NeuroMechFly, a neuromechanical model of adult *Drosophila melanogaster*

Victor Lobato Ríos[1], Pembe Gizem Özdil[*1], Shravan Tata Ramalingasetty[*2], Jonathan Arreguit[2], 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

Abstract

Animal behavior emerges from a seamless interaction between neural network dynamics, musculoskeletal properties, and the environment. Accessing and understanding the interplay between these intertwined elements 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 the 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 to replay real *Drosophila* walking and grooming, (ii) estimate expected contact reaction forces, torques, and tactile signals during these replayed behaviors, and (iii) discover neural network and muscle parameters that enable locomotion using gaits with a variety of speeds and stabilities. 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 contributions to behavior of many neuronal and biomechanical elements is daunting without the complementary guidance of systems-level numerical simulations. These 'bottom-up' tools can be used to consolidate data into a dynamic framework, generate predictions to be tested, and probe the sufficiency of prevailing theories to account for experimental observations [1–6]. Modeling has long played a particularly important role in the study of movement control 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 used in an ongoing dialogue with experimental data. This is possibly due to the limited availability of genetic tools allowing one to repeatedly record from or perturb identified neural circuits in these organisms. By contrast, for a few commonly studied 'model' organisms, a relatively rapid communication between experimental results and computational predictions represents an exciting but largely unrealized opportunity. This is mostly made possible by recent advances in computing power, the realism of physics-based simulations, and improvements in numerical optimization approaches; even with limited experimental data (e.g., [18]) one can reverse-engineer biomechanical systems. Neuromechanical models of some commonly studied organisms have already been developed including several for worms (*Caenorhabditis elegans* [19, 20]), maggots (larval *Drosophila melanogaster* [21]), and rodents [5, 22]. Notably absent among these is a data-driven neuromechanical model for the adult fly, *Drosophila melanogaster*.

[*]equal contribution
[†]corresponding author: pavan.ramdya@epfl.ch

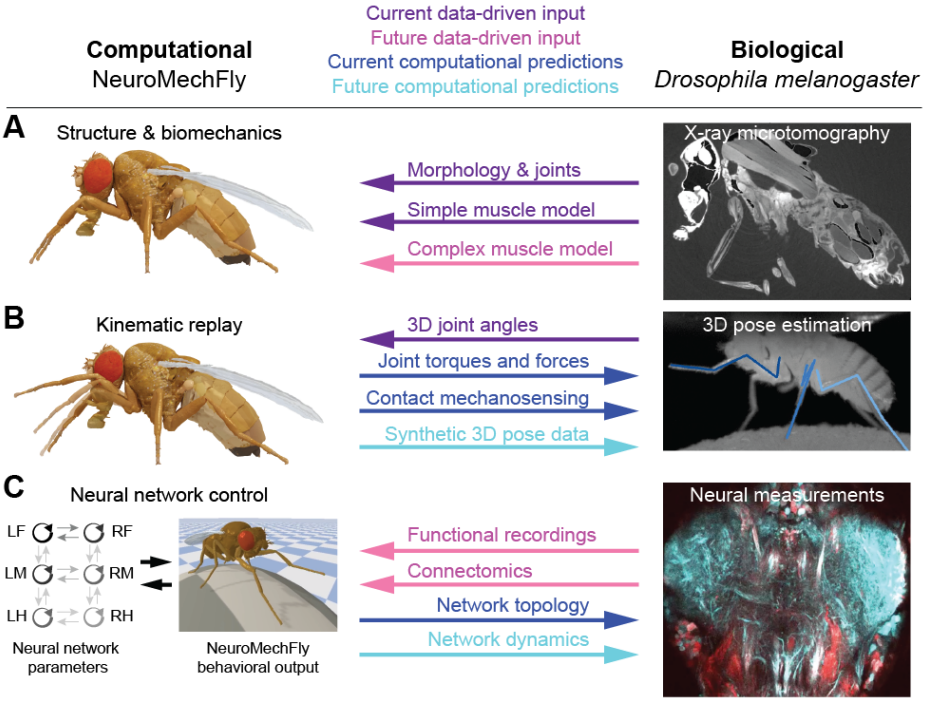

Figure 1: **Data-driven development and applications of NeuroMechFly.** **(A)** Body structures—morphology, joint locations, and degrees-of-freedom—are defined by x-ray microtomography. Future work may incorporate more complex muscle models. **(B)** Real 3D poses can be replayed using the model allowing one to predict unmeasured collisions, contact reaction forces, and joint torques. Future work may also use this model to generate synthetic data for deep learning-based 3D pose estimation. **(C)** Neuromuscular parameters are discovered by optimization to satisfy high-level objectives like fast and stable locomotion. Optimized networks may be more deeply analyzed to predict neural dynamics orchestrating behavior in real animals. Future work may use neural recordings and connectomes to constrain artificial neural networks.

Adult flies are an ideal organism for establishing a synergy between experimental and computational neuroscience. First, flies generate a large repertoire of complex behaviors including grooming [23], courtship [24], flight [25], and walking [26, 27] which they use to navigate complex environments [28]. These behaviors can now be quantified precisely using deep learning-based computer vision tools [29, 30] in 3-dimensions (3D) [31, 32]. Second, flies have a relatively small number of neurons that can be repeatedly genetically targeted [33] to perform recordings or perturbations in tethered, behaving animals [34–36]. These neurons can also be placed within their circuit context using recently acquired brain and ventral nerve cord (VNC) connectomes [37, 38]. Although we previously developed a simple physics-based simulation of adult *Drosophila melanogaster* to investigate hexapod locomotor gaits [39], this model has several important limitations that restrict its widespread use. Specifically, it lacks (i) the morphological accuracy required to simulate mass distributions, compliance, and physical constraints, (ii) muscle models and their associated passive dynamical properties, and (iii) neural networks or other control architectures.

Here we describe NeuroMechFly, a neuromechanical model of adult *Drosophila* that fills this methodological gap. NeuroMechFly is based on a detailed CT-scan of an adult female fly that was subsequently digitally rendered and rigged (Figure 1*A*). Using this model, we first show how a previously unreported coxa-trochanter limb degree-of-freedom (DoF) is required to accurately replay limb movements measured from real flies during walking and grooming (Figure 1*B*). Then, by replaying these limb kinematics in NeuroMechFly, we demonstrate how one can estimate ground reaction forces (GRFs), joint torques, and tactile contacts—quantities that cannot currently be experimentally measured in behaving flies. Finally, we illustrate how one can optimize the parameters of a central pattern generateor (CPG)-inspired coupled-oscillator network and associated torsional spring and damper-based muscle model to discover and more deeply investigate emergent locomotor gaits (Figure 1*C*).

