NeuroMechFly 2.0, a framework for simulating embodied sensorimotor control in adult *Drosophila*

Sibo Wang-Chen[1,*], Victor Alfred Stimpfling[1], Pembe Gizem Özdil[1,2], Louise Genoud[1], Femke Hurtak[1], and Pavan Ramdya[1,*]

[1]Neuroengineering Laboratory, Brain Mind Institute & Interfaculty Institute of Bioengineering, EPFL, Lausanne, Switzerland
[2]Biorobotics Laboratory, EPFL, Lausanne, Switzerland
[*]Correspondence: sibo.wang@epfl.ch, pavan.ramdya@epfl.ch

Abstract

Discovering the principles underlying the neural and biomechanical control of animal behavior requires a tight dialogue between real experiments and data-driven neuromechanical models. Until now, such models have primarily been used to further our understanding of lower-level motor control. For most whole-animal simulations, we still lack an effective framework for studying how the brain processes environmental signals to regulate motor behavior. The adult fly, *Drosophila melanogaster*, is well-suited for data-driven modeling and can be simulated using the neuromechanical model, NeuroMechFly. However, until now this simulation framework did not permit the exploration of full hierarchical sensorimotor loops. Here we present NeuroMechFly 2.0, a framework that greatly expands whole-animal modeling of *Drosophila* by enabling visual and olfactory processing as well as complex three-dimensional environments that can be navigated using leg adhesion. To illustrate its capabilities we explore the effectiveness of biologically-inspired leg controllers for navigating diverse terrain, and show how one can build and use Reinforcement Learning to train an end-to-end hierarchical model with multimodal sensory processing, descending commands, and low-level motor control in closed loop. NeuroMechFly 2.0 can accelerate the discovery of explanatory models of the nervous system and the development of machine learning models to control autonomous artificial agents and robots.

Introduction

Achieving a holistic understanding of how the nervous system controls behavior will require solving a deeply entangled problem: Biological feedback loops occur at multiple hierarchical levels ranging from the physiology of individual neurons to the dynamics of neural circuits; physical feedback loops include the impact of biomechanical interactions with the environment as well as the flow of sensory information while navigating through the world. Thus, the investigation and modeling of any of these elements in isolation, although a useful starting point, will always be fundamentally incomplete. To overcome this gap, neuroscience requires integrated simulation frameworks that enable the exploration of hierarchical feedback loops in an end-to-end manner. This will permit a deeper understanding of how biological systems achieve efficient autonomous control through sensory processing, decision making, and motor execution.

Numerous detailed neuromechanical models have been developed to explore how animals control low level motor programs like walking[1,2,3,4,5,6,7], swimming[8,9], and transitions between them[10,11]. By contrast, in the field of Reinforcement Learning (RL) more abstract and simplified "creatures"[12] have been widely used to model high-order neural processes including visuomotor coordination[13,14], decision making[15,16], and learning algorithms[17,18]. These latter models often lack realistic motor control and biomechanics and, as a consequence, typically generate only simplified control signals that drive unrealistically limited categorical variables or joint degrees of freedom[19,20]. Thus, although progress is being made[21,22], a significant gap remains at the interface between neuromechanical and machine-learning models of most animals. Ideally, a simulation environment would enable the exploration of *hierarchical* controllers[23]—like those found in biological agents—which include higher-order (i.e., brain-like) systems that integrate multimodal sensory inputs and internal states to make appropriate decisions, as well as lower-level (i.e., spinal cord- or ventral nerve cord-like) systems that decode descending brain commands and execute behaviors **(Fig. 1a)**.

The adult fruit fly, *Drosophila melanogaster*, is an ideal organism for modeling biological hierarchical controller in its entirety. It has only $\sim$ 200,000 neurons in its brain[24] (compared with $\sim$ 70 million in the mouse brain[25] and $\sim$ 80 billion in the human brain[26]). Its principal motor system (the ventral nerve cord or VNC) comprises $\sim$ 15,000 neurons[27] (compared with $\sim 10^6$ in the mouse spinal cord[28] and $\sim 10^8$ in the human spinal cord[29]). Using this small number of neurons, flies can nevertheless generate complex behaviors including walking over complex terrain[30], rapid corrective maneuvers during flight[31], courtship sequences[32], aggression towards competitors[33], and learning[34]. The global organization of the fly's nervous system resembles that of vertebrates[35,36] yet it can be examined at the synaptic level using connectomes—synapse-resolution neural wiring diagrams—of the whole brain[37] and VNC[38,39]. Additionally, thousands of transgenic fly lines have been generated that allow investigators to repeatedly target sparse sets of neurons[40] for optogenetic activation[41] and recording[42]. These resources and tools have led to the emergence of data-constrained models of neural circuits (e.g.,[43,44]). Therefore, an integrated, whole-body simulation of the fly is urgently needed to expand, test, and iteratively improve whole brain and VNC models in order to uncover fundamental principles governing biological intelligence and autonomous behavioral control.

Towards this goal, we previously developed NeuroMechFly, the first morphologically realistic bio- and neuromechanical model of the adult fly, *Drosophila melanogaster*[22]. Using this biomechanical model, we could infer unmeasured forces and collisions from the *in silico* replay of real 3D limb kinematics measured from tethered flies walking and grooming on a spherical treadmill. Furthermore, with this model we showed how simple, mathematical central-pattern generators (CPGs) could be used to control fast and stable tethered locomotion. These efforts established the core motor system functionality of our data-driven simulation of *Drosophila*. However, they did not enable full end-to-end modeling of multi-level sensorimotor control. First and foremost, brain-level processing of sensory signals were absent. Second, the simulation framework did not confront the model with salient sensory objects and challenging terrain. Third, the physics engine and biomechanical system were not outfitted for effective untethered behavior including using leg adhesion to navigate three-dimensional terrain.

Here we describe NeuroMechFly 2.0, a simulation framework that fills these important gaps by (i) improving the biomechanical model, (ii) adding leg adhesion, (iii) simulating sensing by the retinas, antennae, and maxillary palps for vision and olfaction, and (iv) adding environmental features like odor sources, obstacles, moving objects, and rugged terrain **(Fig. 1a-b)**. We illustrate how these can be used to explore the control of locomotion over challenging terrain, visual object chasing, and

olfactory chemotaxis taxis. Finally, we show how our simulation framework can be used as a test bed for building machine learning models by solving an integrative multimodal task—avoiding a visual obstacle while approaching an attractive odor source over rugged terrain—using an artificial neural network controller trained through Reinforcement Learning (RL). By providing an implementation compliant with a standard RL task interface, we hope to facilitate a deeper dialogue between neuroscience, machine learning and robotics **(Fig. 1c)**. To accelerate its adoption in both research and education, NeuroMechFly 2.0 is modular, allowing users to flexibly interact with the simulation at multiple levels of abstraction.

Results

The FlyGym package: A standardized simulation framework implemented in MuJoCo

To improve its usability, we made three fundamental changes to the NeuroMechFly simulation framework (released as the FlyGym Python package, `https://neuromechfly.org/`). First, this package fully complies with Gymnasium[12], a standard interface for controller-environment interaction in robotics and RL **(Fig. 1c)**. Second, we moved the simulation framework from PyBullet to MuJoCo[45], a more widely used physics simulator. We changed the underlying physics engine because (i) MuJoCo is known for better stability and performance[46], (ii) MuJoCo supports a wide range of actuators including adhesion actuators that can be used to model insect-like adhesive properties at the leg tips, and (iii) MuJoCo has been made open-source since our previous results were published. Third, we expanded the interface for the arena (the physical environment that the fly model interacts with) to enable complex and dynamic environmental features within the simulation.

We framed the control problem as a partially observable Markov Decision Process (MDP). At each time step, the simulation provides the controller with an observation and, optionally, a user-defined reward. Then, the simulation receives an action from the controller and steps the physics forward accordingly. The observation space is a user-configurable subset of the state space including visual inputs, olfactory inputs, ground contacts, joint states (angles, angular velocities, and torques), and the position and velocity of the fly model within the arena. The action space includes the control signal (e.g., angles for position control) for every actuated joint (e.g., seven degrees-of-freedom (DoFs) per leg $\times$ six legs) and the on/off signal for leg adhesion. This framework is easily extendable: the user can incorporate additional layers of sensory preprocessing or premotor computation into the MDP **(Fig. 1c)**. For example, in a visual taxis example described below, we programmed the centroid calculation to reduce the observation space to two dimensions (the azimuth of the object seen from each eye) and used a CPG network to reduce the action space to a two-dimensional descending command.

Improved morphological accuracy

Morphologically accurate biomechanical models like NeuroMechFly allow researchers to study precise, fine-grained body movements that would not be faithfully represented by simpler geometric ("ball-and-stick") models. One major opportunity afforded by such models is the ability to replay real recorded 3D body part kinematics in simulation to infer unmeasured collisions, contacts, and forces[22]. Behaviors based on different body-part contacts like grooming (as opposed to world-body contact like walking) require precise kinematic replay as well as more fine-grained read-out of where and how individual segments interact with one another. Therefore, we updated the accuracy and granularity of individual body part meshes to better match the morphology of the real fly. First, we adjusted the placement and default angles of the joints between the thorax and front leg coxae as well as between the thorax and head **(Extended Data Fig. 1a–b)**. These adjustments were made based on high-magnification video data collected with a multi-camera keypoint tracking system[47]. Second, to better facilitate the control and the accurate readout of mechanosensory signals in the antennae, we split each antennal mesh into three independent segments: pedicel, funiculus, and arista. We then added degrees of freedom (DoFs) between these segments to allow each to be separately actuated or passively moved, and to measure their angular displacement **(Extended Data Fig. 1c)**.

Leg adhesion enables locomotion in three dimensions

Insects, including flies, have evolved highly specialized adhesive structures to facilitate locomotion over complex 3D terrain[48,49]. Adhesive forces are in part mediated by adhesive pads[50]. Substantial normal forces (10–100 times body weight[51,52,53]) and frictional forces emerge from interactions between the substrate and the adhesive pad[49,54]. These allow insects to navigate 3D terrain with ease. Because we cannot fully represent the physics underlying real, biological adhesion, we added more abstract leg adhesion to our model by injecting an additional normal force to the pretarsus of each leg when it is in contact with a surface **(Fig. 2a, Supplementary Video 1)**. This adhesive force increases the normal force toward the object and the frictional force (in insects, normal and frictional forces are also correlated[52]). We opted for a relatively high adhesion force (20–40 mN) to be consistent with the high, but somewhat variable, forces reported across insect species[52,51]. In this basic form, adhesion provides mechanical coupling between the legs during locomotion on flat terrain and improves force transduction to the ground.

Despite the huge forces generated by adhesive pads, insects can still lift their legs, seemingly without effort. The mechanisms for lifting off are known for some insect species—the beetle *Gastrophysa viridula* seems to utilize the anisotropy (i.e., direction-dependence) of the adhesion pads[51] and the fly *Calliphora vicina* uses lifting and twisting strategies to overcome adhesive forces[49]—but not for *Drosophila melanogaster*. Therefore, we abstracted the mechanisms used by other insects for lifting by turning adhesion forces on during stance and off during swing.

Leg adhesion expands the behavioral repertoire of our model. To illustrate this, we simulated tripod locomotion[55] over 3D surfaces[56,57] inclined at different angles up to 120° **(Fig. 2b, Supplementary Video 2)**. Individual steps were generated by replaying the kinematics of real steps measured from a tethered, behaving fly[47] **(Extended Data Fig. 2, Supplementary Video 3, see Methods)**. Without adhesion, the fly failed to sustain forward locomotion at angles as small as 5 degrees. However, as the maximal adhesion force was increased, the fly could locomote over increasingly inclined terrain even occasionally beyond 90° **(Fig. 2c)**.

Complex terrain demonstrates the importance of feedback in locomotion

Previously, we showed that replaying the real leg kinematics of a tethered walking fly could generate stable walking in NeuroMechFly on flat ground[22]. Although this was an important proof-of-concept, locomotion over smooth terrain is an under-constrained problem that can be solved using a variety of different control strategies. We previously also used an open-loop, abstract network based on central pattern generators (CPGs) to generate fast and stable tethered walking. However, this approach did not allow one to investigate the role played by leg mechanosensory feedback for navigating over more challenging terrain. Therefore, to enable the exploration of how sensory feedback can facilitate motor control over rugged surfaces, we developed three rugged terrain types that complement our baseline smooth terrain **(Fig. 2d, "Flat terrain")**: one with gaps perpendicular to the initial heading of the fly **(Fig. 2d, "Gapped terrain")**, one with blocks of variable height **(Fig. 2d, "Blocks terrain")**, and one that is a mixture of these two **(Fig. 2d, "Mixed terrain")**. Using these new terrain types, we could examine the efficacy of different bioinspired control strategies.

Two prominent mechanisms have been proposed for insect locomotion. The first is a centralized mechanism that depends on coupled CPGs (i.e., circuits producing rhythmic motor output without rhythmic input[58]). The second is a more decentralized approach whereby locomotion emerges from the application of sensory feedback-based rules that dictate for each leg when to lift, swing, or remain in stance phase[1]. Evidence for each of these control approaches has been found across species[58,59,60,61], motivating their application toward robotic motor control[62,63]. To explore the efficacy of these distinct strategies for navigating complex terrain, we built three controllers: (i) CPG-based **(Fig. 2e, "CPG controller", Supplementary Video 4)**, (ii) sensory feedback-based **(Fig. 2e, "Rule-based controller", Supplementary Video 5)**, and (iii) a hybrid controller that modulates CPGs with sensory feedback to recover from challenging positions **(Fig. 2e, "Hybrid controller", Supplementary Video 6)**. As before, the kinematics of individual steps were based on experimentally recorded tethered walking but these were influenced by CPGs and sensory feedback rules (see Methods). In later sensorimotor tasks, we couple the CPG and hybrid controllers with a two-dimensional asymmetric descending signal to control turning by modulating oscillator frequencies and amplitudes **(Fig. 2f)**.

We benchmarked the performance of these controllers by testing them over flat and rugged terrains starting from different spawn positions and initial states. We found that the CPG controller led to fast locomotion while rule-based locomotion is more robust over rugged terrain. By leveraging both CPGs and mechanosensory feedback, the hybrid controller had a higher baseline speed than the rule-based controller **(Fig. 2g, left, Supplementary Video 7)** while also covering greater distances on rugged terrain **(Fig. 2g, right, Supplementary Video 7)**. These results demonstrate that rugged terrains are important for the study of locomotor control: they can expose failure modes of controllers that otherwise work well on flat terrain.

Vision and olfaction enrich behavior in complex sensory environments

Animals typically navigate over rugged terrain to reach attractive objects (e.g., potential mates, food sources) and to avoid repulsive features (e.g., pheromones from predators) and obstacles. To achieve these goals, a controller must be hierarchical: processing higher-order sensory signals, using them to select the next action, and translating these decisions into descending commands that drive lower-level motor systems. We aimed to simulate this sensorimotor hierarchy by adding vision and olfaction to NeuroMechFly **(Fig. 3a)**.

