatest at full co-activation and decreases as the equilibrium position approaches either attachment point.

Parameter matching Flexor-extensor control is specified by 4 spring parameters per joint $(k_{\text{flx}}, k_{\text{ext}}, q_{\text{flx}}, q_{\text{ext}})$, whereas the other interfaces are specified by an equilibrium position, stiffness, and torque. The spring parameters can be chosen so that flexor-extensor control matches per-joint parameters shared with the other interfaces, which enables comparing interfaces on equal footing. These shared parameters are equilibrium position q_{eq}, stiffness k_{eq}, and maximum flexor and extensor torques $\tau_{\text{flx}}^{\text{max}}, \tau_{\text{ext}}^{\text{max}}$, which serve as fixed parameters or control ranges. Matching the 4 shared parameters provides 4 conditions, which uniquely determine the 4 spring parameters. First, at full co-activation ($a_{\text{flx}} = a_{\text{ext}} = 1$), the net spring has the shared stiffness, and the 2 spring torques balance at the shared equilibrium position. Second, each muscle produces its maximum torque when fully active alone at the opposite joint limit, where it is stretched furthest. These conditions are:

$$\begin{aligned} k_{\text{ext}} + k_{\text{flx}} &= k_{\text{eq}} & k_{\text{ext}} \cdot (q_{\text{ext}} - q_{\text{eq}}) &= k_{\text{flx}} \cdot (q_{\text{eq}} - q_{\text{flx}}) \\ k_{\text{ext}} \cdot (q_{\text{ext}} - q_{\text{min}}) &= \tau_{\text{ext}}^{\text{max}} & k_{\text{flx}} \cdot (q_{\text{max}} - q_{\text{flx}}) &= \tau_{\text{flx}}^{\text{max}} \end{aligned}$$

Solving these conditions yields:

$$\begin{aligned} k_{\text{ext}} &= \frac{k_{\text{eq}} \cdot (q_{\text{max}} - q_{\text{eq}}) + \tau_{\text{ext}}^{\text{max}} - \tau_{\text{flx}}^{\text{max}}}{R}, & q_{\text{ext}} &= q_{\text{min}} + \frac{\tau_{\text{ext}}^{\text{max}}}{k_{\text{ext}}}, \\ k_{\text{flx}} &= \frac{k_{\text{eq}} \cdot (q_{\text{eq}} - q_{\text{min}}) + \tau_{\text{flx}}^{\text{max}} - \tau_{\text{ext}}^{\text{max}}}{R}, & q_{\text{flx}} &= q_{\text{max}} - \frac{\tau_{\text{flx}}^{\text{max}}}{k_{\text{flx}}}. \end{aligned}$$

The derived parameters are valid when $k_{\text{ext}}, k_{\text{flx}} > 0$ and the attachment points lie beyond the joint limits, which requires $\tau_{\text{ext}}^{\text{max}} \geq k_{\text{ext}} R$ and $\tau_{\text{flx}}^{\text{max}} \geq k_{\text{flx}} R$. That is, each maximum torque must be large enough relative to its stiffness that the spring remains stretched, and thus pulls, across the full range of motion. In this work, q_{eq} is the midpoint of the joint range and $\tau_{\text{ext}}^{\text{max}} = \tau_{\text{flx}}^{\text{max}}$, so that $k_{\text{ext}} = k_{\text{flx}} = k_{\text{eq}}/2$ and the attachment points are placed symmetrically about the range. Per-joint values of the shared parameters are provided in the released configuration files.