Importantly, the NeuroMechFly framework is modular and open-source. This will enable further extensions including the use of more detailed neural and muscle models that allow one to make interpretable experimental predictions concerning real *Drosophila* neural circuits. Thus, NeuroMechFly represents an important step towards achieving an understanding of how behaviors emerge from a complex interplay between neural dynamics, musculoskeletal biomechanics, and physical interactions with the environment.

2 Results

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

Behavior depends heavily on the body's physical constraints and its interactions with the environment. Therefore, morphological realism is critical to accurately model limb movements and their associated self-collisions, joint ranges of motion, mass distributions, and loading. To achieve this level of realism in our simulation, we measured the morphology of an adult female fly using x-ray microtomography **(Video 1)**. We first embedded the animal in resin to reduce blurring associated with scanner movements (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 [40] to generate a polygon mesh 3D reconstruction of the animal's exoskeleton (Figure 2*D*).

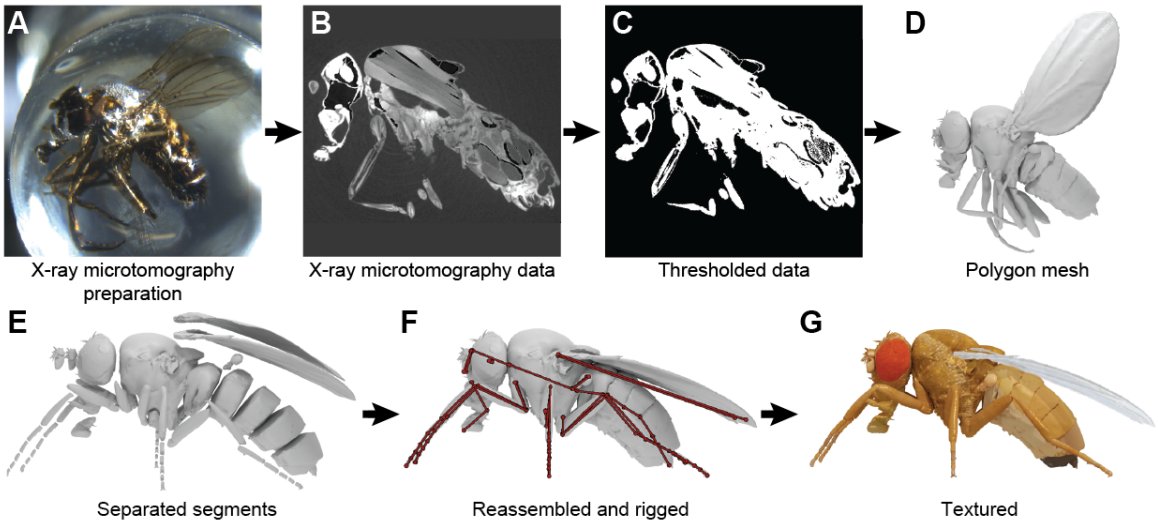

Figure 2: **Constructing a data-driven biomechanical model of adult *Drosophila*. (A)** An adult female fly is encased in resin for x-ray microtomography. **(B)** Cross-section of the resulting x-ray scan. Cuticle, muscles, nervous tissues, and internal organs are visible. **(C)** A threshold is applied to these data to separate the foreground (white) from the background (black). **(D)** A 3D 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 resting pose. Joint locations are defined and constraints are introduced to create an articulated body (dark red). **(G)** Textures are added to increase the visual realism of the model.

Subsequently, to articulate appendages from this polygon mesh, we separated the body into 65 segments (see Table 3)(Figure 2*E*) and reassembled them into an empirically defined natural resting pose. 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 [31], and previously reported limb DoFs [41–43](Table 3)(Figure 2*F*). To facilitate the control of each DoF in the physics engine, we used hinge-type joints to connect each of the body parts. Therefore, to construct thorax-coxa joints with three DoFs, we combined three hinge joints along the yaw, pitch, and roll axes of the base link. This approach also 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 was morphologically accurate enough for neuromechanical studies as well as for model-based computer vision tasks [44–48].

2.2 Replaying real *Drosophila* limb kinematics using NeuroMechFly

After constructing an articulating digital model of an adult fly, we next asked whether the six reported and implemented limb DoFs—(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 [41,42]—would be sufficient to accurately replay real fly limb kinematics. We did not add a trochanter-femur (TrF) joint Because the *Drosophila* trochanter is thought to be fused to the femur [42]. Additionally, because, for the middle and hind legs, ThC protraction/retraction occurs along a different axis than similarly named movements of the front legs, we instead use the notations 'roll', 'pitch', and 'yaw' to refer to rotations around the anterior/posterior, medial/lateral, and dorsal/ventral axes of articulated segments, respectively (**Video 2**).

For our studies, we focused on forward walking and grooming, two of the most common fly behaviors. We used DeepFly3D [31] to acquire 3D poses from recordings of tethered flies behaving spontaneously on a spherical treadmill. Due to inter-animal morphological variability and 3D pose estimation noise, we observed variations in predicted joint positions. Thus, without fixed base ThC reference positions and stable limb segment lengths, direct application of 3D poses to actuate NeuroMechFly would be impossible. To overcome this issue, we fixed the location of ThC joints as stable reference points and set each body part's length as its mean length for a given experiment. Then, we scaled ThC positions and the body part lengths using our neuromechanical model as a template. Thus, instead of using 3D cartesian coordinates, we could now calculate joint angles that were invariant across animals and that could match the DoFs available in NeuroMechFly. At first we calculated these joint angles for the six reported DoFs [41,42] 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).

Interestingly, when only these six DoFs were used to replay walking and grooming in NeuroMechFly, we consistently observed a large discrepancy between 3D pose-derived cartesian joint locations and those computed from joint angles via forward kinematics (Figure 3, Base DoF Dot product). Visualization of these errors showed significant out-of-plane movements of the tibia and tarsus (**Video 3**, top-left). This was surprising given that each limb is assumed to consist of a ball-and-socket joint (3 DoFs in the ThC joint) followed by a series of one DoF hinge joints which should result in limb segments distal to the coxa residing in the same plane. Therefore, we next tried to identify alternative limb configurations that might better match 3D poses. First we performed an inverse kinematics optimization of joint angles instead of dot product operations. This would allow us to identify angle configurations that minimize error at the most distal tip of the kinematic chain—in this case, the pretarsus. Although inverse kinematics yielded a lower discrepancy (Figure 3, Base DoF Inverse kinematics), we still observed consistent out-of-plane movements (**Video 3**, top-middle).

