NeuroMechFly, a neuromechanical model of adult *Drosophila melanogaster*

Victor Lobato Ríos*[1], Pembe Gizem Özdil†[1], Shravan Tata Ramalingasetty†[2], Jonathan Arreguit[2], Stéphanie Clerc Rosset[3], Graham Knott[3], Auke Jan Ijspeert[2], and Pavan Ramdya*[1]

[1]Neuroengineering Laboratory, Brain Mind Institute & Interfaculty Institute of Bioengineering, EPFL, Lausanne, Switzerland
[2]Biorobotics Laboratory, EPFL, Lausanne, Switzerland
[3]Biological Electron Microscopy Facility, EPFL, Lausanne, Switzerland

Abstract

Animal behavior emerges from a seamless interaction between musculoskeletal elements, neural network dynamics, and the environment. Accessing and understanding the interplay between these intertwined systems requires the development of integrative neuromechanical simulations. Until now, there has been no such simulation framework for the widely studied model organism, *Drosophila melanogaster*. Here we present NeuroMechFly, a data-driven computational model of an adult female fly that is designed to synthesize rapidly growing experimental datasets and to test theories of neuromechanical behavioral control. NeuroMechFly combines a set of modules including an exoskeleton with articulating body parts—limbs, halteres, wings, abdominal segments, head, proboscis, and antennae—muscle models, and neural networks within a physics-based simulation environment. Using this computational framework, we (i) predict the minimal limb degrees-of-freedom needed for real *Drosophila* behaviors, (ii) estimate expected contact reaction forces, torques, and tactile signals during replayed *Drosophila* walking and grooming, and (iii) discover neural network and muscle parameters that can drive tripod walking. Thus, NeuroMechFly is a powerful testbed for building an understanding of how behaviors emerge from interactions between complex neuromechanical systems and their physical surroundings.

1 Introduction

Uncoupling the relative contributions of large numbers of neuronal and biomechanical elements to behavior is experimentally daunting without the complementary guidance of systems-level mathematical models and numerical simulations. These "bottom-up" tools simultaneously serve to consolidate experimental data into a dynamic framework, generate predictions for future experiments, and allow us to test the extent to which theories can account for observed biological phenomena [1–6]. Modeling has long played a particularly important role in the study of how neural circuits control movement in vertebrates [7–9] and invertebrates, including stick insects [10–13], cockroaches [14, 15], praying mantises [16], and ants [17].

However, many of these models have not yet been able to achieve an ongoing dialogue with experimental data, perhaps owing to the limited tools available for studying these organisms. By contrast, for a few commonly studied model organisms, rapid iterative interactions between experiments and neuromechanical modeling represents an exciting but still unrealized opportunity: recent advances in computing power, physics-based simulation, and numerical optimization now make it possible to reverse engineer biomechanical systems with even limited experimental data (for an example, see [18]). Neuromechanical models of some commonly studied organisms have already been developed including several for *Caenorhabditis elegans* [19, 20], larval *Drosophila melanogaster* [21], and rodents [5, 22].

*corresponding authors: victor.lobatorios@epfl.ch; pavan.ramdya@epfl.ch
†equal contribution

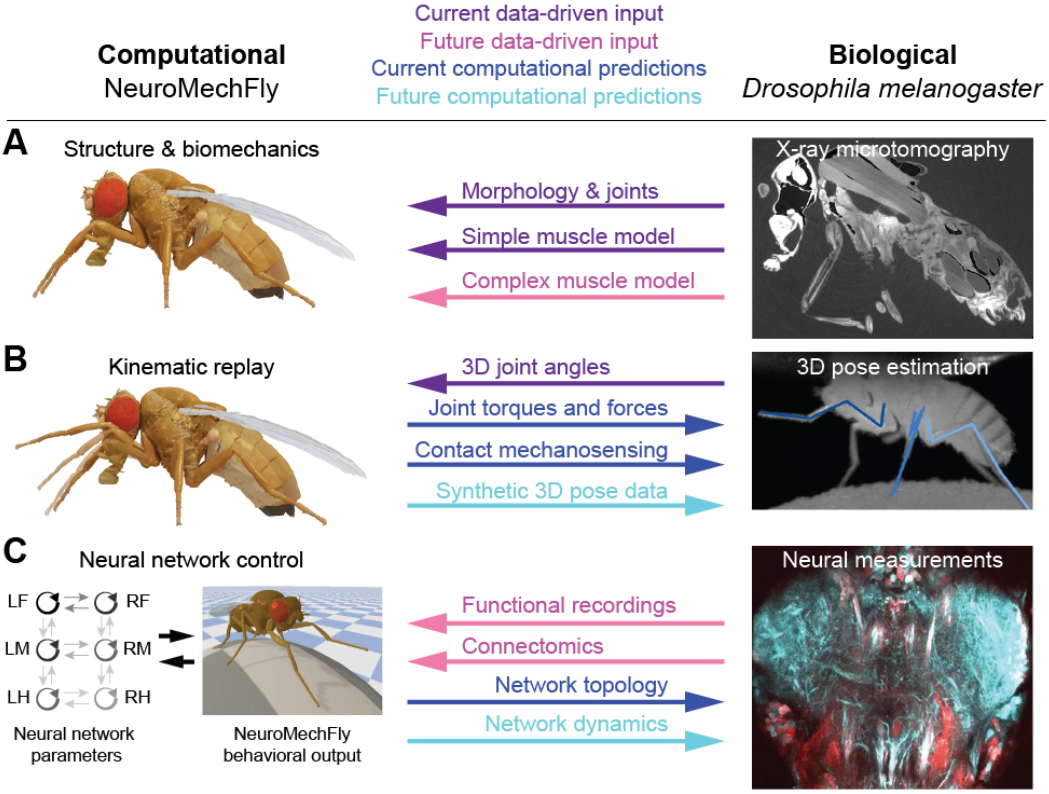

Figure 1: **Data-driven development and potential applications of NeuroMechFly. (A)** Body structures—morphology, joint locations, and degrees-of-freedom—are defined by x-ray microtomography. Future work may use more complex muscle models. **(B)** Real 3D poses can be replayed using the model to predict contact reaction forces and joint torques. Future work may use this actuated model to generate synthetic data for training deep networks to perform markerless 3D pose estimation. **(C)** Neural network and muscle parameters are discovered using multi-objective optimization to generate fast and stable locomotion. Optimized networks may be further studied to predict neural dynamics underlying behaviors in real flies. Future work may use neural recordings and connectomic datasets to constrain artificial networks.

Notably absent among these is a data-driven neuromechanical model for the adult fly, *Drosophila melanogaster*.

Adult flies are an ideal organism for building bridges between experimental and computational neuroscience. First, they generate a large repertoire of complex behaviors including flight [23], escape [24], courtship displays [25], walking [26,27], grooming [28], and navigation of complex environments [29]. These behaviors can now be measured with exquisite precision using deep learning-based computer vision tools [30,31] in 3-dimensions [32,33]. Second, flies have a relatively small number of neurons that can be genetically targeted [34], placed within their circuit context using connectomics [35,36], and reliably recorded in tethered, behaving animals [37–39]. We previously developed a simple physics-based simulation of adult *Drosophila melanogaster* to investigate hexapod locomotor gaits [40]. However, this model has several limitations that restrict its ability to serve as a standard model for building dialogues between experiments and simulations. Specifically, it lacks (i) the morphological accuracy needed to emulate mass distributions, compliance, and physical constraints, (ii) muscle models, and (iii) neural network controllers for exploring control architectures.

To overcome these gaps, we have developed NeuroMechFly, a neuromechanical simulation of adult *Drosophila* that is grounded in physics. NeuroMechFly is based on detailed measurements of an adult female fly that were then rendered digitally and rigged. This model's limb degrees-of-freedom (DoF) include a previously unreported coxa-femur DoF. We discovered that this additional DoF is required to accurately replay measured fly limb movements (Figure 1*A*) during walking and grooming, allowing us to estimate ground reaction forces (GRFs), joint torques, and tactile contacts (Figure 1*B*). Finally,

by optimizing the parameters of a coupled-oscillator network as well as a spring and damper model to represent central pattern generators (CPGs) and muscles, respectively, NeuroMechFly could generate biorealistic locomotor gaits. Importantly, this simulation is modular and open-source, enabling further extensions including more realistic neural and muscle models that enable predictions about the dynamics of real *Drosophila* neural networks (Figure 1*C*). Thus, NeuroMechFly represents an important step towards synthesizing increasing large amounts of experimental data to generate new predictions and will accelerate our understanding of how behavior emerges from an interplay between neural dynamics, biomechanics, and physical interactions with the environment.

2 Results

2.1 Constructing a data-driven biomechanical model of adult *Drosophila*

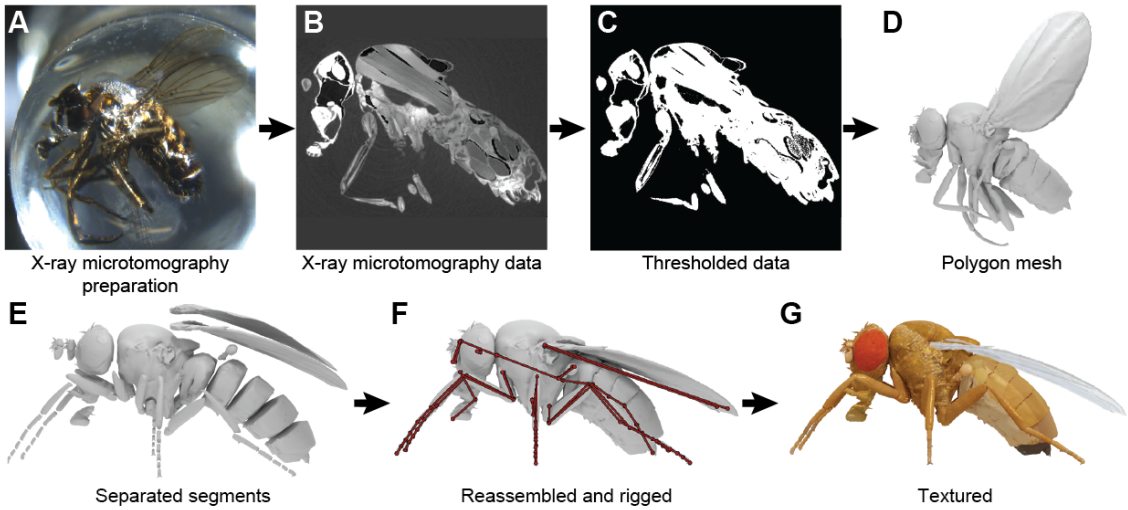

Figure 2: **Constructing a data-driven biomechanical model of adult *Drosophila*. (A)** An adult female fly is encased in resin for x-ray microtomography. **(B)** Cross-section of the resulting x-ray scan. Cuticle, muscles, nervous tissues, and internal organs are visible. **(C)** A threshold is applied to these data to separate the foreground (fly) from the background, and then to segment the exoskeleton. **(D)** A 3-dimensional polygon mesh of the exoskeleton and wings is constructed. **(E)** Articulated body segments are separated from one another. **(F)** These segments are reassembled into a natural pose. Joints locations are identified and constraints are introduced to create a articulated body (red). **(G)** Textures are added to the model.

Behavior depends heavily on the body's physical constraints and its interactions with the environment. Therefore, morphological realism is critical to accurately replicate limb movements, self-collisions, joint ranges of motion, body mass distributions, and limb loading. To achieve this realism in our simulation, we performed x-ray microtomography to digitize an adult female fly **(Video 1)**. We embedded the animal in resin to reduce scanner movement-induced blurring (Figure 2*A*). Then we processed the resulting microtomography data (Figure 2*B*) by binarizing it to discriminate between foreground (fly) and background (Figure 2*C*). Finally, we applied a Lewiner marching cubes algorithm [41] to generate a polygon mesh 3D reconstruction of the animal (Figure 2*D*).

Subsequently, to articulate appendages from this polygon mesh, we separated the body in 65 segments (see Table 1)(Figure 2*E*) and reassembled them into an empirically defined natural resting pose. Bones and joints were added to permit actuation of the antennae, proboscis, head, wings, halteres, abdominal segments, and limb segments based on observations from high-resolution videography [32], and previously reported limb DoF [42–44](Table 1)(Figure 2*F*). To connect these body parts, we used hinge-type joints. For example, to construct thorax-coxa joints (three DoF), we combined three hinge joints along the yaw, pitch, and roll axes of the base link. This approach also facilitated the control of each DoF in the physics engine and increased the interpretability of joint angle readings. Finally, we textured the model for visualization purposes (Figure 2*G*). This entire process yielded a rigged model

of adult *Drosophila* that could then be used for neuromechanical studies as well as for model-based computer vision methods [45–49].