Comparison to Ekeberg model The most closely related abstract muscle model is that of Ekeberg [1993], as formulated in Lobato-Rios et al. [2022]. Up to sign conventions and omitting passive and damping terms, it sums a differential term with gain k_a and a co-activation-dependent stiffness term with gain k_s around a rest position q_{rest}. Grouping terms by activation yields 2 linear springs:

$$\begin{aligned} \tau &= k_a \cdot (a_{\text{ext}} - a_{\text{flx}}) + k_s \cdot (a_{\text{ext}} + a_{\text{flx}}) \cdot (q_{\text{rest}} - q) \\ &= a_{\text{ext}} \cdot k_s \cdot \big((q_{\text{rest}} + k_a/k_s) - q\big) + a_{\text{flx}} \cdot k_s \cdot \big((q_{\text{rest}} - k_a/k_s) - q\big) \end{aligned}$$

This is the flexor-extensor control law with symmetric stiffnesses $k_{\text{flx}} = k_{\text{ext}} = k_s$ and attachment points $q_{\text{ext}}, q_{\text{flx}} = q_{\text{rest}} \pm k_a/k_s$. The difference is that k_a and k_s are independent in the Ekeberg model, so its attachment points are not constrained to lie beyond the joint limits. When an attachment point lies within the range of motion, the muscle reverses its torque past that point. For example, if $q_{\text{flx}} > q_{\text{min}}$, flexor activation produces extension torque at angles between q_{min} and q_{flx}, so the circuit must account for a state-dependent sign of each muscle's effect. Our flexor-extensor model rules out this sign reversal by construction and permits asymmetric flexor and extensor strengths.

Extensions The flexor-extensor interface is a starting point for more realistic muscle models. Since each joint has explicit flexor and extensor activations, Hill-type muscle properties, such as activation dynamics, force-length curves, and force-velocity relationships, can be added one at a time. Their contributions to circuit function and behavior can then be tested against the linear springs used here.

D Training Setup

D.1 Tasks

Initialization Each episode begins with the rat body in the default standing pose at the origin, with zero velocity. The control timestep is $\Delta t_{\text{env}} = 0.03$ s, and the physics timestep is $\Delta t_{\text{sim}} = 0.001$ s (30 substeps per control step). Training episodes are truncated after 100 control steps (3 s), and test episodes after 200 control steps (6 s).

Termination An episode terminates early if the trunk or upper limbs touch the ground, or if roll or pitch exceeds $45°$. These conditions detect falls and prevent prone or rolling behaviors.

Subtasks The locomotion task has 6 subtasks, each with the goal of matching a fixed target speed ($v^{\text{target}} \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$ m/s). Each population member is evaluated on all 6 subtasks in parallel, and its fitness aggregates returns across subtasks. The target speed is provided to the model at each timestep through the brainstem command $c_t = v_t^{\text{target}}$.

Rewards For moving subtasks ($v^{\text{target}} > 0$), the reward is a weighted sum of 3 terms:

$$r_t = w_{\text{alive}}\, r_{\text{alive}} + w_{\text{dense}}\, r_{\text{dense}}(v_t, v_t^{\text{target}}) + w_{\text{sparse}}\, r_{\text{sparse}}(v_t, v_t^{\text{target}})$$

with forward velocity v_t and weights $w_{\text{alive}} = 0.2$, $w_{\text{dense}} = 0.6$, and $w_{\text{sparse}} = 0.2$. For the standing subtask ($v^{\text{target}} = 0$), the reward is a pose term $r_t = r_{\text{pose}}$. All terms lie in $[0, 1]$.

An **alive reward** $r_{\text{alive}} = 1$ is given at every non-terminated timestep, which discourages early termination without shaping behavior.

A **dense velocity reward** is a triangular function centered at the target speed:

$$r_{\text{dense}}(v_t, v_t^{\text{target}}) = \max\left(0,\, 1 - \frac{|v_t - v_t^{\text{target}}|}{\max(v_t^{\text{target}},\, \epsilon)}\right)$$

with margin $\epsilon = 0.1$ m/s. It reaches 1 when $v_t = v_t^{\text{target}}$ and declines linearly to 0 at a deviation of $\max(v_t^{\text{target}}, \epsilon)$, providing a graded signal toward the target speed.