We next examined whether an extra DoF might be needed at the CTr joint to accurately replicate real fly limb movements. This analysis was motivated by the fact that: (i) other insects use additional stabilizing rotations at or near the TrF joint [49–52], (ii) unlike other insects, the *Drosophila* trochanter and femur are fused, and (iii) *Drosophila* hosts reductor muscles of unknown function near the CTr joint [41]. To ensure that any improvements did not result simply from overfitting by increasing the number of DoFs, we also tested the effect of adding one roll or yaw DoF to each of the more distal hinge-type joints (CTr, FTi and TiTa)(**Video 2**). Indeed, for both walking (**Video 3**, top-right) and foreleg/antennal grooming (**Video 4**, top-right), we observed that adding a CTr roll DoF to the six previously reported ('base') DoFs significantly and markedly reduced the discrepancy between 3D pose-derived and forward kinematics-derived joint positions, even when compared with improvements from inverse kinematics (Figure 3, Base DoF & CTr roll; for statistical analysis, see Table 1 and Table 2). This improvement was not seen when an additional DoF was added to joints downstream of the CTr (Figure 3, Base DoF & CTr yaw, Base DoF & FTi roll, Base DoF & FTi yaw, Base DoF & TiTa roll, Base DoF & TiTa yaw). These results demonstrate that accurate kinematic replay of *Drosophila* limb movements requires seven DoFs per leg: the previously reported six DoFs [41,42] along with a roll DoF near the CTr joint. Thus, by default, NeuroMechFly incorporates this additional DoF for each leg (Table 3).

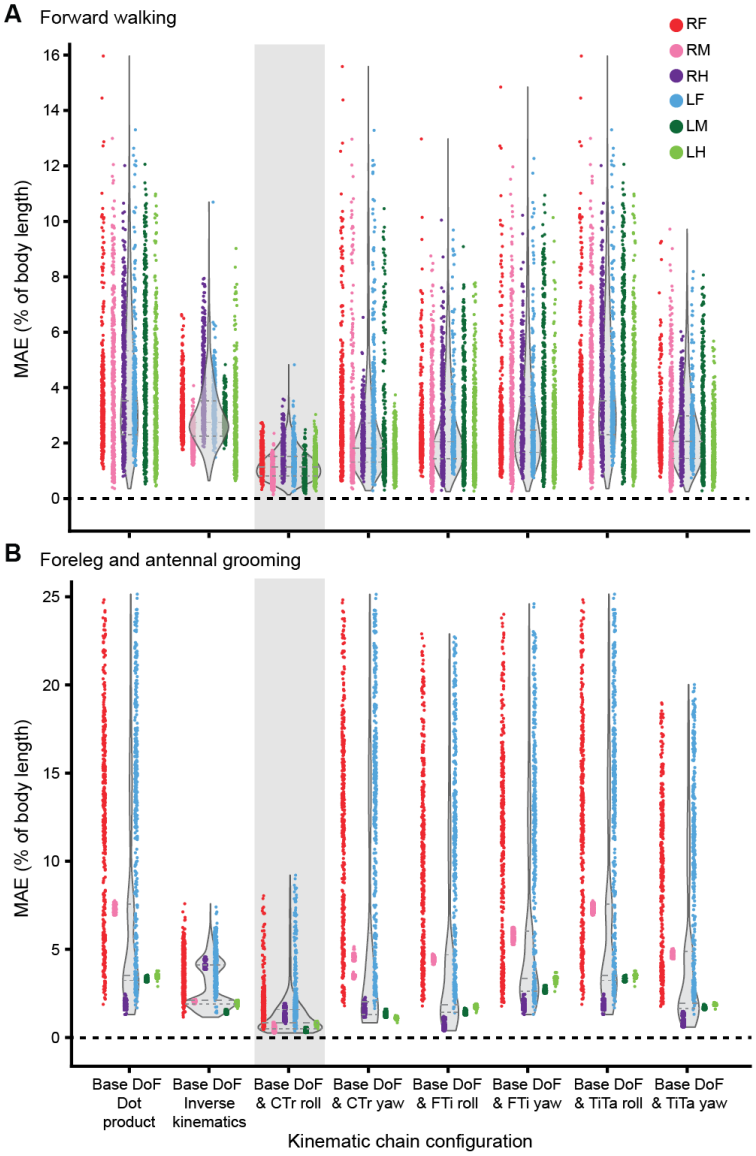

Figure 3: **Adding a CTr roll DoF to base DoFs enables the most accurate kinematic replay of real walking and grooming.** Body-length normalized mean absolute errors (MAE) comparing measured 3D poses and angle-derived joint positions for various DoF configurations. Measurements were made for representative 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. Data for each leg are color-coded. 'R' and 'L' indicate right and left legs, respectively. 'F', 'M', and 'H' indicate front, middle, and hind legs, respectively. Violin plots indicate median, upper, and lower quartiles (dashed lines). Results from adding a coxa-trochanter roll DoF to based DoFs are highlighted in light gray.

2.3 Estimating joint torques and contact forces during kinematic replay

Having identified a suitable set of limb DoFs for replaying recorded *Drosophila* behaviors, we next investigated the extent to which we might use kinematic replay to infer otherwise inaccessible physical quantities in PyBullet [53], the physics-based simulation environment. PyBullet's physics engine gives access to physical quantities of motion including torques, and contact forces like collisions and ground reaction forces (GRFs). Thus, because experimental measurements of these quantities remains technically challenging in small insects like *Drosophila* [17,54], one might instead infer these values by replaying limb kinematics in a physics-based simulation. Although kinematic replay may not provide

information about internal forces, estimates of collisions and forces remain a good first approximation of an animal's proprioception and mechanosensation.

We explored this possibility by using a proportional-derivative (PD) controller implemented in PyBullet to actuate NeuroMechFly's leg joints, replaying measured limb kinematics during forward walking and foreleg/antennal grooming. We used calculated joint angles and angular velocities as the target signals for the controller. Because, when applying this kind of 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—we first quantified how sensitive torque and force estimates would be to changes in PD controller gains. Based on this sensitivity analysis, we chose gain values that optimized the precision of kinematic replay (Figure S1) while also preventing large variations in measured physical quantities (Figure S2). We also maintained a naturalistic pose for the abdominal segments, wings, halteres, head, proboscis, and antennae (Table 4).