2.2 Matching kinematics between *Drosophila* and NeuroMechFly

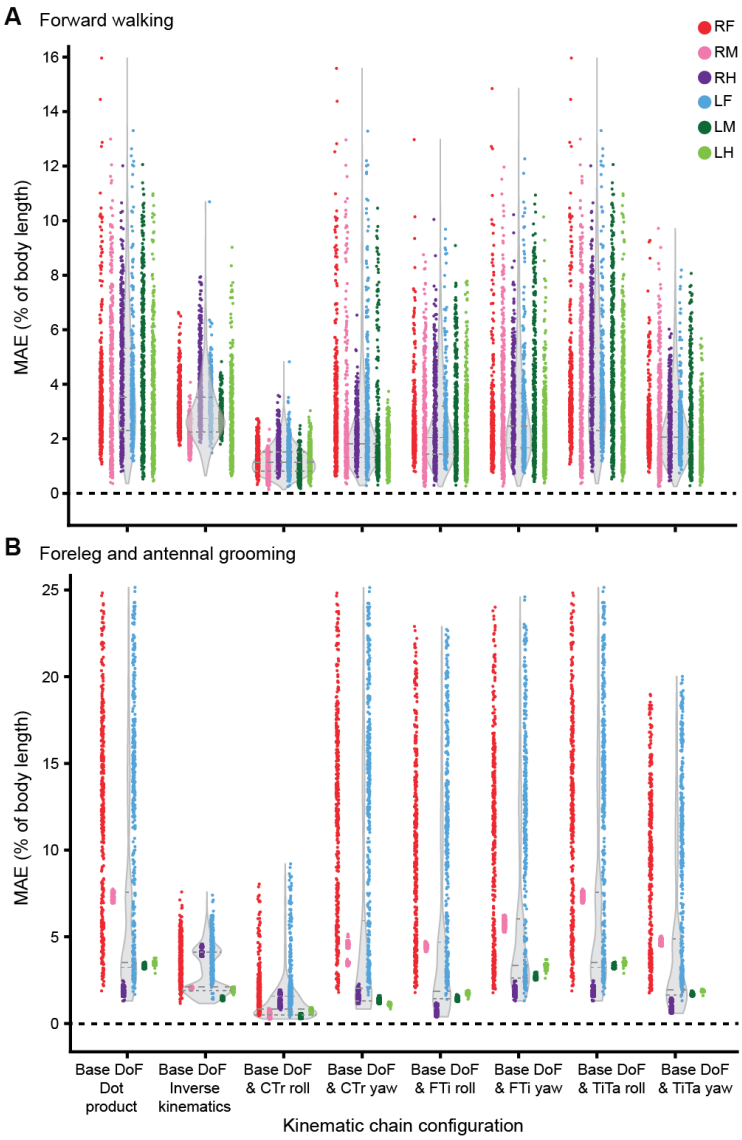

Figure 3: **Adding a coxa-trochanter roll to base DoF uniquely permits accurate kinematic replay of limb movements during forward walking and foreleg/antennal grooming.** Body-length normalized mean absolute errors (MAE) comparing tracked 3D poses and angle-derived joint positions for various DoF configurations. Measurements were made for representative examples of **(A)** forward walking, or **(B)** foreleg/antennal grooming. For each condition, n = 2400 samples were computed for all six legs across 4s of 100 Hz video data. 'R' and 'L' indicate right and left legs, respectively. 'F', 'M', and 'H' indicate front, middle, and hind legs, respectively.

After constructing NeuroMechFly's physical hull, we next investigated whether our limb DoFs would be sufficient to accurately replay limb kinematics recorded during forward walking and grooming, two of the most common fly behaviors. It has been reported that the fly has six DoFs per limb—(i-iii) thorax-coxa (ThC) elevation/depression, protraction/retraction, and rotation, (iv) coxa-trochanter (CTr) flexion/extension, (v) femur-tibia (FTi) flexion/extension, and (vi) tibia-tarsus (TiTa) flexion/extension [42, 43]. However, this notation may be misleading when discussing rotations of the ThC joints. For example, for the middle and hind legs, ThC protraction/retraction

occurs along a different axis than similarly named movements of the front legs. Therefore, we instead used the notations 'roll', 'pitch', and 'yaw' to refer to rotations around the anterior/posterior, medial/lateral, and dorsal/ventral axes of articulated segments, respectively.

We first developed a computational pipeline that could convert 3D pose estimates—obtained using DeepFly3D [32]—into joint angles that would be compatible with the DoFs available in NeuroMechFly. We acquired 3-dimensional poses from recordings of tethered flies spontaneously behaving on a spherical treadmill. Due to inter-animal variability, we observed some noise in predicted limb positions. Without fixed joint positions and stable limb segment lengths, direct application of pose estimates to actuate NeuroMechFly's joints would be impossible. Therefore, we anchored the thorax-coxa joints as stable reference points and calculated joint angles that would be invariant across animals by computing the dot product between adjacent limb segments joined by single-rotational joints, or between the global rotational axes and limb segments for multi-rotational joints (see Materials and Methods). We could then compute joint angles for each reported DoF [42,43]. Because the *Drosophila* trochanter is thought to be fused to the femur [43], we did not add a trochanter-femur (TrF) joint.

Interestingly, when only these 6 DoFs were used to replay walking and grooming, we consistently observed a large discrepancy between joint positions obtained using 3D poses versus those computed from joint angles via forward kinematics (Figure 3A-B; Base DoF Dot product). Visualization of these errors showed significant out-of-plane movements of limb segments, primarily of the tibia and tarsus. This was surprising given that each limb is assumed to consist of a ball-in-socket joint (3 DoFs in the thorax-coxa joint) followed by a series of one DoF hinge joints. Thus, limb segments distal to the coxa are expected to be in the same plane. Furthermore, we tried to identify alternative limb configurations that might better match 3D poses, for example, by using an inverse kinematic optimization of joint angles instead of dot product operations. This optimization approach identifies angle configurations that minimize error at the most distal segment of a kinematic chain—in this case, the pretarsus. Although this yielded a lower discrepancy (Figure 3; Base DoF Inverse kinematics), we still observed consistent out-of-plane movements.

Because (i) these findings were consistent with studies of other insects that propose additional stabilizing rotations at or near the TrF joint [50–53], (ii) in *Drosophila* the trochanter and femur are fused, and (iii) *Drosophila* hosts reductor muscles of unknown function near the CTr joint [42], we next examined the possibility that an extra DoF might be needed at the CTr joint to accurately replicate real fly limb movements. To ensure that any improvements would not simply stem from overfitting—simply increasing the number of DoFs—we also tested adding one roll or yaw DoF to each of the more distal hinge-type joints (CTr, FTi and TiTa)(**Video 2**). Both for walking (**Video 3**, top-right) and for foreleg/antennal grooming (**Video 4**, top-right), we found that adding a roll DoF at the CTr joint most markedly reduced the difference between 3D pose-derived and angle-derived joint positions, even compared with the inverse kinematics approach (Figure 3; Base DoF & CTr roll). This was not the case when a CTr yaw, or other DoFs in the joints downstream of the CTr were added to the six previously reported (base) limb DoFs (Figure 3; Base DoF & CTr yaw, Base DoF & FTi roll, Base DoF & FTi yaw, Base DoF & TiTa roll, Base DoF & TiTa yaw). These results suggest, in addition to the previously reported six DoF per leg, an accurate kinematic recapitulation of *Drosophila* limb movements requires an additional roll DoF near the CTr joint (Table 1). Thus, as a default for NeuroMechFly and for the remainder of this study, we incorporated this additional DoF for each leg.

2.3 Estimating torques and contact forces during kinematic replay

We next sought to illustrate how our model could be used to gain new insights into real biological experiments. Having defined necessary limb DoFs, we next investigated to what extent we could infer otherwise inaccessible physical quantities from real animal poses by replaying them in our physical simulation. To simulate its body movements, NeuroMechFly's various modules—controller, muscle model, body structure, and physics environment—continuously interact with one another (Figure 4). Depending on one's needs, these modules can achieve different levels of realism ranging from simple proportional-derivative (PD) controllers that drive basic Ekeberg-type muscle models [54] to more sophisticated Hodgkin-Huxley type neurons [14] that drive more complex Hill-type muscle models [55]. The physics-based simulation environment in which we run our model, PyBullet, also provides a simple proportional-derivative (PD) position controller. Thus, as a first proof-of-concept we used PD position control to actuate NeuroMechFly's joints during kinematic replay.

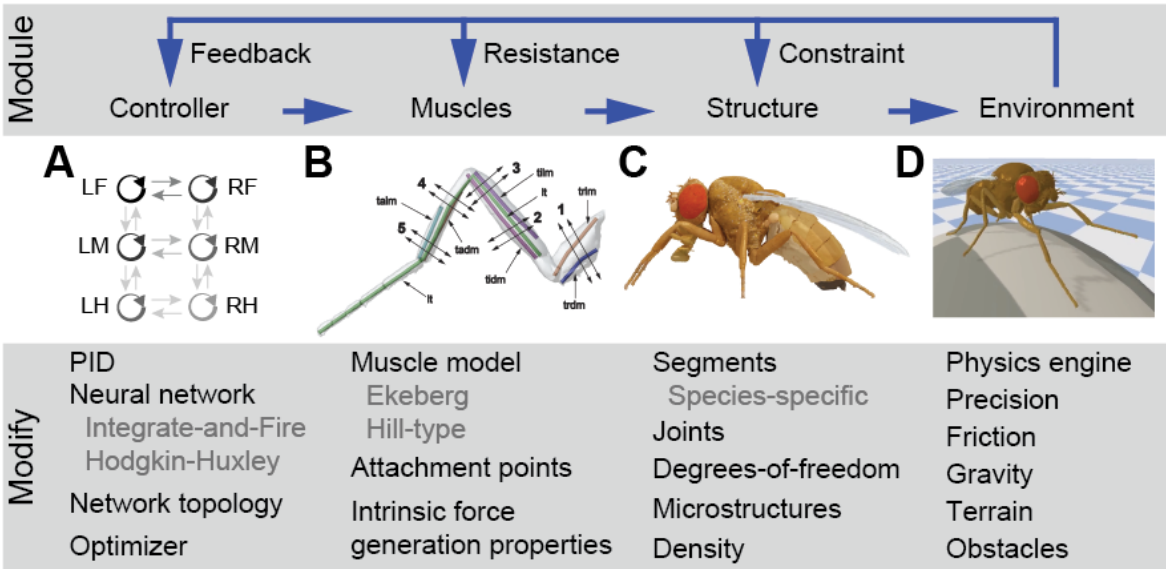

Figure 4: **Modules that can be independently modified in NeuroMechFly.** A controller's output drives muscles in an embodied system to generate movement while in continuous interaction with the physical environment. Each of these modules can be independently modified within the NeuroMechFly framework. **(A)** Controllers can range from abstract to physiologically realistic models. Computational methods like Evolutionary Algorithms and Reinforcement Learning allow users to tune controller parameters. **(B)** Muscle contractions generate forces that enable joint movements depending on musculoskeletal geometry. Elaborate models such as the Hill-type muscle model require more parameters including muscle lengths and attachment points. **(C)** The morphology of the model—including the center-of-mass, joint DoFs, and mass distribution—shapes the dynamical control of movement and introduces articulation constraints. **(D)** Rigid body dynamics including collisions and forces acting on the body are simulated using a physics engine that can also alter properties of the environment including terrain complexity and gravity.

PyBullet provides access to physical quantities of motion including torques, and contact forces (collision and ground reaction forces). Considering the small scale of *Drosophila*, these quantities can be challenging to measure by experimentation. We reasoned that, using a physics-based simulation, one might instead infer these values. We note, however, that when applying a position controller there is no unique set of contact solutions that match forces and torques to prescribed kinematics—experimental validation of force estimates is ultimately necessary. However, this remains a technical challenge for studying small insects [17, 56]. Therefore, we reasoned that even if kinematic replay might not provide information about internal forces, these estimates of collisions and forces would serve as good first approximations of proprioceptive and mechanosensory signals experienced by an animal. To explore this possibility, using a PD controller, we actuated each leg DoF to angles measured by 3D pose estimation during forward walking and foreleg/antennal grooming. Because we used a PD controller, we first quantified how sensitive torque and force estimates would be to controller gains. Based on this sensitivity analysis, we chose gain values that optimized the precision of kinematic replay while also preventing large variation in the measured physical quantities (Figure S1). To maintain a naturalistic pose, we also fixed the positions of the non-actuated abdominal segments, wings, halteres, head, proboscis, and antennae (Table 2).

We found that, during replayed walking (Figure 5 A-C)(**Video 5**) and foreleg/antennal grooming (Figure 6 A-C)(**Video 6**), limb movements were largely identical to those measured from *Drosophila*. By visual inspection, we also noticed striking similarities between simulated and real spherical treadmill rotations. This was surprising given that the ball's rotations were not explicitly controlled but emerged from tarsal contacts and forces in our simulation. These observations support the accuracy of our computational pipeline in processing and replaying measured joint angles.

We next examined collisions and forces computed within the PyBullet physics-based simulation environment. From kinematic replay of joint angles (Figure 5E, top), we measured rich, periodic joint torque dynamics (Figure 5E, middle). These were accompanied by ground reaction forces (GRFs)

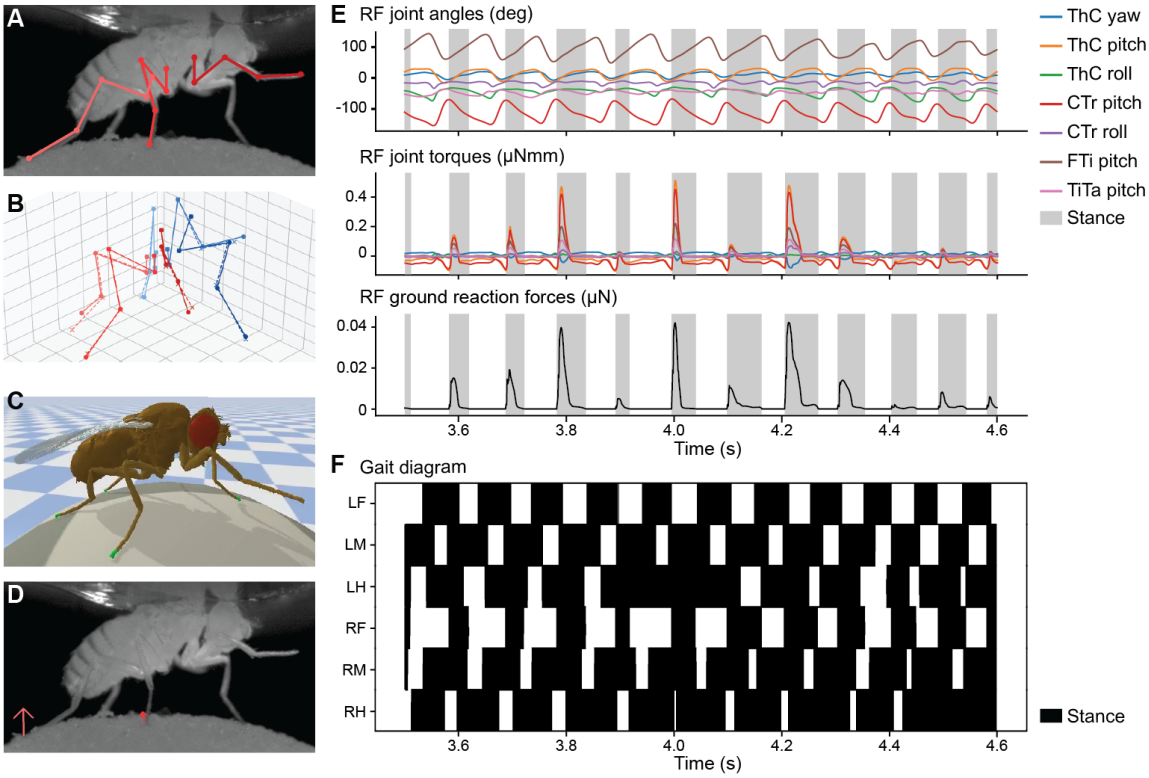

Figure 5: **Kinematic replay estimates ground contacts and reaction forces during forward walking.** **(A)** Multiple cameras and deep network-based 2D pose estimation are used to track the positions of each leg joint while a tethered fly is walking on a spherical treadmill. **(B)** Multiview 2D poses (solid lines) are then triangulated and processed to obtain 3D joint angles (dashed lines) for seven DoFs per leg. **(C)** 3D joint angles are replayed with NeuroMechFly. Green body segments are in contact with the ground. **(D)** Simulated ground collisions are used to visualize estimated ground reaction forces (red arrows). **(E, top)** Kinematic replay of real 3D joint angles permits estimation of unmeasured **(E, middle)** joint torques, and **(E, bottom)** ground reaction forces. Only data for the right front leg (RF) are shown. Grey bars indicate stance phases when the leg is in contact with the ground. Joints are color-coded. **(F)** A gait diagram illustrating stance (black) and swing (white) phases for each leg are computed by measuring simulated pretarsus contacts with the ground.