We first implemented vision. A fly's compound eye consists of ∼700–750 individual units called ommatidia arranged in a hexagonal pattern[64]. To emulate this, we attached a color camera to each of our model's compound eyes **(Fig. 3a, green)**. We then transformed each camera image into 721 bins, representing ommatidia **(Fig. 3b)**. Based on previous studies[65,66], we assumed a 270° combined azimuth for the fly's field of view, with a $\sim 17°$ binocular overlap **(Extended Data Fig. 3)**. Visual sensitivity has evolved to highlight ethologically relevant color spectra at different locations in the environment (see Methods). Here, as an initial step toward enabling this heterogeneity in our model, we implemented yellow- and pale-type ommatidia—sensitive to the green and blue channels of images rendered by the physics simulator—randomly assigned at a 7:3 ratio[67]. Users can substitute the green and blue channel values with the desired light intensities sensed by yellow- and pale-type ommatidia[68] to achieve more biorealistic chromatic vision.

In addition to vision, we also made it possible for our model to detect odors emmited by objects in the simulation environment. The olfactory system in *Drosophila* consists of specialized olfactory sensory neurons (OSNs) located in the antennae and maxillary palps. These detect specific odorant molecules[69] and convey this information to the brain's antennal lobe, where their signals are further processed[70]. We emulated peripheral olfaction by attaching virtual odor sensors to the antennae and maxillary palps of our biomechanical model **(Fig. 3a)**. The user has the option of configuring additional sensors at more precise locations on these olfactory organs. These virtual sensors can detect odor intensities across a multi-dimensional space that can be thought of as representing, for example, the concentrations of monomolecular chemicals sensed by OSNs in the antennae, or the intensities of composite odors co-activating numerous projection neurons in the antennal lobe. The modularity of our framework makes it possible for users to implement additional signal processing by downstream olfactory centers (e.g., lateral horn or mushroom body[71,72]).

To illustrate how our model can use visual and olfactory sensing to interact with higher-order features in the environment, we implemented two behaviors: visual object chasing and olfactory chemotaxis. In our visual object chasing task, the fly model was required to track and follow a black sphere moving in an S-shaped curve—akin to a male fly chasing a female during courtship[73]. We first processed the object's visual location by computing it's centroid position on the retina. Then, these visual features were linearly transformed into a two-dimensional descending signal **(Fig. 3c, left)**. These modulated the frequencies and amplitudes of CPG-based oscillators on either side of the body **(Fig. 2f, "Turning controller")**, allowing the fly to effectively track the object moving through the environment **(Fig. 3c, right; Supplementary Video 8)**.

Similarly, to illustrate olfactory chemotaxis—akin to odor-based food search, but simplified by the absence of complex plumes[74]—we required the fly model to seek an attractive odor source while avoiding two aversive odor sources. To achieve this, the controller used the antennae and the maxillary palps to compare the relative intensities of attractive and aversive odors across the two sides of the head[75] and multiplied them by the appropriate weights (opposite signs for attractive versus aversive odors). This left-right bias was used to asymmetrically control the descending signal **(Fig. 3d, left)**, yielding effective odor-driven navigation through the simulation environment **(Fig. 3d, right; Supplementary Video 9)**.

Finally, we sought to demonstrate the capacity of our simulation framework to use machine learning to learn a controller for whole-body, multi-modal behavior in a modular fashion. Specifically, we trained the fly to avoid an obstacle while also seeking an attractive odor source by locomoting on rugged terrain **(Fig. 1b)**. Our controller consisted of the following: (i) a vision module (implemented as a convolutional neural network on the hexagonal lattice) that extracts the object's direction, distance from the fly, location on the retinas, and size on the retinas **(Extended Data Fig. 4)**; (ii) a decision-making module that receives preprocessed visual features from the vision model and odor intensities detected from each antenna as inputs to compute a turning bias; (iii) a two-dimensional descending representation, transformed from the turning bias, that modulates locomotor CPGs to drive turning; and (iv) a hybrid controller integrating CPGs and sensory feedback to enable locomotion on rugged terrain. We trained the vision module in a supervised manner by randomly placing the fly and obstacle in the arena to collect training data and learned the decision making module through RL. This approach enabled multimodal visual-olfactory navigation despite the challenges of locomoting over rugged terrain **(Fig. 3e, right; Supplementary Videos 10 & 11)**. This integrated task demonstrates that with NeuroMechFly 2.0, one can define individual components in a modular fashion and combine them to investigate a hierarchical sensorimotor task in closed loop.

Although we have used simpler models in the examples above, the user has full access to the raw light intensities sensed by the retinas as well as the raw concentration of chemicals or odorants sensed by the antennae and maxillary palps. Thus, one can implement one's own sensory processing model with the level of biological realism required to address the question at hand (e.g., visual optic flow algorithms[76], connectome-constrained visual feature extraction[43], olfactory gain control[77], odor motion sensing algorithms[78], olfactory memory formation[72]; see **Table 1**).

Discussion

Here we have introduced NeuroMechFly 2.0, a framework for performing integrated sensorimotor neuromechanical simulations of the adult fly *Drosophila melanogaster*. This new framework includes fundamental updates including (i) the development of "FlyGym," an open-source Python package based on the MuJoCo physics simulator that interfaces easily with user-specified neural controllers and RL algorithms, (ii) complex, rugged terrain types that can be modified and used to benchmark the efficacy and robustness of different motor control strategies, (iii) improved morphological realism of the biomechanical model, (iv) leg adhesion enabling 3D locomotion, (v) visual and (vi) olfactory systems that allow the simulated fly to interact with higher-order sensory features in the environment, (vii) a descending interface between higher-level sensing and lower-level locomotion, and (viii) the ability to use machine learning approaches to perform whole-body, closed-loop simulation of multimodal navigation. Beyond the particular examples presented here, our generalizable framework allows researchers to develop more biologically realistic models. We have provided a summary of what our simulation framework currently supports, what we have used to implement the examples in this paper, and some potential opportunities for future work **(Table 1)**. Importantly, our simulation framework is modular, allowing the community to build whole-animal models in an interoperable manner: Researchers can choose the appropriate level of detail for each part of the model in a way that suits the scientific question under consideration. For example, one can use more abstract baselines or existing models for elements of neural control that are not the focus of investigation. Notably, important behaviors, including those involving the control of wings/halteres (e.g., flight), abdomen (e.g., egg-laying), and proboscis (e.g., feeding) are not yet demonstrated but are supported within this framework. Additionally, signals like ascending motor-related feedback[79] and those encoding internal states[33] or learning-based[72] changes in sensory valence can be easily incorporated into user-defined controllers.

With enriched sensory feedback and improved biomechanics, our simulation framework makes a number of new modeling and experimental paradigms possible. First, it can enable the modeling of complex behaviors requiring controllers that span sensing, navigation, learning, and motor control within a whole-body simulation framework. We envision that these explorations will facilitate our understanding of the hierarchical organization of information processing in the nervous system. In the future, a similar simulation framework could be integrated into closed-loop experiments. For example, NeuroMechFly could be used during experiments to replay an animal's kinematics as captured by pose estimation methods[80], enabling the real-time inference of dynamic variables such as forces and

torques and informing experimental perturbations in closed-loop[81].

Beyond this current implementation, we anticipate that in the future our simulation framework can be further improved in several ways. First, we anticipate that recent developments in physics simulation, particularly GPU-acceleration[82,83] and differentiable simulation[84,85,83], will enable advances in FlyGym that facilitate the training of larger models through RL. Second, as more detailed, possibly connectome-constrained models of specific neural circuits emerge, they can be added to a corpus of controllers in our modular simulation framework. Furthermore, its compatibility with the Gymnasium API will ensure that changes are implemented relatively easily without disrupting the established user interface. Third, the careful measurement and analysis of the *Drosophila* musculoskeletal system (i.e., tendons and muscles) will improve the interface between connectome-inspired neural networks and control of the biomechanical model. These efforts will bring the field closer to achieving the ultimate goal of uncovering neuromechanical mechanisms giving rise to flexible and adaptive animal behaviors in a sensory-rich and physically complex world.

Figures

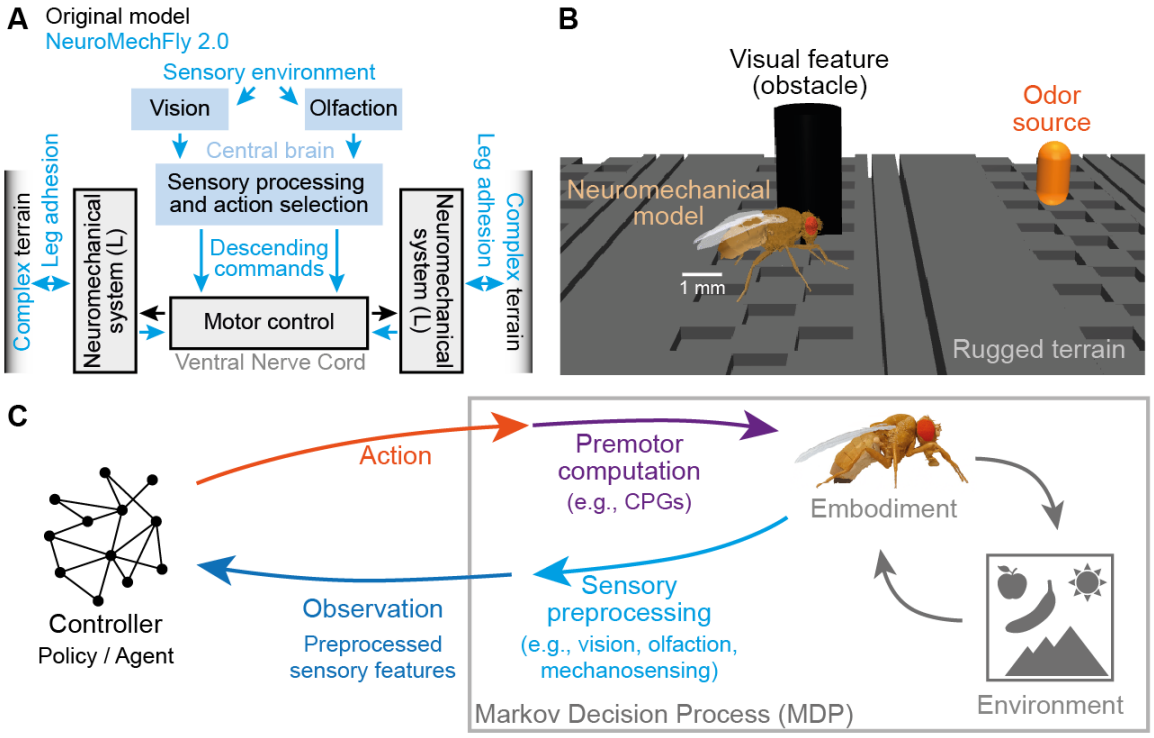


Fig. 1: **Schematic overview of the NeuroMechFly 2.0 modeling framework.** **(a)** Diagram highlighting the integrative and hierarchical aspects of the newly expanded NeuroMechFly framework. Sensory inputs—visual and olfactory—are processed in the central brain to select an appropriate action as driven by a left-right pair of descending inputs to lower motor centers (Ventral Nerve Cord or VNC). Motor output drives joint torques and mechanosensory feedback signals are sent back to the motor system to inform the next movement. Indicated are components that were present in the original model but improved (black) as well as components that are entirely new (blue). **(b)** Simulation camera view of the neuromechanical model, NeuroMechFly, locomoting over complex terrain, using vision to avoid a pillar, and using olfaction to reach an attractive odor source (orange). **(c)** The biomechanical model and its interaction with the environment are encapsulated as a Markov Decision Process (MDP) task. A user-defined controller interfaces with the task through actions (red) and observations (blue). The user can extend the MDP task by adding preprogrammed processing routines for sensory inputs (purple) and motor outputs (light blue), to modify the action and observation spaces handled by the controller.

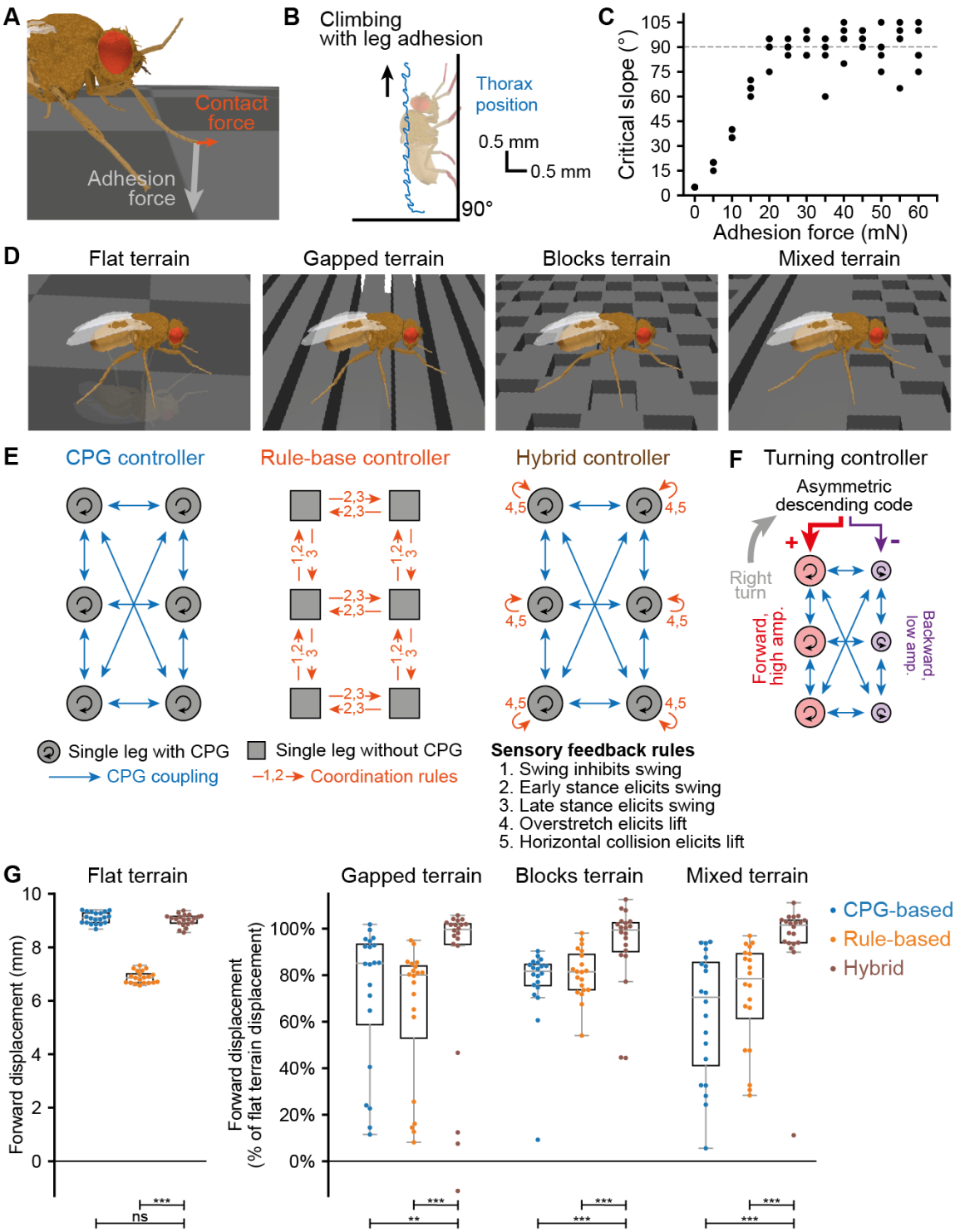


Fig. 2 - See Figure Legend on next page.