A **sparse velocity reward** is a quadratic function centered at the target speed:

$$r_{\text{sparse}}(v_t, v_t^{\text{target}}) = \max\left(0,\, 1 - \left(\frac{|v_t - v_t^{\text{target}}|}{\epsilon}\right)^2\right)$$

It reaches 1 when $v_t = v_t^{\text{target}}$ and declines quadratically to 0 at a deviation of ϵ, rewarding consistency near the target speed.

A **pose reward** decays with the largest joint-angle deviation from the default standing pose:

$$r_{\text{pose}} = \frac{1}{1 + 4\,(d_t/\epsilon_{\text{pose}})^2}, \qquad d_t = \max_j |q_{t,j} - q_j^{\text{stand}}|$$

with joint angles $q_{t,j}$, standing-pose joint angles q_j^{stand}, and margin $\epsilon_{\text{pose}} = 10°$.

D.2 Training algorithm

We optimize the trainable AF and PF parameters $\boldsymbol{\theta}$ with Augmented Random Search (ARS) [Mania et al., 2018], implemented on GPU alongside the vectorized environments. Evolutionary strategies such as ARS suit biologically grounded architectures because they require neither backpropagation through recurrent neural dynamics and simulator contacts nor a differentiable model, and they handle sign constraints and frozen parameters directly. ARS in particular has few hyperparameters and is competitive with deep reinforcement learning on continuous control benchmarks [Mania et al., 2018].

The return of parameters $\boldsymbol{\theta}$ aggregates rewards over all subtasks and timesteps of an episode:

$$R(\boldsymbol{\theta}) = \frac{1}{|\mathcal{V}|} \sum_{v^{\text{target}} \in \mathcal{V}} \frac{1}{T} \sum_{t=1}^{T} r_t$$

with set of target speeds $\mathcal{V}$, episode length T, and rewards r_t obtained by the model with parameters $\boldsymbol{\theta}$. Each epoch samples random directions $\boldsymbol{\delta}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, evaluates the perturbed parameters $\boldsymbol{\theta} \pm \nu\boldsymbol{\delta}_i$, and updates parameters along the b best-performing directions:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \frac{\alpha}{b\,\sigma_R} \sum_{i \in \text{top-}b} \big(R(\boldsymbol{\theta} + \nu\boldsymbol{\delta}_i) - R(\boldsymbol{\theta} - \nu\boldsymbol{\delta}_i)\big)\,\boldsymbol{\delta}_i$$

with perturbation scale ν, step size α, and standard deviation σ_R of the returns used in the update. Sign and range constraints are re-applied after each update. For hyperparameters, see Table 3.

Table 3 | **ARS hyperparameters.**

Hyperparameter	Value
Population size	1024
Top directions b	256
Perturbation scale ν	0.02
Step size α	0.03
Epochs	4000
Training rollout length	100 steps (3 s)
Test rollout length	200 steps (6 s)
Subtasks per population member	6

D.3 Computational resources

All experiments were run on a Linux cluster with 4 CPU cores, 16 GB of RAM, and 1 GPU (NVIDIA L40S, 48 GB VRAM) per run. Each training run takes about 36 h. The reported results comprise 19 training runs: Rat NCAP with 3 seeds for each of the 4 interfaces, 2 DOF ablations of Rat NCAP with 1 seed each, the ES-trained and PPO-trained MLPs with 1 seed each for the position and flexor-extensor interfaces, and Quadruped NCAP with 1 seed. Training totals about 684 GPU-hours, and bumpy-terrain tests and circuit ablations add about 4 GPU-hours.

E Analysis Methods

E.1 Behavioral measures

Evaluation protocol Behavioral measures are computed for 3 seeds from 15-s evaluation rollouts, using the final 10 s after a 5-s settling period (Figure 5 and Figure 7), and from the 6-s test episodes (Figure 6).