that closely tracked subtle differences in limb placement during sequential walking cycles (Figure 5E, bottom). Although experimental data would be required to fully validate these measurements, they fall within the range of previously observed maximum forces measured at the tip of the tibia ($\sim 100 \mu N$) [57]. Superimposing these GRF vectors upon raw video recordings of the fly allowed us to visualize expected tarsal forces (Figure 5D) which could also be used to generate predicted gait diagrams during tethered walking (Figure 5F).

Similarly, for foreleg/antennal grooming (Figure 6 A-C), we observed that measured joint angles (Figure 6E, top) gave rise to cyclically varying joint torques (Figure 6E, middle). Associated limb and antennal contact forces (Figure 6D and E, bottom) could also then be used to generate grooming diagrams—akin to locomotor gait diagrams—that illustrate predicted contacts across the distal limb segments and antennae (Figure 6F). These collision data can provide a more rich description of grooming kinematics beyond classifying the body part that is being cleaned [28].

Because we obtained 3D pose estimates from a tethered fly behaving on a spherical treadmill, our simulation was also 'tethered'—the thorax position was fixed. Therefore, to illustrate the extent to which our model might also be able to walk without body support, we replayed the same 3D kinematics from a tethered fly while NeuroMechFly could move freely (untethered) on a flat surface. We found that our model could walk on the ground stably, even in the face of external perturbations (**Video 7**). Although an animal would be expected to position its front and hind legs differently on a curved versus a flat surface, the flexibility of its tarsal segments allowed NeuroMechFly to walk

with a naturalistic pose. Thus, one may also infer physical quantities from simulating untethered, freely behaving flies, including predicted torques, loading, and proprioception during behaviors like locomotion, jumping and landing among many others. In summary, kinematic replay of real 3D poses in NeuroMechFly allows one to infer physical quantities that serve as predictions for further experimental validation and exploration.

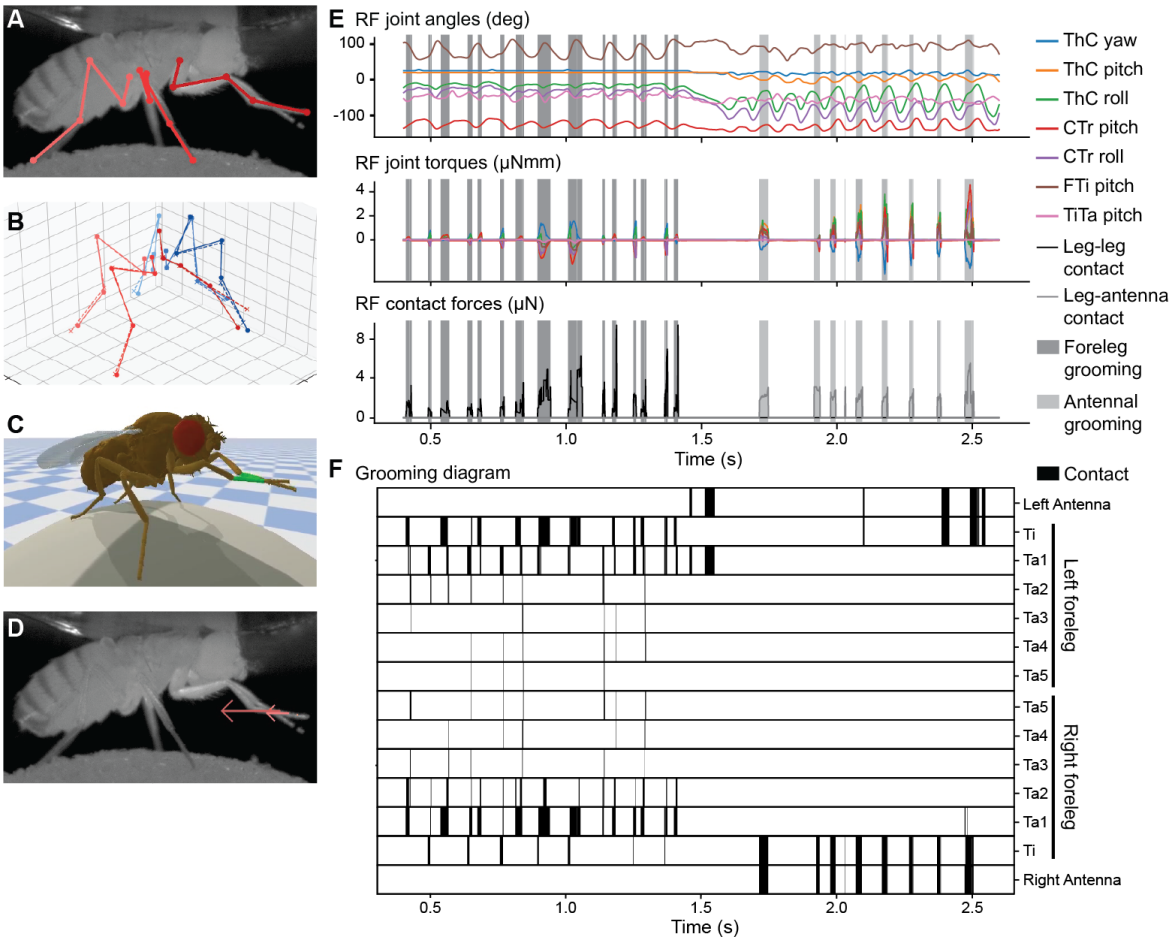

Figure 6: **Kinematic replay estimates ground contacts and reaction forces during foreleg/antennal grooming.** **(A)** Multiple cameras and deep network-based 2D pose estimation are used to track the positions of each leg joint while a tethered fly is grooming its forelegs and antennae on a spherical treadmill. **(B)** Multiview 2D poses (solid lines) are then triangulated and processed to obtain 3D joint angles (dashed lines) for seven DoFs per leg. **(C)** 3D joint angles are replayed with NeuroMechFly. Green body segments are having collisions. **(D)** Simulated leg-leg and leg-antennae collisions are used to estimate contact forces (red arrows). **(E, top)** Kinematic replay of real 3D joint angles permits estimations of unmeasured **(E, middle)** joint torques, and **(E, bottom)** contact forces. Only data for the right front (RF) leg are shown. Dark grey bars indicate leg-leg contacts. Light grey bars indicate leg-antenna contacts. Joints are color-coded. **(F)** A grooming diagram illustrating contacts (black) made by the front leg's five tarsal segments ('Ta'), tibia ('Ti'), and both antennae ('Ant').

2.4 Optimizing oscillator networks and muscles to generate tripod walking

Beyond estimating mechanosensory signals like limb loading, proprioception, and contact forces, NeuroMechFly is a testbed for exploring controllers that can drive *Drosophila*-like behaviors. Insect walking gaits are thought to emerge from the connectivity and dynamics of networks of central pattern generators (CPGs) within the insect's ventral nerve cord [14, 15, 58, 59], although alternative, decentralized approaches have also been proposed [13, 60]. Thus, using our model, one might inves-

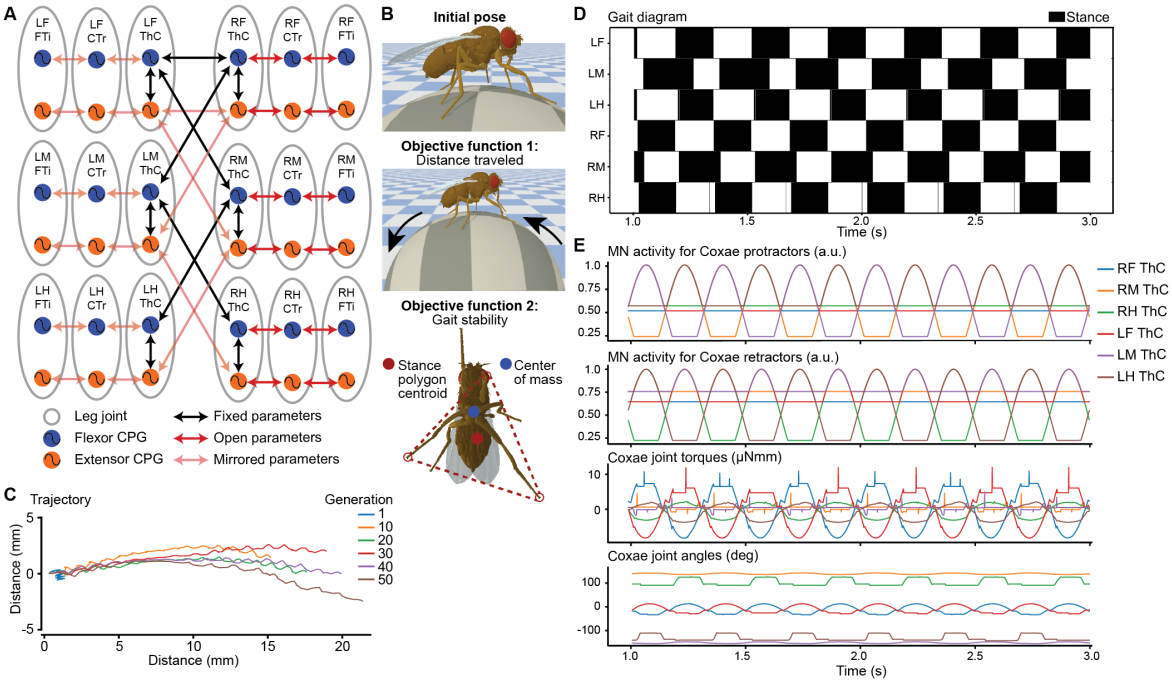

Figure 7: **Optimizing networks of coupled oscillators and muscle parameters to generate tripod locomotion. (A)** A network of coupled oscillators modeling CPG-based intra- and interlimb circuits within the ventral nerve cord of *Drosophila*. Oscillator pairs control specific antagonistic leg DoFs (gray). Network parameter values are either fixed (black), modified during optimization (red), or based on contralateral oscillators on the other body half (pink). **(B)** Multi-objective optimization of network and muscle parameters maximizes the forward distance traveled and gait stability. **(C)** Over 50 optimization epochs, the fly's locomotor trajectory gets longer. **(D)** A gait diagram indicating stance (black) and swing (white) phases is generated for a final optimized gait based on pretarsus ground contacts for each leg. **(E)** Motor neuron activity, joint torques, and joint angles for each leg's coxa during a tripod gait obtained from optimization.

tigate the respective roles of CPGs and leg sensory feedback in generating stable locomotor gaits. As well, the topology and dynamics of successful controllers may shine light on the architecture and dynamics of real *Drosophila* motor circuits.

As a proof-of-concept, we tested the hypothesis that coupled oscillatory circuits alone can account for hexapodal locomotion in *Drosophila*. First, we designed a simple neural network controller consisting of CPG-like coupled oscillators (Figure 7*A*). These were connected to a Ekeberg muscle models for each joint [54]; a more complex muscle model (e.g., Hill-type) would have required careful tuning using as-of-yet unavailable data [57, 61]. We then developed a multi-objective optimization approach [62] to discover neural networks and muscle parameters that give rise to tripod-like, or even novel hexapodal walking gaits [40]. To simplify the parameter optimization problem's search space, we limited the controlled DoFs to three per leg—ThC protraction/retraction, CTr flexion/extension, and FTi flexion/extension, DoFs that were sufficient to generate walking in other insect simulations [63]. Additionally, from our kinematic analyses, we observed that these three DoFs had the most pronounced effect on overall leg trajectory (Figure S5). In our network, each joint consisted of two coupled nonlinear phase oscillators with controlled amplitudes—one driving the extensor muscle and the other driving the antagonistic flexor. In total, we performed a parameter search for 36 coupled oscillators, connecting them in a minimal configuration to remove redundancy and to reduce search space dimensionality (Figure 7*A*). To limit the solution space to biorealistic movements, we constrained joint angle ranges to those measured for kinematic replay experiments. Finally, because *Drosophila* gaits are statically stable [40, 64], our objective function maximized both forward walking speed as well as gait stability while penalizing long periods of resting (See Materials and Methods)(Figure 7*B*). To drive limb movements, oscillator network dynamics—whose outputs are considered motor neurons activity (MN) for simplicity—were fed through muscles whose parameters

defined their active and passive stiffnesses. Muscle damping was also tuned during optimization [65] in a similar manner as for previous simulations of stick insect [10] and salamander muscles [66]. Over the course of network optimization, we observed a steady increase in the forward distance traveled by NeuroMechFly on the spherical treadmill (Figure 7C)**(Video 8)**. Access to our simulation's neural and muscle parameters allowed us to quantify CPG outputs to flexors and extensors ('MN'), joint torques, and resulting joint angles (Figure 7E).

