nature methods

Article

https://doi.org/10.1038/s41592-024-02497-y

NeuroMechFly v2: simulating embodied sensorimotor control in adult *Drosophila*

Received: 18 September 2023

Accepted: 30 September 2024

Published online: 12 November 2024

Check for updates

Sibo Wang-Chen[1] ✉, Victor Alfred Stimpfling[1], Thomas Ka Chung Lam[1], Pembe Gizem Özdil[1,2], Louise Genoud[1], Femke Hurtak[1] & Pavan Ramdya[1] ✉

Discovering principles underlying the control of animal behavior requires a tight dialogue between experiments and neuromechanical models. Such models have primarily been used to investigate motor control with less emphasis on how the brain and motor systems work together during hierarchical sensorimotor control. NeuroMechFly v2 expands *Drosophila* neuromechanical modeling by enabling vision, olfaction, ascending motor feedback and complex terrains that can be navigated using leg adhesion. We illustrate its capabilities by constructing biologically inspired controllers that use ascending feedback to perform path integration and head stabilization. After adding vision and olfaction, we train a controller using reinforcement learning to perform a multimodal navigation task. Finally, we illustrate more bio-realistic modeling involving complex odor plume navigation, and fly–fly following using a connectome-constrained visual network. NeuroMechFly can be used to accelerate the discovery of explanatory models of the nervous system and to develop machine learning-based controllers for autonomous artificial agents and robots.

How the nervous system controls behavior is a deeply entangled problem: nested feedback loops occur at multiple levels including the physiology of individual neurons, the recurrent dynamics of neural circuits, biomechanical interactions with the environment and sensory signals resulting from one's own actions. Therefore, investigating and modeling any of these elements in isolation, although useful, will always be fundamentally incomplete. To overcome this gap, neuroscience requires simulation frameworks that enable the exploration of hierarchical feedback loops in an end-to-end manner. Numerous detailed neuromechanical models have been developed to explore how animals control motor programs like walking[1,2], swimming[3] and transitions between them[4]. As well, in the field of reinforcement learning more abstract and simplified 'creatures'[5] have been widely used to model visuomotor coordination[6], decision-making[7] and learning algorithms[8]. 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 (DoFs)[9]. Thus, although progress is being made[10–15], a substantial gap remains at the interface between machine learning models and morphologically realistic neuromechanical models of most animals. An ideal model would enable the exploration of hierarchical controllers[16]—like those found in biological agents—which include higher-order brain-like systems that integrate multimodal sensory inputs, ascending motor feedback and internal states as well as lower-level motor systems that decode descending brain commands and execute behaviors (Fig. 1a).

The adult fruit fly, *Drosophila melanogaster*, is an ideal animal for comprehensively modeling hierarchical control. It has only ~200,000 neurons in its brain[17] (about 10^3–10^6 times smaller than the mouse and human brains); its principal motor system (the ventral nerve cord or VNC) has only ~15,000 neurons[18] (about 10^2–10^4 times smaller than the mouse and human spinal cords). Using this small number of neurons, flies can nevertheless generate complex behaviors. They can walk over complex terrain[19], make rapid corrective maneuvers during flight[20], execute courtship sequences[21], fight off competitors[22] and learn[23]. The organization of the fly's nervous system resembles that of vertebrates,

[1]Neuroengineering Laboratory, Brain Mind Institute & Interfaculty Institute of Bioengineering, EPFL, Lausanne, Switzerland. [2]Biorobotics Laboratory, EPFL, Lausanne, Switzerland. ✉e-mail: sibo.wang@epfl.ch; pavan.ramdya@epfl.ch

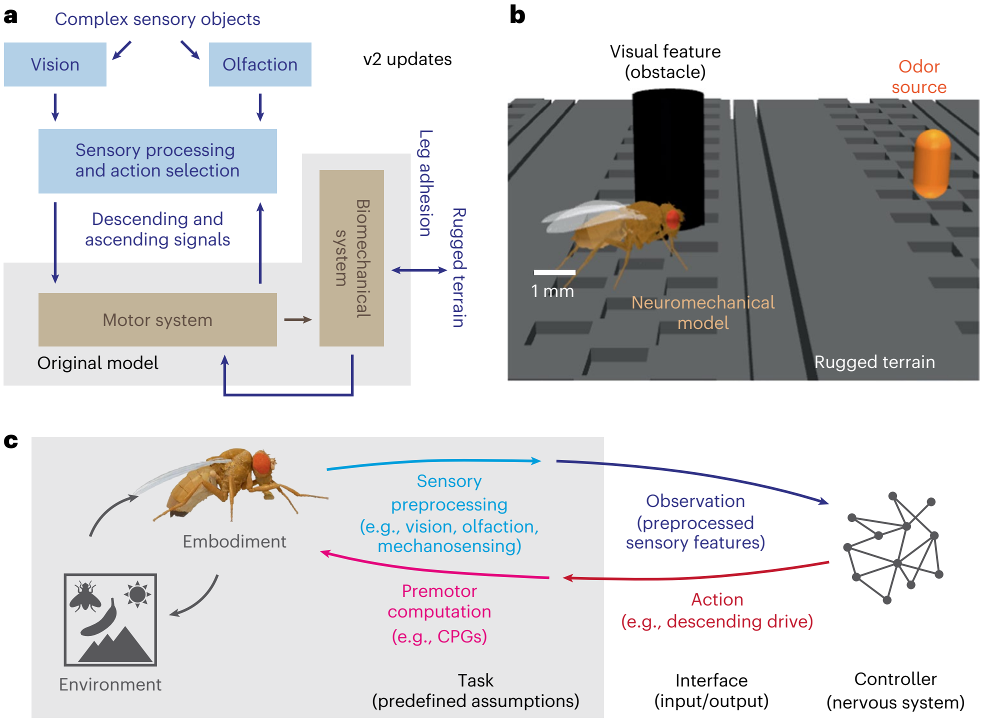

a
Complex sensory objects
Vision
Olfaction
v2 updates
Sensory processing and action selection
Descending and ascending signals
Leg adhesion
Biomechanical system
Rugged terrain
Motor system
Original model
b
Visual feature (obstacle)
Odor source
1 mm
Neuromechanical model
Rugged terrain
c
Embodiment
Sensory preprocessing (e.g., vision, olfaction, mechanosensing)
Observation (preprocessed sensory features)
Premotor computation (e.g., CPGs)
Action (e.g., descending drive)
Environment
Task (predefined assumptions)
Interface (input/output)
Controller (nervous system)

Fig. 1 | Schematic overview of the NeuroMechFly v2 modeling framework. **a**, Diagram highlighting the integrative and hierarchical aspects of the expanded NeuroMechFly framework. Sensory inputs—visual and olfactory—are processed by the brain to select an appropriate action. Instructions to execute the selected action are then communicated to the lower-level motor centers (VNC) through descending signals. The VNC executes the motor program via joint actions and mechanosensory feedback signals are sent back to the motor system to inform the next movement and to the brain via ascending pathways. Indicated are components that were present in the original model but improved (brown, in shaded box) as well as new elements (blue, outside shaded box). **b**, Simulation camera view of the neuromechanical model, NeuroMechFly, walking 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 POMDP task. A user-defined controller interfaces with the task through actions (red) and observations (dark blue). The user can extend the POMDP task by adding preprogrammed processing routines for sensory inputs (light blue) and premotor outputs (magenta), to modify the action and observation spaces handled by the controller.

yet it has been completely mapped in connectomes (that is, wiring diagrams) of the brain[24] and VNC[25,26]. Additionally, thousands of transgenic fly lines enable the repeated targeting of sparse sets of neurons[27] for optical activation[28], silencing[29] and recordings[30]. These resources and tools have led to the development of connectome-constrained models of neural circuits[31,32]. However, we still lack an integrative simulation framework of the fly, embedded in a physics simulator, to embody and explore these models to identify the principles governing biological intelligence and autonomous behavioral control.

Toward this goal, we previously developed NeuroMechFly, a morphologically realistic model of the adult fly[13]. With its biomechanical hull, we could infer unmeasured forces and collisions during the simulated replay of recorded limb kinematics from walking and grooming flies. Furthermore, we could optimize simple coupled oscillators to control fast and stable tethered walking. Although foundational, this previous simulation framework could not fully model hierarchical sensorimotor control: complex environments, brain-level sensory processing and the physics and biomechanics required for untethered behavioral control were lacking. Here we describe NeuroMechFly v2, a simulation framework that addresses these gaps by (i) improving biomechanics and stepping, (ii) adding leg adhesion, (iii) simulating visual and olfactory sensing, and (iv) enriching the fly's environment with rugged terrain, obstacles and sensory objects including other flies (Fig. 1a,b). We illustrate the exploration of locomotor control over rugged terrain, simple visual object tracking and simple odor taxis. Next, we demonstrate the use of ascending limb motor signals to perform path integration and head stabilization. Then, we combine these elements to build integrated hierarchical machine learning models that solve a multimodal task—visually avoiding an obstacle to reach an attractive odor source over rugged terrain. We show how this artificial neural network controller can be trained using reinforcement learning. Finally, we illustrate more biological realism by modeling a *Drosophila* odor-taxis strategy to navigate a complex odor plume and by using a connectome-constrained visual system network to perform fly–fly following. The modularity of NeuroMechFly v2 allows users to flexibly interact with the simulation at multiple levels of abstraction and facilitates its widespread adoption for research and education. Our implementation's compliance with a standard reinforcement learning task interface can also facilitate a dialogue between neuroscience, machine learning and robotics (Fig. 1c and Supplementary Note 1).

Results

The FlyGym package: a standardized simulation framework

To improve the usability of the NeuroMechFly simulation framework (released as the FlyGym Python package; https://neuromechfly.org/), we made three fundamental changes. First, the package fully complies with Gymnasium[5], a standard interface for controller–environment interaction in robotics and reinforcement learning (Fig. 1c). Second, we moved the simulation framework from PyBullet to MuJoCo[33], a more intensively maintained and widely used physics simulator. MuJoCo is known for better stability and performance[34] and supports a wider range of actuators including those for leg adhesion. It was made an open-source tool after NeuroMechFly v1 was published. Third, to facilitate implementing custom environmental features within the simulation, we expanded the interface for the fly model's arena.

In NeuroMechFly v2, we clarify the framing of the control problem as a partially observable Markov decision process (POMDP). 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 input, olfactory input, ground contacts, joint states (angles, angular velocities and torques) and the states (for example, position, orientation) of potentially multiple fly models in the arena. The action space includes the control signals (for example, target angles) for each actuated DoF and the on/off signal for leg adhesion. Users can easily extend this framework by incorporating additional sensory or premotor processing into the Markov decision process (Fig. 1c). For example, in a visual taxis example described below, we programmed the centroid calculation to reduce the observation space to fewer dimensions (such as the azimuth of the object seen from each eye) and used a network of central coupled oscillators to reduce the action space to a two-dimensional descending command. This increased flexibility and usability allows one to rapidly adapt the simulation to their own research question.

Improved morphological accuracy and kinematic realism

A promise of biomechanical models such as NeuroMechFly is to allow researchers studying body movements to, for example, infer unmeasured collisions, contacts and forces during the replay of real recorded body-part kinematics[13]. However, doing so at a high precision is only possible with a morphologically realistically rigged body model. In particular, modeling behaviors that depend on self-contact (such as grooming) requires precise kinematic replay to read out where and how individual body parts interact with one another. Therefore, we improved the morphological accuracy and granularity of several body-part meshes. First, we adjusted the placement and default angles of 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. Second, to better facilitate control of the antennae and readout of their mechanosensory signals, we split their meshes into three segments: pedicel, funiculus and arista. We added DoFs between these segments, allowing each to be separately actuated or passively moved and its angular displacement to be measured (Extended Data Fig. 1c). We note that users can simplify body geometries if morphological accuracy is of less importance than computational speed.

We also improved the realism of limb kinematics during walking. Leg kinematics in NeuroMechFly v1 (ref. 13) were based on data from tethered flies walking on a spherical treadmill. In our simulation these kinematics appeared unnatural during untethered walking, turning and locomotion over rugged terrain. Therefore, to obtain realistic untethered three-dimensional (3D) leg kinematics, we designed a system to record three views of a fly walking straight through a corridor. We annotated and triangulated key points from these data and extracted individual steps. For each pair of legs, we segmented and processed a step while enforcing symmetry, closure and equal lengths (Extended Data Fig. 2 and Supplementary Video 1). Replaying and looping these steps in NeuroMechFly (Supplementary Video 2) drove straight walking more closely resembling that of a real fly.

Leg tip adhesion enables locomotion in three dimensions

Insects, including flies, use highly specialized adhesive structures to walk over complex 3D terrain with ease. These include adhesive pads with substantial normal forces (>100 times body weight) and frictional forces[35]. Adhesion provides mechanical coupling between the legs during locomotion and improves force transduction with the ground. We cannot easily model the physics of real adhesion. Therefore, we modeled leg adhesion as an additional normal force when a leg tip (that is, pretarsus) is in contact with the surface (Fig. 2a and Supplementary Video 3). As for insects[35], this normal force also increases the frictional forces. Despite the huge forces generated by adhesive pads, insects appear to be able to lift their legs without much effort. Although liftoff mechanisms are known for some insects[36,37], they are not known for *D. melanogaster*. Therefore, we abstracted the mechanisms used by other insects and lifted the legs during walking by turning off adhesion forces during the swing phase. To illustrate how leg adhesion expands the behavioral repertoire of our model, we simulated tripod-gait walking[38,39] over terrain with up to 180° of inclination (Fig. 2b and Supplementary Video 4). Without adhesion, the fly slipped at as low as 30° inclination. By contrast, with adhesion the fly could locomote over terrain with sometimes more than >90° of inclination (Fig. 2c). We expect experimental recordings of real inverted walking kinematics to enable the simulation of locomotion at even higher inclinations.