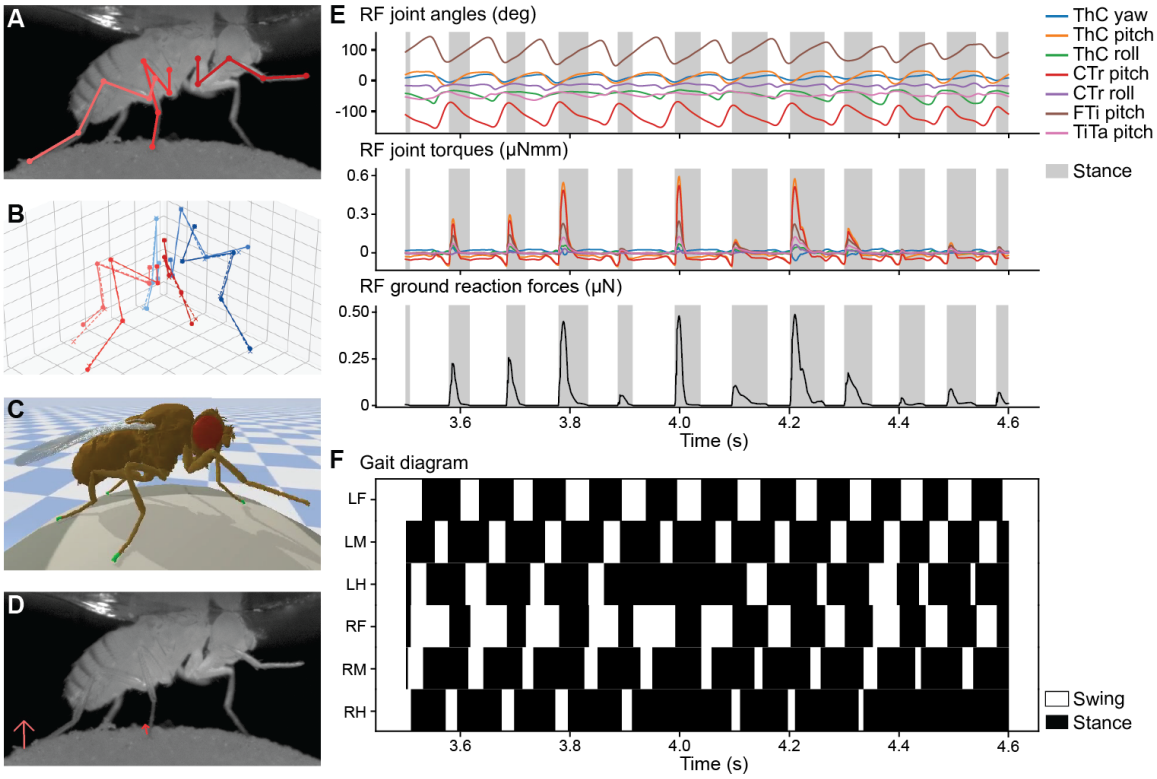

Figure 4: **Kinematic replay of forward walking estimates ground contacts and reaction forces.** **(A)** Multiple cameras and deep learning-based 2D pose estimation are used to track the positions of each leg joint while a tethered fly is walking on a spherical treadmill. **(B)** Multiview 2D poses (solid lines) are triangulated and processed to obtain 3D joint positions (dashed lines). These are further processed to compute joint angles for seven DoFs per leg. **(C)** Joint angles are replayed in NeuroMechFly. Body segments in contact with the ground are indicated (green). **(D)** Estimated ground reaction force vectors (red arrows) are superimposed on original video data. **(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. Joint DoFs are color-coded. **(F)** A gait diagram illustrating stance (black) and swing (white) phases for each leg as computed by measuring simulated tarsal contacts with the ground.

For both replayed walking (Figure 4*A*-*C*)(**Video 5**) and foreleg/antennal grooming (Figure 5*A*-*C*)(**Video 6**), we found that NeuroMechFly limb movements were largely identical to those measured from *Drosophila*. By visual inspection, we also noticed striking similarities between real and simulated 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 recorded joint positions.

We next examined collisions and forces computed within the PyBullet physics-based simulation environment. From kinematic replay of joint angles during walking (Figure 4*E*, top), we measured rich, periodic joint torque dynamics (Figure 4*E*, middle). These were accompanied by ground reaction forces (GRFs) that closely tracked subtle differences in limb placement across walking cycles (Figure 4*E*, bottom). Superimposing these GRF vectors upon raw video recordings of the fly allowed us to visualize expected tarsal forces (Figure 4*D*)(**Video 5**, top-left) which could also be used to generate predicted gait diagrams during tethered walking (Figure 4*F*). Similarly, for foreleg/antennal grooming (Figure 5*A*-*C*), we observed that measured joint angles (Figure 5*E*, top) gave rise to closely associated joint torque dynamics (Figure 5*E*, middle). Associated limb and antennal contact forces (Figure 5*D, E*, bottom) reached magnitudes about twice the fly's weight. 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 ($\sim 100\mu N$) [55]. These limb and antennal contact forces were used to generate grooming diagrams—akin to locomotor gait diagrams—that illustrate predicted contacts across the distal limb segments and antennae (Figure 5*F*). In this case, during leg-leg grooming, we observed collisions moving continuously along the limb segments in proximal to distal sweeps. These collision data provide a richer description of grooming behaviors beyond simply classifying the body part that is being cleaned [23] and open up a more precise physical quantification of this and many other behaviors.

Because we obtained 3D pose estimates from a tethered fly behaving on a spherical treadmill, we also 'tethered' our simulation by fixing the thorax position. Next, we tested the extent to which our model might otherwise walk without body support, keeping its balance while carrying its entire body weight. To do this, we replayed 3D kinematics from tethered walking (Figure 4)(**Video 5**) while NeuroMechFly could move freely (untethered) on flat terrain. Indeed, we observed that our model could walk stably on the ground, even in the face of external perturbations (**Video 7**). Notably, although an animal would be expected to position its front and hind legs differently on a curved versus a flat surface, the flexibility of NeuroMechFly's tarsal segments allowed it to walk with a naturalistic pose on flat terrain using 3D poses acquired from an animal walking on a spherical treadmill. In summary, here we have shown how NeuroMechFly can be used to replay real 3D poses in order to infer otherwise unmeasured physical quantities like joint torques, collisions, and reaction forces. These predictions may then be validated and further explored experimentally.