In conclusion, using a high-level objective function, we could identify tripod-like locomotor gaits without limb sensory feedback. This does not preclude a role for sensory feedback in gait stabilization but rather serves as a first proof-of-concept that NeuroMechFly can be a testbed for exploring potential controller networks and configurations. It is also worth noting that coupled oscillators are only one of many approaches that could be used to control leg movements in NeuroMechFly [17,22,66]. Because NeuroMechFly can locomote without body support, another exciting step would be to optimize neural network controllers that can drive stable locomotion on the ground.

3 Discussion

Here we have introduced NeuroMechFly, a neuromechanical model of adult *Drosophila*. This open-source simulation framework consists of a data-driven fly exoskeleton, independently controlled body parts, and limb joints actuated using DoFs that were established to accurately replay the kinematics of real *Drosophila* walking and grooming. This model can be applied to at least two major neuroscience use cases. First, by using NeuroMechFly to replay measured fly behaviors, one can estimate joint torques and contact forces including limb ground-reaction forces, collisions, tactile and proprioceptive feedback. These estimations can in the future be validated by direct force measurements [67, 68], or indirectly by recording the activity of proprioceptive and tactile neurons within the ventral nerve cord of behaving [39], or immobile [69] animals. Second, NeuroMechFly provides passive dynamics and physical constraints within a rich physical simulation environment, making it an ideal testbed for discovering and exploring neural network topologies that can generate fly-like behaviors in diverse physical conditions (i.e. altering friction, damping, and terrain complexity). Optimal network parameters can be identified using any one of a number of approaches in a rapidly expanding machine learning toolkit including reinforcement learning and evolutionary algorithms. Resulting networks are predictions that can be tested using data from connectomics [35, 36] and neural recordings [39]. In the longer term, NeuroMechFly might be applied to experimental-computational synergies. For example, researchers might integrate NeuroMechFly into their experimental pipeline in real-time: (i) recording an animal's 3D poses, replaying these limb kinematics in NeuroMechFly to estimate joint torques, and then perturbing the animal based on these estimates.

Beyond these immediate use cases, we anticipate a number of avenues for increasing the biorealism and utility of NeuroMechFly. These improvements benefit from the inherent modularity of our simulation pipeline (Figure 4). First, biomechanical deformations might be modeled after measuring the mechanical stress and strain properties (i.e. Young's modulus) of *Drosophila* cuticle [70, 71]. Modifying the model's exoskeleton would involve tailoring meshes based on these measurements. Second, more realistic muscle dampening and nonlinear force generation properties might be achieved by using Hill-type muscle models. Physiological properties that characterize these muscles—slack tendon lengths, attachment points, maximum forces, and pennation angles—would first need to be measured [57, 72]. Third, although NeuroMechFly is accurately modeled based on numerical and physical properties of a real fly, we did not add forces observed at small scales. Thus,Van der Waals and attractive capillary forces of footpad hairs [73] might later be added to achieve additional physical accuracy. Fourth, here we have focused on limb-related behaviors and their potential control architectures. Future work can expand the scope of simulated behaviors to include take-off [74], flight [75], odor plume following [76], courtship [25], and even collective behaviors [77]. NeuroMechFly can already fundamentally support related movements: one can already articulate the wings, halteres, abdominal segments, proboscis, and antennae.

One main advantage of NeuroMechFly is that it is built using an open-source and modular software that invites contributions from different research communities. Each module can be removed, adjusted, or substituted to achieve various levels of abstraction. One might, for instance, design new neural network controllers using Hodgkin-Huxley-type neuron models rather than PD controllers. Similarly, a Hill-type muscle model may be preferred rather than an Ekeberg spring and damper

muscle model. This same flexibility also applies to the complexity of the physical environment. Here we illustrated two implementations: an animal behaving on a spherical treadmill and walking freely on flat ground. One can easily modify the simulation environment by varying physical quantities such as friction, gravity, or damping, as well as introduce complex structures like heightfield terrains [78].

Ultimately, the NeuroMechFly framework can be used to synthesize experimental data, test the sufficiency of control models, and formulate predictions for further experimental evaluation. There are no restrictions to the scope of individual modules, allowing further work to include feedback from peripheral sensors (e.g., visual, gustatory, olfactory, and tactile) as well as central circuits that process these signals. These efforts are expected to be accelerated by connectomics studies of adult *Drosophila* brains and ventral nerve cords [35, 36]. In summary, NeuroMechFly can be used to accelerate our understanding of how passive biomechanics and active neural control give rise to whole-organism behavior, also serving as a bridge between basic neuroscientific discovery and applications in artificial intelligence and robotics.

4 Materials and Methods

4.1 Constructing an adult *Drosophila* neuromechanical model

4.1.1 Preparing adult flies for x-ray microtomography

The protocol used to prepare flies for microtomography was designed to avoid distorting the exoskeleton. We observed that traditional approaches for preparing insects for either archival purposes or for high resolution microscopy, including scanning electron microscopy [79], result in the partial collapse or bending of some leg segments and dents in the exoskeleton of the thorax and abdomen. These alterations mostly occur during the drying phase and while removal of ethanol by using supercritical carbon dioxide drying reduces these somewhat, it is still not satisfactory. We therefore removed this step altogether, and instead embedded flies in a transparent resin. This resulted in only a small surface artifact over the dorsal abdominal segments A1, A2, and A3.

Flies were heavily anaesthetized with CO_2 gas, then carefully immersed in a solution of 2% paraformaldehyde in phosphate buffer (0.1M, pH 7.4) containing 0.1% Triton 100, to ensure fixative penetration, and left for 24 h at 4°C. Care was taken to ensure the flies did not float on the surface, but remained just below the meniscus. They were then washed in 0.1M cacodylate buffer (2 x 3 min washes), and placed in 1% osmium tetroxide in 0.1M cacodylate buffer, and left at 4°C for an additional 24 h. Flies were then washed in distilled water and dehydrated in 70% ethanol for 48 h, followed by 100% ethanol for 72 h, before being infiltrated with 100% LR White acrylic resin (Electron Microscopy Sciences, US) for 24 h at room temperature. This was polymerised for 24 h at 60°C inside a closed gelatin capsule (size 1; Electron Microscopy Sciences) half filled with previously hardened resin to ensure the insect was situated in the center of the final resin block, and away from the side.

4.1.2 X-ray microtomography

We glued the sample onto a small carbon pillar and scanned it using a 160 kV open type, microfocus X-ray source (L10711/-01; Hamamatsu Photonics K.K., Japan). The X-ray voltage was set to 40 kV and the current was set to 112 uA. The voxel size was 0.00327683 mm. To perform the reconstruction, we used X-Act software from the microtomography system developer (RX-solutions, Chavanod, France) obtaining a stack of 982 tiff images of size 1046x1636 pixels.

4.1.3 Building a polygonal mesh volume from processed microtomography data

First, we isolated cuticle and wings from the microtomography data using Fiji [80]. We selected 360 images from the tiff stack as the region of interest (ROI) beginning from slice 300. The tiff stack with the ROI was then duplicated. The first copy was binarized using a threshold value of 64 to isolate the cuticle. The second copy was cropped to keep the upper half of the image—where the wings are—and then binarized using a lower threshold value of 58. Finally, we applied a closing morphological operation to isolate the wings. Both binarized stacks were stored as tiff files.

We developed custom Python code to read the tiff stacks, and to fill empty holes within the regions of interest (body or wings). Finally, we used the Lewiner marching cubes algorithm [41] (implemented

in the scikit-image package [81]) to obtain a polygonal mesh for each stack. Both meshes were then exported to a standard compressed mesh storage format.

4.1.4 Separating and reassembling articulated body parts

We used the Blender Foundation version 2.81 [82] to clean and manipulate the polygon meshes obtained from microtomography data.

After importing the meshes into Blender, we removed noise in the meshes by selecting all vertices linked to the main body (or wings), inverting the selection, and deleting these vertices. We explored the resulting meshes, looking for spurious features, and then manually selected and deleted the related vertices. To obtain 65 body segments (Table 1), we manually selected and deleted vertices from our imported 3D models (body and wings). Segments were then separated at joint locations based on published morphological studies [83].

Each wing was separated into an individual segment from the wing model. The body model was separated into 63 segments as explained below. The abdomen was divided into five segments following the tergite divisions. The first and second tergites were combined as the first segment (A1, A2), and the last segment (A6) included the sixth to tenth tergites. Each antenna was considered a single segment and separated from the head capsule at the antennal foramen. Both eyes and the proboscis were separated from the head. The latter was divided into two parts, the first containing the rostrum (Rostrum), and the second containing the haustellum and labellum (Haustellum). Each leg was divided in eight parts: the coxa, trochanter/femur, tibia, and five tarsal segments. The thorax was considered a single segment and only the halteres were separated from it.

Each segment was stored as a *.blend file and processed in Blender to obtain closed meshes. First, a remesh modifier was used in 'smooth mode', with an octree depth of 8, and a scale of 0.9 to close the gaps generated in the meshes after been separated from the original model. Smooth shading was enabled and all disconnected pieces were removed. Then, we used 'sculpt mode' to manually compensate for depressions/collapses resulting from the X-ray microtomography preparation, or from separating body segments.

Then, all segments were copied into a single *.blend file and rearranged into a naturalistic resting pose (Figure 2F). We made the model symmetric to avoid inertial differences between contralateral legs that could cause unexpected behaviors during simulation. We used the more detailed microtomography data from the right side of the fly. The model was split along the longitudinal plane using the bisect tool. Then the left side was eliminated and the right side was duplicated and mirrored. Finally, the mirrored half was repositioned as the left side of the model, and both sides of the head capsule, rostrum, haustellum, thorax, and abdominal segments were joined.

At this point, the model consisted of about nine million vertices, an intractable number for commonly used simulators. We therefore used the decimate tool to simplify the mesh and collapse its edges at a ratio of 1% in every segment. This resulted in a model with 87 thousand vertices that conserved most of the important details and discarded only some bristles and cuticular textures.

4.1.5 Rigging the Blender model

We added an Armature object alongside our model in Blender to build the skeleton of the fly. To actuate the model, we created a bone—a tool in Blender that is used for animating the characters—for each segment. Bones are created such that the thorax is the root of the skeleton and each bone is the child of its proximal bone, as shown in Table 1. Then, the bones are positioned along the longitudinal axis of each segment with its head and tail over the proximal and distal joints, respectively. Each joint was positioned where segments were separated from each other. Each bone inherited the name of its corresponding mesh and is defined as its parent.

We developed custom Python code for Blender to add custom properties to each bone. These properties can be used later in a simulator, e.g., to define the maximum velocity, or maximum effort of each link. Furthermore, this code adds a limit rotation constraint (range of motion) to each axis of rotation (DoF) of every bone. The range of motion for each rotation axis per joint was defined as $-180°$ to $180°$, because, to the best of our knowledge, there are no reported angles for these variables. The DoF for each bone (segment) were based on previous studies [42, 75, 84] and listed in Table 1. Any bone can be rotated in Blender to observe the constraints imposed upon each axis of rotation. These axes are defined locally for each bone.

Finally, a 'zero-position' was defined for our model in which most bones are positioned in the direction of an axis of rotation (S4). Each leg segment and the proboscis are positioned along the Z axis. Each abdominal segment and the labellum are along the X axis. Both wings, eyes, and halteres are positioned along the Y axis. The head and the antennae are the only bones out of a rotational axis: the head is rotated 20° around the Y axis, and the antennae are at 90° with respect to the head bone. Positioning the bones along the axis of rotation makes it easier to predict a segment's position based on its angular information and standardizes the direction of movements.

4.1.6 Exporting the Blender model into the Bullet simulation engine

First, we obtained the name, location, global rotation axis, range of motion, and custom properties for each bone in the model using a custom Python script run in Blender. As mentioned above, the axes of rotation are defined locally for each bone. Therefore, our code also transforms this information from the local to the global reference system, obtaining the rotation matrix of each bone. These information for each bone are stored in a dictionary data structure (*.yaml file). This YAML file is read by another Python script run in Blender which finds meshes with the same names as the bones. These meshes are then exported as *.stl files. Meshes have to be translated to the origin of the global reference system before they are exported.

We used a Simulation Description Format (SDF, http://sdformat.org/) to store the model's information. This format consists of an *.xml file for describing objects and environments in terms of their visualization and control. This description file was created from the information stored in the *.yaml file extracted from Blender using a custom Python script.

The SDF file contains all of the information related to the joints (rotational axes, limits, and hierarchical relations) and segments (location, orientation, and corresponding paths of the stl mesh) of the biomechanical model. We can modify this file to add or remove segments, joints, or to modify features of existing segments and joints. To construct joint DoF, we used hinge-type joints because they offer more freedom to control individual joint rotations. Therefore, for biological joints with more than 1 degree-of freedom, we positioned in a single location as many rotational joints as degrees-of freedom required to describe its movement. The parenting hierarchy among these extra joints was defined as roll-pitch-yaw. The mass and collision mesh were related to the segment attached to the pitch joint—present in every biological joint of our model. The extra segments were defined with a zero mass and no collision shape.

Our model is based upon the physical properties of a real fly. The full body length and mass of the model are set to $2.8\,mm$ and $1\,\mu g$, respectively. Rather than having a homogeneous mass distribution that would decrease the stability of our model without body support, we used different mass values (and densities) obtained from experimental data for the following body parts: head ($0.125\,\mu g$), thorax ($0.31\,\mu g$), abdomen ($0.45\,\mu g$), wings ($0.005\,\mu g$), and legs ($0.11\,\mu g$). Therefore, the center of mass and the rigid-body dynamics of the model were more similar to those of a real fly.

Since small-sized objects (e.g. less than $15cm$ in PyBullet) are susceptible to simulation errors due to numeric overflows during computations, we scaled up the units of mass and length while setting up the physics of the simulation environment, and scaled down the calculated values while logging the results. Therefore, the physics engine was able to compute the physical quantities without numeric errors, and the model also accurately reflects the physics of a real fly.