Fig. 2: **Locomotion in three-dimensions and over challenging terrain.** **(a)** A schematic of contact forces on the fly's leg with the addition of adhesion. **(b)** Trajectory (blue) of the fly as it walks up a vertical wall. Fly image is not to scale. The legs are controlled by a CPG controller with adhesion enabled (F=20 mN). **(c)** Critical slope (angle) at which the fly falls or does not proceed forward as a function of leg adhesion force. **(d)** Challenging terrains for exploring the efficacy of distinct locomotor strategies. We tested four terrains: a flat surface, a surface with gaps, a surface with blocks, and a mixture of all surface types. **(e)** Three controllers tested across terrains: a controller with six coupled CPGs controlling the swing and stance of the legs, a rule-based controller in which the phase of one leg's movements influences the movements of neighboring legs, and a hybrid controller consisting of coupled CPGs with sensory feedback-based corrective mechanisms to lift the leg when it may be stuck. See Methods for a detailed explanation of rules based on sensory feedback. **(f)** Turning is controlled by the asymmetric modulation of a two-dimensional descending command signal that regulates the frequencies and amplitudes of oscillators on each side of the body. **(g)** The performance (forward displacement) of each locomotor control strategy walking over four types of terrain. For flat terrain, the absolute displacement is shown. For rugged terrain types, the displacement is normalized to the mean displacement of the controller over flat terrain. $N = 20$ single trials with a random spawn location and controller initialization are shown (colored circles). Overlaid are box plots indicating the median, upper and lower quartiles, and whiskers extending to the furthest points excluding outliers that are more than $1.5\times$ the interquartile range (ICR) beyond the ICR. One-sided, asymptotic Mann-Whitney U rank test was used to generate the statistics: "ns" not significant; ** $p < 0.01$, *** $p < 0.001$ (see **Table 6** for complete statistics).

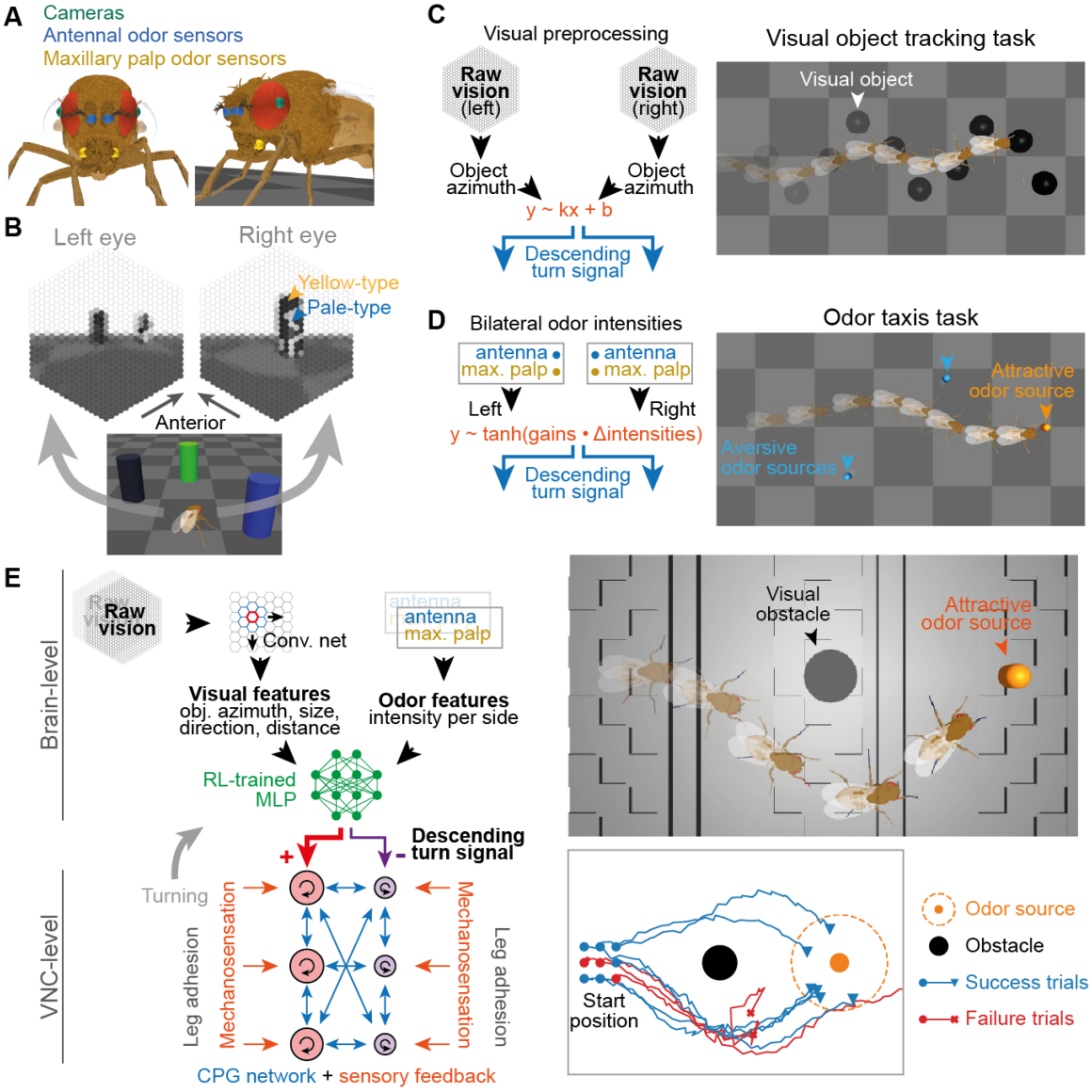


Fig. 3 - See Figure Legend on next page.

Fig. 3: **Vision and olfaction enable closed-loop sensorimotor control.** **(a)** The placement of cameras for vision (green), and odor sensors—antennae (blue) and maxillary palps (yellow)—on the fly model's head. **(b)** A simulation of the visual inputs perceived by the fly while facing three different colored pillars. Ommatidia perceive the same pillar in two different intensities due to the two-dimensional color space. Ommatidia types are randomly assigned. **(c)** The visual object tracking task. (left) First, the azimuth of the centroid of the object on the retina is extracted for each eye. These azimuths are then linearly transformed and capped to generate the appropriate descending turn signal to CPG-based controllers. (right) This controller allows the fly to follow a black sphere moving in S-shaped curve. **(d)** The odor taxis task. (left) First, the differences in mean odor intensities sensed by the antennae and maxillary palps on either side of the head are multiplied by the gains of the corresponding odor type. The product is then passed through a nonlinear function restraining its range and supplied to the descending controller as a turn signal (see Methods). (right) This controller allows the fly to seek an attractive odor source while avoiding two aversive odor sources. Note that the orange odor marker is for visualization purposes only and is not seen by the fly model. **(e)** The multimodal RL-based navigation task. (left) Visual features are extracted using a convolutional neural network. Olfactory features are processed as in panel **d**. These features are passed to an artificial neural network as inputs. The neural network is trained through RL to predict the appropriate turn commands. These commands are executed by a hybrid lower-level motor controller integrating CPGs and sensory feedback and using leg adhesion. (right) This controller allows the fly to seek an attractive odor source while avoiding a visually-detected obstacle over complex terrain. (top) An example of a successful trial. The orange odor marker is for visualization purposes only and is not visible to the fly. (bottom) The trajectory of the fly in 9 examples (6 successful, blue; 3 failed, red) beginning from different spawn positions (circle markers). Videos of these examples are compiled in **Supplementary Video 10**.

Tables

Visual system

Currently Supported	Shown in Examples	Future Potential
• Raw retina readings processed by any user-defined visual system model	• Object size / location on retina • Artificial neural network (convolution on a hexagonal lattice)	• Algorithmic visual system model[76] • Connectome and physiology-constrained visual system model[43] • Robotics/AI: Vision models at variable levels of abstraction

Olfactory system

Currently Supported	Shown in Examples	Future Potential
• Intensity readout in an n-dimensional space according to a user-defined diffusion relationship • User can decide where sensors are placed on the antennae and/or maxillary palps • Signals can be processed with any user-defined olfactory system model	• Readout of 2D (attractive vs. aversive) odor intensities by antennae and maxillary palps • Simple model computing the difference in odor intensities between bilateral olfactory organs	• Integration with dispersion models of odor plumes[86] • Algorithmic odor plume processing model[78,87,88] • Connectome and physiology-constrained olfactory system model[71] • Robotics/AI: Odor search models at variable levels of abstraction

Action selection model

Currently Supported	Shown in Examples	Future Potential
• Arbitrary user-defined process with access to visual/olfactory signals, ascending motor signals, and internal states	• Simple calculations based asymmetry in odor intensities or object position multiplied by gains • Black-box Multilayer Perceptron (MLP) trained by RL	• Algorithmic action selection model • Connectome and physiology-constrained action selection model[89,37] • Robotics/AI: Action selection models at variable levels of abstraction

Table 1: **(part 1 of 3) Summary of features supported by our modeling framework, implementations used in the examples presented in this paper, and future possibilities.** Continued on the next page.

Descending controller

Currently Supported	Shown in Examples	Future Potential
• n-dimensional descending space defined by the user	• 2D descending drive modulating CPGs on each side of the body	• High-dimensional descending representation allowing command-like and population-based coding[90]

VNC (motor control) model

Currently Supported	Shown in Examples	Future Potential
• Arbitrary user-defined VNC model/process with access to mechanosensory signals, internal states, and descending signals	• CPG-based controller[10] • Sensory feedback rule-based controller[91] • Hybrid controller integrating CPGs and sensory feedback rules	• Additional algorithmic motor system models • Connectome and physiology-constrained motor system model[38,92] • Robotics/AI: Hexapod motor system models at variable levels of abstraction[93]

Ascending feedback

Currently Supported	Shown in Examples	Future Potential
• n-dimensional ascending space defined by the user	• None	• High-dimensional ascending representation encoding behavioral and internal states at variable levels of abstraction[79] • Robotics/AI: Ascending motor signals at variable levels of abstraction[94]

Behavioral kinematics

Currently Supported	Shown in Examples	Future Potential
• User-defined limb kinematics, either estimated from recordings of real animals, or generated by models • User-defined sets of actuated versus passive DoFs • User-defined rules to turn adhesion on/off	• Each walking step: 42 actuated leg DoFs (7 per leg) playing out preprogrammed stepping sequences • Stepping coordination: see VNC model • Adhesion on/off periods defined as a part of preprogrammed stepping sequences	• Inclusion of motor neuron and muscle models[95,96] • Real-time replay of animal behavior enabling closed-loop experimentation[81] • Robotics/AI: Body movement patterns learned by imitating animals[97]

Table 1 (part 2 of 3). Continued on the next page.

Environment

Currently Supported	Shown in Examples	Future Potential
• User-defined arena with static (e.g., rugged terrain, obstacles, odor sources) or dynamic (e.g., moving objects, moving odor sources) features	• Low-level features: Rugged terrain for benchmarking locomotor controllers • Higher-order features: Odors, obstacles, and a moving sphere for sensory-guided locomotion	• Aerodynamics for flight and odor plume propagation • Compliant objects • Complex, visually realistic terrain • Additional fly models for social behavior[32,33]

Parametrization / training algorithm

Currently Supported	Shown in Examples	Future Potential
• Manual parameter tuning or arbitrary user-defined optimization algorithm (e.g., genetic algorithms, RL algorithms)	• Manual parameter tuning • Gradient-based RL algorithm (SAC[98])	• Larger-scale simulation, optimization, and neural architecture search

Physics simulator

Currently Supported	Shown in Examples	Future Potential
• PyBullet (original version[22]) • MuJoCo[45]	• MuJoCo[45]	• GPU-accelerated and/or gradient-aware simulators for large-scale model training[82,84,85,83]

Table 1 (part 3 of 3).

Variable	Value
ϕ_{ij}	See Table 3, row i, column j
w_{ij}	10 if $i \neq j$, otherwise 0
ν_i	12Hz, unless modulated for turning (see "Control of turning")
R_i	1, unless modulated for turning (see "Control of turning")
α_i	20

Table 2: **Parameters of the CPG-based controller.**

	RF	**RM**	**RH**	**LF**	**LM**	**LH**
RF	0	π	0	π	0	π
RM	π	0	π	0	π	0
RH	0	π	0	π	0	π
LF	π	0	π	0	π	0
LM	0	π	0	π	0	π
LH	π	0	π	0	π	0

Table 3: **Phase biases between legs** (tripod gait). "L": left, "R": right; "F": front leg, "M": middle leg, "H": hind leg.

	Weight
Rule 1	-10,000 if in swing
Rule 2 (ipsilateral)	25,000/s, cumulative
Rule 2 (contralateral)	10,000/s, cumulative
Rule 3 (ipsilateral)	30,000/s, cumulative
Rule 3 (contralateral)	20,000/s, cumulative

Table 4: **Weights used in the rule-based controller.**

	Degree of freedom	**Increment per adjustment**	**Adjustment interval**			
			Retraction rule		**Stumbling rule**	
			Lifting	Recovery	Lifting	Recovery
Front leg	Coxa-trochanter pitch	-0.02 rad.	2ms	0.5ms	3ms	2ms
	Femur-tibia pitch	0.016 rad.				
Middle leg	Thorax-coxa pitch	0.015 rad.				
	Coxa-trochanter pitch	0.04 rad.				
	Femur-tibia pitch	0.01 rad.				
	Tibia-tarsus pitch	-0.008 rad.				
Hind leg	Coxa-trochanter pitch	-0.01 rad.				
	Femur-tibia pitch	0.005 rad.				

Table 5: **Parameters of mechanosensation-based joint angle adjustment rules.** Joint angles are adjusted by the specified increment at the specified interval during leg lift and recovery.