2.4 Optimizing oscillator networks and muscles to explore locomotor gaits

Beyond estimating physical quantities from kinematic replay of measured fly behaviors—a top-down application—one can also use NeuroMechFly to discover and investigate controllers that can orchestrate high-level behaviors—a bottom-up approach. Insect walking gaits are thought to emerge from the connectivity and dynamics of networks of CPGs within the insect's ventral nerve cord [14, 15, 56, 57], although alternative, decentralized approaches have also been proposed [13, 58]. Thus, using NeuroMechFly, one might investigate how networks of coupled CPGs and muscles generate fast and stable locomotion. Ultimately, the topology and dynamics of identified controllers can inform our understanding of real *Drosophila* motor circuits.

To illustrate this approach, we aimed to discover controllers that give rise to fast and statically stable locomotion. First, we designed a neural network controller consisting of a CPG-like coupled oscillator [59] for each joint (Figure 6*A*). For simplicity, we denoted the output of each coupled oscillator as motor neuron (MN) activity. These MNs, in turn, were connected to Ekeberg-type muscles [60], a muscle model that has been used successfully in simulations of the stick insect [10], and salamander [61]. The large number of DoFs in our model (Table 3) suggested that it might be challenging to identify suitable solutions during optimization. Therefore, we next aimed to minimize the parameter optimization's search space. To do this, we first limited the controlled DoFs to just those that were (i) sufficient to generate walking behaviors in other insect simulations [62] and (ii) had the most pronounced effect on overall leg trajectories in our kinematic analysis (Figure S5). These were just three per leg—ThC pitch for forelegs and ThC roll for middle and hind legs, CTr pitch, and FTi pitch. Additionally, to encourage the optimizer to discover more biorealistic solutions, we constrained joint ranges of motion to those measured from kinematic replay experiments. Each DoF was driven by two coupled nonlinear phase oscillators with controlled amplitudes—one driving the extensor muscle and the other driving the antagonistic flexor. Finally, we treated both sides of the fly as symmetric. We optimized phase differences between intraleg joints and muscle parameters

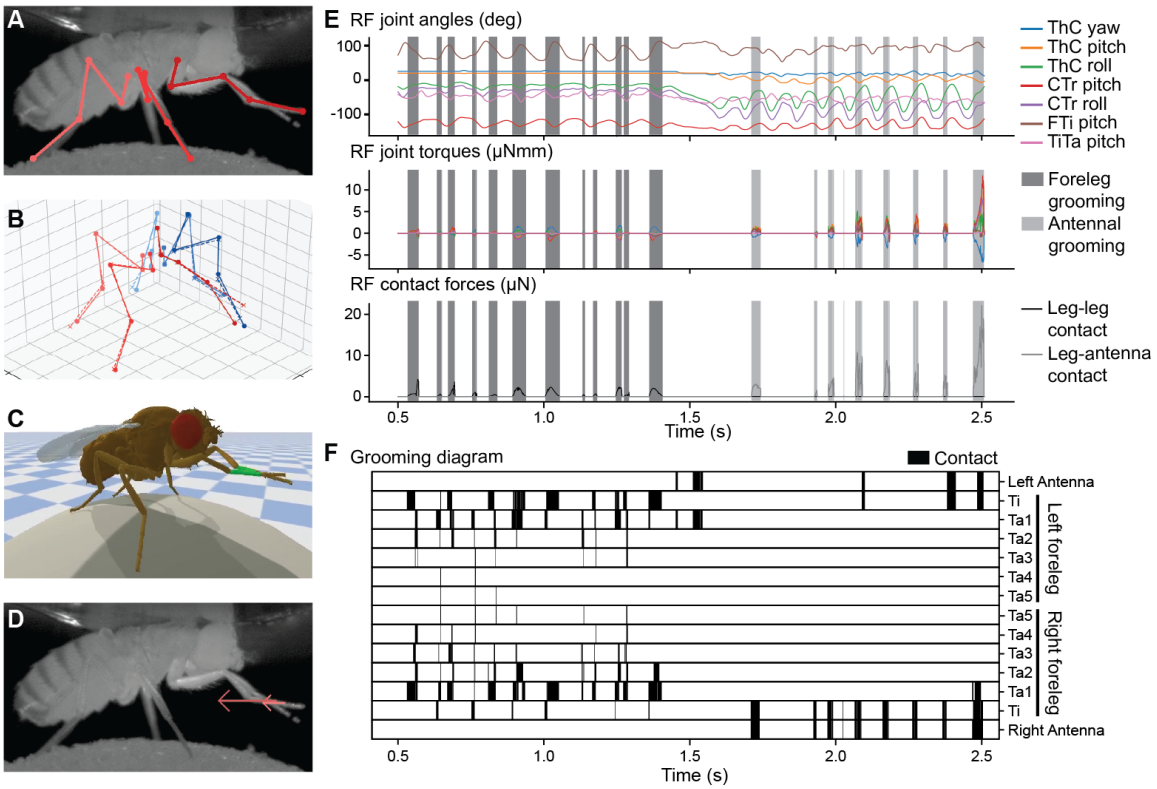

Figure 5: **Kinematic replay estimates ground contacts and reaction forces during foreleg/antennal grooming. (A)** Multiple cameras and deep learning-based 2D pose estimation are used to track the positions of each leg joint while a tethered fly grooms its forelegs and antennae. **(B)** Multiview 2D poses (solid lines) are triangulated and processed to obtain 3D joint positions (dashed lines). These are further processed to compute joint angles for seven DoFs per leg. **(C)** Joint angles are replayed in NeuroMechFly. Body segments undergoing collisions are indicated (green). **(D)** Estimated leg-leg and leg-antennae contact forces (red arrows) are superimposed on original video data. **(E, top)** Kinematic replay of real 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 ('Ta1' and 'Ta5' being the most proximal and the most distal, respectively), tibia ('Ti'), and both antennae ('Ant').

for actuated right legs and then mirrored these results for the left legs. In the same manner, we mirrored the phase difference parameters between flexors and extensors for inter-coxae connections (Figure 6*A*, pink arrows), i.e., both muscles should be equally out of phase across coxae. Thus, our 36 coupled oscillators were connected in a minimal configuration to remove redundancy and to reduce the dimensionality of the optimization search space (Figure 6*A*).