Step statistics Stance is defined as paw touch-sensor contact associated with the limb's RG extension phase, which removes brief contacts during swing and fills brief lift-offs during stance. A step cycle spans consecutive touchdowns of the same paw. Stance and swing durations are measured within each cycle, step frequency is the inverse of cycle duration, and step length is the horizontal distance traveled by the paw during swing (Figure 5A–C). Values outside 1.5 times the interquartile range at each target speed are excluded as outliers.

Gait classification Interlimb phases are computed for each step cycle as the touchdown time of each limb relative to the right forelimb, normalized by cycle duration. Each cycle is classified by comparing its interlimb phases with idealized phases: diagonal synchrony with left-right alternation for trot, left-right synchrony for bound, and sequential quarter-cycle offsets for walk, within a tolerance of 0.25 cycles for trot and bound and 0.125 cycles for walk. Gait classification reports the fraction of step cycles matching each gait (Figure 7).

Kinetics and kinematics Ground reaction forces are computed from paw contact forces and normalized by body mass (Figure 5F). Ground reaction forces and joint angles (Figure 5G) are resampled to 100 points over a normalized step cycle from touchdown to touchdown, then averaged across cycles.

Mechanical power The mechanical power of actuator j at timestep t is $p_{j,t} = \tau_{j,t}\,\dot{q}_{j,t}$, with actuator force $\tau_{j,t}$ and actuator velocity $\dot{q}_{j,t}$. Positive and negative mechanical power are the sums of $\max(p_{j,t}, 0)$ and $\min(p_{j,t}, 0)$ over limb actuators, averaged over timesteps (Figure 6C, D). Positive power reflects propulsive work, and negative power reflects energy absorbed by the joints, such as during braking.

E.2 Circuit perturbations

Evaluation protocol Perturbations are applied to trained models at test time without retraining, at 0.4 m/s, where the intact model trots, with 30 trials for each of 3 seeds.

RG weight noise Each RG synaptic weight w is perturbed multiplicatively as $(1 + \sigma\epsilon)\,w$ with $\epsilon \sim \mathcal{N}(0, 1)$, clipped so that each weight preserves its sign (Figure 7A). Noise is sampled independently for each trial, and σ ranges from 0 to 0.5 in increments of 0.025.

Cell-type activity scaling The outputs of targeted RG populations are multiplied by an activity scale k before projecting to their targets, with $k = 1$ for the intact circuit and $k = 0$ for complete ablation. Scaling is applied jointly to V0v and V0d populations (Figure 7B) or to V3f and V3a populations (Figure 7C), with k ranging from 1 to 0 in increments of 0.05.

F Architecture Baselines

Generic architectures serve as scientific controls rather than benchmarks, since a generic controller with higher return would still not map onto experimentally identified cell types or connectivity [Cao and Yamins, 2024]. MLPs test whether the interface or training algorithm alone explains Rat NCAP's behavior, and Quadruped NCAP tests whether its architectural additions are critical.

F.1 MLP training

We train 2 MLPs with the same observations, interfaces, rewards, and target speeds as Rat NCAP. The ES-trained MLP uses the same ARS algorithm and hyperparameters as Rat NCAP, isolating the effect of architecture. The PPO-trained MLP uses Proximal Policy Optimization (PPO) [Schulman et al., 2017], a standard deep reinforcement learning algorithm, providing a reference for reasonable return. The ES-trained MLP uses 2 hidden layers of 256 units with ReLU activations and outputs clipped to the action range. The PPO-trained MLP uses an actor with the same architecture and a Gaussian policy with a learned, state-independent standard deviation, and a separate critic with 2 hidden layers of 256 units with tanh activations. Both MLPs receive the target speed concatenated with proprioceptive and tactile observations. For PPO hyperparameters, see Table 4.