		CPG < hybrid	Rule-based < hybrid
Raw displacement	**Flat terrain**	$u = 231$, $p = 8.0 \times 10^{-1}$	$u = 0$, $p = 3.4 \times 10^{-8}$
	Gapped terrain	$u = 105$, $p = 5.3 \times 10^{-3}$	$u = 70$, $p = 2.3 \times 10^{-4}$
	Blocks terrain	$u = 57$, $p = 5.8 \times 10^{-5}$	$u = 38$, $p = 6.3 \times 10^{-6}$
	Mixed terrain	$u = 39$, $p = 7.1 \times 10^{-6}$	$u = 20$, $p = 6.0 \times 10^{-7}$
Normalized displacement	**Flat terrain**	$u = 196$, $p = 4.6 \times 10^{-1}$	$u = 185$, $p = 3.5 \times 10^{-1}$
	Gapped terrain	$u = 98$, $p = 3.0 \times 10^{-3}$	$u = 78$, $p = 5.1 \times 10^{-4}$
	Blocks terrain	$u = 53$, $p = 3.7 \times 10^{-5}$	$u = 73$, $p = 3.1 \times 10^{-4}$
	Mixed terrain	$u = 32$, $p = 2.9 \times 10^{-6}$	$u = 35$, $p = 4.3 \times 10^{-6}$

Table 6: **Statistics on the controller benchmark.** The u-statistics and p-values in the one-sided, asymptotic Mann-Whitney U rank test used to generate the statistics. "Raw displacement": displacement in the forward direction over 1 s. N=20 for each combination of controller and terrain type. "Normalized displacement": displacement in the forward direction over 1 s as a percentage of the mean forward displacement of the same controller over flat terrain.

Variable	Value
$\bar{\rho}$	40
k	-3
b	1
δ_{min}	0.2
δ_{max}	1

Table 7: **Parameters used in the controller for visual object chasing.**

Variable	Value
$\gamma_{\text{attractive}}$	-500
γ_{aversive}	80
δ_{min}	0.2
δ_{max}	1

Table 8: **Parameters used in the controller for olfactory chemotaxis.**

	Layer
1	GCNConv: 2 in channels, 4 out channels
2	Tanh
3	GCNConv: 4 in channels, 2 out channels
4	Tanh
5	Concatenation of left/right outputs
6	Linear: $2 \times 2 \times 721 = 2884$ inputs, 16 outputs
7	Tanh
8	Linear: 16 inputs, 16 outputs
9	Tanh
10	Linear: 16 inputs, 7 outputs

Table 9: **Parameters used in the vision module of the multimodal navigation model.**

Model	Variable	Value
Vision	$\bar{\rho}$	40
	Loss weights	$k_{\text{dir}} = 0.5,\ k_{\text{dist}} = 2,\ k_{\text{class}} = 0.1,\ k_{\text{azi}} = 4,\ k_{\text{size}} = 2$
	Learning rate	0.001
	Architecture	See Table 9
RL	m	4
	ϵ	2
	q	3
	Reward weights	$k_{\text{prox}} = 1,\ k_{\text{avoid}} = 7,\ k_{\text{attract}}^{(\text{near})} = 10,\ k_{\text{attract}}^{(\text{far})} = 1$
	Termination rewards	$R_{\text{success}} = 10,\ R_{\text{fail}} = -5$
	Learning rate	0.01
	Architecture	MLP with 2 hidden layers of size 32
DN/VNC	DN range	$\delta \in [0.2, 1]$

Table 10: **Parameters used in the multimodal navigation model.**

Methods

The FlyGym package

FlyGym is implemented based on MuJoCo[45] and the dm_control[99] library and complies with the Gymnasium API for MDP[12]. The user interacts with the simulation through *actions* and *observations* **(Fig. 1, Extended Data Fig. 5)**. While more detailed, low-level information can be accessed using dm_control directly, in the typical use case, the core simulation class `NeuroMechFlyMuJoCo` offers the following action and observation spaces:

- Action space:
  - *Joint actuation signals:* The target angle, velocity, or force (as specified by the user) of the n actuated joint degrees of freedom (by default all 42 leg degrees-of-freedom (DoFs)), $\mathbb{R}^n$.
  - *Adhesion on/off signals (if adhesion is enabled):* Whether adhesion is turned on or off at the tip of each leg, $\{0, 1\}^6$.
- Observation space:
  - *Joint states:* The angle, velocity, and force at each of the n actuated joint DoFs (by default all 42 leg DoFs), $\mathbb{R}^{3\times n}$.
  - *Fly state:* The linear (x-y-z) position, linear velocity, angular (pitch-roll-yaw) position, and angular velocity of the base of the fly (thorax) in the arena, $\mathbb{R}^{4\times 3}$.
  - *Fly orientation:* The thorax's mesh z axis (unit vector aligned with the anterior-posterior axis of the fly), $\mathbb{R}^3$..
  - *Ground reaction forces*: The ground reaction forces in all three dimensions when experienced by the m body segments (by default the tibia and all five tarsal segments of all legs), $\mathbb{R}^{m\times 3}$.
  - *Visual inputs (if vision is enabled):* Light intensity readings from all 721 ommatidia on each eye, $\mathbb{R}^{2\times 721\times 2}$. Note that 70% of the ommatidia are yellow-type and 30% are pale-type. The last dimension indicates the color channel (for each ommatidium, only one of the two values is nonzero).
  - *Odor inputs (if olfaction is enabled):* Odor intensities sensed by the antennae and maxillary palps on either side of the fly in a k-dimensional odor space, $\mathbb{R}^{k\times 4}$.

We configured the meshes at 1000x scale in MuJoCo to obtain measurements in mm and mN in the observation. The user can build preprogrammed premotor computations and sensory preprocessing by extending the base `NeuroMechFlyMuJoCo` class **(Fig. 1c)**. This will modify the action and observation spaces accordingly. For example, for the odor taxis, visual taxis, and navigation tasks we used a CPG-based network to reduce the action space to a two-dimensional descending command encoding the intended walking direction (see "Control of turning").

Updated rigging of the biomechanical model

In *Drosophila*, the antenna consists of three main segments—the scape, pedicel, and funiculus—and the arista[100]. The fly has four muscles that can actively control the joint between the scape and pedicel[101]. By contrast, the funiculus and the arista move or deform passively in response to external forces (e.g., wind, limb contact during grooming). In the original NeuroMechFly, the entire antenna could move relative to the head with one DoF. We improved the model by separating each antennal mesh into three different meshes in Blender. In the biomechanical model, "bones" determine how objects move with respect to one another. We positioned the bones to accurately replace joints based on anatomical features such as the stalk-like structures connecting the funiculus to the pedicel[102]. We then constructed a kinematic chain connecting these segments: scape-pedicel-funiculus-arista from proximal to distal. Instead of simulating the arista as a soft body (which is computationally expensive), we emulated the compliance of the arista by adding three DoFs between the funiculus and the arista. The passive movement of the arista can be fine-tuned by modifying the stiffness and

damping coefficients of these DoFs. We gave the remaining joints (i.e., head-pedicel and pedicel-funiculus) three DoFs because the real number of DoFs in these antennal joints remains unknown. Future users can modify each DoF (e.g., fix/unfix or stiffen) in the model file to emulate the measured, real movement dynamics of the antennae.

The position of the neck joint affects the translation of the segments on the head, such as the proboscis, antennae, and eyes. The neck is located ventral to the hair plate behind the head. In our previous model, the neck had one unactuated pitch DoF. Here we modified the location of the neck joint by comparing head rotations of the model with those of the real fly and added two known DoFs (yaw and roll) to the neck. Furthermore, we spaced the head away from the thorax to emulate the space filled by the neck. The size of the neck was determined by measuring the proportion of head size to neck size in real animals (data not shown). We confirmed that the rotation center of the neck joint fits the original pose from the NeuroMechFly CT scan by actuating the neck joint to match the original pose. Next, we adjusted the positions of the front legs based on the distance between the front leg thorax-coxa position and anatomical landmarks (e.g., humeral bristles) and an overlay of camera images of real animals with images of the model (data not shown). Finally, we changed the resting pose of the model such that the angle of the scutellum would resemble that of real animals standing freely (untethered) on flat terrain. We used FARMS simulation framework to generate the MJCF file of the updated model[103].

Implementation of leg adhesion

Leg adhesion was added using built-in MuJoCo actuators. Adhesion takes the form of an artificial force injected normal to the point of contact. This force is oriented toward the object colliding with the body part containing the actuator. If multiple contacts occur with external objects and the adhesion actuated body, the force is equally divided between these contact points.

In our model, adhesion is actuated and can be turned on and off during locomotion. We manually defined the adhesion on/off periods within the preprogrammed stepping pattern **(Extended Data Fig. 2)**. Adhesion is on during the stance phase and off during the swing phase **(see "Stepping pattern")**. We controlled adhesion in a binary fashion but it is possible to use an intermediate level of adhesion by modulating the input to the adhesion actuator at every time step.

Critical angle calculation

To quantify the impact of maximal adhesive force on the ability of the fly model to climb **Fig. 2c**, we measured the critical slope—the angle in degrees at which the fly could no longer maintain forward locomotion, or flipped—as a function of the maximal adhesion force. Flipping is defined as when the absolute roll and pitch angles of the fly are above $\pi/2$. A fly has failed to maintain forward locomotion if its position along the surface is negative compared to its initial position after $1s$.

Stepping pattern

We derived the kinematics for each individual step from recordings of real animal behavior[47], beginning with the kinematics of a tethered fly walking on a spherical treadmill[22]. From this initial kinematic replay, we recorded contact forces and end effector positions. To isolate individual steps for each pair of legs, we identified peaks in end effector z-positions. Each peak indicates the middle of a swing phase. To identify the duration of steps, we used the contact data and defined a step as the period of time from the first loss of contact preceding the swing to the next loss of contact following the swing. The contact data was median filtered with a centered window of size $0.0501s$ such that, despite leg dragging sometimes resulting in small jumps of the leg, contacts could still be properly labelled as swing and stance. We excluded all steps that were shorter than 0.1 s and longer than 0.2 s. In this manner, we selected steps for the front, middle, and hind legs. We pooled steps from left and right leg data and selected the ones with the largest z elevation. We then cropped each joint angle trace to the length of the shortest one ($0.1273s$, middle leg): The selected hind leg step was cropped by $0.0266s$ (17.3% of the initial hind leg step duration); the selected front leg step was cropped by $0.0180s$ (12.4% of the initial front leg step duration). All steps differed mostly in the lengths of their stance phases. To ensure that steps were cyclical, we extended the step duration by 1% through linear interpolation of joint angles between the last and first time-points of the step. For

each leg pair, we extracted swing and stance onsets by stepping each leg in place and visually defining the swing and stance periods. All steps, including swing and stance periods, are left-right symmetric.

CPG-based controller

Central Pattern Generators (CPGs) are neural circuits that generate rhythmic output without receiving rhythmic input[58]. Through interactions, coupled CPGs can synchronize with given phase offsets. As for the previous version of NeuroMechFly, we implemented CPGs by adapting those used to model limb actuation in salamanders[10]. All DoFs for a given leg of the fly were controlled by a single CPG. The oscillatory output of a given CPG was then interpreted non-linearly as the phase and amplitude of the swing-to-stance cycle. The gait pattern emerges from the phase biases of the different CPGs. More precisely,

$$\dot{\theta}_i = 2\pi\nu_i + \sum_j r_j w_{ij} \sin(\theta_j - \theta_i - \phi_{ij}) \tag{1}$$

$$\dot{r}_i = \alpha_i (R_i - r_i) \tag{2}$$

$$\psi_{i,\mathrm{dof}} = \mathbf{\Psi}_{i,\mathrm{dof}}(0) + r_i \left[\mathbf{\Psi}_{i,\mathrm{dof}} \left(\frac{\theta_i}{2\pi} |\mathbf{\Psi}_{i,\mathrm{dof}}| \right) - \mathbf{\Psi}_{i,\mathrm{dof}}(0) \right] \tag{3}$$

where θ_i and r_i are the state variables describing the phase and amplitude of the oscillator for leg i; ν_i and R_i are the intrinsic frequency and amplitude of leg i; α_i is a constant determining the rate of convergence to the intrinsic amplitude; w_{ij} and ϕ_{ij} are the coupling weight and phase bias between legs i and j; $\psi_{i,\mathrm{dof}}$ is the angle of the degree of freedom dof on leg i; $\mathbf{\Psi}_{i,\mathrm{dof}}(t)$ gives the angle of dof at time t of the preprogrammed step cycle; $\mathbf{\Psi}_{i,\mathrm{dof}}(0)$ is the neutral position; $|\mathbf{\Psi}_{i,\mathrm{dof}}|$ is the duration of the preprogrammed step. We used an idealized tripod gait for walking in our CPG model. The parameters of the CPG controller are detailed in **Table 2** and **Table 3**.

Rule-based controller

The rule-based controller was inspired by the first three rules described in Walknet[104,1,105]. The first rule ensures stability by inhibiting swing onset in the rostral neighbor of a swinging leg. The second rule ensures the propagation of the wave by eliciting a swing in the rostral and contralateral neighbor of a leg entering stance phase. The third rule enforces temporal coherence by eliciting a swing in the caudal and contralateral neighbor of a leg approaching the end of its stance phase. The rules modify a stepping likelihood score for each leg, and a step is initiated on the leg in stance phase with the highest positive score. If all legs have negative scores, no step is initiated. If multiple legs have similar scores ($< 0.1\%$ of the highest score), a leg is selected at random to avoid artifacts resulting from small numerical errors. The contributions of these rules are weighted (**Table 4**): Rule 1 is weighted most heavily as it is crucial to maintain stability. Rules 2 and 3 are given different weights for ipsi- and contralateral connections. To maintain synchrony, we ensured that the duration of the swing and stance periods were identical across all legs.

Hybrid controller

The hybrid controller is a CPG controller with two additional rules that can be activated depending on leg mechanosensory signals. These rules allow the fly to recover when a leg becomes stuck in a gap (e.g., in gapped terrain) or hits an obstacle (e.g., in block terrain) by adjusting the leg in question. The first rule ("retraction rule") is activated when a leg is extended farther than expected along the z-axis (indicating that the leg may have fallen into a gap). More precisely, this rule becomes active when the tip of a leg is > 0.05 mm lower than the third most extended leg in the z axis. Due to numerical errors and physics instabilities, the z positions of the tips of the legs read out from the physics simulator are sometimes slightly below 0 when the legs are on the ground. A 0.05 mm margin was therefore added to avoid spurious leg retraction. If multiple legs meet this criterion, only the leg that extends the lowest is corrected. The second rule ("stumbling rule") is activated when a leg comes into unexpected contact with an object, resulting in a horizontal force against the direction of locomotion. More precisely, this rule becomes active when the tibia or the two most proximal segments of the tarsus have a contact force greater than 1 mN opposing the heading of the fly while

the leg is in swing. When either rule is activated, a shift is progressively added to a subset of joints on the leg in question such that the leg lifts up further than normal. When the condition is no longer met, the joint angles progressively reset. **Table 5** provides a summary of the joints involved and the rate at which they increment or decrement. Adhesion is turned off when one of the rules is active.