Next, we designed a multi-objective optimization approach [63] to discover network and muscle parameters that give rise to hexapodal walking gaits [39]. We used two high-level objective functions: distance traveled and static stability. Notably, under some conditions, these objectives are inversely correlated in that we might expect fast gaits to be less static stable and vice versa. Forwrd speed was defined as the number of backward ball rotations within a fixed period of time. This was quantified as a fictive distance traveled (Figure 6*B*, top). Static stability was quantified as the minimal distance between the model's center-of-mass and the convex hull formed by the legs in stance phase [39,64](Figure 6*B*, bottom). Additionally, we defined penalties to discourage unrealistic solutions (See Materials and Methods). Because the optimizer minimizes the objective functions, we inverted the sign for both functions. Thus, during network and muscle optimization, we observed that the pareto front of best solutions evolved towards more negative values (Figure S6*A*). This was reflected by a steady increase in the forward distance traveled by NeuroMechFly over optimization

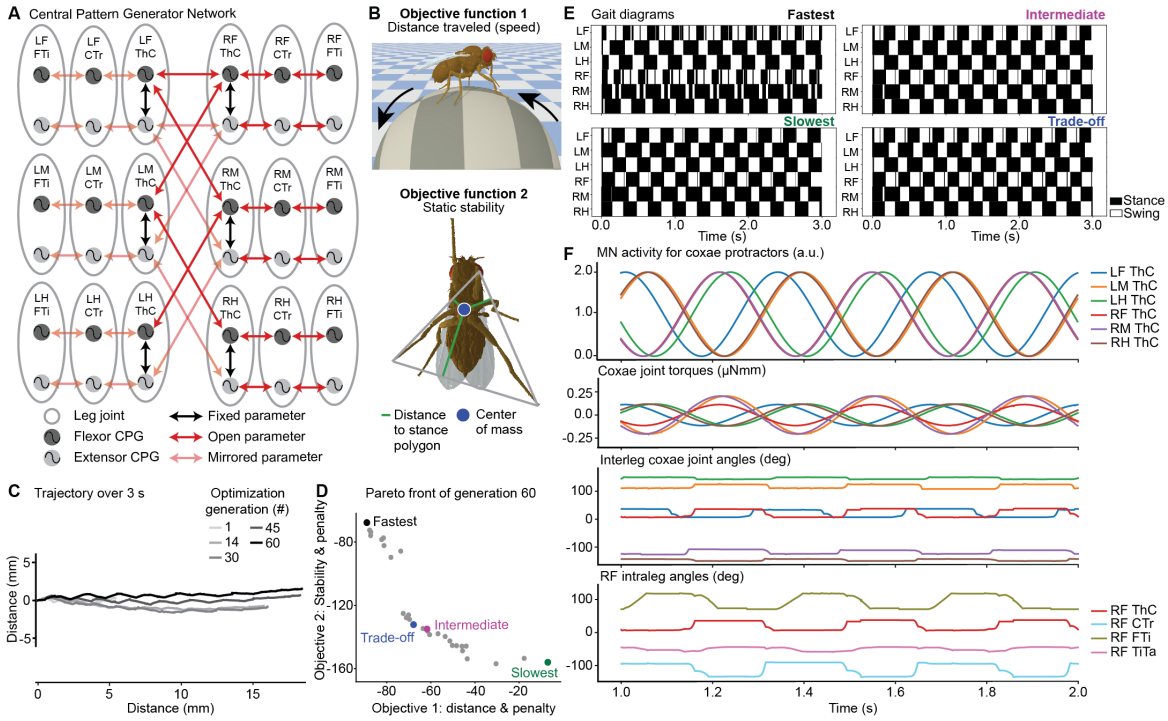

Figure 6: **Exploring optimal oscillator network and muscle parameters to achieve fast and stable locomotion. (A)** A network of coupled oscillators modeling CPG-based intra- and interlimb circuits within the ventral nerve cord of *Drosophila*. Oscillator pairs control specific antagonistic leg DoFs (gray). Network parameter values are either fixed (black), modified during optimization (red), or based on oscillators on the other half of the body (pink). **(B)** Multi-objective optimization of network and muscle parameters maximizes the forward distance traveled (speed) and static stability of locomotion. **(C)** Over 60 optimization epochs, an individual solution's locomotor trajectory gets longer. Ultimately, this solution achieves a trade-off between speed and static stability. **(D)** Pareto front of solutions from the final (60th) optimization generation. Four individuals were selected from the population using different criteria: the longest distance traveled (fastest, black), the shortest distance travel (slowest, green), the closest to the pareto front's center (intermediate, purple), and the best trade-off to achieve both objectives (trade-off, blue). **(E)** Gait diagrams for selected solutions from generation 60. Stance (black) and swing (white) phases were determined based on tarsus ground contacts for each leg. **(F)** Physics-based predictions for the 'trade-off' solution. Motor neuron (MN) activity, joint torques, and joint angles for each leg's coxa are shown, as well as intraleg joint angles for the right foreleg. Joints are color-coded.

generations (Figure 6*C*)**(Video 8)**.

To more deeply investigate our results, we selected four individual solutions from the final optimization generation. These were: (i) the fastest solution which traveled the longest distance, (ii) the slowest solution that traveled the shortest distance, (iii) a solution with intermediate speed and stability, and (iv) a solution having the best trade-off between speed and static stability (Figure 6*D*). We calculated and visualized gait diagrams for each of these solutions. We observed that the the left hind and right front legs of the fastest solution spent reduced time in stance phase (Figure 6*E*, top-left). The slowest solution had the most statically stable gait, one that closely resembled a naturalistic insect tripod-gait (Figure 6*E*, bottom-left). The solution with intermediate speed and stability exhibited characteristics of both the fastest and slowest solutions—it was tripod-like but with slightly shorter stance periods (Figure 6*E*, top-right). Finally, the 'trade-off' solution was very similar but not identical to the intermediate solution; it exhibited small differences related to hind leg movements (Figure 6*E*, bottom-right). Notably, although we restricted our optimization search space, our stability penalty could favor many possible gaits that have some combination of three and four legs in stance phase. Thus, no specific solution would be expected. Therefore, the fact that optimization converged upon a tripod-like gait supports the notion that this gait may have evolved

to achieve a trade-off between speed and static stability in insects [39, 64].