Table 4 | **PPO hyperparameters.**

Hyperparameter	Value	Hyperparameter	Value
Parallel environments	6144	KL early-stop threshold	0.05
Steps per update	40	Optimizer	Adam
Discount	0.97	Learning rate	3×10^{-4}
GAE λ	0.95	Entropy coefficient	0
Minibatch size	4096	Gradient clipping	None
Minibatch iterations	4	Advantage normalization	Yes
Clip ratio	0.2		

Rat NCAP achieves return comparable to or higher than both MLPs (Figure F.1). With flexor-extensor control, Rat NCAP reaches the highest return, while the ES-trained MLP plateaus lower and the PPO-trained MLP improves more slowly to a similar final return. With position control, the PPO-trained MLP eventually reaches slightly higher return than Rat NCAP, while the ES-trained MLP plateaus substantially lower. With both interfaces, Rat NCAP reaches high return early in training, consistent with the inductive bias of its architecture. Since the same ARS algorithm yields lower return for an MLP, the performance of Rat NCAP does not arise from the training algorithm alone.

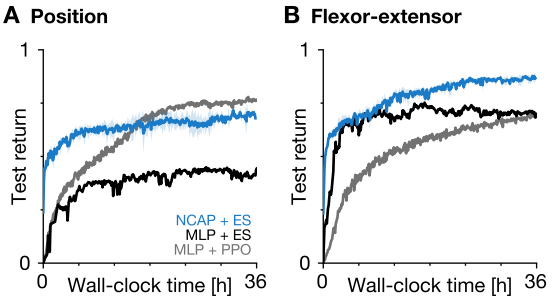


Figure F.1 | **MLP training.** Test return over wall-clock training time for Rat NCAP and ES- or PPO-trained MLPs. (**A**) Position control. The PPO-trained MLP eventually slightly exceeds Rat NCAP, while the ES-trained MLP remains well below. (**B**) Flexor-extensor control. Rat NCAP reaches the highest return. Rat NCAP data over 3 seeds with mean $\pm$ SD; MLP data from 1 seed each.

F.2 MLP behavior

Despite competitive return in some conditions, both MLPs produce markedly less naturalistic behavior than Rat NCAP with both interfaces (Figure F.2). Rat NCAP stands still at 0 m/s and transitions from walk to trot to bound as target speed increases, with mechanical power rising with speed. The MLPs instead step continuously even at 0 m/s, with irregular footfall patterns and no distinct gait transitions. The ES-trained MLP adopts abnormal postures with small, stereotyped steps, while the PPO-trained MLP produces large, erratic limb movements and expends substantially more mechanical power than Rat NCAP at every target speed. These results are consistent with prior work in this rat, in which generic architectures trained without imitation data produced unrealistic movement [Merel et al., 2019]. Together, these controls indicate that neither the neuromechanical interface nor the training algorithm alone explains the naturalistic behavior of Rat NCAP.