Complex terrains demonstrate the use of locomotor feedback

A variety of mechanisms have been proposed for insect locomotion ranging across a spectrum from those depending purely on central pattern generators (CPGs, namely circuits in the central nervous system that produce rhythmic motor output without rhythmic input[40]) to those relying on sensory feedback-based rules[1]. Evidence for each of these control strategies has been found across species, motivating their application in robotics[40–42]. Although walking over flat terrain can be solved using a variety of feedback-independent control strategies, leg mechanosensory signals are thought to be required to navigate rugged terrain. To demonstrate that NeuroMechFly can serve as a test bed to evaluate different control strategies in complex environments, we developed three rugged terrain types to compare with smooth terrain: one with gaps perpendicular to the initial heading of the fly, one with blocks of alternating height and one that is a mixture of the previous two (Fig. 2d).

We next built controllers and benchmarked them over flat and rugged terrains. The control strategies tested include purely CPG-based (Fig. 2e and Supplementary Video 5), purely sensory feedback rule-based (Fig. 2e and Supplementary Video 6) or intermediate to these two, with CPGs but also sensory feedback rules to recover from challenging positions (Fig. 2e and Supplementary Video 7). At baseline (that is, on flat terrain), CPG and hybrid controllers were fastest (Fig. 2f). However, on rugged terrain the CPG-based controller struggled compared with the rule-based controller (Fig. 2f). The hybrid controller leveraging both CPGs and sensory feedback rules overcame this trade-off: it remained fast over rugged terrain (Fig. 2f) (Supplementary Video 8) while still being able to overcome obstacles. These results demonstrate the importance of rugged terrains in studying locomotor control: they expose the failure modes of controllers that otherwise work on flat terrain. Hereon, for our more complex sensorimotor tasks, we use the hybrid controller driven by a two-dimensional descending signal to control walking speed and turning by modulating oscillator frequencies and amplitude asymmetries, respectively (Fig. 2g).

Vision and olfaction enable sensory navigation

To reach attractive objects (for example, potential mates, or food sources), avert from repulsive features (for example, pheromones from predators) and avoid obstacles, animals use hierarchical controllers: higher-order brain systems must process sensory signals, use them to select the next course of action, and then transmit directives via descending pathways to lower-level motor systems. To simulate this sensorimotor hierarchy, we next added vision and olfaction to NeuroMechFly (Fig. 3a).

A fly's compound eye consists of ~700–750 individual units called ommatidia that are arranged in a hexagonal pattern[43]. We emulated this by attaching a color camera to each of our model's compound eyes (Fig. 3a) and then transformed each camera image into 721 bins on a

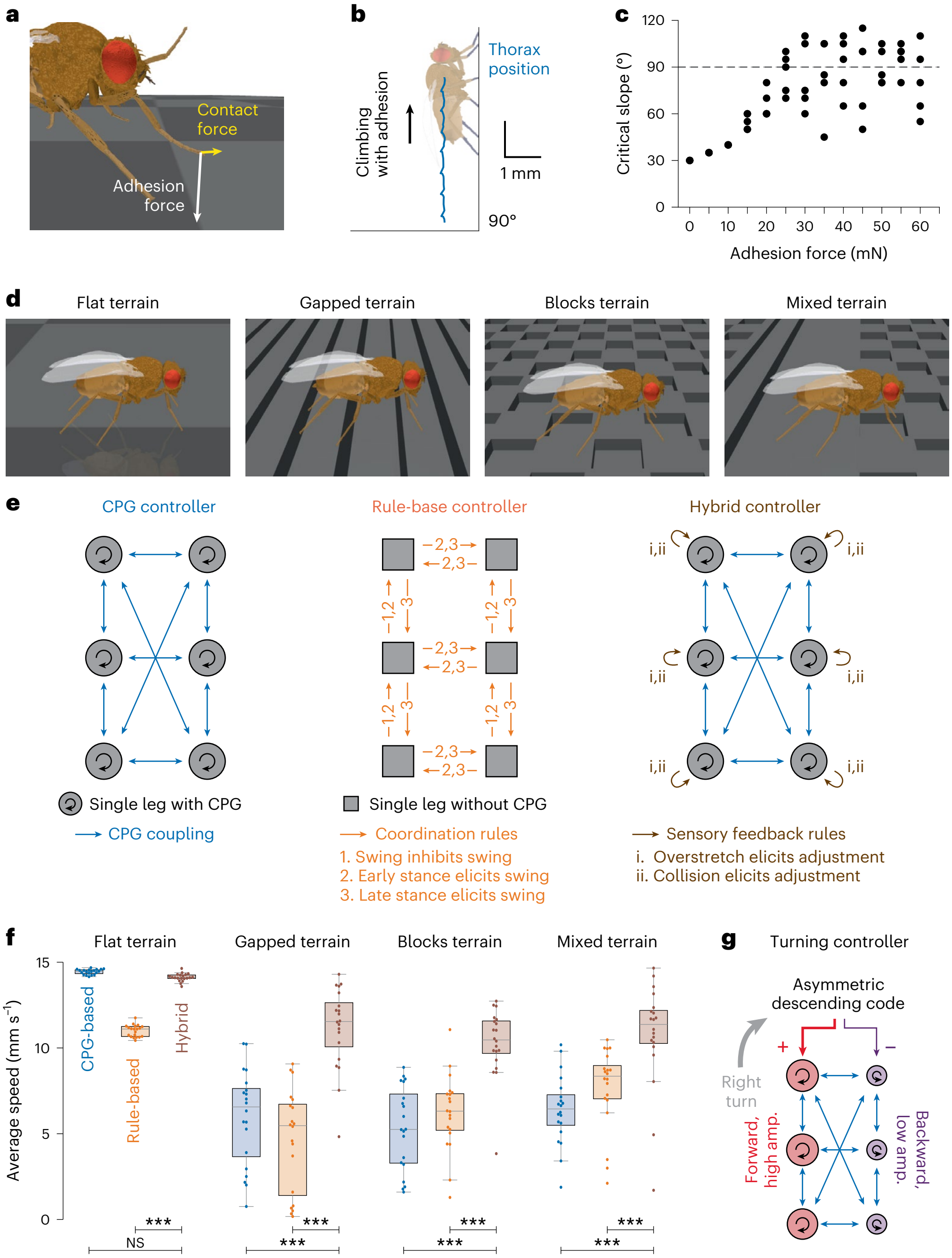


Fig. 2 | Locomotion in three dimensions and over rugged terrains. a, A schematic of contact forces on the fly's leg with the addition of adhesion. **b**, Trajectory (blue) of the fly as it walks up a vertical wall. The legs are controlled by a CPG controller with adhesion enabled (F = 40 mN). **c**, Critical slope (angle) at which the fly falls or does not proceed forward at different magnitudes of leg adhesion force. **d**, 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 that execute stepping phase-dependent adjustments when the leg might be stuck. In all cases, leg adhesion is present. **f**, The performance (average speed) of each locomotor controller while walking over four types of terrain. Shown as dots are $N = 20$ trials, each with a random spawn location and controller initialization. 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 (IQR) beyond the IQR. A one-sided, asymptotic Mann–Whitney *U* test was used to generate the statistics: NS, not significant; $^{**}P < 0.01$, $^{***}P < 0.001$ (see Supplementary Table 1 for complete statistics). **g**, Turning is controlled by the asymmetric modulation of a two-dimensional descending command signal that regulates the directions and amplitudes of oscillators on each side of the body.

hexagonal grid[31] (Fig. 3b). We assumed a 270° combined azimuth for the fly's field of view, with a ~17° binocular overlap (Extended Data Fig. 3). As an initial step toward enabling heterogeneous color sensitivity in our model, we implemented yellow-type and pale-type ommatidia—sensitive to the green and blue channels of images rendered by the physics simulator. Users can substitute the green and blue-channel values with the desired light intensities sensed by yellow-type and pale-type ommatidia to achieve more bio-realistic chromatic vision.

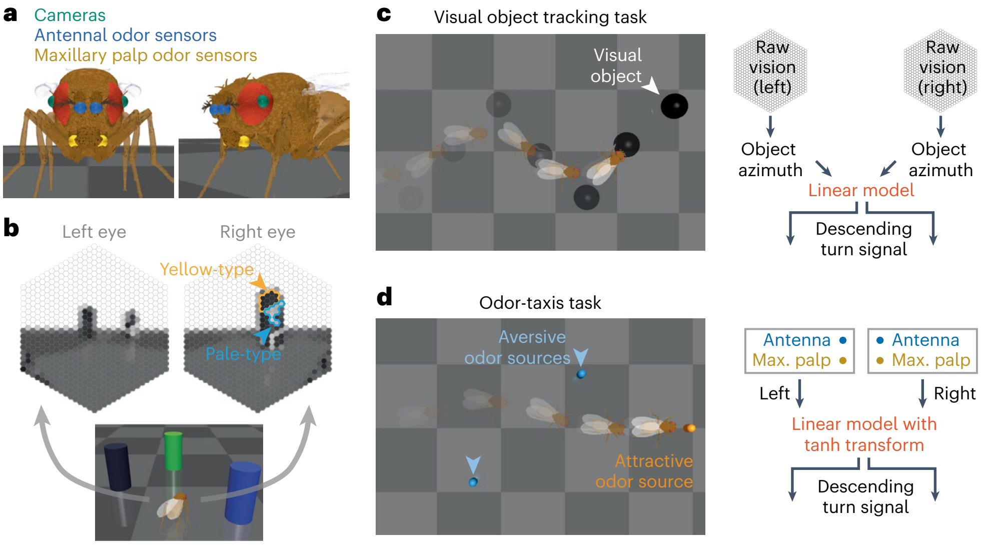


Fig. 3 | Vision and olfaction enable closed-loop sensorimotor control. a, The placement of cameras for vision (green), and odor sensors—antennae (blue) and maxillary palps (yellow)—on the fly model's head. **b**, A simulation of the visual inputs perceived by the fly while facing three different colored pillars. Ommatidia perceive the same natural color in two different intensities due to selective sensitivity to different wavelengths of the two ommatidia types. For example, the blue pillar is perceived as darker by the yellow-type and as brighter by pale-type ommatidia. Indicated are some example yellow-type and pale-type ommatidia. The positions of ommatidia types are randomly assigned. **c**, A visual object tracking task. The fly must follow a black sphere moving in an S-shaped trajectory (left). To execute this task, we first extract the azimuth of the centroid of the object on the retina for each eye (right). These azimuths are then linearly transformed and capped to generate the appropriate descending command to CPG-based controllers. **d**, An odor-taxis task. The fly must seek an attractive odor source while avoiding two aversive odor sources (left). Note that the blue and orange odor markers are for visualization purposes only and are not seen by the simulated fly. To execute this task, we first multiply the differences in mean odor intensities sensed by the antennae and maxillary palps on either side of the head by the gains of the corresponding odor types (right). The product is then passed through a nonlinear function restraining its range, which is supplied to the descending controller.

In addition to vision, we also made it possible for our model to detect odors in the simulation environment. Flies have olfactory receptor neurons (ORNs) in their antennae and maxillary palps. ORNs detect specific odorant molecules and convey this information to the brain's antennal lobe, where it is further processed[44]. We emulated olfaction by attaching virtual odor sensors to our model's antennae and maxillary palps (Fig. 3a). These virtual sensors can detect odor intensities across a multidimensional space that can be thought as representing, for example, the concentrations of monomolecular chemicals sensed by ORNs, 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 add more sensors to specific head locations and to implement additional signal processing by downstream olfactory centers (for example, lateral horn or mushroom body[45]).

To illustrate the use of visual and olfactory sensing, we implemented visual object tracking and olfactory chemotaxis. In our object tracking task, the fly model had to visually track and follow a black sphere moving along an S-shaped trajectory in the environment. The controller processed the object's visual location by computing its centroid position on the retina. Then, these visual features were linearly transformed into a two-dimensional descending signal (Fig. 3c) that modulated the frequencies and amplitudes of CPG-based oscillators on each side of the body (Fig. 2g). This strategy allowed the fly to effectively track the sphere (Fig. 3c and Supplementary Video 9). In our odor-seeking task, the simulated fly had to reach an attractive odor source while avoiding two aversive odor sources. The controller used sensors on the antennae and maxillary palps to compare the relative intensities of attractive and aversive odors across the left and right sides of the head[46]. These intensity values were multiplied by weights of opposite signs for attractive-versus-aversive odors. This left–right bias was used to asymmetrically control the descending signal (Fig. 3d), yielding effective odor-based navigation through the environment (Fig. 3d and Supplementary Video 10).

Ascending signals for path integration and head stabilization

Thus far, we have demonstrated how brain-level sensory processing can drive the motor system via descending control. The inverse, ascending signals are thought to convey information back to the brain for action selection, motor planning and sensory contextualization[47]. We next investigated how ascending feedback enables the modeling of important behaviors like path integration and head stabilization.

To effectively navigate the world, many animals, including flies[48], perform path integration wherein they constantly estimate and keep track of their own heading and distance traveled ('odometry'). The source of these idiothetic cues for path integration is unknown but may, in principle, be derived from ascending leg proprioceptive and tactile signals. We next explored how ascending proprioceptive and tactile feedback might be used to inform the brain of the change in body orientation and displacement. For each leg, we accumulated stride lengths by computing the forward translation of the leg tip relative to the thorax when the leg was in contact with the ground. Then, we computed the differences and sums of the left and right total stride lengths for each pair of legs within a short time window. These left–right differences and sums were used to predict the change in heading and forward displacement, respectively (Fig. 4a). Despite being linear, our model could give accurate predictions of the change in heading and forward displacement (Fig. 4b). These signals could be integrated over time to accurately estimate the fly's true two-dimensional position (Fig. 4c). We note, however, that the heading of the fly could sometimes be wrongly estimated (Extended Data Fig. 4) in rare instances when the heading change prediction was off by a large margin. Thus, position estimates based on idiothetic cues alone can be prone to errors in heading integration even with an exceptionally well-performing internal model ($r^2 = 0.96$; Fig. 4b). This suggests that calibration using external sensory (for example, visual) cues may be crucial.