Locomotor controller benchmarking

We benchmarked locomotor controllers by running 20 simulations, starting from different spawn positions and initial states, each for 1 s and computing the average displacement per second along the x axis (the anterior-posterior axis of the fly's original spawn orientation). The controllers are given 0.2 extra seconds in the beginning of the simulation to stabilize (i.e., to stand stably in its initial pose without movement). The "gapped terrain" consists of horizontal 1 mm-wide blocks separated by 0.4 mm-wide gaps. The "blocks terrain" consists of 1.3×1.3 mm blocks configured in a checkerboard pattern, with half of the blocks 0.35 mm higher than the others. A small overlap is added between the blocks to avoid extremely thin surfaces near the corners that can lead to physics instabilities. The "mixed terrain" consists of alternating flat, gapped, and block patterns along the x axis. We used a joint position gain k_p of 30 and adhesion forces of 40 mN.

Control of turning

Walking flies use different strategies to turn depending on the sharpness of the turn[55]. For smoother turns (i.e., rotations $< 20°$) the fly mostly increases the stroke amplitude of its outer legs. For sharper turns (20° - 50°) the fly additionally decreases the stroke amplitude of its inner legs. For very sharp in-place turns (>50°), the fly steps its inner legs backward. Our controller receives a two-dimensional descending input that controls turning. On each side, the descending signal $\delta \in [-1, 1]$ modifies the intrinsic frequency ν_i and maximum amplitude R_i of each oscillator i as follows:

$$R'_i(\delta) = |\delta| \qquad \nu'_i(\delta) = \begin{cases} \nu_i & \text{if } \delta > 0 \\ -\nu_i & \text{otherwise} \end{cases} \tag{4}$$

where R'_i and ν'_i are the modified maximum amplitude and intrinsic frequency.

Implementation of vision

Flies have three major types of ommatidia. These differentiate colors and polarization properties by using different combinations of photoreceptors. Yellow- and pale-type ommatidia are stochastically arranged throughout the eye and enable two-dimensional chromatic sensitivity in the UV$_{\sim 300\text{nm}}$-to-yellow range[68]. The yellow- and pale-type ommatidia are found at 7:3 ratio[67]. A third type is found in the eye's dorsal rim area (DRA) facing the sky and specialized for polarization detection during navigation[106]; this type of ommatidia is not implemented in our model. In our implementation yellow- and pale-type ommatidia are instead made sensitive to the green and blue channels of the physics simulator. For a more biologically accurate representation of color, the green- and blue-channel display color can be set as the product of the actual surface reflectance spectrum of the object and the spectral response curve of the appropriate photoreceptors[68].

Calibration of visual input

To calibrate vision, we surrounded the model with color-coded pillars in the arena **(Extended Data Fig. 3a–b)**. This allowed us to identify the limits and midlines of each eye's (camera's) field of view (FOV) to configure the camera's orientation and FOV correctly. MuJoCo's built-in camera renders a rectilinear image with a distortion: the periphery of the image is enlarged to keep lines straight. This distortion is especially severe at large FOV, such as those used to simulate fly vision. To remove this distortion, we applied a fish-eye effect on the raw camera readings **(Extended Data Fig. 3c–d)** to represent the same angular span approximately equally in the rendered image. More precisely, the pixel on row r, column c in the transformed image is assigned the value of the pixel on row r', column c' of the original image, where r', c' are determined by

$$r' = \frac{r/z}{1 - k(r^2 + c^2) + \epsilon} \qquad c' = \frac{c/z}{1 - k(r^2 + c^2) + \epsilon} \tag{5}$$

where r, r' and c, c' are each normalized to the range $[-1, 1]$ (i.e., the pixel in the center of the image would have the coordinates $(0, 0)$); k describes the extent of the fisheye distortion, and z describes the zoom level of the resulting image. A small constant ϵ is added to avoid numerical issues. Where r', c' are outside the range $[-1, 1]$, an empty value is assigned. Finally, we binned the pixels by the ommatidium they belong to and took the mean intensity per ommatidum (**Extended Data Fig. 3**e).

Visual object chasing task

In our visual object chasing task, the fly follows a black sphere moving in an S-shaped curve at 8 mm/s. To achieve this, we first used a thresholding rule to detect the object (intensity below $\bar{\rho}/255$). Then, we computed the position and size of the object (both normalized) seen from each eye. Finally, we linearly adjusted the descending signal on each side depending on the object's azimuth as seen from the ipsilateral eye. The turning bias is updated every 0.05 s of simulated time. More precisely,

$$\delta_i = \min(\max(ka_i + b, \delta_{\min}), \delta_{\max}) \tag{6}$$

where δ_i is the descending signal on side i; a_i is the azimuth expressed as the deviation from the anterior edge of the eye's field of view, normalized to $[0, 1]$; k, b describe the response curve; $\delta_{\min}, \delta_{\max}$ are the minimal and maximal allowed values for the descending signal. The values used in our controller are detailed in **Table 7**.

Olfactory chemotaxis task

In the olfactory chemotaxis task, the fly seeks an attractive odor source while avoiding two aversive odor sources. To achieve this, we first calculate the odor intensities sensed at the location of the antennae and maxillary palps based on a diffusion function, $I(d)$, where I is the intensity and d is the distance from the odor source. The odor diffusion relationship can be defined by the user. In our example, we used the inverse square relationship $I(d) = I_{\text{peak}}/d^2$ where I_{peak} is the peak intensity. If there are multiple sources for the same odor, their intensities are summed. Then, for the attractive odor, we averaged intensities sensed by the antenna and the maxillary palp weighted by 9:1 (roughly comparable to the ratio of OSNs in the antenna and maxillary palps [107]). By contrast, to demonstrate the possibility of using different sensors for different odors, we used only the intensity sensed by the antenna for the aversive odor to emulate odorants that can only be sensed by one but not both organs. We performed this process for olfactory organs on each side of the head and multiplied the relative differences in intensities between both sides with a gain factor. Next, we summed up this product for each odor and nonlinearly transformed it into a turning bias. This bias modulates descending signals that drive turning. The turning bias is updated every 0.05 s of simulated time. More precisely,

$$s = \sum_o \gamma_o \frac{I_{\text{left},o} - I_{\text{right},o}}{(I_{\text{left},o} + I_{\text{right},o})/2} \tag{7}$$

$$b = \tanh(s^2) \tag{8}$$

$$\delta_{\text{left}} = \begin{cases} \delta_{\max} & \text{if } s > 0 \\ \delta_{\max} - b(\delta_{\max} - \delta_{\min}) & \text{otherwise} \end{cases} \qquad \delta_{\text{right}} = \begin{cases} \delta_{\max} - b(\delta_{\max} - \delta_{\min}) & \text{if } s > 0 \\ \delta_{\max} & \text{otherwise} \end{cases} \tag{9}$$

where s is the weighted sum of bilateral differences in odor intensities, b is the nonlinearly transformed turning bias, $I_{\text{side},o}$ is the mean intensity of odor o sensed by the antenna and the maxillary palp on the specified side; γ_o is the gain of odor o; $\delta_{\min}, \delta_{\max}$ are the minimal and maximal allowed values for the descending signal. The values used in our controller are detailed in **Table 8**.

Multimodal navigation task solved using reinforcement learning (RL)

In the multimodal navigation task, the fly seeks an attractive odor source and avoids a visual obstacle in its path while locomoting over rugged terrain. To achieve this we used a hierarchical controller consisting of (i) a vision module that extracts lower-dimensional visual features from retinal inputs, (ii) a decision-making module that predicts the appropriate turning bias given pre-extracted visual features and odor intensities, and (iii) a descending interface passing the turning bias to a downstream (iv) hybrid motor controller that integrates CPG states with leg mechanosensory feedback. We used a simplified "mixed terrain" environment with gap widths reduced to 0.2 mm and block heights reduced to 0.3 mm.

We started by training a visual model that performs convolution on the retinal image. Because the retinal image is on a hexagonal lattice, we built a graph representation of the lattice and performed graph convolution[108] on it using PyTorch[109] and PyTorch Geometric[110]. The precise architecture of the model is detailed in **Table 9**. Given the ommatidia intensity readings, this model predicts (i) the direction of the object relative to the fly, (ii) the distance of the object from the fly, (iii) whether the object is within the fly's eyes FOVs, (iv) the azimuth of the object seen from each eye, and (v) the percentage size of the object on the retina. We gathered $10,000$ training samples by randomly spawning the fly and the obstacle (within 10 mm of the fly) and injecting Gaussian noise (standard deviation 0.05x the intensity range) into the visual inputs. We then trained the model in a supervised manner using the Adam optimizer[111] with the loss L defined as follows

$$L = k_{\text{dir}}\text{MSE}(\hat{\vartheta}, \vartheta) + k_{\text{dist}}\text{MSE}(\hat{d}_{\text{fo}}, d_{\text{fo}}) + k_{\text{class}}\text{BCE}(\hat{p}, p) + k_{\text{azi}}\text{MSE}(\hat{a}, a) + k_{\text{size}}\text{MSE}(\hat{s}, s) \tag{10}$$

where $k_{\text{dir}}, k_{\text{dist}}, k_{\text{class}}, k_{\text{azi}}, k_{\text{size}}$ are the coefficients for the obstacle direction, obstacle distance, obstacle presence classification, obstacle azimuth, and obstacle size terms; MSE is the Mean Squared Error, BCE is the Binary Cross Entropy; ϑ is the direction of the obstacle from the fly; d_{fo} is the distance of the obstacle from the fly; p is the predicted probability on whether the obstacle is within the fields of view of the fly's eyes; a is the azimuth of the obstacle on the retina from each eye; s is the relative size of the obstacle on the retina from each eye; $\hat{\vartheta}, \hat{d}_{\text{fo}}, \hat{p}, \hat{a}, \hat{s}$ are the predictions of $\vartheta, d_{\text{fo}}, p, a, s$. The exact hyperparameters chosen are detailed in **Table 10**. We obtained satisfactory model performance after 150 epochs of training (**Extended Data Fig. 4**).

Next, we calculated the odor intensity sensed on each side of the head by averaging the intensities sensed by the antenna and ipsilateral maxillary palp at a relative weight of 9:1. We then transformed the odor intensity with the square root operator to give it a more linear response as a function of distance. We then concatenated the odor intensities on each side of the head with the extracted visual features (described above) and the turning bias b from the previous time step (described below). This feature vector was given to a Multilayer Perceptron (MLP, 2 hidden layers, 32 units each) as the input. The MLP predicts a scalar turning bias b which is converted to a descending representation according to **Equation 9**. The turning bias is updated every 0.05 s of simulated time. We train this MLP using RL: If the fly has reached within 3 mm of the target, a reward of R_{success} is given and the simulation is terminated. If the fly has flipped or touched the obstacle, a reward of R_{fail} is given and the simulation is terminated. If none of these special conditions are met, we calculate a score S_{obstacle} describing whether the fly is facing the obstacle: $S_{\text{obstacle}} = 1$ when $\vartheta = 0$; $S_{\text{obstacle}} = 0$ when $\vartheta > m \arctan(r/d_{\text{fo}})$ where r is the radius of the obstacle; d_{fo} is the fly-to-obstacle distance. In other words, this score is 1 when the fly is perfectly facing the obstacle and 0 when the fly is facing away from the obstacle by a margin of m. When the fly's heading is within this range, S_{obstacle} is linearly interpolated. Similarly, we calculated a second score S_{target} that is 1 when the fly is facing any point within ϵ mm of the target; by contrast, $S_{\text{target}} = 0$ when the shortest distance from the fly's heading direction to the target is more than $q\epsilon$. S_{target} is also linearly interpolated when the shortest distance from the target to the fly's heading vector is between ϵ and $q\epsilon$. Then, we calculated the reward R defined as follows:

$$k_{\text{attract}} = \begin{cases} k_{\text{attract}}^{(\text{near})} & \text{if fly has passed obstacle} \\ k_{\text{attract}}^{(\text{far})} & \text{otherwise} \end{cases} \tag{11}$$

$$R = k_{\text{dist}}\Delta d_{\text{ft}} - k_{\text{attract}}\Delta S_{\text{target}} + k_{\text{avoid}}\Delta S_{\text{obstacle}} \tag{12}$$

where $k_{\text{dist}}, k_{\text{attract}}, k_{\text{avoid}}$ are the weights of the proximity, target facing, and obstacle avoiding terms of the reward; d_{ft} is the distance from the fly to the target; $\Delta d_{\text{ft}}, \Delta S_{\text{target}}, \Delta S_{\text{obstacle}}$ are the current values of $d_{\text{ft}}, S_{\text{target}}, S_{\text{obstacle}}$ subtracted from their values in the previous decision-making time step. We trained the model using the Soft Actor Critic (SAC) algorithm [112] provided by the Stable Baselines 3 library [113], which uses gradient backpropagation to update the weights and biases of the MLP. We trained the model for 266,000 decision steps (approximately 3.7 simulated hours). We consider this training time reasonable as the network is randomly initialized and the simulated fly has to "evolve" the entire circuit for decision making. Parameters used in the model and the training process are expressed in **Table 10**.