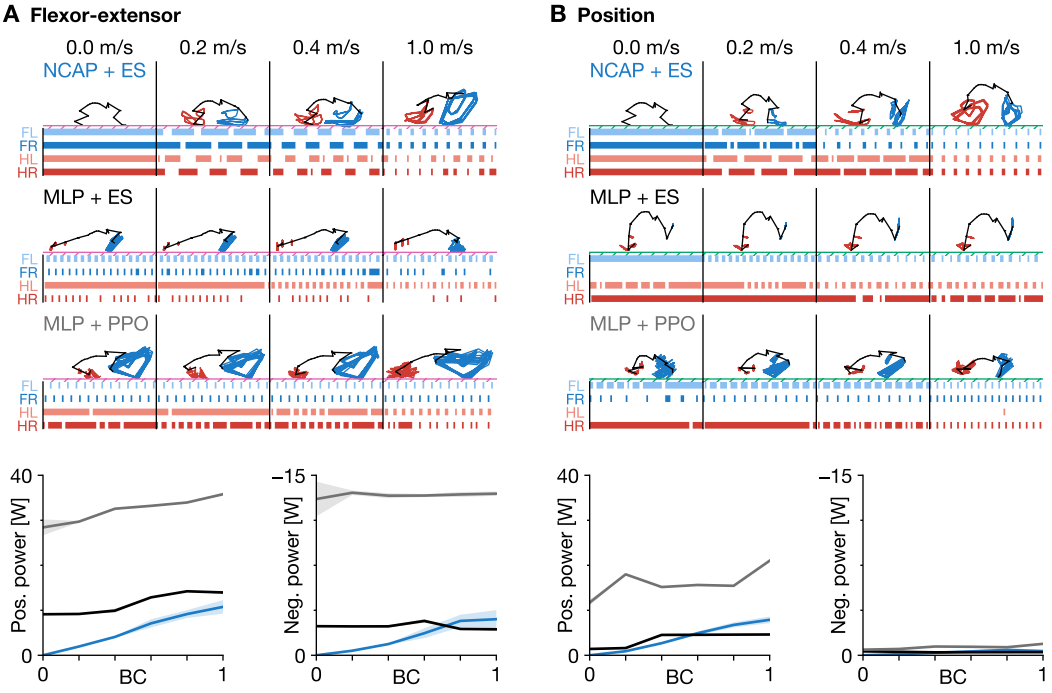


Figure F.2 | **MLP behavior.** Limb trajectories and footfall patterns at representative target speeds (top) and mean positive and negative mechanical power as brainstem command increases (bottom) for Rat NCAP (blue) and MLPs trained with ES (black) or PPO (gray). (**A**) Flexor-extensor control. (**B**) Position control. Rat NCAP stands at 0 m/s and shifts gaits with speed, while the MLPs step irregularly at all speeds and the PPO-trained MLP expends far more mechanical power.

F.3 MLP generalization

Rat NCAP generalizes to unseen bumpy terrain at least as well as both MLPs with both interfaces (Figure F.3). All models are trained only on flat ground and evaluated on terrain with randomly placed bumps of increasing height. The terrain is a heightfield constructed by drawing uniform random heights on a 5-cm grid and upsampling them with cubic-spline interpolation to a 2-cm mesh, producing smooth bumps about 5 cm across. Heights are scaled so that bump peaks lie at ground level and troughs lie one bump height below, with bump heights from 0 to 4 cm in 0.5-cm increments. With position control, Rat NCAP performs comparably to the PPO-trained MLP, while the ES-trained MLP collapses on bumps above about 1 cm. With flexor-extensor control, Rat NCAP achieves the highest return at all bump heights, while the ES-trained MLP drops sharply even on small bumps.

This experiment evaluates transfer to bumpy terrain, not general robustness to perturbations, obstacles, or sensory conditions. Such settings would likely also require vestibular and visual feedback and additional circuits such as the cerebellum, which are outside the scope of the present spinal model.

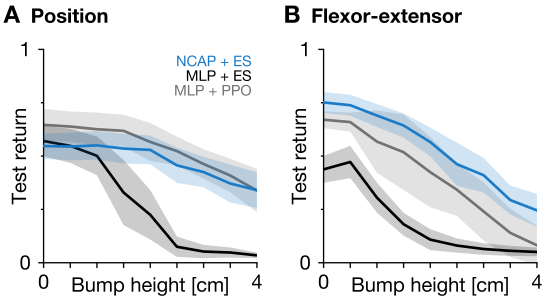


Figure F.3 | **MLP generalization.** Test return of models trained on flat ground and evaluated on terrain with increasing bump height. (**A**) Position control. Rat NCAP performs comparably to the PPO-trained MLP, while the ES-trained MLP collapses. (**B**) Flexor-extensor control. Rat NCAP retains the highest return at all bump heights. Mean ± SD over 128 episodes per bump height.