In addition to informing path integration, ascending signals are well poised to perform other important tasks including visually

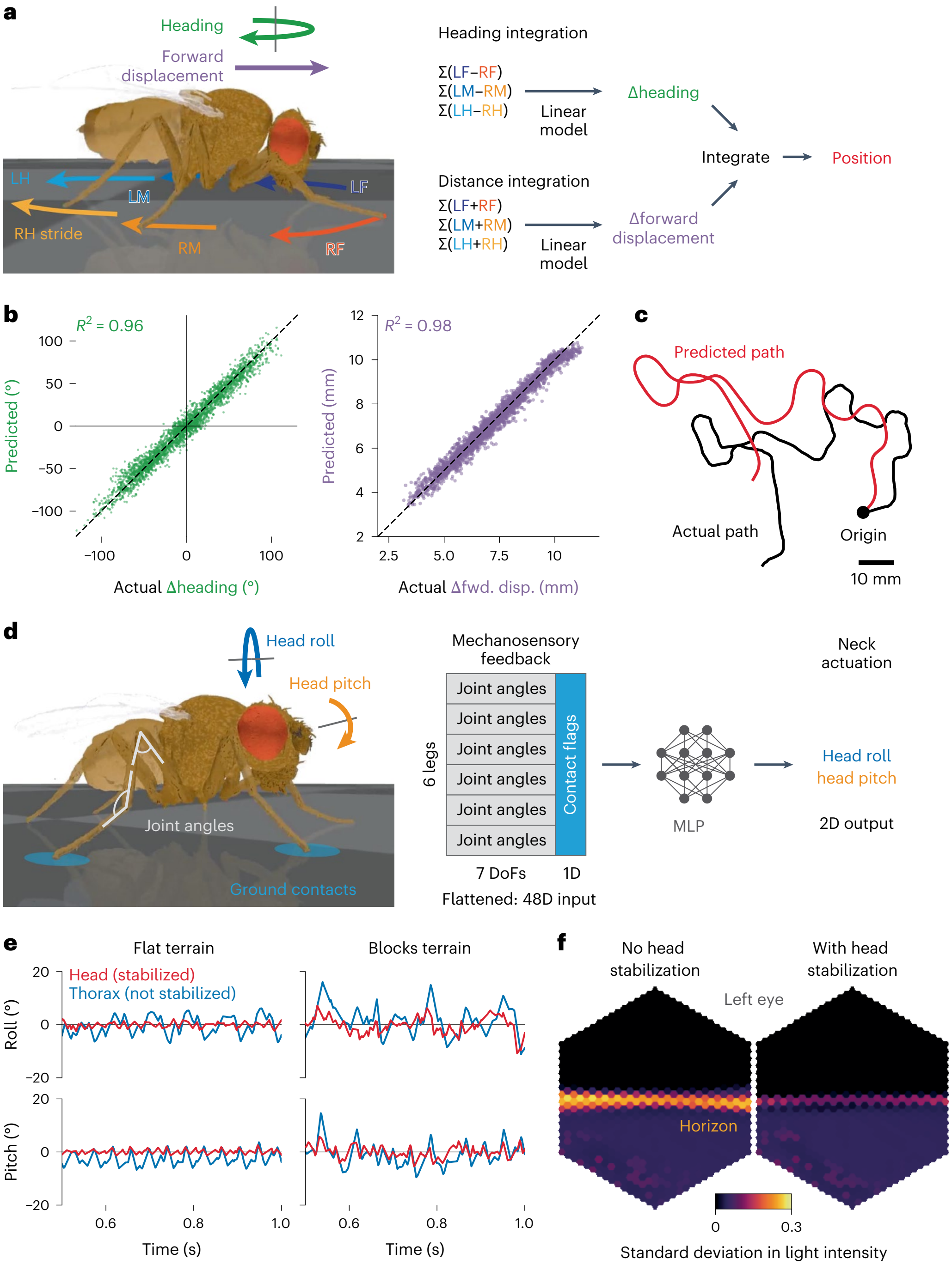


Fig. 4 | Ascending motor feedback for path integration and head stabilization. **a**, Schematic of the path integration task. The simulated fly walks in a random trajectory over flat terrain. It must use proprioceptive signals from leg strides to predict the change in heading and forward displacement and integrate these signals over time to estimate its own position. **b**, Scatterplot for 15 test trials (only 0.1% of all points are shown to facilitate visualization). Shown are the actual-versus-predicted changes in heading (left) or forward displacement (right). **c**, An example of an actual-versus-predicted walking path. **d**, Schematic of the head stabilization task. The fly model walks over flat terrain. An MLP is trained to use leg joint angles and ground contacts to predict a neck roll and pitch actuation to compensate for thoracic movements and to stabilize the visual scene. **e**, Time series of head and thoracic roll and pitch with respect to the horizon when walking over either flat or blocks terrain. Shown are data with head stabilization. Note that without head stabilization the head and the thorax are coupled. **f**, The standard deviation of ommatidia readings from the left eye while walking over flat, featureless terrain without (left) or with (right) head stabilization. Note the high variability in light intensity near the horizon when the head is not stabilized. This is due to more pronounced self-motion of the head.

tracking landmarks or targets (for example, potential mates) while navigating over rugged terrain. In this context head stabilization may be controlled using leg sensory feedback signals[49] to compensate for body pitch and roll[50]. To explore this possibility in our embodied model, we designed a controller in which leg joint angles (that is, proprioceptive signals) and ground contacts (that is, tactile signals) were fed into a multilayer perceptron (MLP). This MLP was trained to predict the appropriate neck joint actuation (that is, head roll and pitch) required to cancel visual rotations caused by the animal's own body movements during locomotion (using the hybrid controller) over either flat or blocks terrain (Fig. 4d). These predictions made from mechanosensory signals could indeed be used to dampen head movements in the roll

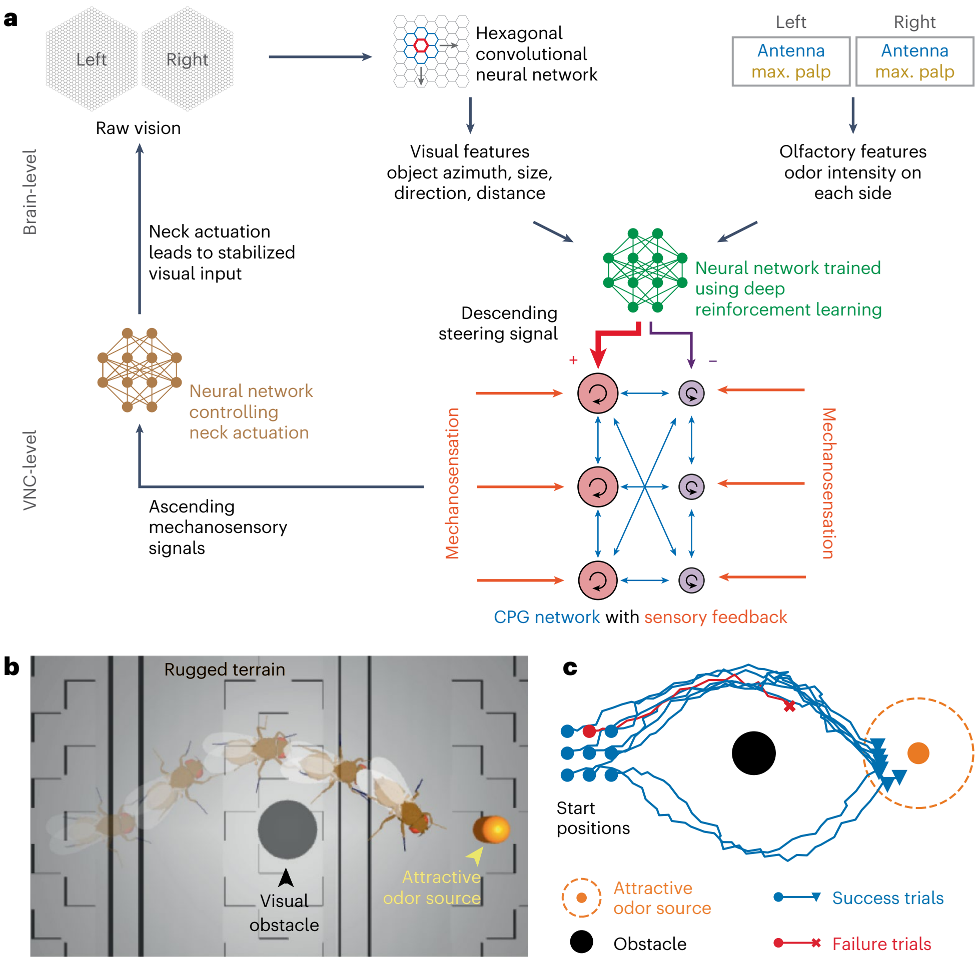

a
Left
Right
Raw vision
Hexagonal convolutional neural network
Left
Right
Antenna
max. palp
Antenna
max. palp
Visual features object azimuth, size, direction, distance
Olfactory features odor intensity on each side
Brain-level
Neck actuation leads to stabilized visual input
Neural network trained using deep reinforcement learning
Descending steering signal
+
−
Neural network controlling neck actuation
VNC-level
Ascending mechanosensory signals
Mechanosensation
Mechanosensation
CPG network with sensory feedback
b
Rugged terrain
Visual obstacle
Attractive odor source
c
Start positions
Attractive odor source
Obstacle
Success trials
Failure trials

Fig. 5 | Using reinforcement learning to train a hierarchical controller for a multimodal task. a, Schematic of the multimodal reinforcement learning-based navigation controller. Visual features are extracted using a convolutional neural network. Olfactory features are processed as in Fig. 3d. These features are input to an artificial neural network, which is trained through reinforcement learning to output appropriate descending turning commands. These commands are executed by a hybrid low-level motor controller integrating CPGs and sensory feedback and using leg adhesion. Ascending motor feedback signals are used to perform head stabilization for visual (and to a lesser extent olfactory) processing. **b**, The decision-making network is trained using reinforcement learning to enable the fly to seek an attractive odor source while avoiding a visually detected obstacle over complex terrain. An example of one successful trial is shown. The orange odor marker is for visualization purposes only and is not visible to the simulated fly. **c**, The trajectories of the fly in nine examples (eight successful; one failed) beginning from different spawn positions (circle markers).

and pitch axes compared to thoracic movements (Fig. 4e) in a manner reminiscent of data from blowflies[50]. As a result, visual inputs, especially near the horizon, were more stable (that is, exhibiting less variance in light intensity when walking in a featureless environment; Fig. 4f, Extended Data Fig. 5a,b and Supplementary Video 11). Restricting the number and kinds of leg proprioceptive inputs to the MLP shows that ascending feedback concerning multiple DoFs appears to be necessary to estimate body orientation for head stabilization (Extended Data Fig. 5c,d).

Hierarchical controller trained with reinforcement learning

With the elements of a hierarchical controller in place, it becomes possible to leverage modern machine learning approaches to train a network to accomplish more complex tasks. To illustrate this, we trained our fly model to avoid an obstacle while searching for an attractive odor source over rugged terrain. In total, our hierarchical controller[16] (Fig. 5a) consisted of: (i) a vision module (a convolutional neural network on a hexagonal lattice) that extracts the object's direction, distance, locations and sizes on the retinas (Extended Data Fig. 6); (ii) a decision module (an MLP) that receives as inputs preprocessed visual features and odor intensities from each antenna and computes a turning bias; (iii) a two-dimensional descending signal that modulates locomotor CPGs that drive walking and turning; (iv) a hybrid walking controller (Fig. 2g); and (v) an ascending feedback module that performs head stabilization. We trained the vision module in a supervised manner by randomly placing the fly and obstacle in the arena to collect training data and trained the decision module through reinforcement learning. This hierarchical controller could achieve multimodal visual–olfactory navigation over rugged terrains (Fig. 5b,c and Supplementary Video 12). This integrative task demonstrates how one can define individual components in a modular fashion and combine them to investigate a hierarchical sensorimotor task in closed loop using NeuroMechFly v2.