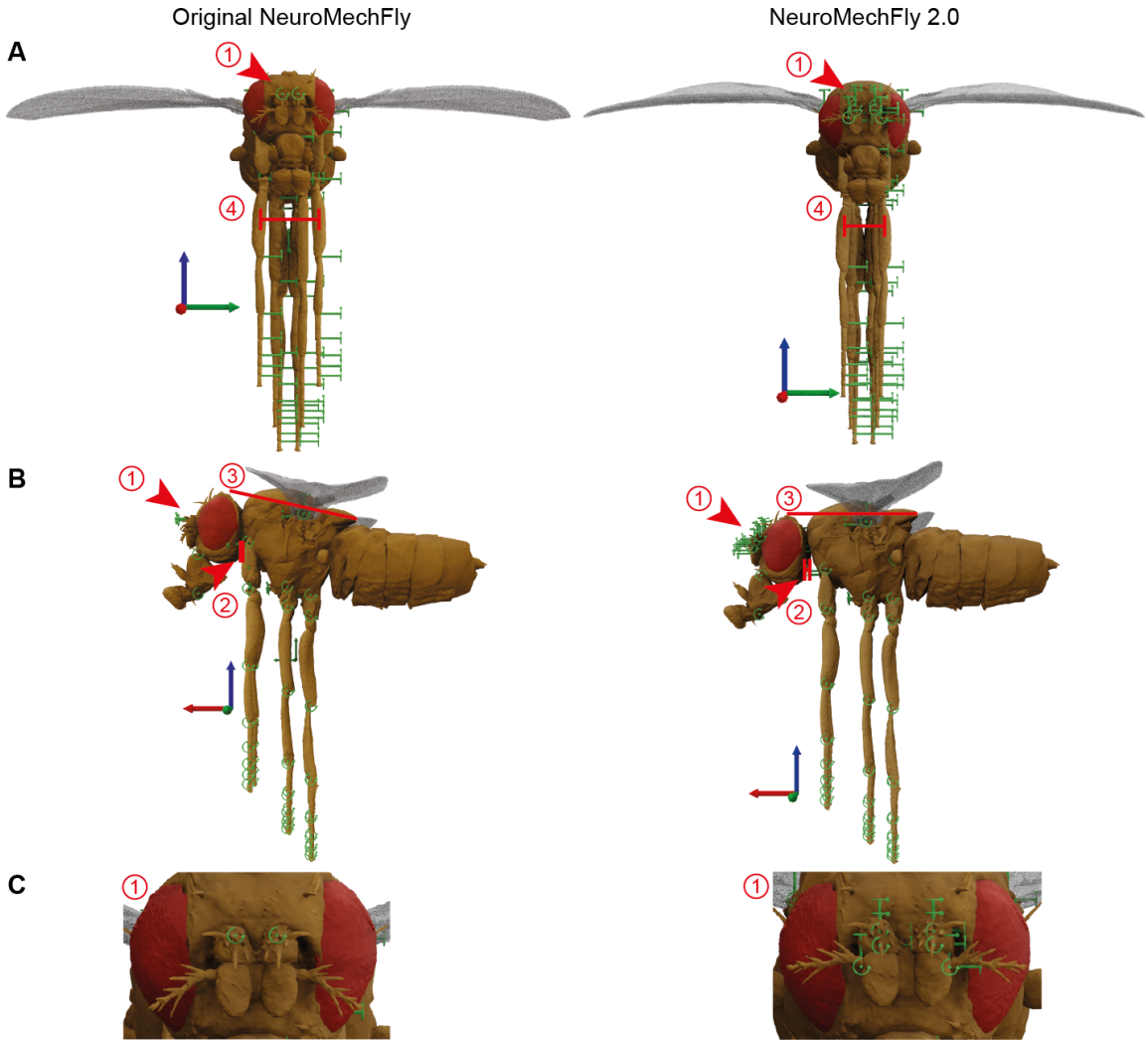
Data availability

Data are available at: https://doi.org/10.7910/DVN/3MCEYR
This repository includes the parameter files of the trained visual processing and RL models used in the multimodal navigation task, the training data for the visual processing model, and the graph representation of the ommatidia lattice used to perform convolution on the visual input.

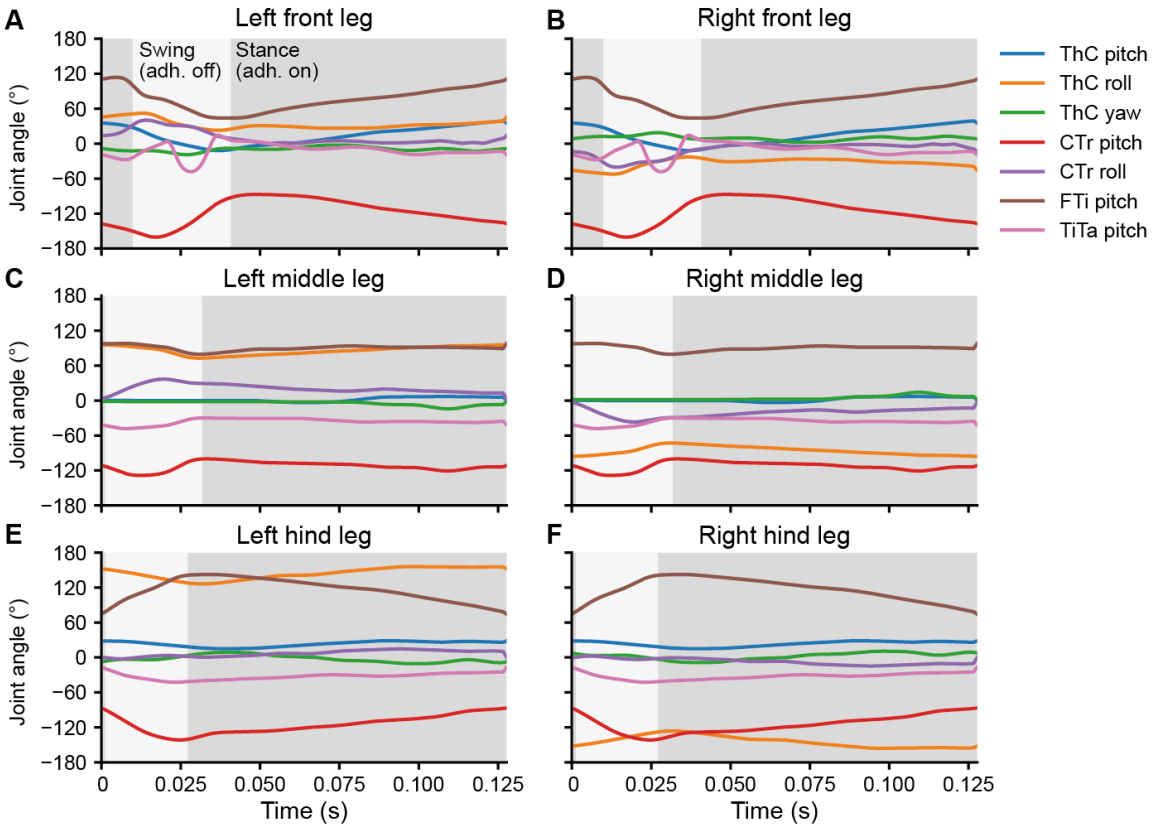
Code availability

The FlyGym package is available at: https://github.com/NeLy-EPFL/flygym.
Its documentation is available at: https://neuromechfly.org/
The code used to generate data presented in this paper are available at:
https://github.com/NeLy-EPFL/nmf2-paper

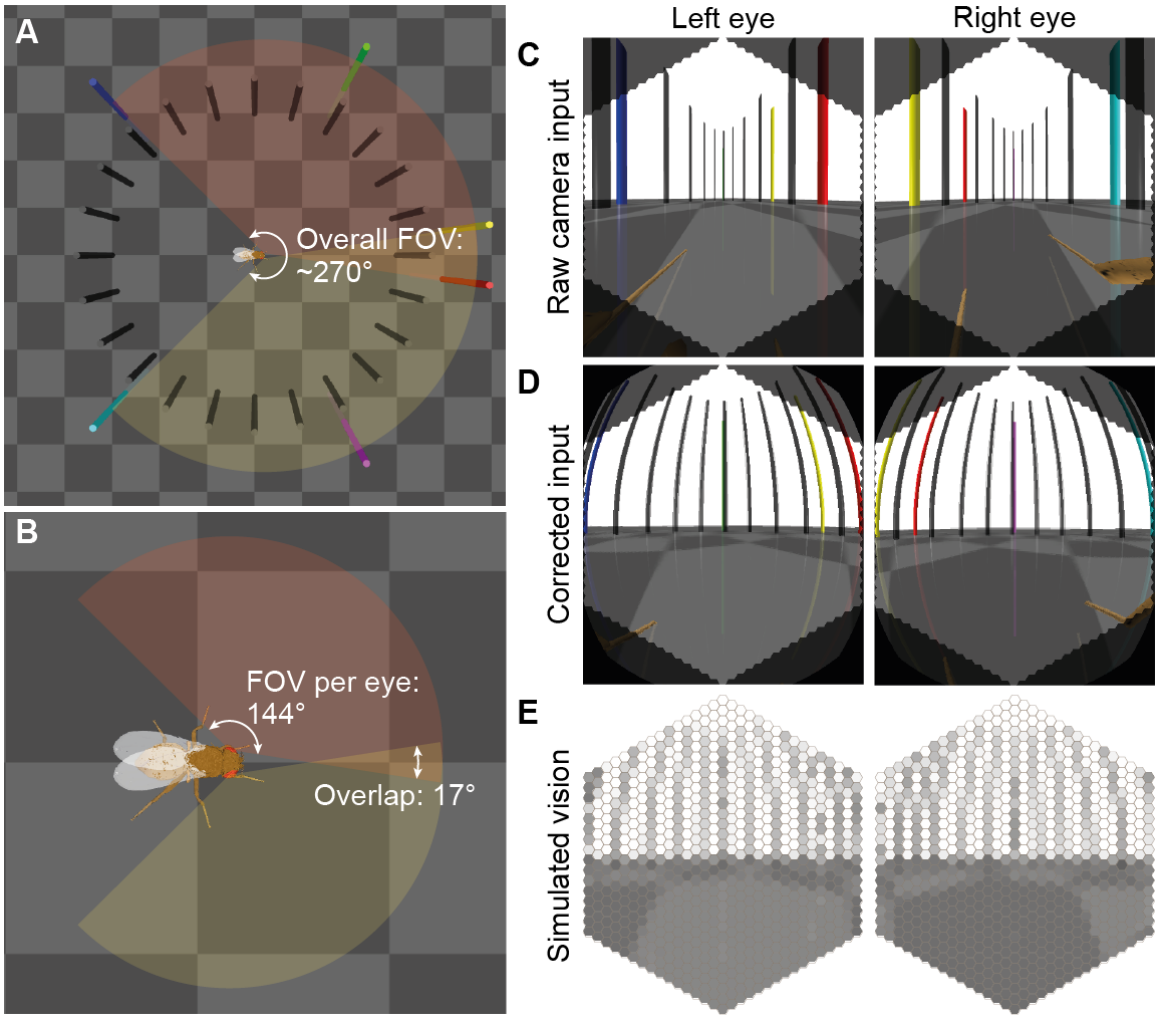
Extended data



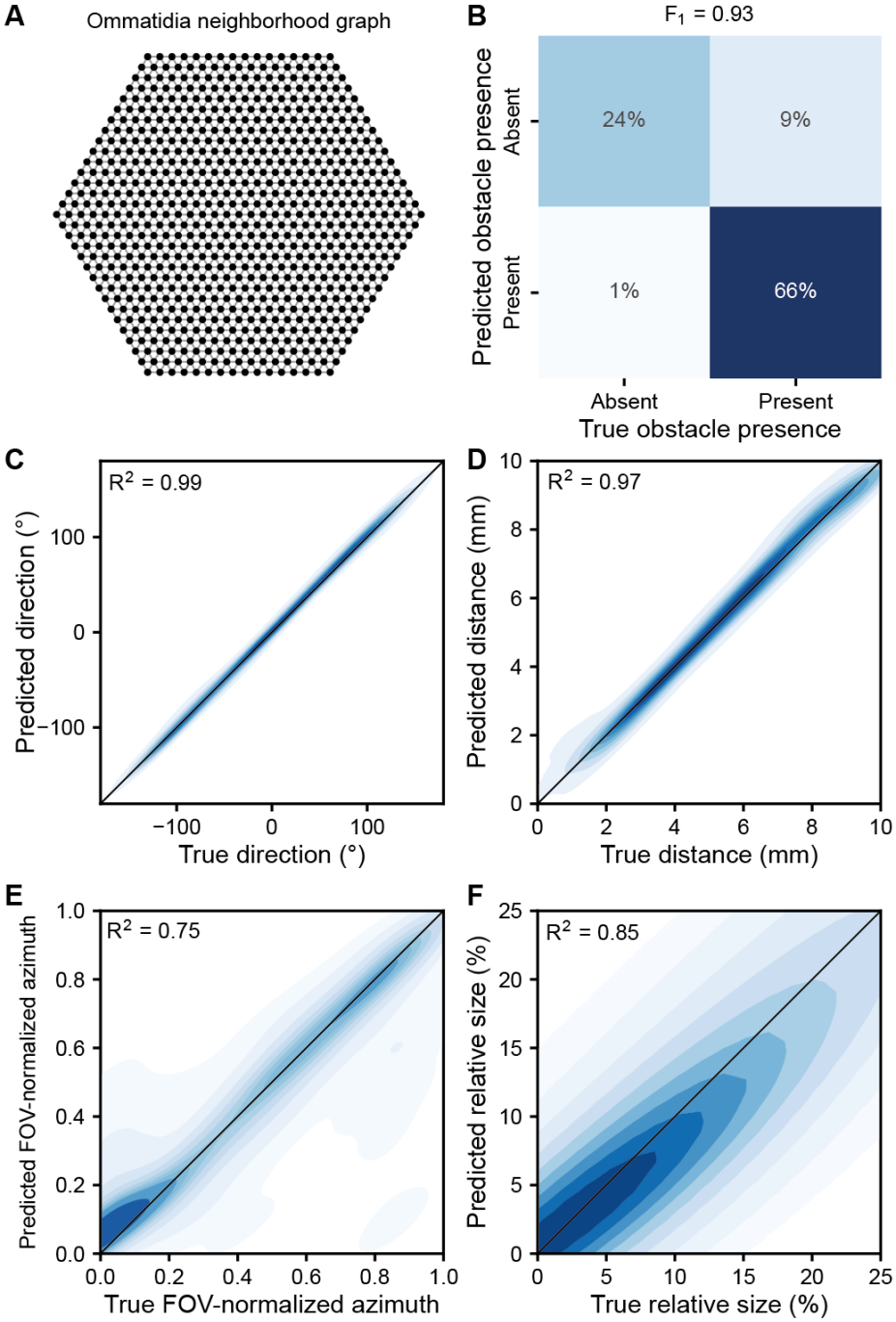
Extended Data Fig. 1: **Improvements to the biomechanical model.** A comparison of the the original (left) and updated (right) NeuroMechFly biomechanical model from the **(a)** front and **(b)** side views as well as a **(c)** zoomed-in view of the head, highlighting antennal DoFs. DoFs are indicated (green). The highlighted differences are: (1) additional DoFs in the antennae, (2) a gap for the neck between the head and the thorax, (3) angles of the thorax and the position of the head relative to it, and (4) the placement of the legs on the thorax.



Extended Data Fig. 2: **Preprogrammed stepping based on experimentally recorded data.** Joint kinematics for each leg during preprogrammed stepping. Kinematic patterns derived from behavioral recordings **(see Methods)**. Time series for each joint are color-coded. ThC: thorax-coxa joint; CTr: coxa-trochanter joint; FTi: femur-tibia jointl; TiTa: tibia-tarsus joint. Note the left-right symmetry in roll and yaw DoFs. Indicated are periods when adhesion is turned off during swing to facilitate lifting each leg (light grey).