4.2 Kinematic replay and analysis

4.2.1 Measuring 3D poses during forward walking and foreleg/antennal grooming

Data for kinematic replay during forward walking and foreleg/antennal grooming were taken from a previously published study describing DeepFly3D, a deep learning-based 3D pose estimation tool [32]. These data consist of images from seven synchronized cameras obtained at 100 fps (https://dataverse.harvard.edu/dataverse/DeepFly3D). Time axes (Figure 5E,F and Figure 6E,F) correspond to time points from the original, published videos. Forward walking replay data were obtained from experiment #5 of a control animal (#8) expressing aDN-GAL4 only. Foreleg and antennal grooming replay data were obtained from experiment #3 of an experimental animal #6 expressing aDN-GAL4 driving UAS-CsChrimson. We used DeepFly3D v0.4 [32] to obtain 3D poses from these images. 2D poses were examined using the GUI to perform manual corrections on 36 frames during walking and 128 frames during grooming.

4.2.2 Processing real 3D pose data

DeepFly3D, like many other 3d pose estimators, uses a local reference system based on the cameras' positions to define the animal's pose. Therefore, we first defined a global reference system for NeuroMechFly from which we could compare different experiments with different animals (see Figure S4).

Aligning both reference systems consisted of five steps. First, we defined the mean position of each Thorax-Coxa (ThC) joint as their fixed positions. Second, we calculated the orientation of the vectors formed between the hind and middle coxae on each side of the fly with respect to the global x-axis along the dorsal plane. Third, we treated each leg independently and defined their ThC joints as their origin. Fourth, we rotated all data points on each leg accordingly with its side (i.e., left or right) and the orientations obtained previously. Finally, we use the NeuroMechFly exoskeleton as a template for positioning all coxae within our global reference system because the exoskeleton has global location information for all of the joints. Next, we translated each data point for each leg (i.e. CTr, FTi, and TiTa joints) with respect to the ThC position based on the template.

Since the body size of NeuroMechFly and the fly being recorded can differ, we scaled the real fly's leg lengths for each experiment to fit the size of our model. We calculated a scaling factor for each leg segment as the ratio between its mean length throughout the experiment and the template's segment length. Then, each data point was scaled using the corresponding scaling factor.

4.2.3 Calculating joint angles from 3D poses

We considered each leg as a kinematic chain and calculated the angle of each degree of freedom (DoF) to reproduce real poses in NeuroMechFly. We refer to this as 'kinematic replay'. Angles were obtained by computing the dot product between two vectors with a common origin. We obtained 42 angles in total, seven per leg. The angles' names correspond to the rotational axis of the movement—roll, pitch, or yaw—for rotations around the anterior-posterior, mediolateral, and dorsoventral axes, respectively.

The thorax-coxa joint (ThC) has three DoFs. The yaw angle is measured between the dorsoventral axis and the coxa's projection in the transverse plane. The pitch angle is measured between the dorsoventral axis and the coxa's projection in the sagittal plane. To calculate the roll angle, we aligned the coxa to the dorsoventral axis by rotating the kinematic chain from the Thorax to the Femur-Tibia joint (FTi) using the yaw and pitch angles. Then, we measured the angle between the anterior-posterior axis and the projection of the rotated FTi in the dorsal plane.

Initially, we considered only a pitch DoF for the Coxa-Trochanter/femur joint (CTr). This was measured between the coxa and femur's longitudinal axis. Afterward, we identified an important role for a roll DoF on the CTr to match the kinematic chain. To calculate this angle, we rotated the tibia-tarsus joint (TiTa) using the inverse angles from the coxa and femur and measured the angle between the anterior-posterior axis and the projection of the rotated TiTa in the dorsal plane.

The pitch angle for the FTi was measured between the femur and tibia's longitudinal axis, and the pitch angle for the TiTa was measured between the tibia and tarsus's longitudinal axis. The direction of rotation was calculated by the determinant between the vectors forming the angle and its rotational axis. If the determinant was negative, the angle was inverted.

To validate that the original six DoF were not enough for an accurate kinematic replay, we additionally used inverse kinematics to assess whether an optimizer could find a set of angles that were capable of precisely matching our kinematic chain using these six DoFs. To compute inverse kinematics for each leg, we used the optimization method implemented in the Python IKPy package (L-BFGS-B from Scipy). We defined the zero pose as a kinematic chain and used the angles from the first frame as an initial position (seed) for the optimizer.

4.2.4 Calculating forward kinematics and errors with respect to 3D poses

To quantify the contribution of each DoF to the kinematic replay, we used the forward kinematics method to compare the original and the reconstructed poses. Since noise in pose estimation causes each 3D pose-based leg segment length to vary, we set the length of each segment as the mean length of that segment across all frames.

We first calculated the angles from 3D pose estimates with the addition of each DoF as explained in the previous section. Next, we formed a new kinematic chain including the new DoF using the

IKPy's chain module. This kinematic chain allowed us to compute the forward kinematics from joint angles, which is then compared with the 3D pose estimates to calculate the error. We performed an exhaustive search to find the angle that minimizes the overall distance between each 3D pose joint position and that joint's position reconstructed through forward kinematics. The search spanned $-90°$ to $90°$ from the 'zero pose' in $0.5°$ increments.

The error between 3D-based and angle-based poses per leg was calculated as the average distance across every joint. To normalize error with respect to body length, we measured the distance between the antenna and genitalia in our Blender model: $2.88\,mm$. Errors were computed based on 400 frames: frames 300-699 for forward walking and frames 0-399 for foreleg/antennal grooming.

4.2.5 Transferring 3D pose data into the NeuroMechFly reference frame

We incorporated a CTr roll DoF into NeuroMechFly as it was necessary for accurate kinematic replay of the 3D pose data. To do this, we enabled rotations along the Z axis of the CTr joints. Then, we recreated the configuration files (i.e. YAML and SDF) using our custom Python scripts such that the resulting files would contain a CTr roll DoF in each leg.

To simulate the fly tethering stage used in our experiments, we added three support joints (one per axis of movement) that would keep our model in place. However, we removed these supports during the ground walking experiments (**Video 7**). To create a spherical treadmill in PyBullet, we added a spherical object with a diameter of $10\,mm$. This measurement was taken from our experimental setup. The sphere has three joints along the x, y, and z axes, allowing the ball to rotate freely. The mass of each link was empirically set to $50\,\mu g$ to allow the fly to rotate the ball with its legs.

We used position control for each joint in the model. We fixed the position of non-actuated joints with the values shown in Table 2. The actuated joints, i.e. the leg joints, were controlled to achieve the angle calculated from 3D pose data. The simulation was run with a time step of $1\,ms$. Since the fly recordings were captured at 100 fps, we interpolated pose estimations prior to calculating joint angles to match the time step of simulation with that of joint angle data.

4.2.6 Application of external perturbations

To test the stability of the untethered model walking over flat ground, we introduced external perturbations by propelling solid spheres at the model governed by the following equation of motion,

$$\vec{p} = \vec{r} + \vec{u}t + \frac{1}{2}\vec{a}t^2 \tag{1}$$

where, $\vec{p}$ is the 3D target position(fly's center of mass), $\vec{r}$ is the 3D initial position of the sphere, $\vec{u}$ is the initial velocity vector, $\vec{a}$ is the external acceleration vector (only external acceleration is due to gravity in the z-direction), t is the time taken by the sphere to reach the target position $\vec{p}$ from $\vec{r}$ with an initial velocity $\vec{u}$. The mass of the sphere was $3mg$. Spheres were placed at a distance of $2mm$ from the fly's center of mass in the y-direction. With t set to $20ms$, the initial velocity of the projectile was computed using Equation 1. The spheres were shot at the model every $0.5s$ during the simulation. Finally, at $3s$ into the simulation, a sphere with $3g$ was shot at the fly to topple it.

4.2.7 Analyzing NeuroMechFly contact and collision data

The PyBullet physics engine provides forward dynamics simulations and collision detections. We plotted joint torques as they were calculated from PyBullet. To infer ground reaction forces (GRFs), we computed and summed the magnitude of normal forces resulting from contact of each tarsal tip with the ball. Gait diagrams were generated by thresholding GRFs; a leg was considered to be in stance phase if its GRF was greater than zero.

Self-collisions are disabled by default in PyBullet. Therefore, for the kinematic replay of grooming, we enabled self-collisions between the segments of interest (i.e. tarsal segment of the leg and antenna.) We recorded normal forces generated by the collision between the right front leg's tarsus or tibia against those of the left front leg. Additionally, we recorded normal forces generated by collisions between the left front leg and the left antenna, as well as those between the right front leg and the right antenna. Grooming collision diagrams were calculated using the same rationale as for gait diagrams: a segment was experiencing a contact/collision if it reported a normal force greater than zero.

4.3 Controller gain sensitivity analysis

We achieved kinematic replay using a built-in Proportional-Derivative (PD) controller in PyBullet [78] given by

$$error = K_p(\theta_r - \theta_a) + K_d(\omega_r - \omega_a) \tag{2}$$

where θ_r and θ_a denote reference and actual positions, ω_r and ω_a are desired and actual velocities, and K_p and K_v are proportional and derivative gains, respectively.

Because the outputs of our model, i.e., dynamics of the motion, depend on controller gains, K_p and K_v, we first systematically searched for optimal gain values. To do this, we ran the simulation's kinematic replay for K_p and K_v pairs, ranging from 0.1 to 1.0 with a step size of 0.1. We then calculated the mean squared error (MSE) summed from each joint between the ground truth—pose estimates of joints obtained by running DeepFly3D [32] on real fly recordings—and joint positions obtained in PyBullet. To compare joint velocities, we used the first-order numerical differentiation method with a time-step of 1 ms to compute the velocity from position data. We compared these with joint velocities calculated in PyBullet. Our results (Figure S1) show that our biomechanical model can clearly replicate the poses of a real fly while also closely matching the velocity. In particular, an MSE of 84 rad^2/sec corresponds approximately to 1.4 rad/sec per joint. This is acceptable given the rapid, nearly 30 Hz, leg movements of the real fly.

After validating the accuracy of kinematic replay, we then performed a sensitivity analysis to measure the impact of varying controller gains on the estimated torques and ground reaction forces. This analysis showed that torques and ground reaction forces are highly sensitive to changing proportional gains (K_p) (Figure S2) but are robust to variations in derivative gain (K_v). These results are expected since high proportional gains cause "stiffness" in the system whereas derivative gains affect the "dampening" in a system's response. We observed rapid changes in estimated torques and ground reaction forces at high K_p values (Figure S2).

Although high values of K_p enable more accurate kinematic replay, we kept this value low enough to reduce stiffness in the motion of NeuroMechFly. Specifically, we chose 0.4 and 0.9 for K_p and K_v respectively to ensure smoothness in the movements of our model.

4.4 Neural network parameter optimization

4.4.1 CPG network architecture

To discover tripod-like locomotor gaits in our neuromechanical model, we designed a Central Pattern Generator (CPG) based controller composed of 36 nonlinear oscillators (Figure 7) that was previously applied to studying salamander locomotion [85]. The CPG model is governed by the following system of differential equations:

$$\dot{\theta}_i = 2\pi\nu_i + \sum_j r_j w_{ij} \sin(\theta_j - \theta_i - \phi_{ij}) \tag{3}$$

$$\ddot{r}_i = a_i(R_i - r_i) \tag{4}$$

$$M_i = r_i(1 + \sin(\theta_i)) \tag{5}$$

where the state variables—phase and amplitude of the oscillator i—are denoted as θ_i and r_i, respectively; ν_i and R_i represent the oscillator i's intrinsic frequency and amplitude, a_i is a constant. The coupling strength and phase bias between the oscillator i and j are denoted as w_{ij} and ϕ_{ij}.

For the entire network of coupled oscillators, we set the intrinsic frequency(ν) to 3 Hz, the intrinsic amplitude(R) to 1, and the constant a_i to 25. To ensure a stable phase-locked regime between oscillators, we set coupling strengths to 1000 [86]. M_i represents the cyclical activity pattern of neuronal ensembles activating the muscles. We solved this system of differential equations using the explicit Runge-Kutta method of 5th-order with a time step of $1ms$.

Each oscillator pair sends cyclical bursts to flexor and extensor muscles that antagonistically move the corresponding revolute joint. Since only three degrees of freedom are controlled during the optimization, there are 3 pairs of oscillators per leg, summing up to 36 in total. We only considered three DoF per leg because this has been shown to be sufficient for locomotion in previous hexapod models [63]. Therefore, we selected the three joint DoFs that have the most pronounced joint angles

(Figure S5). These DoFs include ThC protraction/retraction, CTr elevation/depression, and FTi flexion/extension. We coupled (i) the intraleg oscillators in a proximal to distal chain, (ii) the interleg oscillators in a tripod-like fashion (the ipsilateral front and hind legs to the contralateral middle leg from anterior to posterior), and (iii) both front legs to one another, (iv) coxa extensor and flexor oscillators. We visualized the resulting network architecture in an XML-based file format. We fixed interleg connections in the network during optimization such that the coxa joints of the front and hind legs on one side will synchronously move with the contralateral middle leg. Intraleg couplings (i.e. phase relation within a leg) were determined by the optimizer.

4.4.2 Muscle model

To generate torques on the joints, we adapted the Ekeberg muscle model [54]. This model simulates muscles as a spring and damper system, allowing torque control of any joint as a linear function of the motor neuron activities driving antagonist flexor (M_F) and extensor (M_E) muscles controlling that joint. The torque exerted on a joint is given by the equation:

$$T = \alpha(M_F - M_E) + \beta(M_F + M_E + \gamma)\Delta\varphi + \delta\dot{\varphi} \tag{6}$$

where α, β, γ, and δ represent the gain, stiffness gain, tonic stiffness, and damping coefficient, respectively [66]. $\Delta\varphi$ is the difference between the current angle of the joint and its resting pose and $\dot{\varphi}$ is the angular velocity of the joint. This muscle model makes it possible to control the static torque and stiffness of the joints based on the muscle coefficients (i.e., $\alpha, \beta, \gamma, \delta$, and $\Delta\varphi$) which are set through optimization.