Using more bio-realistic algorithms for sensorimotor control

With full access to raw light intensities and odor concentrations, users can build their own sensory-rich environments and process these with models of even higher levels of biological realism. We first illustrate this by using a *Drosophila* olfactory taxis algorithm to navigate a complex odor plume. We simulated a plume embedded in airflow (Fig. 6a) representing the propagation of, for example, an attractive food odor in the real world. Unlike the simplified controller used in Fig. 3d, here we implemented a previously proposed *Drosophila* plume navigation

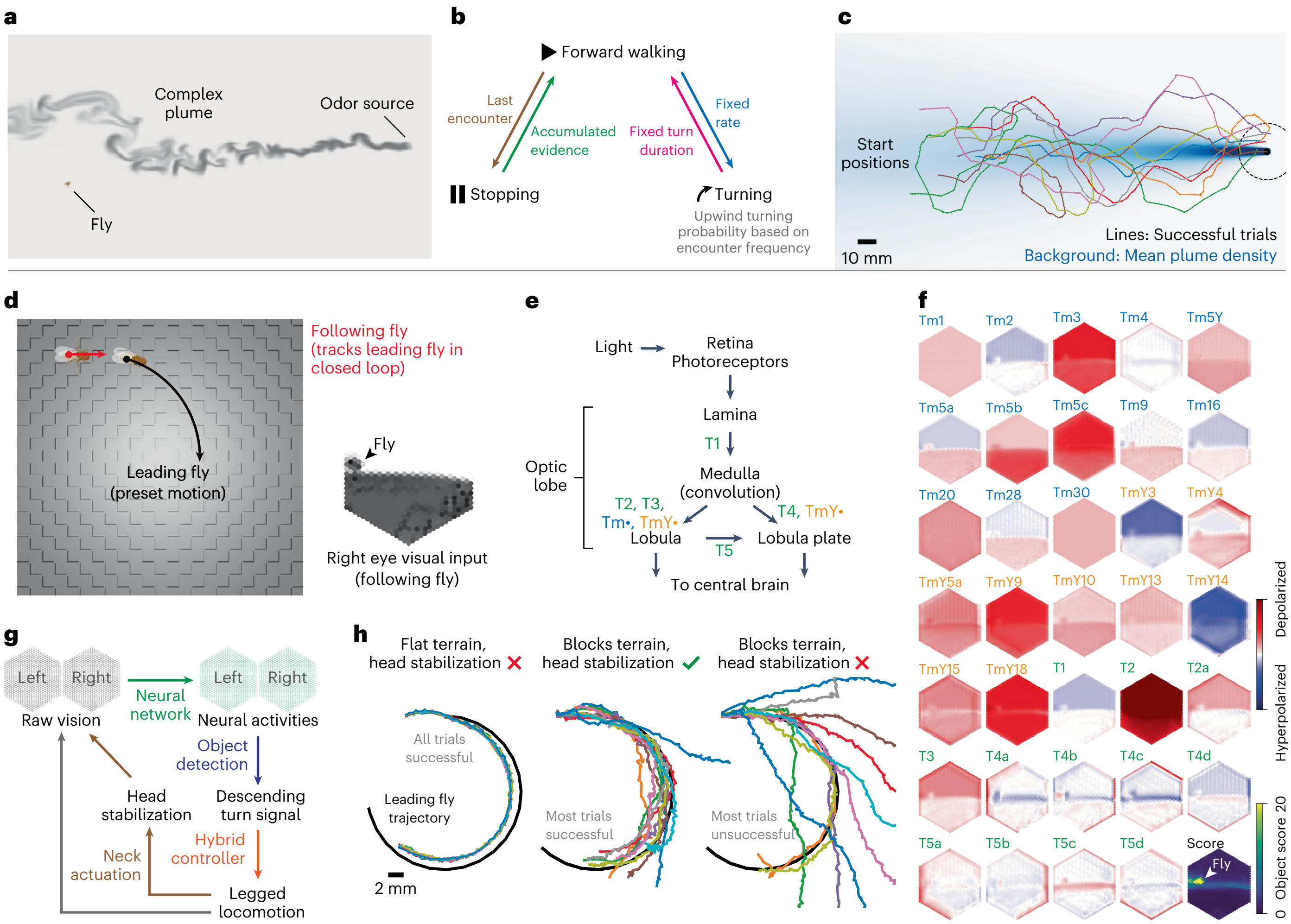


Fig. 6 | Incorporating more bio-realistic environments and controllers. **a**, Modeling a complex odor plume to simulate taxis toward an attractive odor source (from ref. 51). **b**, Transitions between locomotion states based on plume encounters. The transition from forward walking to stopping is governed by a Poisson process whose rate changes based on the time since the last encounter. The transition from stopping to forward walking is governed by a Poisson process whose rate changes based on a history of recent encounters (accumulated evidence). The transition from forward walking to turning is governed by a Poisson process with a fixed rate. Each turn has a fixed duration. The probability of turning upwind increases as the encounter frequency increases. **c**, Fly trajectories for nine successful trials when the odor source (dashed circle) was reached. Trajectories are color coded. The mean plume density is shown in blue on the background. **d**, Overview of the fly following task. Two fly models are spawned. The first ('leading') fly has a preset circular walking trajectory. The second ('following') fly uses a hierarchical controller to follow the leading fly. **e**, Block diagram of the *Drosophila* optic lobe. Indicated in blue and green are the neurons used to detect the leading fly. Only connections directly relevant to the neurons used in the fly-tracking task are shown. **f**, Object detection score and activity patterns of 34 putative output columnar neuron types (of 65 total) drawn from a connectome-constrained visual system network[31] when the leading fly is moving in the field of view of the 'following' fly's right eye. **g**, Schematic of a hierarchical controller using visual inputs, a connectome-constrained visual system network, processing of population activity, object detection, descending control of a hybrid controller with leg adhesion and ascending feedback for head stabilization. **h**, Performance of this more bio-realistic visual system network on fly following with or without ascending feedback-based head stabilization. Shown are 11 trials each.

algorithm[51] to control locomotion. Using this algorithm, to reach the odor source, the fly randomly switches between forward walking, pausing and turning (Fig. 6b). These actions are governed by Poisson processes, with Poisson rates and turning direction bias modulated based on odor encounters to favor navigation toward the target. In our simulation, the fly successfully reached within 15 mm of the odor source (Fig. 6c and Supplementary Video 13) with a low (9 of 100 trials), albeit similar, success rate comparable to what was seen in real flies in a larger arena over a longer time period[51].

Ultimately, to gain insights into how the real fly brain works, one would explore controllers with artificial neurons that can be mapped to real neurons or neuronal cell types. This may be achieved by building artificial neural networks with architectures constrained by the connectivity of the brain[24] and the VNC[25,26]. To illustrate how such models might be embodied and studied in the context of autonomous behavior, we designed a 'fly following' task in which a fly must use a realistic visual system model (Fig. 6d) to follow another fly—akin to chasing behaviors during courtship. We used a recently constructed connectome-constrained model[31] for this task. We interfaced NeuroMechFly with this visual system model to emulate layered visual processing in the fly brain. Concretely, we passed the visual experience of the 'following' fly as inputs to this pretrained connectome-constrained model and used the activities of T1–T5, all Tm neurons and all TmY neurons (indicated as putative output cells types[31]) to perform object detection (Fig. 6f). Then, based on the position of the detected object, we modulated the descending turning signal (Fig. 6g) to drive the hybrid controller (Fig. 2g), which controls walking. We asked to what extent ascending feedback-driven head stabilization (Fig. 4d) is necessary to enable reliable fly following. We found that, although over flat terrain the 'following' fly could successfully track the 'leading' fly without head stabilization (Extended Data Fig. 7), stabilization was crucial for tracking over rugged terrain (Fig. 6h and Supplementary Video 14). We obtained similar results even when using a smaller subset of neurons that provide inputs to LC9 and LC10–LC neurons implicated in courtship in particular[52] (Extended Data Fig. 7b). These results highlight how realistic neural networks can be coupled with embodied models to close the sensorimotor control loop.

Discussion

Here we have introduced NeuroMechFly v2, a framework for performing integrated sensorimotor neuromechanical simulations of the adult fly, *D. melanogaster*. In Supplementary Note 1, we provide a summary of general features currently supported in NeuroMechFly v2, specific modeling choices concretely demonstrated in this paper and opportunities for future work. Because our simulation framework is modular, researchers can build integrated models in an interoperable manner by choosing the appropriate level of detail for each part of the model to suit the scientific question under consideration. For example, one can use more abstract baselines or existing models for control elements outside the focus of investigation. Although important behaviors like those involving the control of wings/halteres (for example, flight[14]), abdomen (for example, egg laying) and proboscis (for example, feeding) are not yet implemented, they are supported within this framework. With enriched sensory feedback and improved biomechanics, NeuroMechFly v2 enables the whole-body simulation of complex behaviors requiring controllers that span sensing, navigation, internal states[22], learning[45] and motor control.

In the future, our simulation framework is likely to be further improved in a number of ways. First, we anticipate that recent developments in physics simulation, particularly GPU acceleration and differentiable simulation will facilitate the training of larger models through reinforcement learning. Second, careful measurements and analyses of the *Drosophila* musculoskeletal system (that is, tendons and muscles) could improve the interface between neural network controllers and the biomechanical embodiment. Third, as additional connectome-constrained neural circuit models become available, they can be added to the corpus of controllers in our modular simulation framework. FlyGym's compatibility with the Gymnasium API will ensure that changes are implemented relatively easily without disrupting the established user interface. In the more distant future—following substantial improvements in modeling infrastructure enabling high-throughput, low-latency simulations—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, enabling the real-time inference of dynamic variables such as contacts and informing experimental perturbations in closed loop[53]. These efforts will bring the field closer to achieving the ultimate goal of uncovering neuromechanical mechanisms giving rise to adaptive animal behaviors in a sensory-rich and physically complex world.

Online content

Any methods, additional references, Nature Portfolio reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at https://doi.org/10.1038/s41592-024-02497-y.

References

1. Cruse, H., Kindermann, T., Schumm, M., Dean, J. & Schmitz, J. Walknet—a biologically inspired network to control six-legged walking. *Neural Netw.* **11**, 1435–1447 (1998).
2. Schumacher, P. et al. Natural and robust walking using reinforcement learning without demonstrations in high-dimensional musculoskeletal models. Preprint at https://arxiv.org/abs/2309.02976 (2023).
3. Thandiackal, R. et al. Emergence of robust self-organized undulatory swimming based on local hydrodynamic force sensing. *Sci. Robot.* **6**, eabf6354 (2021).
4. Ijspeert, A. J., Crespi, A., Ryczko, D. & Cabelguen, J. From swimming to walking with a salamander robot driven by a spinal cord model. *Science* **315**, 1416–1420 (2007).
5. Towers, M. et al. Gymnasium. *Zenodo* https://doi.org/10.5281/zenodo.11232524 (2024).
6. Levine, S., Finn, C., Darrell, T. & Abbeel, P. End-to-end training of deep visuomotor policies. *J. Mach. Learning Res.* **17**, 1–40 (2016).
7. Shi, H., Lin, Z., Hwang, K., Yang, S. & Chen, J. An adaptive strategy selection method with reinforcement learning for robotic soccer games. *IEEE Access* **6**, 8376–8386 (2018).
8. Ho, J. & Ermon, S. Generative adversarial imitation learning. In *Proc. 30th International Conference on Neural Information Processing Systems*. 4572–4580 (Curran Associates, 2016).
9. Nagabandi, A., Kahn, G., Fearing, R. S. & Levine, S. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation* 7559–7566 (IEEE Press, 2018).
10. Merel, J. et al. Deep neuroethology of a virtual rodent. In *Proc. International Conference on Learning Representations* (2020); https://openreview.net/forum?id=SyxrxR4KPS
11. Choi, S. et al. Learning quadrupedal locomotion on deformable terrain. *Sci. Robot.* **8**, eade2256 (2023).
12. Caggiano, V., Wang, H., Durandau, G., Sartori, M. & Kumar, V. MyoSuite: a contact-rich simulation suite for musculoskeletal motor control. *Proc. Mach. Learn. Res.* **168**, 492–507 (2022).
13. Lobato-Rios, V. et al. NeuroMechFly, a neuromechanical model of adult *Drosophila melanogaster*. *Nat. Methods* **19**, 620–627 (2022).
14. Vaxenburg, R. et al. Whole-body simulation of realistic fruit fly locomotion with deep reinforcement learning. Preprint at *bioRxiv* https://doi.org/10.1101/2024.03.11.584515 (2024).
15. Aldarondo, D. et al. A virtual rodent predicts the structure of neural activity across behaviors. *Nature* **632**, 594–602 (2024).
16. Merel, J., Botvinick, M. & Wayne, G. Hierarchical motor control in mammals and machines. *Nat. Commun.* **10**, 5489 (2019).
17. Raji, J. I. & Potter, C. J. The number of neurons in *Drosophila* and mosquito brains. *PLoS ONE* **16**, e0250381 (2021).
18. Azevedo, A. et al. Connectomic reconstruction of a female *Drosophila* ventral nerve cord. *Nature* **631**, 360–368 (2024).
19. Pick, S. & Strauss, R. Goal-driven behavioral adaptations in gap-climbing *Drosophila*. *Curr. Biol.* **15**, 1473–1478 (2005).
20. 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).
21. Pavlou, H. J. & Goodwin, S. F. Courtship behavior in *Drosophila melanogaster*: towards a 'courtship connectome'. *Curr. Opin. Neurobiol.* **23**, 76–83 (2013).
22. Hoopfer, E. D. Neural control of aggression in *Drosophila*. *Curr. Opin. Neurobiol.* **38**, 109–118 (2016).
23. Wolf, R. et al. *Drosophila* mushroom bodies are dispensable for visual, tactile, and motor learning. *Learn. Mem.* **5**, 166–178 (1998).
24. Dorkenwald, S. et al. Neuronal wiring diagram of an adult brain. *Nature* **634**, 124–138 (2024).
25. Phelps, J. S. et al. Reconstruction of motor control circuits in adult *Drosophila* using automated transmission electron microscopy. *Cell* **184**, 759–774 (2021).
26. Takemura, S. et al. A connectome of the male *Drosophila* ventral nerve cord. *eLife* **13**, RP97769 (2024).