NeuroMechFly gives us access to neuromuscular dynamics and physical interactions with the environment that were used to drive optimized gaits. Thus, we next analyzed these underlying quantities to infer how they may have orchestrated locomotion in NeuroMechFly. We focused on the 'trade-off' solution discovered during optimization (Figure 6*F*) and began by analyzing MN activity for the coxae protractors of each leg. As expected for tripod locomotion, neural activity was mostly either in phase or perfectly antiphase among pairs of legs. Notably, however, the left front (LF) ThC joint MN activity is shifted with respect to the others (Figure 6*F*, top row, blue). This phase offset also propagated through the muscle model to shift LF ThC torques (Figure 6*F*, second row, blue), causing a premature movement of the LF leg—as indicated by its ThC joint angle (Figure 6*F*, third row, blue). Additionally, we asked to what extent tripod gaits might be robust against variations in intraleg coordination. For the 'trade-off' solution, although we observed that RF ThC and CTr joints showed normal coordination compared with that exhibited during walking (Figure 4*E*, top), the FTi joint was antiphase. Thus, although tripod coordination depend critically on interleg phase differences [57], it may be robust to intraleg coordination patterns. Interestingly, we observed naturalistic phase dynamics for the TiTa joint as well, even though its controlled angle was nominally fixed during these experiments. This suggests that while TiTa joint control attempted to maintain a constant position during walking, its movements were influenced by the dampening properties of muscles interacting with the environment. Finally, we explored how internal variables shift to favor each objective—speed and static stability—by comparing unilaterally similar solutions. When comparing solutions with similar speeds but different stabilities, we observed that for the more stable solution the coxae joints' torques were better synchronized to generate tripod-like gaits (Figure S6*B, C*, middle and right). On the other hand, solutions with similar stabilities but varying speeds, showed an increase in coxae joint torques for faster locomotion (Figure S6*B, C*, left and middle). In conclusion, NeuroMechFly can be used in a bottom-up manner to identify neuromuscular solutions to high-level objectives. Here we discovered that tripod-like locomotor gaits emerge when optimizing for speed and static stability, even in the absence of limb sensory feedback.

3 Discussion

NeuroMechFly is an open source simulation framework whose various modules—body morphology, controller, muscle model, and physics environment—constantly interact with one another allowing it to generate rich *in silico* behaviors (Figure 7). Depending on one's needs, we can envision several avenues for improving our model's biological interpretability. First, to increase biomechanical realism, (i) cuticular deformations might also be modeled by modifying the exoskeleton mesh in accordance with measurements of cuticular responses to mechanical stresses and strains (i.e. Young's modulus) [65, 66], and (ii) forces observed at small scales like Van der Waals and attractive capillary forces of footpad hairs [67] might also be modeled. Second, to increase the realism of movement control, (i) Hill-type muscle models with accurate dampening and nonlinear force generation properties might be applied based on species-specific physiological muscle properties—slack tendon lengths, attachment points, maximum forces, and pennation angles [55, 68], and (ii) controllers using integrate-and-fire, or Hodgkin-Huxley type neurons [14] might be used. These can also include sensory feedback (e.g., visual, gustatory, olfactory, and tactile) as well as central circuits for processing these signals. Third, to simulate behaviors more complex than locomotion on smooth terrain and grooming, (i) one can easily modify physical quantities like friction, gravity, and damping, (ii) introduce complex physical structures like heightfield terrains [53], (iii) and articulate wings, halteres, abdominal segments, proboscis, or antennae to model, for example, take-off [69], flight [70], odor plume following [71], courtship [24], or collective behavior [72].

Here we have illustrated how NeuroMechFly can be applied to two major neuroscience use cases. First, by replaying measured fly walking and grooming, we have shown how one can estimate joint torques and contact forces including ground-reaction forces, collisions, and tactile feedback. In the future, these estimates should be validated through direct force measurements [73, 74], or indirect recordings from proprioceptive and tactile neurons within the ventral nerve cord of behaving [36], or immobile [75] animals. Second, we have shown how high-level network and muscle parameter optimization can be used to discover and explore controller topologies that allow NeuroMechFly to generate fly-like behaviors. Any one of a number of approaches in a rapidly expanding machine

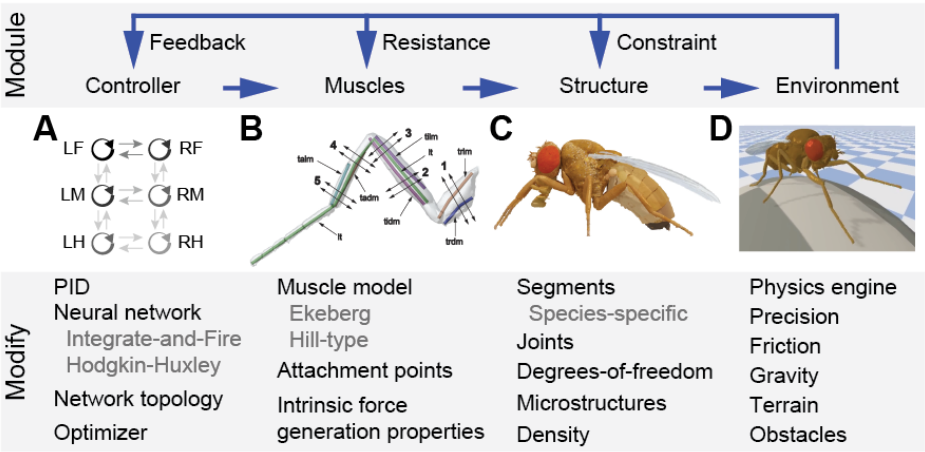

Figure 7: **Modules that can be independently modified in NeuroMechFly.** A controller's output drives muscles to move an embodied system. This system undergoes continuous interactions with the physical environment. Each of these modules can be independently modified within the NeuroMechFly simulation framework. **(A)** Controllers can range from abstract to physiologically realistic models. Optimization tools like Evolutionary Algorithms and Reinforcement Learning allow users to discover neuromuscular parameters that achieve high-level objectives. **(B)** Muscle contractions generate forces to drive joint movements. These dynamics depend on musculoskeletal geometry. Elaborate muscle models (e.g., Hill-type) require more parameters including muscle lengths and attachment points. **(C)** The morphology of the model—including its center-of-mass, joint DoFs, and mass distribution—shapes movement dynamics 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 like terrain complexity, friction, and gravity.

learning toolkit (e.g., reinforcement learning and evolutionary algorithms) may be applied to this goal. The topologies and dynamics of discovered networks might then be compared with connectomics data [37, 38] and neural recordings [36]. Although here we optimized for speed and static stability during tethered locomotion, NeuroMechFly can also locomote without body support, allowing one to optimize neural networks for free behaviors. Finally, our model can also be used to create animations of behaviors that improve the training of 3D pose estimation networks for applications in computer vision [46, 48].