F.4 Quadruped NCAP training

Rat NCAP achieves about twice the return of Quadruped NCAP on Rat v2 (Figure F.4). Quadruped NCAP [Bhattasali et al., 2024] is trained with the same rewards and ARS algorithm, and its AF and PF modules are adapted to the rat joints using the same sign-constraint priors as Rat NCAP. It plateaus early in training. This gap indicates that the architectural additions of Rat NCAP, including the 2-layer AF module, phase-specific synergies, and trunk control, are collectively critical for scaling biologically grounded architectures from a simplified quadruped to a biorealistic rat body.

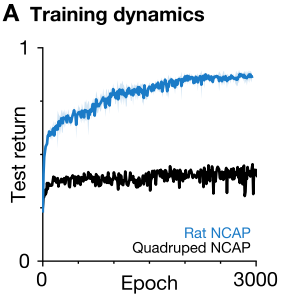


Figure F.4 | **Quadruped NCAP training.** Test return over training epochs for Rat NCAP and Quadruped NCAP on Rat v2. Quadruped NCAP plateaus early at about half the return of Rat NCAP. Rat NCAP data over 3 seeds with mean ± SD; Quadruped NCAP data from 1 seed.

G Body Ablations

Rat v2 has non-planar limb DOFs and lumbar DOFs that Quadruped NCAP did not control. Quadruped NCAP controlled only the planar hip and knee DOFs of each leg, since planar control suffices for locomotion in a quadruped robot and actuating hip abduction was neither necessary nor desirable [Bhattasali et al., 2024]. Ablating each DOF group tests its contribution to locomotion.

G.1 DOF ablations

We ablate 2 DOF groups: non-planar limb DOFs (scapula supination and abduction, shoulder supination, and hip supination and abduction) and 3 lumbar DOFs (extension, bending, and twisting). Each group is held near the default pose by passive spring-damper torques, as for the head, cervical, and tail DOFs, and Rat NCAP is retrained.

Constraining non-planar limb DOFs substantially lowers test return (Figure G.1B). Unlike in the quadruped robot, these DOFs are therefore important for locomotion in a biorealistic body. In contrast, constraining lumbar DOFs does not lower test return (Figure G.1A).

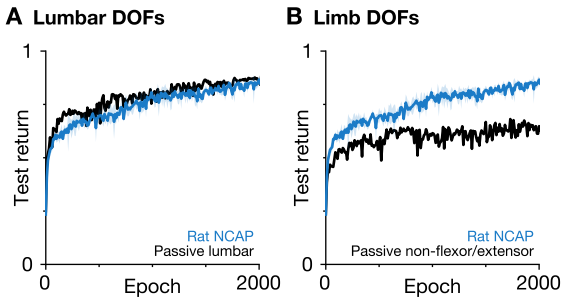


Figure G.1 | **DOF ablations.** Test return over training epochs for Rat NCAP with all DOFs active and with a DOF group held by passive springs. (**A**) Lumbar DOFs. Return is unaffected, suggesting compensation through other DOFs. (**B**) Non-planar limb DOFs. Return drops substantially, indicating that these DOFs are important for locomotion. Rat NCAP data over 3 seeds with mean ± SD; DOF ablation data from 1 seed each.

G.2 Lumbar kinematics

Although constraining lumbar DOFs does not lower return, Rat NCAP uses them in a speed-dependent manner when available (Figure G.2A). At low speeds, lumbar bending and twisting oscillate once per step cycle through alternating flexor and extensor activations. As speed increases, bending and twisting diminish and vanish in bound, while lumbar extension oscillates with larger amplitude. With passive lumbar DOFs, these oscillations are largely absent (Figure G.2B), so Rat NCAP likely compensates through other DOFs.

Thus, some DOFs admit compensation while others do not, and test return alone may not reveal how the body is used. Comparing these lumbar kinematics with experimental measurements of trunk motion in rats is an important direction for future work.