27. Jenett, A. et al. A GAL4-driver line resource for *Drosophila* neurobiology. *Cell Rep.* **2**, 991–1001 (2012).
28. Klapoetke, N. C. et al. Independent optical excitation of distinct neural populations. *Nat. Methods* **11**, 338–346 (2014).
29. Mohammad, F. et al. Optogenetic inhibition of behavior with anion channelrhodopsins. *Nat. Methods* **14**, 271–274 (2017).
30. Chen, T. et al. Ultrasensitive fluorescent proteins for imaging neuronal activity. *Nature* **499**, 295–300 (2013).
31. Lappalainen, J. K. et al. Connectome-constrained networks predict neural activity across the fly visual system. *Nature* https://doi.org/10.1038/s41586-024-07939-3 (2024).
32. 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. Preprint at bioRxiv https://doi.org/10.1101/2023.05.02.539144 (2023).
33. 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*. https://doi.org/10.1109/IROS.2012.6386109 (IEEE, 2012).
34. 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*. https://doi.org/10.1109/ICRA.2015.7139807 (IEEE, 2015).
35. Drechsler, P. & Federle, W. Biomechanics of smooth adhesive pads in insects: Influence of tarsal secretion on attachment performance. *J. Comp. Physiol. A Neuroethol. Sens. Neural Behav. Physiol.* **192**, 1213–1222 (2006).
36. Bullock, J. M. R., Drechsler, P. & Federle, W. Comparison of smooth and hairy attachment pads in insects: Friction, adhesion and mechanisms for direction-dependence. *J. Exp. Biol.* **211**, 3333–3343 (2008).
37. Gorb, S. N. et al. Structural design and biomechanics of friction-based releasable attachment devices in insects. *Integr. Comp. Biol.* **42**, 1127–1139 (2002).
38. Ramdya, P. et al. Climbing favours the tripod gait over alternative faster insect gaits. *Nat. Commun.* **8**, 14494 (2017).
39. Szczecinski, N. S., Bockemühl, T., Chockley, A. S. & Büschges, A. Static stability predicts the continuum of interleg coordination patterns in *Drosophila*. *J. Exp. Biol.* **221**, jeb189142 (2018).
40. Mantziaris, C., Bockemühl, T. & Büschges, A. Central pattern generating networks in insect locomotion. *Dev. Neurobiol.* **80**, 16–30 (2020).
41. Bellegarda, G. & Ijspeert, A. CPG-RL: learning central pattern generators for quadruped locomotion. *IEEE Robot. Autom. Lett.* **7**, 12547–12554 (2022).
42. Schneider, A., Paskarbeit, J., Schaeffersmann, M. & Schmitz, J. HECTOR, a new hexapod robot platform with increased mobility—control approach, design and communication. In *Advances in Autonomous Mini Robots* (eds. Rückert, U. et al.) 249–264 (Springer, 2012).
43. Cagan, R. Principles of Drosophila eye differentiation. *Curr. Top. Dev. Biol.* **85**, 115–135 (2009).
44. Wilson, R. I. Early olfactory processing in *Drosophila*: mechanisms and principles. *Annu. Rev. Neurosci.* **36**, 217–241 (2013).
45. Cognigni, P., Felsenberg, J. & Waddell, S. Do the right thing: neural network mechanisms of memory formation, expression and update in *Drosophila*. *Curr. Opinion Neurobiol.* **49**, 51–58 (2018).
46. Taisz, I. et al. Generating parallel representations of position and identity in the olfactory system. *Cell* **186**, 2556–2573 (2023).
47. Chen, C. et al. Ascending neurons convey behavioral state to integrative sensory and action selection brain regions. *Nat. Neurosci.* **26**, 682–695 (2023).
48. Kim, I. S. & Dickinson, M. H. Idiothetic path integration in the fruit fly *Drosophila melanogaster*. *Curr. Biol.* **27**, 2227–2238 (2017).
49. Gollin, A. & Dürr, V. Estimating body pitch from distributed proprioception in a hexapod. In *Biomimetic and Biohybrid Systems* (eds Vouloutsi, V. et al.) 187–199 (Springer, 2018).
50. Kress, D. & Egelhaaf, M. Head and body stabilization in blowflies walking on differently structured substrates. *J. Exp. Biol.* **215**, 1523–1532 (2012).
51. Demir, M., Kadakia, N., Anderson, H. D., Clark, D. A. & Emonet, T. Walking *Drosophila* navigate complex plumes using stochastic decisions biased by the timing of odor encounters. *Elife* **9**, e57524 (2020).
52. Cowley, B. R. et al. Mapping model units to visual neurons reveals population code for social behaviour. *Nature* **629**, 1100–1108 (2024).
53. Ramdya, P. & Ijspeert, A. J. The neuromechanics of animal locomotion: from biology to robotics and back. *Sci. Robot.* **8**, eadg0279 (2023).

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Methods

The FlyGym package

FlyGym is implemented based on MuJoCo[33] and the dm_control[54] library and complies with the Gymnasium API[5] for the Markov decision process. The user interacts with the simulation through 'actions' and 'observations' (Fig. 1 and Supplementary Note 2). The definition of the action and observations in the default control task can be found in Supplementary Note 3. More detailed, low-level information can be accessed directly using dm_control and MuJoCo.

We configured the meshes at 1,000× scale in MuJoCo to obtain observation measurements in mm and mN. The user can implement preprogrammed premotor computations and sensory preprocessing by extending the base `Fly` or `Simulation` classes (Fig. 1c). This will modify the action and observation spaces accordingly.

If the user wishes to use a simplified, ball-and-stick model to speed up computation, one can use a MuJoCo feature that approximates body geometries as 'capsules' (that is, cylinders with a hemisphere at each end). To do so, the user should replace `type = "mesh"` with `type = "capsule"` in the `<geom>` tag of the model MJCF file.

Updated rigging of the biomechanical model

In *Drosophila*, the antenna consists of three main segments—the scape, pedicel and funiculus—in addition to the arista[55]. The fly has four muscles that can actively control the joint between the scape and pedicel[56]. By contrast, the funiculus and the arista move or deform passively in response to external forces (for example, wind, limb contact during grooming). In the original NeuroMechFly model, 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 using 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[57]. 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 (that is, head–pedicel and pedicel–funiculus) all three rotational DoFs because the real number of DoFs in these antennal joints remains unknown. Future users can modify each DoF (for example, fix/unfix or stiffen) in the model file to emulate the measured dynamics of the antennae.

The position of the neck joint affects the translation of 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 the 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[58]. 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 (for example, humeral bristles) and an overlay of camera images of real animals with images of the model. 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 the FARMS simulation framework[59] to generate the MJCF file of the updated model.

Leg adhesion and critical climbing angle

Leg adhesion was added using built-in MuJoCo actuators. Adhesion takes the form of an artificial force injected perpendicular to the contact surface at 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 ('Stepping pattern'). We controlled adhesion in a binary fashion but it is possible to use a gradient of adhesion forces by modulating the input to the adhesion actuator at every time step.

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 either the absolute roll or pitch angle of the fly is 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 1 s.

Stepping pattern

We derived the kinematics for each individual step from manually annotated video recordings of a real fly during untethered walking. We recorded bouts of straight walking at 360 Hz in a linear chamber (12-mm long × 4-mm wide × 2-mm tall) with prisms as walls. The video was downsampled to 120 Hz. Then, five leg key points (thorax–coxa, coxa–trochanter, femur–tibia, tibia–tarsus joints and claw), both antennae, neck, thorax and abdomen key points were manually annotated from a 0.3-s episode of straight walking. The recording was performed on a wild-type (PR) female adult *D. melanogaster* raised at 25 °C and 50% humidity on a 12-h light–dark cycle. The fly was recorded 4–5 days after eclosion.

We determined the 3D position of each key point assuming that the prisms are oriented at 90°. We aligned resulting 3D poses to the template of NeuroMechFly's skeleton by scaling the length of the full leg. Finally, we applied inverse kinematics to each kinematic chain to obtain joint angles[60]. From the recording, we then segmented eight swing-to-swing steps and eight stance-to-stance steps. Of these 16 unique steps (five front leg steps, six middle leg steps and five hind leg steps), 7 were insufficiently closed (that is, mean distance between the first and last joint angle in the step greater than 0.17, 0.12 and 0.17 radians for the front, middle and hind legs, respectively) and were, therefore, discarded. The final stepping pattern is composed of three steps that are of the same lengths, are closed, and, when mirrored, yield symmetric and smooth steps.

We obtained the final stepping pattern by (i) segmenting each of the nine selected steps, (ii) stretching or compressing them to the median step length of 0.135 s, (iii) linearly interpolating the difference between the first and last time points of the step through the last 10% of each step to guarantee perfect closure, (iv) modifying all steps so that phase 0 corresponded to the initiation of the swing, and (v) finally generating a complementary dataset with mirrored joint angles so that each step could be replayed in either right or left legs irrespective of their leg of origin. From those nine steps, we obtained 30 combined stepping patterns. We used the combination that maximized the displacement along the fly's initial heading direction and minimized its lateral displacement.

CPG-based controller

CPGs are neural circuits that generate rhythmic outputs without receiving rhythmic input[40]. 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[4]. 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 as the phase and amplitude of the step cycle. The gait pattern emerges from the phase biases of the different CPGs. We used an idealized tripod gait for walking in our CPG model. The precise definition of the CPG network can be found in Supplementary Note 4, and the parameters are detailed in Supplementary Tables 2 and 3.

Rule-based controller

We used a rule-based controller to illustrate a decentralized control architecture. This controller was inspired by the first three rules described in Walknet[1,61,62]. 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 neighbors of a leg entering stance phase. The third rule enforces temporal coherence by eliciting a swing in the caudal and contralateral neighbors 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 (difference < 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 (Supplementary Table 4): rule 1 is weighted most heavily as it is crucial to maintain stability. Rules 2 and 3 are given different weights for ipsilateral and contralateral connections. To maintain synchrony, we ensured that the duration of the swing and stance periods were identical across all legs. To more fairly compare the rule-based controller with the CPG controller, we scaled the duration of steps to match the stepping frequency of the CPG controller.

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 (for example, in gapped terrain) or hits an obstacle (for example, in blocks terrain) by adjusting the leg in question. The first rule ('overstretch 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 along 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 detection of leg overstretch. If multiple legs meet this criterion, only the leg that extends the furthest 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 higher than normal during swing or extends slightly more during stance. The step phase dependence of the adjustment is obtained by using a gain described by a piece-wise linear function reaching a maximum of 0.8 at the swing midpoint, a minimum of −0.1 at the stance midpoint and remaining at 0 from the beginning of the swing to the end of the swing plus one-eighth of a cycle. Supplementary Table 5 provides a summary of the joints involved in leg retraction and their rates of change. Both rules are persistent to ensure proper release of the leg: if one rule was active during the past 0.002 s, the leg enters a persistence period prolonging the adjustment for 0.002 s. Once the persistence period is over and as long as the rules are no longer active, joint angles progressively reset. To avoid overcorrection, the leg's position is adjusted for 0.008 s before the increment is capped. The swing duration was extended by one-eighth of a cycle to delay the initiation of adhesion and give more time for the leg to clear any obstacles.

Benchmarking of locomotor controllers over rugged terrains

We benchmarked locomotor controllers by running 20 simulations, starting from different spawn positions and initial states, for 1.5 s each and computing the average velocity in the horizontal plane. Walking speeds driven by the controllers are comparable because the same preprogrammed step is used for all controllers and walking speed is only influenced by inter-leg coordination. 'Gapped terrain' consists of horizontal 1-mm-wide blocks separated by 0.3-mm-wide, 2-mm-deep gaps. '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 blocks to avoid extremely thin surfaces near the corners that can lead to physics instabilities. 'Mixed terrain' consists of alternating flat, gapped and block patterns along the x axis. We used a joint position gain k_p of 45 and an adhesion force of 40 mN for all controllers.

Control of turning

Walking flies execute turns on a continuum of sharpness. The turning program is controlled by descending neurons[63]. For smoother turns (that is, 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[64]. Our controller receives a two-dimensional descending input that controls turning (Fig. 2g). On each side, the descending signal $\mathrm{DN} \in \mathbb{R}$ modifies the intrinsic frequency ν_i and maximum amplitude R_i of each oscillator i as follows:

$$R_i^{\star}(\mathrm{DN}) = |\mathrm{DN}|, \qquad \nu_i^{\star}(\mathrm{DN}) = \begin{cases} \nu_i & \text{if } \mathrm{DN} > 0, \\ -\nu_i & \text{otherwise}, \end{cases} \tag{1}$$

where $R_i^{\star}$ and $\nu_i^{\star}$ are the modified maximum amplitude and intrinsic frequency, respectively.

Implementation of vision

Flies have three major types of ommatidia—units arranged in a hexagonal pattern to make up the compound eye. These differentiate colors and polarization properties by using different combinations of photoreceptors. Yellow-type and pale-type ommatidia are stochastically arranged throughout the eye and enable two-dimensional chromatic sensitivity in the $\text{UV}_{\sim 300\,\text{nm}}$-to-yellow range[65]. The yellow-type and pale-type ommatidia are found at a 7:3 ratio[66]. A third type is found in the eye's dorsal rim area facing the sky and is specialized for polarization detection during navigation[67]; this type of ommatidia is not implemented in our model. The field of view of each eye is defined based on prior studies[68,69]. In our implementation, yellow-type 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-channel and blue-channel display colors can be set as the inner products of the actual surface reflectance spectrum of the object and the spectral response curves of the appropriate photoreceptors[65]. We corrected the input images to superimpose a 'fish-eye' effect that makes the representation of angles consistent throughout the field of view (Supplementary Note 5 and Extended Data Fig. 3c,d).

Visual object tracking task

In our visual object tracking task, the fly follows a black sphere moving in an S-shaped trajectory at 10 mm s^{-1}. To achieve this, we first used a thresholding rule to detect the object (normalized light intensity $\bar{\rho}$ below 0.2). Then, we computed the position and size of the object (both normalized) as 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, as shown in equation (2),

$$\mathrm{DN}_i = \min\left(\max(ka_i + b, \mathrm{DN}_{\min}), \mathrm{DN}_{\max}\right), \tag{2}$$

where DN_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]; $k = -3$, $b = 1$ describe the response curve; $\mathrm{DN}_{\min} = 0.4, \mathrm{DN}_{\max} = 1.2$ are the minimal and maximal allowed values for the descending signal, respectively.