4.4.3 Defining joint limits for the biomechanical model

We did not define joint limits for kinematic replay experiments because, to the best of our knowledge, joint angle limits have not been reported for adult *Drosophila melanogaster*. Nevertheless, we defined joint limits to reduce the optimization search space and to facilitate the convergence of the optimizer. These limits were based on angles measured for kinematic replay of forward walking. The upper and lower bounds for each joint were defined as the maximum and minimum angles that were achieved during walking, respectively. Using these constraints on joint angles, we created a new pose configuration (SDF) file. In addition, we fixed the position of the non-actuated joints as the mean joint angle values of the corresponding joint DoF during walking (Table 3). We used the same angles with the kinematic replay for the remaining non-actuated joints(Table 2).

4.4.4 CPG network and muscle parameter optimization

To identify neuromuscular networks that could coordinate insect-like locomotion, we performed optimization of (i) the phase difference of each network connection, (ii) five parameters controlling the gains and resting position of each spring and damper muscle (i.e., $\alpha, \beta, \gamma, \delta$, and $\Delta\varphi$), and (iii) two parameters that set the clipping threshold of each oscillator's sinusoidal output. To simplify the problem for the optimizer even further, we (i) fixed the ThC flexor-extensor phase differences to 180°, making them perfectly antagonistic, (ii) mirrored the phase differences from the right leg oscillators to the left leg oscillators, making gaits symmetric, (iii) mirrored muscle and clipping parameters from the right joints to the left joints, and (iv) mirrored phase differences from ThC-ThC flexors to ThC-ThC extensors. To obtain a tripod-like gait, we set a 180° phase difference between the two front legs, and coupled ipsilateral front and hind legs with the contralateral middle leg. This configuration caused a fast convergence to tripod-like gaits by making the front and hind legs on one side synchronous with the middle leg on the other side.

Thus, 75 parameters remained flexible during optimization: 12 phases, 45 muscle parameters, and 18 clipping thresholds. We empirically set the lower and upper bounds for the parameters so that the leg movements would stay stable along the boundaries (Table 4).

We used an NSGA-II optimizer [87], a multi-objective genetic algorithm implemented in Python using the jMetalPy package [88]. We defined two objective functions. First, we aimed to maximize the distance traveled by the fly, as measured by ball rotations (Equation 7) along the Y axis. This function penalised the fly for not completing the simulation. Second, we aimed to maximize gait stability by comparing the fly's center of mass with the centroid of the stance polygon, namely the

wide stability margin (Equation 8) [89]. To avoid stationary solutions, we penalised long stance periods. To reinforce a tripod gait further, we rewarded simulations when the legs in stance made a tripod configuration. In summary, we aimed to identify stable locomotor gaits that allowed the model to travel long forward distances. Because the optimizer works by minimizing objective functions, we simply inverted the sign of the output values.

The optimization was formulated as follows:

$$\min \quad -2000 \cdot R_b \cdot \theta_{b,\|} + p_t \quad \text{(Distance objective)} \tag{7}$$

$$\min \quad 2000 \cdot s + p_s \quad \text{(Stability objective)} \tag{8}$$

with the following penalty terms:

$$p_t = \begin{cases} 0 & \text{if simulation completed} \\ 100 + 100\frac{(t_{stop} - t_{tot})}{t_r} & \text{otherwise} \end{cases} \quad \text{(Time penalty)} \tag{9}$$

$$p_s = \begin{cases} 0 & \text{if} \quad 3 \leq \overline{t_{ms}} \leq 4 \\ 100 \cdot |\overline{t_{ms}} - 3| & \text{otherwise} \end{cases} \quad \text{(Stance penalty)} \tag{10}$$

where R_b is the ball radius, $\theta_{b,\|}$ is the angle of the ball in the direction of walking and t_{tot} is the maximum simulation duration and t_{stop} is the time at which the simulation was terminated by one of the constraints, $\overline{t_{ms}}$ is the average stance time of all legs.

Four constraints were set to end simulations prematurely. First, to make sure the model was always moving, we set a threshold for the angular rotation of the ball, increasing from $-0.3\,rad$ to allow for total rotation of the ball at the end to at least $2.94\,rad$, a value that will guarantee almost one revolution of the ball. If the fly fails to achieve this angular displacement, then the simulation is aborted. Second, if any body part besides the leg tarsal segments touched the ball, the simulation is aborted. Third, if at any moment, there are fewer than two legs on the ball, the simulation is aborted. Fourth, if a joint moves faster than $5 * 10^3\,rad/sec$, the simulation is terminated.

Each optimization generation consisted of a population of 20 individuals. Optimization runs lasted for 50 generations, an acceptable computing time of approximately 2.5 hours per run on Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz. Mutations occur with a probability of 1.0, and distribution index of 1.0. As for cross-over, we set the probability to 1.0 and distribution index to 15 (for more details, see [88]). These values allowed variations in the populations, hence more exploration of possible gait combinations, while not diverging too much from parent networks.

4.4.5 Neural network analysis

Following optimization we analyzed individuals who traveled the longest distances with highest stability in the final generation. We read-out and plotted muscle activation patterns (represented by clipped CPG outputs), joint torques, joint angles, ground reaction forces (GRFs), and ball rotations during this final simulation. GRFs were used to generate gait diagrams as explained in the previous sections. Ball rotations were used to reconstruct the model's path, which is used to compare fly's trajectories across optimization generations. Briefly, the distances made along the longitudinal (x) and transverse (y) axes were calculated from the angular displacement of the ball using the following formula:

$$\Delta x = \Delta\theta_t r \qquad \Delta y = \Delta\theta_l r \tag{11}$$

where $\Delta\theta_t$ and $\Delta\theta_l$ denote, respectively, the angular displacement around the transverse and longitudinal axes, and r is the radius of the ball.

5 Supplementary Tables

Table 1: NeuroMechFly body parts and degrees-of-freedom between each segment and its parent

Body part	**Segment**	**Parent**	**Degrees of freedom**
Abdomen	A1A2	Thorax	1
	A3	A1A2	1
	A4	A3	1
	A5	A4	1
	A6	A5	1
Head	Head capsule	Thorax	3
	Eyes (x2)	Head	0
	Antennae (x2)		1
	Rostrum		1
	Haustellum	Rostrum	1
Legs	Coxa (x6)	Thorax	3
	Trochanter/Femur (x6)	Coxa	2
	Tibia (x6)	Femur	1
	Tarsus1 (x6)	Tibia	1
	Tarsus2 (x6)	Tarsus1	1
	Tarsus3 (x6)	Tarsus2	1
	Tarsus4 (x6)	Tarsus3	1
	Tarsus5-Claw (x6)	Tarsus4	1
Thorax	Halteres (x2)	Thorax	3
	Wings (x2)		3
	Thorax	-	0

Table 2: Fixed angles for body joint during kinematic replay

Body part	**Joint**	**Fixed angle (degrees)**
Abdomen	A1A2	0
	A3	-15
	A4	-15
	A5	-15
	A6	-15
Head	Head capsule roll	0
	Head capsule pitch	10
	Head capsule yaw	0
	Left antenna	35
	Right antenna	-35
	Rostrum	90
	Haustellum	-60

Body part	**Joint**	**Fixed angle (degrees)**
Thorax	Left haltere roll	0
	Left haltere pitch	0
	Left haltere yaw	0
	Right haltere roll	0
	Right haltere pitch	0
	Right haltere yaw	0
	Left wing roll	90
	Left wing pitch	0
	Left wing yaw	-17
	Right wing roll	-90
	Right wing pitch	0
	Right wing yaw	17

Table 3: Fixed joint angles for leg joints during optimization (degrees)

Body Part	**Side**	**ThC yaw**	**ThC pitch**	**ThC roll**	**CTr pitch**	**CTr roll**	**FTi**	**TiTa**
Front	Left	0	actuated	10	actuated	0	actuated	-46
	Right	0	actuated	-10	actuated	0	actuated	-46
Middle	Left	2	-3	actuated	actuated	0	actuated	-56
	Right	2	-3	actuated	actuated	0	actuated	-56
Hind	Left	3	11	actuated	actuated	0	actuated	-50
	Right	3	11	actuated	actuated	0	actuated	-50

Table 4: Upper and lower limits for the optimization parameters

Body Part	**Joint**	α	β	γ	δ	$\Delta\varphi$	**Flexor Clip**	**Extensor Clip**
Front	ThC pitch	$[10^{-2}, 1]$	$[10^{-2}, 1]$	$[10^{-3}, 1]$	$[10^{-4}, 10^{-3}]$	[-0.22, 0.69]	[0, 1.75]	[0, 1.75]
	CTr pitch	$[10^{-2}, 1]$	$[10^{-2}, 1]$	$[10^{-3}, 1]$	$[10^{-4}, 0.92 * 10^{-3}]$	[-2.5, -1.3]	[0, 1.75]	[0, 1.75]
	TiTa	$[10^{-2}, 10^{-1}]$	$[10^{-2}, 10^{-1}]$	$[10^{-3}, 10^{-1}]$	$[5*10^{-4}, 10^{-3}]$	[0.76, 2.19]	[0, 1.75]	[0, 1.75]
Middle	ThC roll	$[10^{-2}, 1]$	$[10^{-2}, 1]$	$[10^{-3}, 1]$	$[10^{-4}, 10^{-3}]$	[-2.2, -1.95]	[0, 1.75]	[0, 1.75]
	CTr pitch	$[10^{-2}, 1]$	$[10^{-2}, 1]$	$[10^{-3}, 1]$	$[10^{-4}, 0.92 * 10^{-3}]$	[-2.35, -1.84]	[0, 1.75]	[0, 1.75]
	TiTa	$[10^{-2}, 10^{-1}]$	$[10^{-2}, 10^{-1}]$	$[10^{-3}, 10^{-1}]$	$[5*10^{-4}, 10^{-3}]$	[1.73, 2.63]	[0, 1.75]	[0, 1.75]
Hind	ThC roll	$[10^{-2}, 1]$	$[10^{-2}, 1]$	$[10^{-3}, 1]$	$[10^{-4}, 10^{-3}]$	[-2.78, 2.44]	[0, 1.75]	[0, 1.75]
	CTr pitch	$[10^{-2}, 1]$	$[10^{-2}, 1]$	$[10^{-3}, 1]$	$[10^{-4}, 10^{-3}]$	[-2.46, -1.31]	[0, 1.75]	[0, 1.75]
	TiTa	$[10^{-2}, 10^{-1}]$	$[10^{-2}, 10^{-1}]$	$[10^{-3}, 10^{-1}]$	$[5*10^{-4}, 10^{-3}]$	[1.12, 2.79]	[0, 1.75]	[0, 1.75]

6 Supplementary Figures

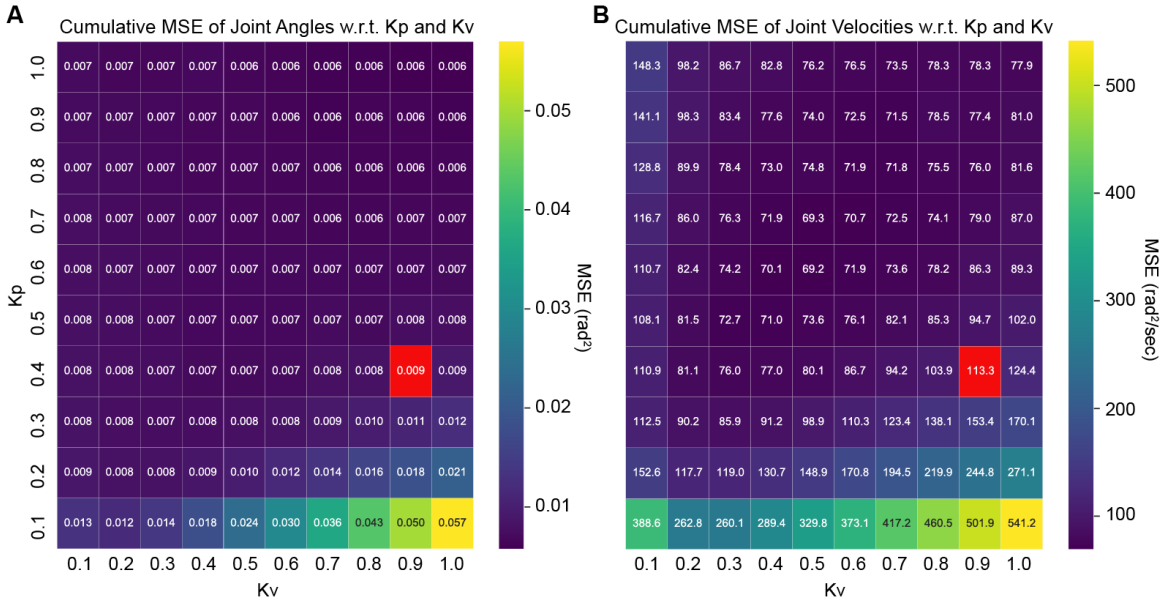

Figure S1: **Dependence of mean squared error between tracked and simulated joint trajectories and velocities on position and velocity gain values.** **(A)** K_p reduces the error in joint positions and **(B)** joint velocities by acting on the error itself at the expense of increasing the stiffness in the system. Chosen K_p and K_v value pairs are shown in red. These values are high enough to keep the error sufficiently small and low enough to avoid stiffness and large fluctuations in joint torques.