Extended Data Fig. 3: **Calibration of vision.** **(a)** The calibration environment has pillars spaced regularly around the fly at 6° intervals (black pillars at 18° intervals). Additionally, red, green, and blue pillars are used to indicate the anterior, midline, and posterior field of view (FOV) limits of the left eye. Yellow, magenta, and cyan pillars indicate the FOV limits of the right eye. **(b)** Each eye has a FOV spanning $\sim 144°$ horizontally. The two eyes overlap by $\sim 17°$, resulting in an overall horizontal FOV of $\sim 270°$. **(c)** A raw camera view of what the fly sees in this environment before applying a fisheye effect. Note that peripheral regions are exaggerated to keep the lines straight. **(d)** A fisheye effect is applied to simulate the roughly spherical arrangement of ommatidia in the fly eye. **(e)** Retinal inputs are simulated by binning the pixels according to the hexagonal grid of ommatidia and taking the average intensity within each ommatidium. Ommatidia are randomly sensitive to green (yellow-type) and blue (pale-type) channels in a 7:3 ratio.



Extended Data Fig. 4: **Vision model used in the multimodal navigation task.** **(a)** The neighborhood graph used to perform convolution on the hexagonal lattice. **(b)** Accuracy of the model in predicting whether the obstacle is present in the fields of view of the fly's eyes. **(c-d)** Accuracy of the model in predicting the **(c)** direction and **(d)** distance of the obstacle from the fly. The angular R^2 score is defined as the R^2 score of $\sin(\vartheta)$ concatenated with $\cos(\vartheta)$, where ϑ is the angle. **(e-f)** Accuracy of the model in predicting the **(e)** azimuth and **(f)** size of the obstacle in the retinal images.

Algorithm 1 The main control loop in FlyGym

```
1: sim_params = MuJoCoParameters(physics parameters, rendering parameters, ...)
2: arena = Arena(arena parameters)
3: sim = NeuroMechFlyMuJoCo(sim_params, arena, initial_pose, actuated_joints, spawn_pos, ...)
4: controller = User-defined controller        ▷ CPG network, neural network, coordination rules, ...
5: observation_history, action_history, reward_history = empty lists
6: observation, info = sim.reset()
7: for a specified maximum number of simulation steps do
8:     action = Apply controller given current observation
9:     observation, reward, terminated, truncated, info = sim.step(action)
10:    Add observation, reward, action to history
11:    if terminated or truncated then
12:        break                          ▷ early termination due to task success/failure/timeout
13:    end if
14: end for
15: sim.save_video("video_path.mp4")
16: sim.close()
```

Extended Data Fig. 5: **The main control loop in the FlyGym package. (Lines 1–3)** First, we define the simulation parameters including the simulation time step, physics parameters such as joint gains and gravity, and parameters specifying how the scenes should be rendered. We then define the arena in which the biomechanical model will be embedded. Several predefined arenas are provided with the package. With the simulation parameters and the arena, we can then define the simulation itself. The user can configure which joints are actuated and where the fly should be spawned within the arena. **(Line 4)** The implementation of the controller is determined by the user. Typically, it has a prediction function that determines the action to take given the current observation. An internal state can be maintained within the controller. It can be a hard-coded model or an artificial neural network (e.g., a network trained through RL). The definition of the controller (or training algorithm) is isolated from the control loop. **(Lines 8–12)** This is the portion within the main MDP loop. Using the user-defined controller, we supply the appropriate action to the simulation and step it forward by one simulation time step. The simulation returns a new observation, optionally a reward (depending on the user's definition), a "terminated" flag indicating whether the simulation has ended due to a factor within the MDP framing (e.g., task success or failure), a "truncated" flag indicating whether the simulation has ended due to a factor outside of the MDP framing (e.g., timeout), and a dictionary containing any additional information.

Supplementary Videos

Supplementary Video 1: Ground reaction forces experienced by the fly's legs. Ground reaction forces during locomotion with a CPG-based controller. Here and in all subsequent videos, the tarsi are color-coded based on whether adhesion is on and the tarsus is in contact with the ground (black) or not in contact with the ground (red). Note that in this video the tarsus is never red because the legs are always in contact with the ground when adhesion is on. Simulation is played back at 0.05x original speed.
Link to Supplementary Video 1

Supplementary Video 2: Locomotion over sloped terrain using leg adhesion. Locomotion is driven by a CPG-based controller. Shown are simulations (left) without or (right) with adhesion. Simulation is played back at 0.1x original speed.
Link to Supplementary Video 2

Supplementary Video 3: Preprogrammed stepping of each leg. Individual legs are stepped in series according to their data-derived joint kinematics. Adhesion is disabled in this video.
Link to Supplementary Video 3

Supplementary Video 4: Control signals of the CPG-based controller. Shown for all legs are the CPG phases (wrapped by 2π) and amplitudes from random initialization. As CPGs synchronize, the amplitude is set low to prevent abrupt movements. The CPGs eventually synchronize, forming a tripod gait. Simulation is played back at 0.1x original speed.
Link to Supplementary Video 4

Supplementary Video 5: Control signals of the rule-based controller. Shown for all legs are the stepping scores and contributions of each of the three coordination rules. Indicated are the initiation of steps (triangles). Simulation is played back at 0.1x original speed.
Link to Supplementary Video 5

Supplementary Video 6: Control signals of the hybrid controller. Shown for all legs are the CPG phases (wrapped by 2π) and amplitudes as well as the activation of the retraction (solid) and stumbling (dashed) rules based on sensory feedback. The tibia is colored (dark blue) when the retraction rule is active. The coxa is colored (red) when the stumbling rule is active. Simulation is played back at 0.1x original speed.
Link to Supplementary Video 6

Supplementary Video 7: Locomotion over multiple terrain types. The fly walks over a flat surface (first column), a surface with gaps (second column), a surface with blocks (third column), and a hybrid surface (fourth column). The fly is either controlled by a CPG-based controller (top), a rule-based controller (middle), or a hybrid controller integrating both CPGs and sensory feedback rules (bottom). Shown are the results from 20 trials for each condition. Simulation is played back at 0.1x original speed.
Link to Supplementary Video 7

Supplementary Video 8: Visual object chasing task. The fly uses vision to follow a sphere that is moving away along an S-shaped trajectory. Shown are visual inputs to the left and right eyes (bottom). Locomotion is regulated using a hybrid controller with leg adhesion. Note that each eye's field of view can observe front leg movements. Simulation is played back at 0.5x original speed.
Link to Supplementary Video 8

Supplementary Video 9: Olfactory chemotaxis task. The fly seeks an attractive odor source (orange) while avoiding two aversive odor sources (blue). Colored bars (bottom) indicate the intensities of attractive (orange) and aversive (blue) odors sensed by antennae on each side of the head. Locomotion is regulated using a hybrid controller with leg adhesion. Simulation is played back at 0.5x original speed.

Link to Supplementary Video 9

Supplementary Video 10: Multimodal navigation task controlled by an RL-trained artificial neural network. The fly seeks an attractive odor source (orange) while visually avoiding an obstacle (gray pillar) over rugged hybrid terrain. Shown are visual inputs to the left and right eyes (bottom-center). Orange bars (bottom-left and bottom-right) indicate the intensity of an attractive (orange) odor sensed by antennae on each side of the head. Locomotion is regulated using a hybrid controller with leg adhesion. Simulation is played back at 0.2x original speed.
Link to Supplementary Video 10

Supplementary Video 11: Multimodal navigation task executed from different spawn positions. Shown are the nine trials illustrated in **Fig. 3e (bottom right)**. Simulation is played back at 0.5x original speed.
Link to Supplementary Video 11

Acknowledgments

We thank Victor Lobato-Rios for his valuable insights and early exploration of the visual inputs. We thank Jonathan Arreguit, Shravan Tata Ramalingasetty, and Auke Jan Ijspeert for the development of FARMS, which we used to generate the MJCF file of the updated model. PR acknowledges support from a Swiss National Science Foundation (SNSF) Project Grant (175667) and an SNSF Eccellenza Grant (181239). SWC acknowledges support from a Boehringer Ingelheim Fonds PhD fellowship. PGÖ acknowledges support from a Swiss Government Excellence PhD Scholarhip and a Google PhD Fellowship. FH acknowledges support from a Boehringer Ingelheim Fonds PhD fellowship.

Author Contributions

S.W.C. — Conceptualization, Methodology, Software, Formal Analysis, Investigation, Data Curation, Validation, Writing – Original Draft Preparation, Writing – Review & Editing, Visualization.
V.A.S. — Conceptualization, Methodology, Software, Formal Analysis, Investigation, Data Curation, Validation, Writing – Original Draft Preparation, Writing – Review & Editing, Visualization.
P.G.O. — Conceptualization, Methodology, Software, Investigation, Data Curation, Validation, Writing – Original Draft preparation, Visualization, Writing – Review & Editing.
L.G. — Methodology, Software, Validation, Investigation, Writing – Review & Editing.
F.H. — Conceptualization, Methodology, Software, Investigation, Writing – Review & Editing.
P.R. — Conceptualization, Methodology, Resources, Writing – Original Draft Preparation, Writing - Review & Editing, Supervision, Project Administration, Funding Acquisition.

Ethical compliance

All experiments were computational and thus performed in compliance with relevant national (Switzerland) and institutional (EPFL) ethical regulations.

Declaration of Interests

The authors declare that no competing interests exist.

References

[1] Cruse, H., Kindermann, T., Schumm, M., Dean, J. & Schmitz, J. Walknet—a biologically inspired network to control six-legged walking. *Neural Networks* **11**, 1435–1447 (1998).

[2] Ekeberg, Ö., Blümel, M. & Büschges, A. Dynamic simulation of insect walking. *Arthropod Structure & Development* **33**, 287–300 (2004).

[3] Schmitz, J., Schneider, A., Schilling, M. & Cruse, H. No need for a body model: Positive velocity feedback for the control of an 18-DOF robot walker. *Applied Bionics and Biomechanics* **5**, 135–147 (2008).

[4] Goldsmith, C., Szczecinski, N. & Quinn, R. Drosophibot: A Fruit Fly Inspired Bio-Robot. In Martinez-Hernandez, U. *et al.* (eds.) *Biomimetic and Biohybrid Systems*, Lecture Notes in Computer Science, 146–157 (Cham, 2019).

[5] Schumacher, P. *et al.* Natural and Robust Walking using Reinforcement Learning without Demonstrations in High-Dimensional Musculoskeletal Models (2023). ArXiv:2309.02976 [cs].

[6] Sigvardt, K. A. & Miller, W. L. Analysis and modeling of the locomotor central pattern generator as a network of coupled oscillators. *Annals of the New York Academy of Sciences* **860**, 250–265 (1998).

[7] Rybak, I. A., Dougherty, K. J. & Shevtsova, N. A. Organization of the mammalian locomotor CPG: review of computational model and circuit architectures based on genetically identified spinal interneurons. *ENeuro* **2** (2015).

[8] Ekeberg, Ö. & Grillner, S. Simulations of neuromuscular control in lamprey swimming. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences* (1999).

[9] Thandiackal, R. *et al.* Emergence of robust self-organized undulatory swimming based on local hydrodynamic force sensing. *Science Robotics* **6**, eabf6354 (2021).

[10] Ijspeert, A. J., Crespi, A., Ryczko, D. & Cabelguen, J.-M. From Swimming to Walking with a Salamander Robot Driven by a Spinal Cord Model. *Science* **315**, 1416–1420 (2007).

[11] Knüsel, J., Crespi, A., Cabelguen, J.-M., Ijspeert, A. J. & Ryczko, D. Reproducing Five Motor Behaviors in a Salamander Robot With Virtual Muscles and a Distributed CPG Controller Regulated by Drive Signals and Proprioceptive Feedback. *Frontiers in Neurorobotics* **14** (2020).

[12] Towers, M. *et al.* Gymnasium (2023). URL https://zenodo.org/record/8127025.

[13] Levine, S., Pastor, P., Krizhevsky, A. & Quillen, D. Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection (2016). ArXiv:1603.02199 [cs].

[14] Levine, S., Finn, C., Darrell, T. & Abbeel, P. End-to-End Training of Deep Visuomotor Policies (2016). ArXiv:1504.00702 [cs].

[15] Hoel, C.-J., Driggs-Campbell, K., Wolff, K., Laine, L. & Kochenderfer, M. J. Combining Planning and Deep Reinforcement Learning in Tactical Decision Making for Autonomous Driving. *IEEE Transactions on Intelligent Vehicles* **5**, 294–305 (2020).

[16] Shi, H., Lin, Z., Hwang, K.-S., Yang, S. & Chen, J. An Adaptive Strategy Selection Method With Reinforcement Learning for Robotic Soccer Games. *IEEE Access* **6**, 8376–8386 (2018).

[17] Sharma, S., Srinivas, A. & Ravindran, B. Learning to Repeat: Fine Grained Action Repetition for Deep Reinforcement Learning (2020). ArXiv:1702.06054 [cs].

[18] Cimurs, R., Suh, I. H. & Lee, J. H. Goal-Driven Autonomous Exploration Through Deep Reinforcement Learning. *IEEE Robotics and Automation Letters* **7**, 730–737 (2022).

[19] Chrabaszcz, P., Loshchilov, I. & Hutter, F. Back to Basics: Benchmarking Canonical Evolution Strategies for Playing Atari (2018). ArXiv:1802.08842 [cs].

[20] Nagabandi, A., Kahn, G., Fearing, R. S. & Levine, S. Neural Network Dynamics for Model-Based Deep Reinforcement Learning with Model-Free Fine-Tuning (2017). ArXiv:1708.02596 [cs].

[21] Merel, J. *et al.* Deep neuroethology of a virtual rodent (2019). ArXiv:1911.09451 [q-bio].

[22] Lobato-Rios, V. *et al.* NeuroMechFly, a neuromechanical model of adult Drosophila melanogaster. *Nature Methods* **19** (2022).

[23] Merel, J., Botvinick, M. & Wayne, G. Hierarchical motor control in mammals and machines. *Nature Communications* **10** (2019).

[24] Raji, J. I. & Potter, C. J. The number of neurons in Drosophila and mosquito brains. *PLOS ONE* **16**, e0250381 (2021).

[25] Herculano-Houzel, S., Mota, B. & Lent, R. Cellular scaling rules for rodent brains. *Proceedings of the National Academy of Sciences* **103**, 12138–12143 (2006).

[26] Azevedo, F. A. *et al.* Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain. *Journal of Comparative Neurology* **513**, 532–541 (2009).

[27] Azevedo, A. *et al.* Tools for comprehensive reconstruction and analysis of Drosophila motor circuits (2022). URL https://www.biorxiv.org/content/10.1101/2022.12.15.520299v1.

[28] Bjugn, R. The use of the optical disector to estimate the number of neurons, glial and endothelial cells in the spinal cord of the mouse — with a comparative note on the rat spinal cord. *Brain Research* **627**, 25–33 (1993).

[29] Bahney, J. & von Bartheld, C. S. The Cellular Composition and Glia–Neuron Ratio in the Spinal Cord of a Human and a Nonhuman Primate: Comparison With Other Species and Brain Regions. *The Anatomical Record* **301**, 697–710 (2018).

[30] Pick, S. & Strauss, R. Goal-Driven Behavioral Adaptations in Gap-Climbing Drosophila. *Current Biology* **15**, 1473–1478 (2005).