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 locations of the antennae and maxillary palps based on a diffusion function, $I(d)$, where d is the distance from the odor source and $I(d)$ gives the odor intensity. The odor diffusion relationship can be defined by the user. In this example, we used the inverse square relationship $I(d) = I_{\mathrm{peak}}/d^2$ 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 antennae and the maxillary palps weighted by 9:1 (roughly comparable to the ratio of ORNs in the antennae and maxillary palps[70]). By contrast, to demonstrate the possibility of using different sensors for different odors, we used only the intensity sensed by the antennae 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 in simulated time. More precisely, according to equations (3)–(6),

$$s = \sum_o \gamma_o \frac{I_{\mathrm{left},o} - I_{\mathrm{right},o}}{(I_{\mathrm{left},o} + I_{\mathrm{right},o})/2}, \tag{3}$$

$$b = \tanh(s^2), \tag{4}$$

$$\mathrm{DN}_{\mathrm{inner}} = b(\mathrm{DN}_{\max} - \mathrm{DN}_{\min}), \tag{5}$$

$$\mathrm{DN}_{\mathrm{outer}} = \mathrm{DN}_{\max}, \tag{6}$$

where s is the weighted sum of bilateral differences in odor intensities, b is the nonlinearly transformed turning bias, $I_{\mathrm{side},o}$ is the mean intensity of odor o sensed by the antenna and the maxillary palp on the specified side; $\mathrm{DN}_{\mathrm{inner}}$, $\mathrm{DN}_{\mathrm{outer}}$ are the DN drives on the inner and outer sides, respectively (when $s > 0$, the fly performs a right turn, vice versa); $\mathrm{DN}_{\min} = 0.2, \mathrm{DN}_{\max} = 1$ define the range of the DN drives; and γ_o is the gain of odor o ($\gamma_{\mathrm{attractive}} = -500$ and $\gamma_{\mathrm{aversive}} = 80$).

Path integration using ascending feedback

To test the degree to which path integration can be performed using ascending feedback, we constructed a scenario in which the fly model performs random exploration of a featureless environment and tries to estimate its position. To collect training data, we make the fly alternate between forward walking and in-place turning. Turning is modeled as a Poisson process with a rate $\lambda_{\mathrm{turn}} = 2\ \mathrm{s}^{-1}$. We deliberately chose a relatively high λ_{turn}, compared to the range of typical fly behavior, to make path integration more difficult. We simulated walking using three walking gaits: tripod gait (three legs in stance at a time), tetrapod gait (four legs in stance at a time) and wave gait (five legs in stance at a time). When the fly executes a turn, we apply a fixed asymmetrical descending drive of $[\mathrm{DN}_{\mathrm{inner}}, \mathrm{DN}_{\mathrm{outer}}]$. For the tripod and tetrapod gaits, $[\mathrm{DN}_{\mathrm{inner}}, \mathrm{DN}_{\mathrm{outer}}] = [-0.2, 1.0]$; for the wave gait, $[\mathrm{DN}_{\mathrm{inner}}, \mathrm{DN}_{\mathrm{outer}}] = [0.4, 1.0]$. These choices led to qualitatively similar turning. The direction of the turn is chosen at random. The duration of the turn (and, therefore, the angle turned) is sampled from a normal distribution centered at 0.4 s with a standard deviation of 0.1 s. The fly receives no visual information—akin to navigating in the dark. We ran $N = 15$ trials with different random seeds for each of the three gaits. Each trial was 20-s long. For simplicity, the correction rules in the hybrid controller were disabled for this experiment. Then, we developed a path integration algorithm that separately predicts the changes in heading and forward displacement using the difference and sum of the cumulative stride lengths on the left and right sides. These signals are then integrated over time to estimate the fly's position. Parameters in this algorithm are fitted to the aforementioned training data. A detailed description of the algorithm can be found in Supplementary Note 6.

Head stabilization using ascending feedback

We first simulated walking over the 'blocks' terrain and recorded movements of the thorax. We recorded joint angles and ground contacts throughout the simulation and calculated the optimal neck roll and pitch angles that would 'cancel out' thoracic rotations. We used these angles as ground truth. Using the joint angles and ground contacts as inputs, we trained an artificial neural network (MLP) to predict the optimal correction angles. These predicted correction angles are then used to actuate the neck joint using a proportional derivative controller. Details of this process can be found in Supplementary Note 7.

Multimodal navigation task solved using reinforcement learning

In the multimodal navigation task, the fly locomotes over rugged terrain to seek an attractive odor source while avoiding a visual obstacle in its path. 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 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. To reduce training time, we slightly simplified the 'mixed terrain' by reducing the gap width to 0.2 mm and the block height to 0.3 mm.

We started by training a convolutional neural network to extract features from the raw visual input, namely the direction of the object relative to the fly, the distance of the object from the fly, whether the object is within the fly's field of view, the azimuth of the object seen from each eye, and the size of the object on the retina. Details of this vision preprocessing model can be found in Supplementary Note 8. Then, we trained an MLP to perform the multimodal navigation task using Soft Actor-Critic (SAC), a reinforcement learning algorithm[71]. Details of the reinforcement learning task are outlined in Supplementary Note 9.

Tracking complex odor plumes

We simulated the complex odor plume using PhiFlow[72] based on existing open-source software[73]. Once simulated, we provided the plume concentration s at the appropriate location to the odor sensors of the simulated fly. The plume is also overlaid onto the rendered image when applicable. Then, we implemented a tracking algorithm similar to the one in ref. 51 wherein the fly makes decisions based on discrete, binarized plume encounters. Briefly, the fly switches between walking and stopping governed by Poisson processes whose rates depend on the time since the last odor encounter and the accumulation of odor encounters, respectively. The fly also turns based on a Poisson process where the direction of turning depends on the encounter frequency.

Details about the odor plume simulation and the odor-tracking algorithm can be found in Supplementary Note 10.

Closed-loop fly following using a connectome-constrained visual system model

To simulate the responses of visual system neurons to the visual experience of the simulated fly, we interfaced NeuroMechFly with a published connectome-constrained visual system model[31] implemented in the FlyVision package (https://github.com/TuragaLab/flyvis/). We modified its implementation to handle one frame at a time rather than the whole dataset at once, enabling closed-loop deployment. Since both FlyGym and FlyVision use a hexagonally arranged ommatidia grid with a side length of 16 ommatidia, there is a one-to-one mapping for ommatidia between the two packages. We sampled the visual experience of the simulated fly at 500 Hz and simulated the FlyVision model at the same rate. The two eyes are simulated independently. We used the best-performing model reported in the visual system modeling study[31] rather than the whole ensemble of models.

From the visual system simulation, we read out the activities of the 34 T-shaped transmedullary neurons, which are considered the outputs of the optic lobe. We compared the activities of these neurons with their baseline activities, obtained by simulating the fly walking in an empty arena. We then detected the object by identifying ommatidia for which neural activities differed substantially from baseline. More precisely, for each cell in the hexagonal ommatidia grid, we computed an object score χ, defined in equation (7),

$$\chi = \frac{1}{|\mathcal{C}|}\sum_{\text{cell}\in\mathcal{C}}\left|\frac{a_{\text{cell}} - a_{\text{cell,mean}}}{a_{\text{cell,s.d.}}}\right|, \tag{7}$$

where $\mathcal{C}$ is the set of neurons used, a_{cell} is the activity of a cell, and $a_{\text{cell,mean}}$, $a_{\text{cell,s.d.}}$ are the mean and standard deviation of the activity of the same cell in the baseline simulation. Then, we selected ommatidia where χ is greater than a threshold $\chi_{\text{thr}} = 7$ as the object. Once the object mask was detected, we calculated a descending signal using the method described in 'Visual object tracking task' with the range of DN drives $[\text{DN}_{\text{min}}, \text{DN}_{\text{max}}]$ set to [0.4, 1.2]. This descending signal was then passed to the hybrid walking controller to navigate either flat or 'blocks' (height reduced to 0.2 mm) terrain.

In the schematic diagram Fig. 6e, the placement of Tm and TmY neurons is based on ref. 74; the placement of T1 neurons is based on ref. 75; the placement of T2 and T3 neurons is based on ref. 76; the placement of T4 and T5 neurons is based on ref. 77.

Software

We used Python 3.12, NumPy 1.26.4, SciPy 1.13.0, OpenCV-Python 4.9.0.80, Numba 0.59.1 and Pandas 2.2.2 for general computing; Gymnasium 0.29.1, MuJoCo 3.1.4, dm_control 1.0.18 and PhiFlow 2.5.3 for physics simulation; PyTorch 2.2.2/2.3.0, PyTorch Lightning 2.2.2 and PyTorch Geometric 2.5.0 for neural networks; Stable Baselines 3 2.3 for reinforcement learning; Nvidia graphics driver 550.54.15 and Nvidia CUDA Toolkit 12.4 for GPU acceleration; FlyVision commit 056e4aa for connectome-constrained visual system simulation; SeqIKPy 1.0.0 for inverse kinematics; and Blender 2.81 for rigging the biomechanical model. Installation is managed automatically by package installers such as pip based on the `setup.py` file of our FlyGym package ('Code availability').

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

Data are available via The Harvard Dataverse Repository at https://doi.org/10.7910/DVN/3MCEYR (ref. 78). This repository includes (i) the experimentally recorded walking kinematics, (ii) trained parameters of the path integration models, (iii) trained parameters of the head stabilization models, (iv) trained parameters of the visual processing and reinforcement learning models in the multimodal navigation task, (v) training data for the visual processing model, and the graph representation of the ommatidia lattice used to perform graph convolution, (vi) the simulated complex plume dataset and (vii) baseline neuron activities in the connectome-constrained visual system model. Source data are provided with this paper.

Code availability

The FlyGym package is available at https://github.com/NeLy-EPFL/flygym/ under the Apache-2.0 license. The documentation for FlyGym, along with detailed tutorials for some experiments in this paper, is available at https://neuromechfly.org/.

The code used to generate some figures is not a part of the FlyGym package but is instead available at https://github.com/NeLy-EPFL/nmf2-paper under the same license.

A frozen snapshot of our code is available via Zenodo at https://doi.org/10.5281/zenodo.12973000 (ref. 79). However, FlyGym is under continued development and we recommend always using the latest version. Additionally, the results might not be bit-for-bit identical to the ones shown in this paper even with an exact copy of the code and its dependencies. This is due to differences in the computing hardware.

References

54. Tunyasuvunakool, S. et al. dm_control: software and tasks for continuous control. *Software Impacts* **6**, 100022 (2020).
55. Schneider, D. Insect antennae. *Annu. Rev. Entomol.* **9**, 103–122 (1964).
56. Suver, M. P., Medina, A. M. & Nagel, K. I. Active antennal movements in *Drosophila* can tune wind encoding. *Curr. Biol.* **33**, 780–789 (2023).
57. Todi, S. V., Sharma, Y. & Eberl, D. F. Anatomical and molecular design of the *Drosophila* antenna as a flagellar auditory organ. *Microsc. Res. Tech.* **63**, 388–399 (2004).
58. Günel, S. et al. DeepFly3D, a deep learning-based approach for 3D limb and appendage tracking in tethered, adult *Drosophila*. *eLife* **8**, e48571 (2019).
59. Arreguit, J., Ramalingasetty, S. T. & Ijspeert, A. J. FARMS: framework for animal and robot modeling and simulation. Preprint at *bioRxiv* https://doi.org/10.1101/2023.09.25.559130 (2023).
60. Ozdil, P. G., Ijspeert, A. & Ramdya, P. Sequential-inverse-kinematics. *Zenodo* https://doi.org/10.5281/zenodo.12601316 (2024).
61. Cruse, H. What mechanisms coordinate leg movement in walking arthropods? *Trends Neurosci.* **13**, 15–21 (1990).
62. Schilling, M., Hoinville, T., Schmitz, J. & Cruse, H. Walknet, a bio-inspired controller for hexapod walking. *Biol. Cybern.* **107**, 397–419 (2013).
63. Yang, H. H. et al. Fine-grained descending control of steering in walking *Drosophila*. *Cell* **187**, 1–19 (2023).
64. Strauss, R. & Heisenberg, M. Coordination of legs during straight walking and turning in *Drosophila melanogaster*. *J. Comp. Physiol. A* **167**, 403–412 (1990).
65. Sharkey, C. R., Blanco, J., Leibowitz, M. M., Pinto-Benito, D. & Wardill, T. J. The spectral sensitivity of *Drosophila* photoreceptors. *Sci. Rep.* **10**, 18242 (2020).
66. Rister, J., Desplan, C. & Vasiliauskas, D. Establishing and maintaining gene expression patterns: insights from sensory receptor patterning. *Development* **140**, 493–503 (2013).
67. Sancer, G. et al. Modality-specific circuits for skylight orientation in the fly visual system. *Curr. Biol.* **29**, 2812–2825 (2019).
68. Hindmarsh Sten, T., Li, R., Otopalik, A. & Ruta, V. Sexual arousal gates visual processing during *Drosophila* courtship. *Nature* **595**, 549–553 (2021).

69. Strother, J. A. et al. Behavioral state modulates the ON visual motion pathway of *Drosophila*. *Proc. Natl Acad. Sci. USA* **115**, E102–E111 (2018).
70. de Bruyne, M., Clyne, P. J. & Carlson, J. R. Odor coding in a model olfactory organ: the *Drosophila* maxillary palp. *J. Neurosci.* **19**, 4520–4532 (1999).
71. 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* (PMLR, 2018).
72. Holl, P., Thuerey, N. & Koltun, V. Learning to control PDEs with differentiable physics. In *International Conference on Learning Representations* (2020); https://openreview.net/forum?id=HyeSin4FPB
73. Koehler, F. Machine learning and simulation. *Zenodo* https://doi.org/10.5281/zenodo.12793324 (2024).
74. Raghu, S. V. & Borst, A. Candidate glutamatergic neurons in the visual system of *Drosophila*. *PLoS ONE* **6**, e19472 (2011).
75. Kolodziejczyk, A., Sun, X., Meinertzhagen, I. A. & Nässel, D. R. Glutamate, GABA and acetylcholine signaling components in the lamina of the *Drosophila* visual system. *PLoS ONE* **3**, e2110 (2008).
76. Keleş, M. F., Hardcastle, B. J., Städele, C., Xiao, Q. & Frye, M. A. Inhibitory interactions and columnar inputs to an object motion detector in *Drosophila*. *Cell Rep.* **30**, 2115–2124 (2020).
77. Shinomiya, K., Nern, A., Meinertzhagen, I. A., Plaza, S. M. & Reiser, M. B. Neuronal circuits integrating visual motion information in *Drosophila melanogaster*. *Curr. Biol.* **32**, 3529–3544 (2022).
78. Wang-Chen, S. et al. Data accompanying 'NeuroMechFly v2, simulating embodied sensorimotor control in adult *Drosophila*'. The Harvard Dataverse Repository. https://doi.org/10.7910/DVN/3MCEYR (2024).
79. Wang-Chen, S. et al. Code accompanying 'NeuroMechFly v2, simulating embodied sensorimotor control in adult *Drosophila*'. *Zenodo* https://doi.org/10.5281/zenodo.12973000 (2024).