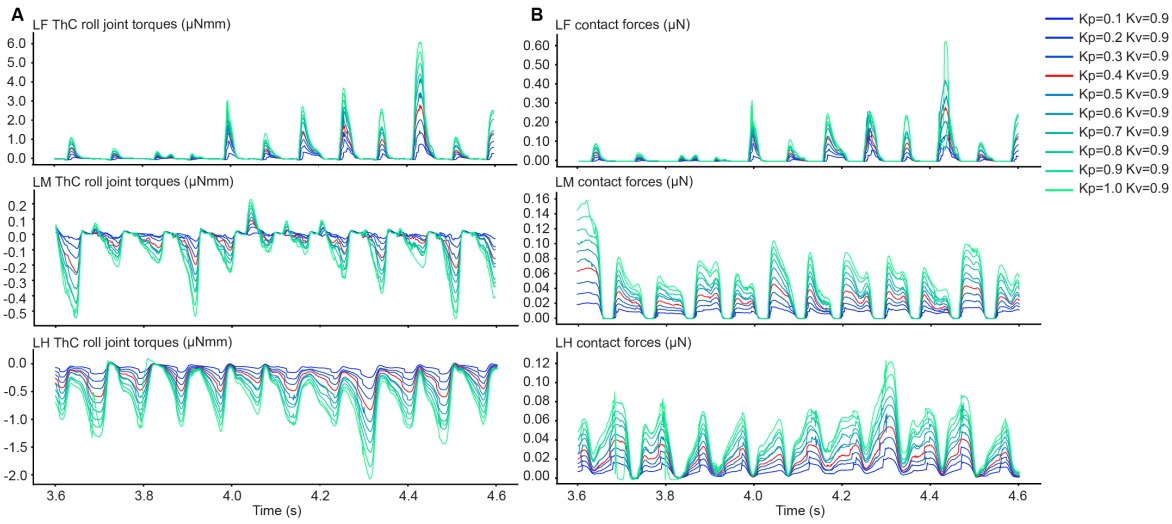

Figure S2: **Sensitivity of estimated torques and ground reaction forces to proportional gain.** **(A)** Estimated torque values during forward locomotion as a function of proportional gain values (K_p) while K_v is fixed at 0.9. Graphs show measurements for the thorax-coxa joint along the pitch axis of the left legs. Measurements for the contralateral legs are almost symmetrical. **(B)** Ground reaction force measurements of ipsilateral legs on the left side of the body during forward walking with varying K_p values. Results corresponding to the selected values of K_p and K_v are shown in red.

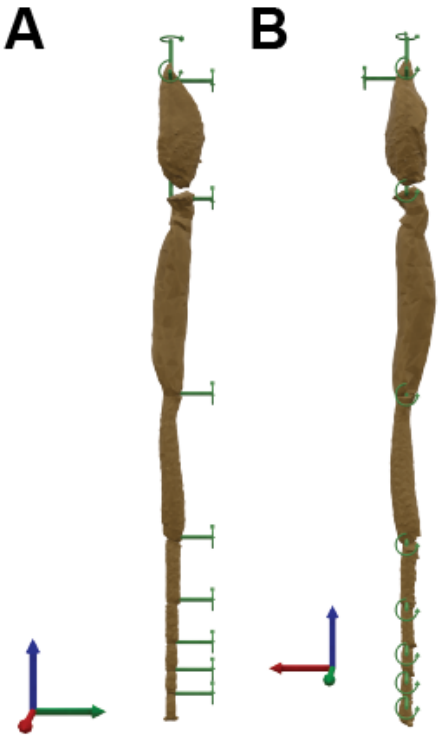

Figure S3: **Limb joint DoF and their rotational axes.** Each leg is composed of 11 hinge joints, with four fixed joints in the tarsal segment. Joints with more than one DoF are modeled as a union of multiple hinge joints. Forelimb observed from **(A)** front and **(B)** side views as well as the global coordinate system where x, y, and z are red, green, and blue, respectively.

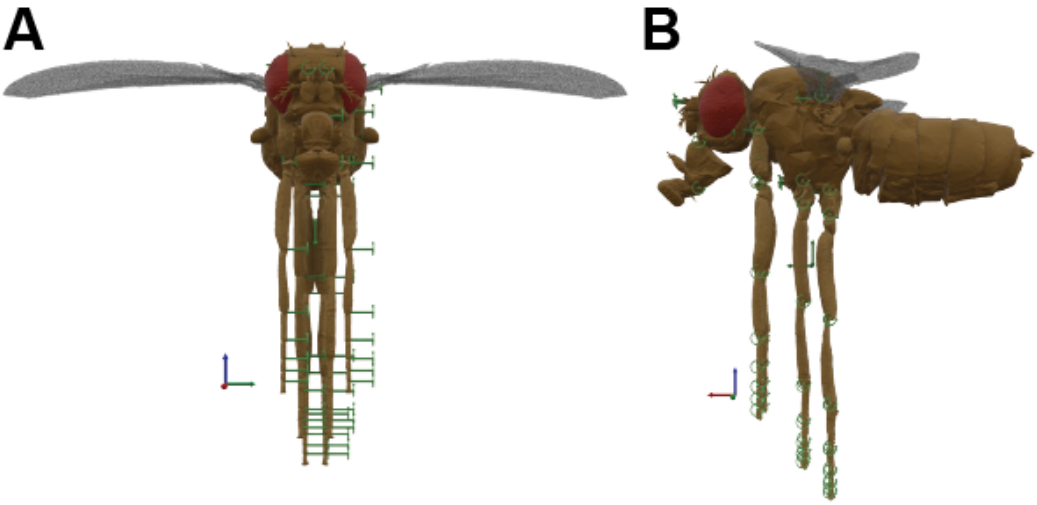

Figure S4: **Zero pose of NeuroMechFly.** Each body segment (Table 1) is aggregated via hinge joints. Rotational axes of joints and the global coordinate system are shown. **(A)** Zero pose from the front and **(B)** side views where x, y, and z are red, green, and blue, respectively.

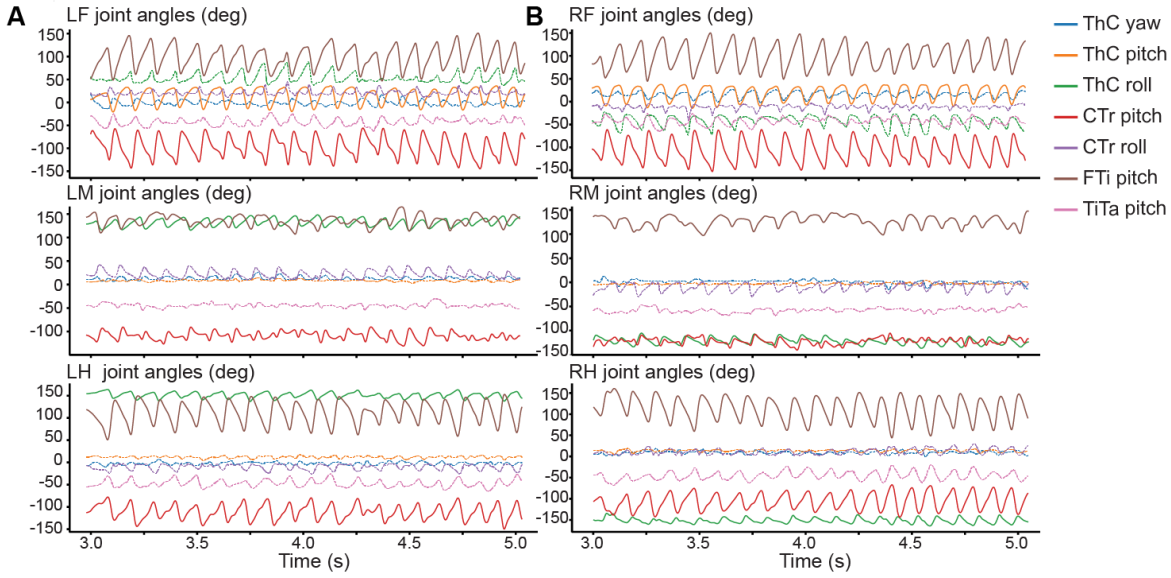

Figure S5: **Joint trajectories during forward locomotion.** Lines in bold represent joint angles of the **(A)** left and **(B)** right legs measured from a real fly during walking. These angles were controlled by a system of coupled oscillators during gait optimization. Thus, three joints are controlled for each leg by the CPG-based controller—for the front legs ThC pitch, CTr pitch, and FTi pitch, and for the mid and hind legs ThC roll, CTr pitch, and FTi pitch joint DoFs. The remaining four DoFs (dashed lines) do not exhibit pronounced angular changes.

7 Supplementary Videos

Video 1: Constructing a data-driven biomechanical model of adult *Drosophila*. An adult female fly is encased in resin for x-ray microtomography. The resulting x-ray microtomography data reveals cuticle, muscles, nervous tissues, and internal organs. These data are first thresholded to separate foreground (fly) from background. Then the exoskeleton is separated. Finally the exoskeleton is voxelized into a 3-dimensional polygon mesh. Articulated body segments are separated from one another and then reassembled into a natural pose. Bones are added and rigged to permit actuation. Finally, textures are added to the model for visualization purposes.
https://www.dropbox.com/s/9ck0ik1g4yyu6lt/Video1.mp4?dl=0

Video 2: Visualization of possible additional limb degrees-of-freedom. NeuroMechFly's left-middle leg is sequentially actuated along individual degrees-of-freedom that may be further required to match real fly limb kinematics. Articulated joint (e.g., 'CTr') and type of movement ('roll') are indicated.
https://www.dropbox.com/s/12882qeczxw3kkp/Video2.mp4?dl=0

Video 3: The effect of additional degrees-of-freedom on replaying forward walking 3D poses using forward kinematics. Measured 3D poses (solid lines) and forward kinematic replay (dashed lines) for forward walking. Forward kinematics are determined either **(top-left)** using no additional degrees-of-freedom (Base DoF, dot product), **(top-middle)** using inverse kinematics to optimize joint angles and minimize error with only base degrees-of-freedom (Base DoF, inverse kinematics), or **(top-right and bottom)** by adding a single new degrees-of-freedom (BaseDoF & 'joint' 'DoF'). Legs are color-coded.
https://www.dropbox.com/s/fama81thvy4y9xr/Video3.mp4?dl=0

Video 4: The effect of additional degrees-of-freedom on replaying foreleg/antennal grooming 3D poses using forward kinematics. Measured 3D poses (solid lines) and forward kinematic replay (dashed lines) for foreleg/antennal grooming. Forward kinematics are determined either **(top-left)** using no additional degrees-of-freedom (Base DoF, dot product), **(top-middle)** using inverse kinematics to optimize joint angles and minimize error with only base degrees-of-freedom (Base DoF, inverse kinematics), or **(top-right and bottom)** by adding a single new degrees-of-freedom (BaseDoF & 'joint' 'DoF'). Legs are color-coded.
https://www.dropbox.com/s/jq9g8tu6fm66eue/Video4.mp4?dl=0

Video 5: Kinematic replay of real *Drosophila* forward walking using NeuroMechFly. (top-left, 'Raw data') A tethered adult fly is shown walking on a spherical treadmill. One of six synchronized camera views is shown. Data are replayed at 0.15x real time. **(bottom-left, '2D tracking')** 2D poses (filled circles) and connecting 'bones' (lines) are superimposed for the proximal three legs. **(bottom-right, '3D reconstruction')** These six 2D poses are triangulated to obtain 3D poses. Overlaid are triangulated 3D poses (solid lines) and 3D joint angles derived using forward kinematics (dashed lines). **(top-right, 'Kinematic replay')** These 3D joint angles actuate NeuroMechFly limb movements as it walks on a simulated spherical treadmill. Tarsal contacts with the ground are indicated (green). Estimated ground reaction force vectors for the proximal three legs are superimposed upon the original raw 2D video data **(top-left)**.
https://www.dropbox.com/s/73w2meqrquxkjeb/Video5.mp4?dl=0

Video 6: Kinematic replay of real *Drosophila* foreleg/antennal grooming using NeuroMechFly. (top-left, 'Raw data') A tethered adult fly is shown grooming on a spherical treadmill. One of six synchronized camera views is shown. Data are replayed at 0.15x real time. **(bottom-left, '2D tracking')** 2D poses (filled circles) and connecting 'bones' (lines) are superimposed for the proximal three legs. **(bottom-right, '3D reconstruction')** These six 2D poses are triangulated to obtain 3D poses. Overlaid are triangulated 3D poses (solid lines) and 3D joint angles derived using forward kinematics (dashed lines). **(top-right, 'Kinematic replay')** These 3D joint angles actuate NeuroMechFly limb movements as it grooms on a simulated spherical treadmill. Leg segment and antennal collisions are indicated (green). Estimated collision force vectors for the proximal three legs are subsequently superimposed upon the original raw 2D video data **(top-left)**.
https://www.dropbox.com/s/j3f4ijxsiknr7ic/Video6.mp4?dl=0

Video 7: Kinematic replay of tethered *Drosophila* forward walking on flat ground without body support. Pose estimates obtained from a real tethered fly walking on a spherical treadmill are replayed in NeuroMechFly walking untethered on the ground without any body support. Balls are projected at the simulation to demonstrate its stability against most perturbations.
https://www.dropbox.com/s/yug5bd0s22vpk13/Video7.mp4?dl=0

Video 8: Walking gaits improve across optimization epochs. Forward walking gaits for four successful optimization runs for generations 1, 10, 20, and 30. Tarsal contacts with the ground are indicated (green).
https://www.dropbox.com/s/l8x6tmu804oi1bh/Video8.mp4?dl=0

8 Code and data availability

Data and code are available at: https://github.com/NeLy-EPFL/NeuroMechFly

9 Funding

PR acknowledges support from an SNSF Project grant (175667), an SNSF Eccellenza grant (181239), and an EPFL iPhD grant. VLR acknowledges support from the Mexican National Council for Science and Technology, CONACYT, under the grant number 709993. ST acknowledges support from the European Union's Horizon 2020 research and innovation programme under grant agreement nos. 720270 (SGA1), 785907 (SGA2). JA acknowledges support from the Human Frontier Science Program (HFSP-RGP0027/2017).

10 Author Contributions

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

11 Competing interests

The authors declare that no competing interests exist.

References

[1] Chiel, H. J. & Beer, R. D. The brain has a body: adaptive behavior emerges from interactions of nervous system, body and environment. *Trends in neurosciences* **20**, 553–557 (1997).

[2] Webb, B. A framework for models of biological behaviour. *International journal of neural systems* **9**, 375–381 (1999).

[3] Pearson, K., Ekeberg, Ö. & Büschges, A. Assessing sensory function in locomotor systems using neuro-mechanical simulations. *Trends in neurosciences* **29**, 625–631 (2006).