[31] Muijres, F. T., Elzinga, M. J., Melis, J. M. & Dickinson, M. H. Flies Evade Looming Targets by Executing Rapid Visually Directed Banked Turns. *Science* **344**, 172–177 (2014).

[32] Pavlou, H. J. & Goodwin, S. F. Courtship behavior in Drosophila melanogaster: towards a 'courtship connectome'. *Current Opinion in Neurobiology* **23**, 76–83 (2013).

[33] Hoopfer, E. D. Neural control of aggression in Drosophila. *Current Opinion in Neurobiology* **38**, 109–118 (2016).

[34] Wolf, R. *et al.* Drosophila Mushroom Bodies Are Dispensable for Visual, Tactile, and Motor Learning. *Learning & Memory* **5**, 166–178 (1998).

[35] Tsubouchi, A. *et al.* Topological and modality-specific representation of somatosensory information in the fly brain. *Science* **358**, 615–623 (2017).

[36] Strausfeld, N. J. & Hirth, F. Deep Homology of Arthropod Central Complex and Vertebrate Basal Ganglia. *Science* **340**, 157–161 (2013).

[37] Dorkenwald, S. *et al.* Neuronal wiring diagram of an adult brain (2023). URL https://www.biorxiv.org/content/10.1101/2023.06.27.546656v1.

[38] Phelps, J. S. *et al.* Reconstruction of motor control circuits in adult Drosophila using automated transmission electron microscopy. *Cell* **184**, 759–774.e18 (2021).

[39] Takemura, S.-y. *et al.* A Connectome of the Male Drosophila Ventral Nerve Cord (2023). URL https://www.biorxiv.org/content/10.1101/2023.06.05.543757v1.

[40] Jenett, A. *et al.* A GAL4-Driver Line Resource for Drosophila Neurobiology. *Cell Reports* **2**, 991–1001 (2012).

[41] Klapoetke, N. C. *et al.* Independent optical excitation of distinct neural populations. *Nature Methods* **11**, 338–346 (2014).

[42] Chen, T.-W. *et al.* Ultrasensitive fluorescent proteins for imaging neuronal activity. *Nature* **499**, 295–300 (2013).

[43] Lappalainen, J. K. *et al.* Connectome-constrained deep mechanistic networks predict neural responses across the fly visual system at single-neuron resolution (2023). URL https://www.biorxiv.org/content/10.1101/2023.03.11.532232v1.

[44] Shiu, P. K. *et al.* A leaky integrate-and-fire computational model based on the connectome of the entire adult Drosophila brain reveals insights into sensorimotor processing (2023). URL https://www.biorxiv.org/content/10.1101/2023.05.02.539144v1.

[45] Todorov, E., Erez, T. & Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 5026–5033 (IEEE, 2012).

[46] Erez, T., Tassa, Y. & Todorov, E. Simulation tools for model-based robotics: Comparison of Bullet, Havok, MuJoCo, ODE and PhysX. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 4397–4404 (2015).

[47] Günel, S. *et al.* DeepFly3D, a deep learning-based approach for 3D limb and appendage tracking in tethered, adult Drosophila. *eLife* **8**, e48571 (2019).

[48] Whitney, H. M. & Federle, W. Biomechanics of plant–insect interactions. *Current Opinion in Plant Biology* **16**, 105–111 (2013).

[49] Gorb, S. N. *et al.* Structural design and biomechanics of friction-based releasable attachment devices in insects. *Integrative and Comparative Biology* **42**, 1127–1139 (2002).

[50] Hosoda, N., Nakamoto, M., Suga, T. & Gorb, S. N. Evidence for intermolecular forces involved in ladybird beetle tarsal setae adhesion. *Scientific Reports* **11**, 7729 (2021).

[51] Bullock, J. M. R., Drechsler, P. & Federle, W. Comparison of smooth and hairy attachment pads in insects: friction, adhesion and mechanisms for direction-dependence. *Journal of Experimental Biology* **211**, 3333–3343 (2008).

[52] Drechsler, P. & Federle, W. Biomechanics of smooth adhesive pads in insects: influence of tarsal secretion on attachment performance. *Journal of Comparative Physiology A* **192**, 1213–1222 (2006).

[53] Federle, W., Rohrseitz, K. & Hölldobler, B. Attachment forces of ants measured with a centrifuge: better 'wax-runners' have a poorer attachment to a smooth surface. *Journal of Experimental Biology* **203**, 505–512 (2000).

[54] Dirks, J.-H. Physical principles of fluid-mediated insect attachment—Shouldn't insects slip? *Beilstein Journal of Nanotechnology* **5**, 1160–1166 (2014).

[55] Strauss, R. & Heisenberg, M. Coordination of legs during straight walking and turning in drosophila melanogaster. *Journal of Comparative Physiology A, Sensory, Neural, and Behavioral Physiology* **167**, 403–412 (1990).

[56] Ramdya, P. *et al.* Climbing favours the tripod gait over alternative faster insect gaits. *Nature Communications* **8**, 14494 (2017).

[57] Szczecinski, N. S., Bockemühl, T., Chockley, A. S. & Büschges, A. Static stability predicts the continuum of interleg coordination patterns in drosophila. *Journal of Experimental Biology* **221**, jeb189142 (2018).

[58] Mantziaris, C., Bockemühl, T. & Büschges, A. Central pattern generating networks in insect locomotion. *Developmental Neurobiology* **80**, 16–30 (2020).

[59] MacKay-Lyons, M. Central Pattern Generation of Locomotion: A Review of the Evidence. *Physical Therapy* **82**, 69–83 (2002).

[60] Guertin, P. A. The mammalian central pattern generator for locomotion. *Brain Research Reviews* **62**, 45–56 (2009).

[61] Ayali, A. *et al.* Sensory feedback in cockroach locomotion: current knowledge and open questions. *Journal of Comparative Physiology A* **201**, 841–850 (2015).

[62] Ijspeert, A. J. Central pattern generators for locomotion control in animals and robots: A review. *Neural Networks* **21**, 642–653 (2008).

[63] Bellegarda, G. & Ijspeert, A. CPG-RL: Learning Central Pattern Generators for Quadruped Locomotion. *IEEE Robotics and Automation Letters* **7**, 12547–12554 (2022).

[64] Cagan, R. Principles of Drosophila Eye Differentiation. In *Current Topics in Developmental Biology*, vol. 89, 115–135 (Academic Press, 2009).

[65] Hindmarsh Sten, T., Li, R., Otopalik, A. & Ruta, V. Sexual arousal gates visual processing during Drosophila courtship. *Nature* **595**, 549–553 (2021).

[66] Strother, J. A. *et al.* Behavioral state modulates the ON visual motion pathway of Drosophila. *Proceedings of the National Academy of Sciences* **115**, E102–E111 (2018).

[67] Rister, J., Desplan, C. & Vasiliauskas, D. Establishing and maintaining gene expression patterns: insights from sensory receptor patterning. *Development* **140**, 493–503 (2013).

[68] Sharkey, C. R., Blanco, J., Leibowitz, M. M., Pinto-Benito, D. & Wardill, T. J. The spectral sensitivity of Drosophila photoreceptors. *Scientific Reports* **10**, 18242 (2020).

[69] Benton, R. Drosophila olfaction: past, present and future. *Proceedings of the Royal Society B: Biological Sciences* **289**, 20222054 (2022).

[70] Wilson, R. I. Early olfactory processing in drosophila: mechanisms and principles. *Annual review of neuroscience* **36**, 217–241 (2013).

[71] Schlegel, P. *et al.* Information flow, cell types and stereotypy in a full olfactory connectome. *eLife* **10**, e66018 (2021).

[72] Cognigni, P., Felsenberg, J. & Waddell, S. Do the right thing: neural network mechanisms of memory formation, expression and update in drosophila. *Current Opinion in Neurobiology* **49**, 51–58 (2018).

[73] Coen, P. *et al.* Dynamic sensory cues shape song structure in drosophila. *Nature* **507**, 233–237 (2014).

[74] van Breugel, F. Correlated decision making across multiple phases of olfactory-guided search in drosophila improves search efficiency. *Journal of Experimental Biology* **224**, jeb242267 (2021).

[75] Taisz, I. *et al.* Generating parallel representations of position and identity in the olfactory system. *Cell* **186**, 2556–2573 (2023).

[76] Creamer, M. S., Mano, O. & Clark, D. A. Visual control of walking speed in drosophila. *Neuron* **100**, 1460–1473 (2018).

[77] Olsen, S. R. & Wilson, R. I. Lateral presynaptic inhibition mediates gain control in an olfactory circuit. *Nature* **452**, 956–960 (2008).

[78] Kadakia, N. *et al.* Odour motion sensing enhances navigation of complex plumes. *Nature* **611**, 754–761 (2022).

[79] Chen, C.-L. *et al.* Ascending neurons convey behavioral state to integrative sensory and action selection brain regions. *Nature Neuroscience* **26**, 682–695 (2023).

[80] Kane, G. A., Lopes, G., Saunders, J. L., Mathis, A. & Mathis, M. W. Real-time, low-latency closed-loop feedback using markerless posture tracking. *eLife* **9**, e61909 (2020).

[81] Ramdya, P. & Ijspeert, A. J. The neuromechanics of animal locomotion: From biology to robotics and back. *Science Robotics* **8**, eadg0279 (2023).

[82] Makoviychuk, V. *et al.* Isaac gym: High performance GPU based physics simulation for robot learning. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)* (2021).

[83] Freeman, C. D. *et al.* Brax – A Differentiable Physics Engine for Large Scale Rigid Body Simulation (2021). ArXiv:2106.13281 [cs].

[84] Murthy, J. K. *et al.* gradsim: Differentiable simulation for system identification and visuomotor control. In *International Conference on Learning Representations* (2021).

[85] Werling, K., Omens, D., Lee, J., Exarchos, I. & Liu, C. K. Fast and Feature-Complete Differentiable Physics for Articulated Rigid Bodies with Contact (2021). ArXiv:2103.16021 [cs, eess].

[86] Farrell, J. A., Murlis, J., Long, X., Li, W. & Cardé, R. T. Filament-Based Atmospheric Dispersion Model to Achieve Short Time-Scale Structure of Odor Plumes. *Environmental Fluid Mechanics* **2**, 143–169 (2002).

[87] Álvarez-Salvado, E. *et al.* Elementary sensory-motor transformations underlying olfactory navigation in walking fruit-flies. *eLife* **7**, e37815 (2018).

[88] Singh, S. H., van Breugel, F., Rao, R. P. N. & Brunton, B. W. Emergent behaviour and neural dynamics in artificial agents tracking odour plumes. *Nature Machine Intelligence* **5**, 58–70 (2023).

[89] von Reyn, C. R. *et al.* A spike-timing mechanism for action selection. *Nature Neuroscience* **17**, 962–970 (2014).

[90] Braun, J., Hurtak, F., Wang-Chen, S. & Ramdya, P. Networks of descending neurons transform command-like signals into population-based behavioral control (2023). URL https://www.biorxiv.org/content/10.1101/2023.09.11.557103v1.

[91] Cruse, H., Bartling, C., Cymbalyuk, G., Dean, J. & Dreifert, M. A modular artificial neural net for controlling a six-legged walking system. *Biological Cybernetics* **72**, 421–430 (1995).

[92] Marin, E. C. *et al.* Systematic annotation of a complete adult male Drosophila nerve cord connectome reveals principles of functional organisation (2023). URL https://www.biorxiv.org/content/10.1101/2023.06.05.543407v1.

[93] Manoonpong, P., Parlitz, U. & Wörgötter, F. Neural control and adaptive neural forward models for insect-like, energy-efficient, and adaptable locomotion of walking machines. *Frontiers in Neural Circuits* **7**, 12 (2013).

[94] Brooks, R. A robust layered control system for a mobile robot. *IEEE Journal on Robotics and Automation* **2**, 14–23 (1986).

[95] Kuan, A. T. *et al.* Dense neuronal reconstruction through X-ray holographic nano-tomography. *Nature Neuroscience* **23**, 1637–1643 (2020).

[96] Azevedo, A. W. *et al.* A size principle for recruitment of drosophila leg motor neurons. *eLife* **9**, e56754 (2020).

[97] Peng, X. B. *et al.* Learning Agile Robotic Locomotion Skills by Imitating Animals (2020). ArXiv:2004.00784 [cs].

[98] Haarnoja, T. *et al.* Learning to Walk via Deep Reinforcement Learning (2019). ArXiv:1812.11103 [cs].

[99] Tunyasuvunakool, S. *et al.* dm_control: Software and tasks for continuous control. *Software Impacts* **6**, 100022 (2020).

[100] Schneider, D. Insect Antennae. *Annual Review of Entomology* **9**, 103–122 (1964).

[101] Suver, M. P., Medina, A. M. & Nagel, K. I. Active antennal movements in drosophila can tune wind encoding. *Current Biology* **33**, 780–789 (2023).

[102] Todi, S. V., Sharma, Y. & Eberl, D. F. Anatomical and molecular design of the drosophila antenna as a flagellar auditory organ. *Microscopy Research and Technique* **63**, 388–399 (2004).

[103] Arreguit, J., Ramalingasetty, S. T. & Ijspeert, A. J. FARMS: Framework for Animal and Robot Modeling and Simulation (2023). URL https://www.biorxiv.org/content/10.1101/2023.09.25.559130v1.

[104] Cruse, H. What mechanisms coordinate leg movement in walking arthropods? *Trends in Neurosciences* **13**, 15–21 (1990).

[105] Schilling, M., Hoinville, T., Schmitz, J. & Cruse, H. Walknet, a bio-inspired controller for hexapod walking. *Biological Cybernetics* **107**, 397–419 (2013).

[106] Sancer, G. *et al.* Modality-Specific Circuits for Skylight Orientation in the Fly Visual System. *Current Biology* **29**, 2812–2825.e4 (2019).

[107] Bruyne, M. d., Clyne, P. J. & Carlson, J. R. Odor Coding in a Model Olfactory Organ: The Drosophila Maxillary Palp. *Journal of Neuroscience* **19**, 4520–4532 (1999).

[108] Kipf, T. N. & Welling, M. Semi-Supervised Classification with Graph Convolutional Networks (2017). ArXiv:1609.02907 [cs, stat].

[109] Paszke, A. *et al.* PyTorch: An Imperative Style, High-Performance Deep Learning Library (2019). ArXiv:1912.01703 [cs, stat].

[110] Fey, M. & Lenssen, J. E. Fast Graph Representation Learning with PyTorch Geometric (2019). ArXiv:1903.02428 [cs, stat].

[111] Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization (2017). ArXiv:1412.6980 [cs].

[112] Haarnoja, T., Zhou, A., Abbeel, P. & Levine, S. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning*, 1861–1870 (PMLR, 2018).

[113] Raffin, A. *et al.* Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research* **22**, 1–8 (2021).