Acknowledgements

We thank V. Lobato-Rios for valuable insights and early exploration of visual inputs to the model. We thank J. Arreguit (EPFL, Switzerland), S. T. Ramalingasetty (EPFL, Switzerland) and A. J. Ijspeert (EPFL, Switzerland) for the development of FARMS, which was used to generate the MJCF file of the updated model. We thank J. K. Lappalainen (Tübingen University, Germany; Tübingen AI Center, Germany; Janelia Research Campus, USA), J. H. Macke (Tübingen University, Germany; Tübingen AI Center, Germany; Max Planck Institute for Intelligent Systems, Germany), S. C. Turaga (Janelia Research Campus, USA) and colleagues for making the FlyVision model available before publication. P.R. acknowledges support from a Swiss National Science Foundation (SNSF) Project Grant (175667) and an SNSF Eccellenza Grant (181239). S.W.-C. acknowledges support from a Boehringer Ingelheim Fonds PhD fellowship. P.G.Ö. acknowledges support from a Swiss Government Excellence PhD Scholarship and a Google PhD Fellowship. F.H. 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 and editing and visualization. V.A.S.—conceptualization, methodology, software, formal analysis, investigation, data curation, validation, writing—original draft preparation, writing—review and editing and visualization. T.K.C.L.—conceptualization, methodology, software, formal analysis, investigation, data curation, validation, writing—review and editing and visualization. P.G.Ö.—conceptualization, methodology, software, investigation, data curation, validation, writing—original draft preparation, visualization and writing—review & editing. L.G.—methodology, software, validation, investigation and writing—review and editing. F.H.—conceptualization, methodology, software, investigation and writing—review and editing. P.R.—conceptualization, methodology, resources, writing—original draft preparation, writing—review and editing, supervision, project administration and funding acquisition.

Competing interests

The authors declare no competing interests.

Additional information

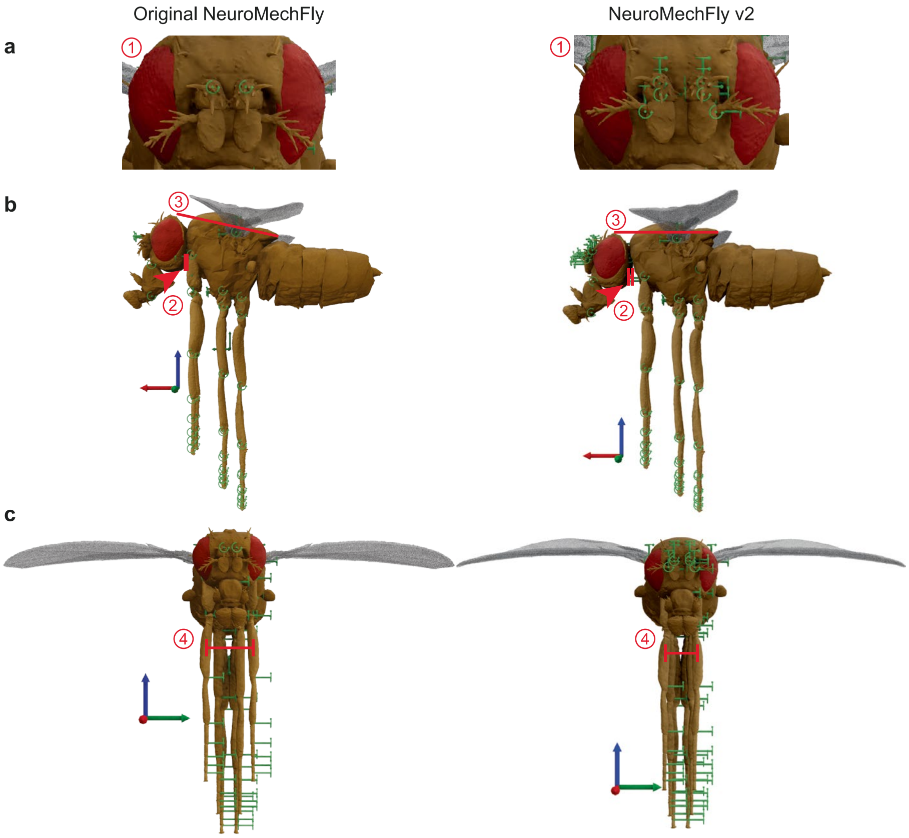
Extended data is available for this paper at https://doi.org/10.1038/s41592-024-02497-y.

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41592-024-02497-y.

Correspondence and requests for materials should be addressed to Sibo Wang-Chen or Pavan Ramdya.

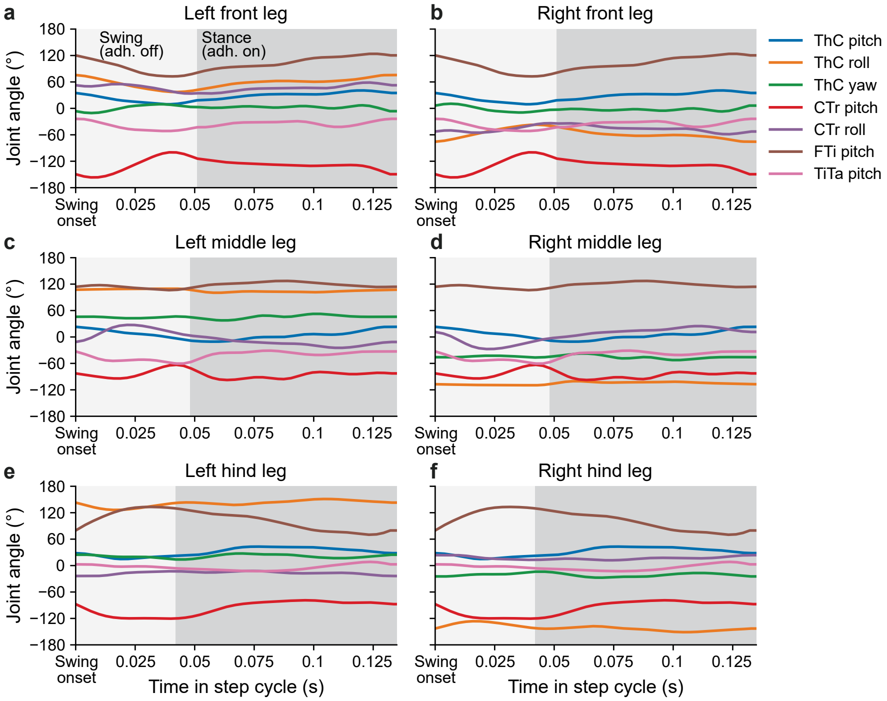
Peer review information *Nature Methods* thanks Stephane Viollet and the other, anonymous, reviewers for their contribution to the peer review of this work. Primary Handling Editor: Nina Vogt, in collaboration with the *Nature Methods* team. Peer reviewer reports are available.

Reprints and permissions information is available at www.nature.com/reprints.

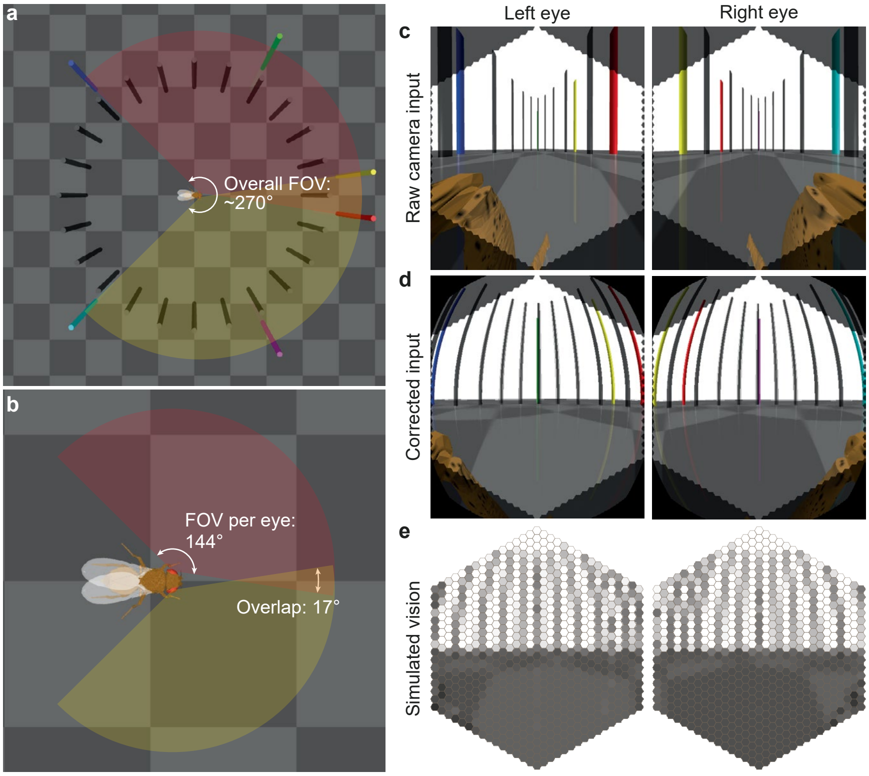


Extended Data Fig. 1 | Improvements to the biomechanical model. A comparison of the original (left) and updated (right) NeuroMechFly biomechanical model from a (**a**) zoomed-in view of the head, highlighting antennal DoFs, (**b**) the side views, and (**c**) the front views. DoFs are indicated in green. 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 placements of the legs on the thorax.

https://doi.org/10.1038/s41592-024-02497-y

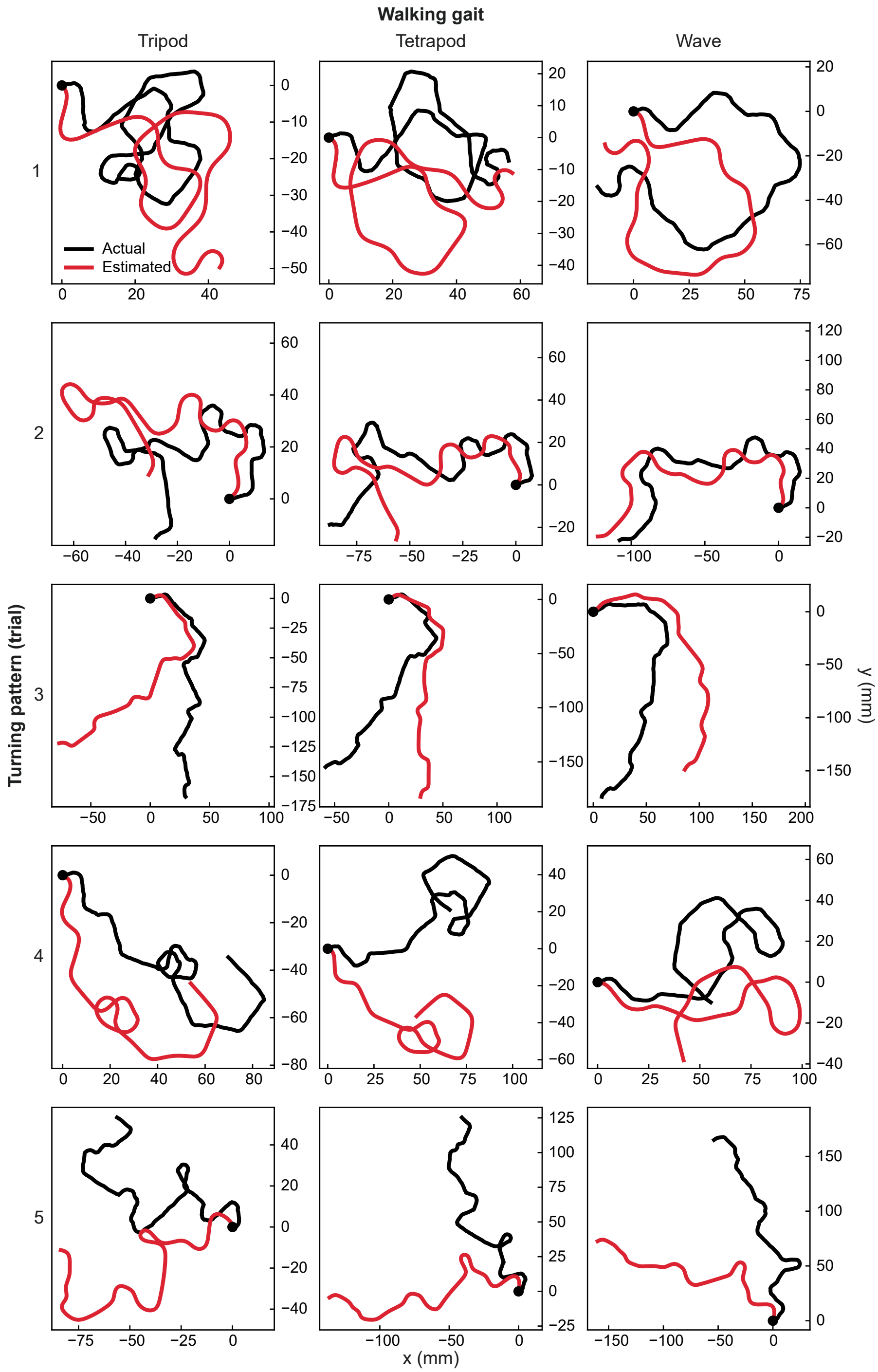


Extended Data Fig. 2 | Preprogrammed stepping based on experimentally recorded data. Joint kinematics for each leg during preprogrammed stepping. Kinematic patterns derived from behavioral recordings. Time series for each joint are color-coded. ThC: thorax-coxa joint; CTr: coxa-trochanter joint; FTi: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