[4] Prilutsky, B. I. & Edwards, D. H. *Neuromechanical modeling of posture and locomotion* (Springer, 2015).

[5] Seth, A. *et al.* Opensim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement. *PLoS computational biology* **14** (2018).

[6] Einevoll, G. T. *et al.* The scientific case for brain simulations. *Neuron* **102**, 735–744 (2019).

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

[8] Lansner, A., Hellgren Kotaleski, J. & Grillner, S. Modeling of the spinal neuronal circuitry underlying locomotion in a lower vertebratea. *Annals of the New York Academy of Sciences* **860**, 239–249 (1998).

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

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

[11] Toth, T. I., Schmidt, J., Büschges, A. & Daun-Gruhn, S. A neuro-mechanical model of a single leg joint highlighting the basic physiological role of fast and slow muscle fibres of an insect muscle system. *PloS one* **8**, e78247 (2013).

[12] Toth, T. I., Grabowska, M., Schmidt, J., Büschges, A. & Daun-Gruhn, S. A neuro-mechanical model explaining the physiological role of fast and slow muscle fibres at stop and start of stepping of an insect leg. *PloS one* **8**, e78246 (2013).

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

[14] Szczecinski, N. S., Brown, A. E., Bender, J. A., Quinn, R. D. & Ritzmann, R. E. A neuromechanical simulation of insect walking and transition to turning of the cockroach blaberus discoidalis. *Biological cybernetics* **108**, 1–21 (2014).

[15] Proctor, J., Kukillaya, R. & Holmes, P. A phase-reduced neuro-mechanical model for insect locomotion: feed-forward stability and proprioceptive feedback. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* **368**, 5087–5104 (2010).

[16] Szczecinski, N. S., Martin, J. P., Bertsch, D. J., Ritzmann, R. E. & Quinn, R. D. Neuromechanical model of praying mantis explores the role of descending commands in pre-strike pivots. *Bioinspiration & biomimetics* **10**, 065005 (2015).

[17] Guo, S., Lin, J., Wöhrl, T. & Liao, M. A neuro-musculo-skeletal model for insects with data-driven optimization. *Scientific reports* **8**, 1–11 (2018).

[18] Nyakatura, J. A. *et al.* Reverse-engineering the locomotion of a stem amniote. *Nature* **565**, 351 (2019).

[19] Szigeti, B. *et al.* Openworm: an open-science approach to modeling caenorhabditis elegans. *Frontiers in computational neuroscience* **8**, 137 (2014).

[20] Izquierdo, E. J. & Beer, R. D. From head to tail: a neuromechanical model of forward locomotion in caenorhabditis elegans. *Philosophical Transactions of the Royal Society B: Biological Sciences* **373**, 20170374 (2018).

[21] Loveless, J., Lagogiannis, K. & Webb, B. Modelling the mechanics of exploration in larval drosophila. *PLoS computational biology* **15**, e1006635 (2019).

[22] Merel, J. *et al.* Deep neuroethology of a virtual rodent. *arXiv:1911.09451* (2019).

[23] Fry, S. N., Sayaman, R. & Dickinson, M. H. The aerodynamics of free-flight maneuvers in drosophila. *Science* **300**, 495–498 (2003).

[24] Card, G. & Dickinson, M. H. Visually mediated motor planning in the escape response of drosophila. *Current Biology* **18**, 1300–1307 (2008).

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

[26] Mendes, C. S., Bartos, I., Akay, T., Márka, S. & Mann, R. S. Quantification of gait parameters in freely walking wild type and sensory deprived drosophila melanogaster. *elife* **2**, e00231 (2013).

[27] Wosnitza, A., Bockemühl, T., Dübbert, M., Scholz, H. & Büschges, A. Inter-leg coordination in the control of walking speed in drosophila. *Journal of experimental biology* **216**, 480–491 (2013).

[28] Seeds, A. M. *et al.* A suppression hierarchy among competing motor programs drives sequential grooming in drosophila. *Elife* **3**, e02951 (2014).

[29] Pick, S. & Strauss, R. Goal-driven behavioral adaptations in gap-climbing drosophila. *Current Biology* **15**, 1473–1478 (2005).

[30] Pereira, T. D. *et al.* Fast animal pose estimation using deep neural networks. *Nature methods* **16**, 117–125 (2019).

[31] Mathis, A. *et al.* Deeplabcut: markerless pose estimation of user-defined body parts with deep learning. *Nature neuroscience* **21**, 1281–1289 (2018).

[32] Günel, S. *et al.* Deepfly3d, a deep learning-based approach for 3d limb and appendage tracking in tethered, adult drosophila. *eLife* **8** (2019).

[33] Gosztolai, A. *et al.* Liftpose3d, a deep learning-based approach for transforming 2d to 3d pose in laboratory animals. *bioRxiv* (2020).

[34] Jenett, A. *et al.* A gal4-driver line resource for drosophila neurobiology. *Cell reports* **2**, 991–1001 (2012).

[35] Maniates-Selvin, J. T. *et al.* Reconstruction of motor control circuits in adult drosophila using automated transmission electron microscopy. *bioRxiv* (2020).

[36] Scheffer, L. K. *et al.* A connectome and analysis of the adult drosophila central brain. *bioRxiv* (2020).

[37] Seelig, J. D. *et al.* Two-photon calcium imaging from head-fixed drosophila during optomotor walking behavior. *Nature methods* **7**, 535 (2010).

[38] Maimon, G., Straw, A. D. & Dickinson, M. H. Active flight increases the gain of visual motion processing in drosophila. *Nature neuroscience* **13**, 393 (2010).

[39] Chen, C.-L. *et al.* Imaging neural activity in the ventral nerve cord of behaving adult drosophila. *Nature communications* **9**, 4390 (2018).

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

[41] Lewiner, T., Lopes, H., Vieira, A. W. & Tavares, G. Efficient implementation of marching cubes' cases with topological guarantees. *Journal of Graphics Tools* **8**, 1–15 (2003).

[42] Soler, C., Daczewska, M., Da Ponte, J. P., Dastugue, B. & Jagla, K. Coordinated development of muscles and tendons of the drosophila leg. *Development* **131**, 6041–6051 (2004).

[43] Sink, H. *Muscle development in drosophila* (Springer, 2006).

[44] Cruse, H., Dürr, V. & Schmitz, J. Insect walking is based on a decentralized architecture revealing a simple and robust controller. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* **365**, 221–250 (2007).

[45] Loper, M., Mahmood, N., Romero, J., Pons-Moll, G. & Black, M. J. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)* **34**, 1–16 (2015).

[46] Zuffi, S., Kanazawa, A., Jacobs, D. W. & Black, M. J. 3d menagerie: Modeling the 3d shape and pose of animals. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 6365–6373 (2017).

[47] Li, S. *et al.* Deformation-aware unpaired image translation for pose estimation on laboratory animals. *arXiv:2001.08601* (2020).

[48] Mu, J., Qiu, W., Hager, G. D. & Yuille, A. L. Learning from synthetic animals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12386–12395 (2020).

[49] Bolaños, L. A. *et al.* A three-dimensional virtual mouse generates synthetic training data for behavioral analysis. *Nature Methods* (2021).

[50] Watson, J. T., Ritzmann, R. E., Zill, S. N. & Pollack, A. J. Control of obstacle climbing in the cockroach, blaberus discoidalis. i. kinematics. *Journal of Comparative Physiology A* **188**, 39–53 (2002).

[51] Frantsevich, L. & Wang, W. Gimbals in the insect leg. *Arthropod structure & development* **38**, 16–30 (2009).

[52] Bender, J. A., Simpson, E. M. & Ritzmann, R. E. Computer-assisted 3d kinematic analysis of all leg joints in walking insects. *PloS one* **5**, e13617 (2010).

[53] Zill, S. N. *et al.* Effects of force detecting sense organs on muscle synergies are correlated with their response properties. *Arthropod structure & development* **46**, 564–578 (2017).

[54] Ekeberg, Ö. A combined neuronal and mechanical model of fish swimming. *Biological cybernetics* **69**, 363–374 (1993).

[55] Buchanan, T. S., Lloyd, D. G., Manal, K. & Besier, T. F. Neuromusculoskeletal modeling: Estimation of muscle forces and joint moments and movements from measurements of neural command. *Human Kinetics* (2004).

[56] Cofer, D., Cymbalyuk, G., Heitler, W. J. & Edwards, D. H. Neuromechanical simulation of the locust jump. *Journal of Experimental Biology* **213**, 1060–1068 (2010).

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

[58] Fuchs, E., Holmes, P., Kiemel, T. & Ayali, A. Intersegmental coordination of cockroach locomotion: adaptive control of centrally coupled pattern generator circuits. *Frontiers in neural circuits* **4**, 125 (2011).

[59] Mantziaris, C. *et al.* Intra-and intersegmental influences among central pattern generating networks in the walking system of the stick insect. *Journal of neurophysiology* **118**, 2296–2310 (2017).

[60] Schilling, M. & Cruse, H. Decentralized control of insect walking: A simple neural network explains a wide range of behavioral and neurophysiological results. *PLoS computational biology* **16**, e1007804 (2020).

[61] Guschlbauer, C., Scharstein, H. & Büschges, A. The extensor tibiae muscle of the stick insect: biomechanical properties of an insect walking leg muscle. *Journal of Experimental Biology* **210**, 1092–1108 (2007).

[62] Oliveira, M., Santos, C. P., Costa, L., Matos, V. & Ferreira, M. Multi-objective parameter cpg optimization for gait generation of a quadruped robot considering behavioral diversity. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2286–2291 (2011).

[63] Daun-Gruhn, S. A mathematical modeling study of inter-segmental coordination during stick insect walking. *Journal of computational neuroscience* **30**, 255–278 (2011).

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

[65] Harischandra, N., Cabelguen, J.-M. & Ekeberg, A 3d musculo-mechanical model of the salamander for the study of different gaits and modes of locomotion. *Frontiers in Neurorobotics* **4** (2010).

[66] Ijspeert, A. J. A connectionist central pattern generator for the aquatic and terrestrial gaits of a simulated salamander. *Biological cybernetics* **84**, 331–348 (2001).

[67] Takahashi, H. *et al.* Maximum force capacity of legs of a fruit fly during landing motion. In *2017 19th International Conference on Solid-State Sensors, Actuators and Microsystems (TRANSDUCERS)*, 1061–1064 (IEEE, 2017).

[68] Elliott, C. J. & Sparrow, J. C. In vivo measurement of muscle output in intact drosophila. *Methods* **56**, 78–86 (2012).

[69] Mamiya, A., Gurung, P. & Tuthill, J. C. Neural coding of leg proprioception in drosophila. *Neuron* **100**, 636–650 (2018).

[70] Vincent, J. F. & Wegst, U. G. Design and mechanical properties of insect cuticle. *Arthropod structure & development* **33**, 187–199 (2004).

[71] Flynn, P. C. & Kaufman, W. R. Mechanical properties of the cuticle of the tick amblyomma hebraeum (acari: Ixodidae). *Journal of Experimental Biology* **218**, 2806–2814 (2015).

[72] Kuan, A. T. *et al.* Dense neuronal reconstruction through x-ray holographic nano-tomography. Tech. Rep., Nature Publishing Group (2020).

[73] Kimura, K.-i., Minami, R., Yamahama, Y., Hariyama, T. & Hosoda, N. Framework with cytoskeletal actin filaments forming insect footpad hairs inspires biomimetic adhesive device design. *Nature News* (2020).

[74] Ache, J. M. *et al.* Neural basis for looming size and velocity encoding in the drosophila giant fiber escape pathway. *Current Biology* **29**, 1073–1081.e4 (2019).

[75] Dickson, W. B., Straw, A. D. & Dickinson, M. H. Integrative model of drosophila flight. *AIAA Journal* **46**, 2150–2164 (2008).

[76] Breugel, F. V., Huda, A. & Dickinson, M. H. Distinct activity-gated pathways mediate attraction and aversion to co2 in drosophila. *Nature* **564**, 420–424 (2018).

[77] Ramdya, P. *et al.* Mechanosensory interactions drive collective behaviour in drosophila. *Nature* **519**, 233–236 (2015).

[78] Coumans, E. Bullet physics simulation. In *ACM SIGGRAPH 2015 Courses*, 1 (2015).

[79] Chaffey, N. Principles and techniques of electron microscopy: biological applications. 4th edn (2001).

[80] Schindelin, J. *et al.* Fiji: an open-source platform for biological-image analysis. *Nature methods* **9**, 676–682 (2012).

[81] van der Walt, S. *et al.* scikit-image: image processing in Python. *PeerJ* **2**, e453 (2014).

[82] Foundation, B. Blender-a 3d modelling and rendering package (2012).

[83] Ferris, G. External morphology of the adult. *Biology of Drosophila* 368–419 (1950).

[84] Geurten, B. R. H., Jähde, P., Corthals, K. & Göpfert, M. C. Saccadic body turns in walking drosophila. *Frontiers in Behavioral Neuroscience* **8**, 365 (2014).

[85] Ijspeert, A. J., Crespi, A., Ryczko, D. & Cabelguen, J.-M. From swimming to walking with a salamander robot driven by a spinal cord model. *Science* **315**, 1416–1420 (2007).

[86] Cohen, A. H., Holmes, P. J. & Rand, R. H. The nature of the coupling between segmental oscillators of the lamprey spinal generator for locomotion: A mathematical model. *Journal of Mathematical Biology* **13**, 345–369 (1982).

[87] Deb, K., Pratap, A., Agarwal, S. & Meyarivan, T. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation* **6**, 182–197 (2002).

[88] Benitez-Hidalgo, A., Nebro, A. J., Garcia-Nieto, J., Oregi, I. & Del Ser, J. jmetalpy: A python framework for multi-objective optimization with metaheuristics. *Swarm and Evolutionary Computation* **51**, 100598 (2019).

[89] Song, S. & Waldron, K. Machines that walk: The adaptive suspension vehicle. *International Journal of Adaptive Control and Signal Processing* **4**, 247–247 (1990).