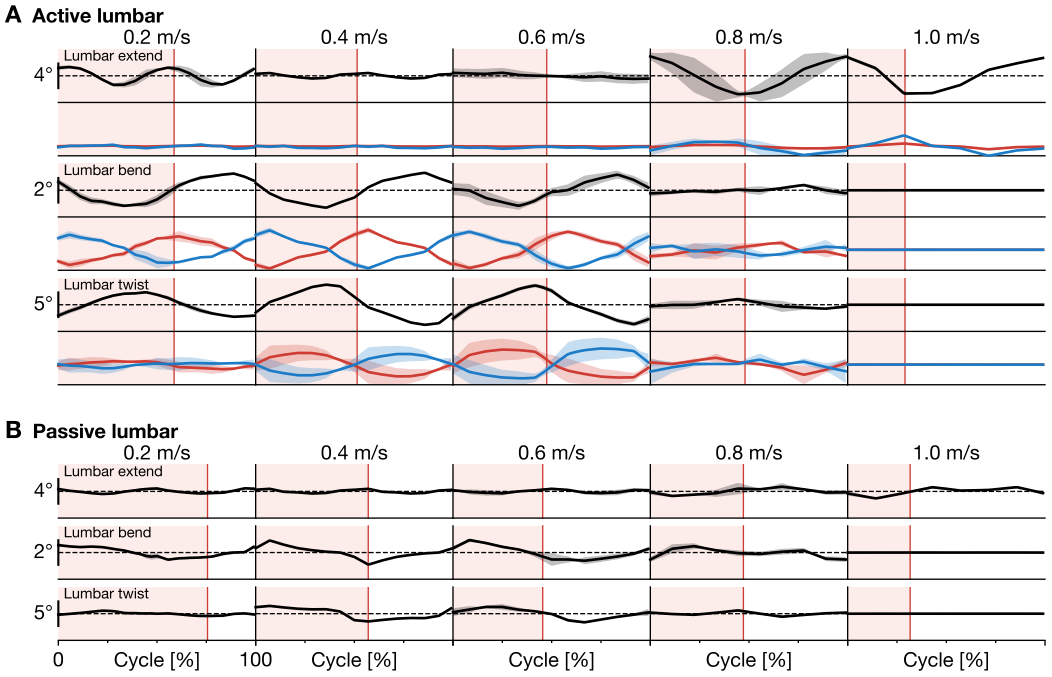


Figure G.2 | **Lumbar kinematics.** Lumbar joint angles (black) and flexor and extensor activations (**A** only) over the hindlimb step cycle at each nonzero target speed, with stance shaded and liftoff marked. (**A**) Active lumbar DOFs. Bending and twisting dominate at low speeds, while extension dominates at high speeds. (**B**) Passive lumbar DOFs. Lumbar motion remains near the default pose throughout the step cycle. Data from 1 seed per panel with mean $\pm$ SD over step cycles.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction enumerate contributions and the main empirical results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in the Discussion (Section 5).

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not include theoretical results; all contributions are empirical.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The neural circuit architecture, neuromechanical interfaces, training algorithm, reward functions, and hyperparameters are described in the Methods (Section 3) and Appendices (Appendix A–Appendix G). Code is released at https://rat-ncap.github.io/.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is released at https://rat-ncap.github.io/.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The details and hyperparameters of the model, algorithm, and environment are described in the Appendices.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Training curves and quantitative comparisons report error bars and figure captions state the source of variability.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Computational resources, including per-run training time, are reported in Appendix D.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: This work involves computational simulation only and raises no ethical concerns.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This is foundational neuroscience research with no foreseeable direct path to harmful applications.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The key released assets are a biomechanical simulation body, a neural circuit model, and simulation and training infrastructure; they pose no foreseeable misuse risk.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Third-party assets are cited; their licenses are permissive and compatible with our use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Rat v2 and Rat NCAP are documented and released with code at https://rat-ncap.github.io/.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were not used as part of the core methods.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.