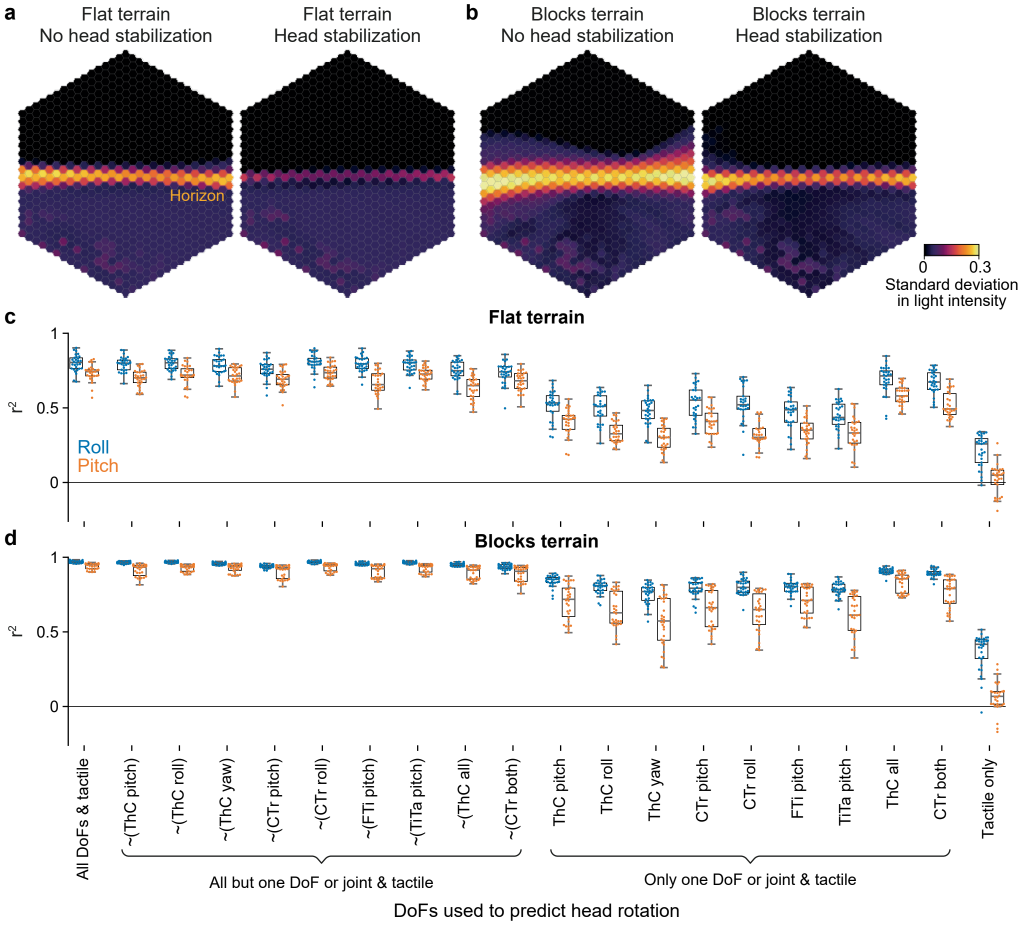


Extended Data Fig. 3 | Calibration of vision. (**a**) The calibration environment has black pillars spaced regularly around the fly at 15° intervals. Additionally, red, green, and blue pillars are used to indicate the anterior, midline, and posterior field of view (FOV) limits of the left eye. Yellow, magenta, and cyan pillars indicate the FOV limits of the right eye. (**b**) Each eye has a FOV spanning ~144° horizontally. The two eyes overlap by ~ 17°, resulting in an overall horizontal FOV of ~270°. (**c**) A raw camera view of what the fly sees in this environment before applying a fisheye effect. Note that by default, the rectilinear camera distorted areas closer to the edges of the FOV 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.

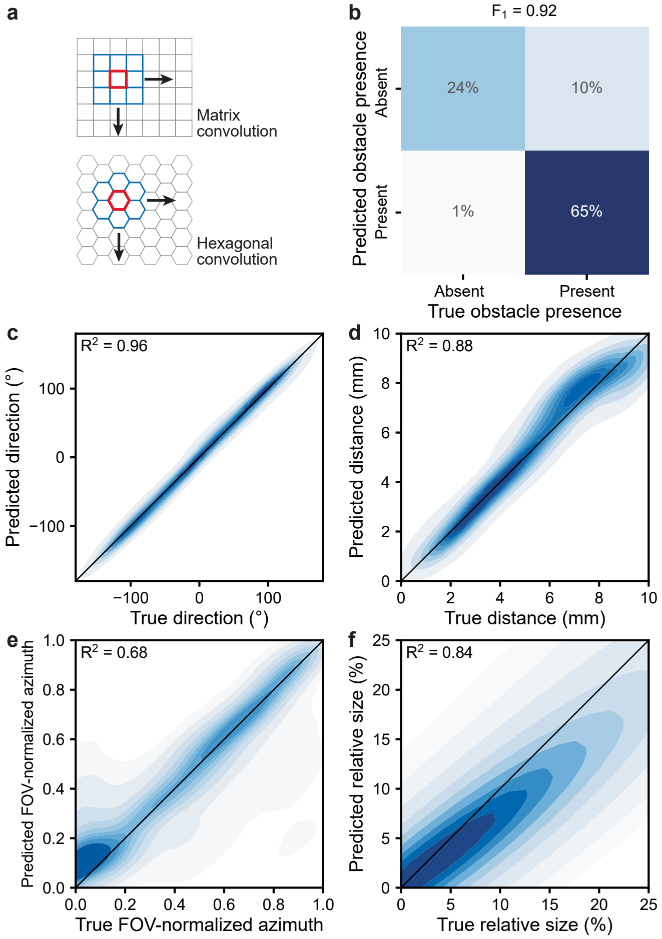
https://doi.org/10.1038/s41592-024-02497-y



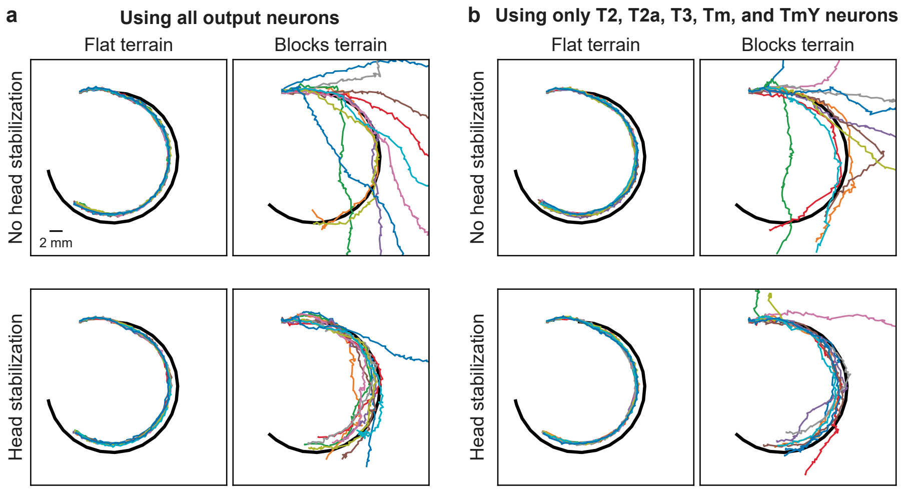
Extended Data Fig. 4 | Trajectories during path integration based on ascending feedback. Actual (black) and ascending feedback-based (red) estimates of walking trajectories for five trials (rows) and three different insect locomotor gaits (columns). Indicated are starting positions of the paths (black circles). For each trial (row), the fly executes the same sequence of straight walking and turns but with different gaits.



Extended Data Fig. 5 | Efficacy of head stabilization as a function of terrain type and ascending signals. (**a**-**b**) The standard deviations of ommatidia readings from the left eye while walking over (**a**) flat or (**b**) blocks terrain without or with ascending feedback-based head stabilization. Note the high variability in light intensity near the horizon when the head is not stabilized; this is due to more pronounced self motion of the head. (**c**-**d**) Coefficient of determination (r^2) between predicted and optimal head roll (blue) and pitch (brown) when performing head stabilization while walking over (**c**) flat or (**d**) blocks terrain and using ascending motor feedback from different sets of leg joint angles. '~ (removed DoF)' indicates all leg DoFs are used except the removed one (or ones at the same joint); 'used DoF' indicates only the indicated DoF (or multiple ones at the same joint) are used. In each case, the same set of DoFs is used in all legs. Note that ground contact information is always used, hence the better-than-chance performance in the cases where no leg DoF is used. 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 × the interquartile range (IQR) beyond the IQR. N = 30 for each box; trials where physics simulation failed due to numerical instabilities are excluded.



Extended Data Fig. 6 | Vision model used in the multi-modal navigation task. (**a**) Illustration of hexagonal convolution compared to standard matrix convolution. (**b**) Accuracy of the model in predicting whether the obstacle is present in the fields of view of the fly's eyes. The reported F1 score is the harmonic mean of the precision and recall. (**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 (ϑ) concatenated with cos (ϑ) for all samples, 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. N = 2,646.



Extended Data Fig. 7 | Performance of the connectome-constrained visual controller in a fly following task. Using (**a**) all T1–T5, Tm, and TmY neurons, or (**b**) T2, T2a, T3, Tm, and TmY neurons-upstream partners of LC9 and LC10 neurons[52]-to perform fly following either without or with head stabilization while walking over flat or blocks terrain. Shown are 11 trials (color-coded) per case.

nature portfolio

Corresponding author(s): Pavan Ramdya, Sibo Wang-Chen

Last updated by author(s): Jul 26, 2024

Reporting Summary

Nature Portfolio wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Portfolio policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed	
☐	☒	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☒	☐	A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☐	☒	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐	A description of all covariates tested
☒	☐	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☒	☐	A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☐	☒	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	Blender v2.81
Data analysis	We used the following software in this study: Python 3.12, NumPy 1.26.4, SciPy 1.13.0, OpenCV-Python 4.9.0.80, Numba 0.59.1, Pandas 2.2.2 for general computing; Gymnasium 0.29.1, MuJoCo 3.1.4, dm_control 1.0.18, PhiFlow 2.5.3 for physics simulation; PyTorch 2.2.2/2.3.0, PyTorch Lightning 2.2.2, PyTorch Geometric 2.5.0 for neural networks; Stable Baselines 3 2.3 for reinforcement learning; Nvidia graphics driver 550.54.15, Nvidia CUDA Toolkit 12.4 for GPU acceleration; FlyVision commit 056e4aa for connectome-constrained visual system simulation; SeqIKPy 1.0.0 for inverse kinematics; and Blender 2.81 for rigging the biomechanical model.

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Portfolio guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A description of any restrictions on data availability
- For clinical datasets or third party data, please ensure that the statement adheres to our policy

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.

Human research participants

Policy information about studies involving human research participants and Sex and Gender in Research.

Reporting on sex and gender	N/A
Population characteristics	N/A
Recruitment	N/A
Ethics oversight	N/A

Note that full information on the approval of the study protocol must also be provided in the manuscript.

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences □ Behavioural & social sciences □ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	We used a sample size of N=20 in the controller performance benchmark because it is a good balance between computational tractability and robustness in statistics. We were able to show robust differences between the controller types at this sample size. We used a sample size of N=9 in the multimodal navigation task to show 9 examples with initial positions on a 3x3 grid, thus covering a large portion of the arena.
Data exclusions	Trials where physics simulation failed due to numerical instabilities are excluded.
Replication	All results can be reproduced on the same computer architecture using the same versions of dependencies. Measures to enhance and measure reproducibility include: (1) setting the random seed whenever possible, (2) the code accompanying this paper saves intermediate steps and logs, (3) the code is version controlled using Git, and (2) continuous integration (CI) workflows are used on GitHub so that a set of tests are run every time there is a code commit, and the developers are notified if the tests fail. However, across different computer architectures, the results are not bit-for-bit reproducible even with an identical copy of our code and its dependencies. This is due to differences in computing hardware. Qualitative results are always reproducible in our attempts to test the software (on Linux, macOS, and Windows, locally and on cloud servers).
Randomization	Analyses are repeated using different random seeds and initial conditions as described in the paper and the supplementary notes.
Blinding	N/A. The study is computational and all analyses are conducted using automated code. Since the same code and procedure is automatically applied to all cases (including controls), there is no human manipulation required and therefore blinding is not necessary.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology and archaeology
☐	☒ Animals and other organisms
☒	☐ Clinical data
☒	☐ Dual use research of concern

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging

Animals and other research organisms

Policy information about studies involving animals; ARRIVE guidelines recommended for reporting animal research, and Sex and Gender in Research

Laboratory animals	Drosophila melanogaster, female, wildtype (PR), 4-5days post eclosion
Wild animals	Study does not involve wild animals.
Reporting on sex	A female fly was used to obtain the behavior kinematics data due to the larger body size. This facilitated behavior tracking.
Field-collected samples	Study does not involve animals collected from the field.
Ethics oversight	All experiments were performed in compliance with relevant national (Switzerland) and institutional (EPFL) ethical regulations.

Note that full information on the approval of the study protocol must also be provided in the manuscript.