Ultimately, we envision that NeuroMechFly can be applied to the synthesis of multimodal experimental data, the testing of theories for neuromechanical behavioral control, and the formulation of predictions for further experimental evaluation. Several of these applications might also be combined in real-time during closed-loop experimentation. For example, ongoing recordings of an animal's 3D poses might be replayed through NeuroMechFly to predict appendage collisions. These predictions might then direct the delivery of limb perturbations to test how proprioceptive feedback is used to achieve robust movement control. In summary, NeuroMechFly promises to accelerate our understanding of how passive biomechanics and active neural control give rise to organismic behavior, and can serve as a bridge linking fundamental biological discoveries to 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 [76], 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 1046x1636 pixels each.

4.1.3 Building a polygonal mesh volume from processed microtomography data

First, we isolated cuticle and wings from the microtomography data using Fiji [77]. 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 body and wings. Finally, we used the Lewiner marching cubes algorithm [40] (implemented in the scikit-image package [78]) to obtain a polygon 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 Blender (Foundation version 2.81 [79]) to clean and manipulate polygon meshes obtained from microtomography data.

After importing these meshes into Blender, we removed noise 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 3), we manually selected and deleted vertices from our imported 3D body and wing models. Segments were then separated at joint locations based on published morphological studies [80].

Each wing was separated into an individual segment from the wing model. The body model was separated into 63 segments as described below. The abdomen was divided into five segments according to tergite divisions. The first and second tergites were combined as the first segment (A1A2), 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 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 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 and body parts. For this, we used the more detailed microtomography data containing the right side of the fly. First, 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 approximately 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% for every segment. This resulted in a model with 87,000 vertices that conserved the most important details but eliminated some bristles and cuticular textures.

4.1.5 Rigging the Blender model

We added an Armature object alongside our model to build the skeleton of the fly. To actuate the model, we created a 'bone'—a tool in Blender that is used to animate characters—for each segment. Bones were created such that the thorax would be the root of the skeleton and each bone would be the child of its proximal bone, as indicated in Table 3. Then, the bones were positioned along the longitudinal axis of each segment with their heads and tails over the proximal and distal joints, respectively. Each joint was positioned at a location between neighboring segments. Each bone inherited the name of its corresponding mesh.

We used the Custom Properties feature in Blender to modify the properties of each bone. These properties can be used later in a simulator to e.g., define the maximum velocity, or maximum effort of each link. Furthermore, we added a limit rotation constraint (range of motion) to each axis of rotation (DoF) for every bone. The range of motion for each rotation axis per joint was defined as $-180°$ to $180°$ to achieve more biorealistic movements. Because, to the best of our knowledge, there are no reported angles for these variables, these ranges of motion should be further refined once relevant data become available. The DoF of each bone (segment) were based on previous studies [41, 70, 81] (see Table 3). 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, we defined a 'zero-position' for our model. Most bones were positioned in the direction of an axis of rotation (Figure S4). Each leg segment and the proboscis were positioned along the Z axis. Each abdominal segment and the labellum were positioned along the X axis. Wings, eyes, and halteres were positioned along the Y axis. The head and the antennae are the only bones not along a rotational axis: the head is rotated $20°$ along the Y axis, and the antennae are rotated $90°$ with respect to the head bone. Positioning the bones along axes of rotation makes it easier to intuit a segment's position with its angular information and also more effectively standardizes the direction of movements.

4.1.6 Exporting the Blender model into the Bullet simulation engine

We used a custom Python script in Blender to obtain the name, location, global rotation axis, range of motion, and custom properties for each bone. As mentioned above, the axes of rotation are defined locally for each bone. Therefore, our code also transforms this information from a local to a global reference system, obtaining the rotation matrix of each bone.

We used a Simulation Description Format (SDF, `http://sdformat.org/`) file to store the model's information. This format consists of an *.xml file that describes objects and environments in terms of their visualization and control. 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 meshes) 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 DoFs, we used hinge-type joints because they offer more freedom to control individual joint rotations. Therefore, for joints with more than 1 DoF, we positioned in a single location as many rotational joints as DoFs needed 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 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) measured in a previous study [64] 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 ($0.11\,\mu g$). In this way, the center of mass and the rigid-body dynamics of the model were made more similar to those of a real fly.

Since small-sized objects (e.g. less than $15\,cm$ in PyBullet) are susceptible to simulation errors due to computational numeric overflows, we scaled up the units of mass and length when setting up the physics of the simulation environment, and then scaled down the calculated values when logging the results. Therefore, the physics engine was able to compute the physical quantities without numerical errors, and the model could also more accurately reflect 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 of forward walking and foreleg/antennal grooming were taken from a previous study describing DeepFly3D, a deep learning-based 3D pose estimation tool [31]. These data consist of images from seven synchronized cameras obtained at 100 fps (`https://dataverse.harvard.edu/dataverse/DeepFly3D`). Time axes (Figure 4*E, F* and Figure 5*E, F*) correspond to time points from the original, published videos. Forward walking replay data were obtained from experiment #5 of animal (#8) expressing aDN-GAL4 only. Foreleg and antennal grooming replay data were obtained from experiment #3 of animal #6 expressing aDN-GAL4 driving UAS-CsChrimson. We used DeepFly3D v0.4 [31] 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 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 taken from different animals (see Figure S4).

Aligning both reference systems consisted of six steps. First, we defined the mean position of each Thorax-Coxa (ThC) keypoint as fixed joint locations. 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 according to its side (i.e., left or right) and the previously obtained orientations. Fifth, we scaled the real fly's leg lengths for each experiment to fit NeuroMechFly's body size. A scaling factor was calculated 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. Finally, we use the NeuroMechFly exoskeleton as a template to position all coxae within our global reference system; the exoskeleton has global location information for every joint. 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.

4.2.3 Calculating joint angles from 3D poses

We considered each leg as a kinematic chain and calculated the angle of each DoF to reproduce real poses in NeuroMechFly. We refer to this process 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 FTi joint 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 CTr joint. This was measured between the coxa and femur's longitudinal axis. Subsequently, we discovered that a CTr roll DoF would be required 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. 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 demonstrate that the base six DoFs were not sufficient for accurate kinematic replay, we also compared these results to angles obtained using inverse kinematics. In other words, we assessed whether an optimizer could find a set of angles that could precisely match our kinematic chain using only 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 kinematic replay, we used the forward kinematics method to compare original and the reconstructed poses. Since noise in 3D pose estimation causes leg segment lengths to vary, we first fixed the length of each segment as its mean length across all video frames.

We then calculated joint angles from 3D pose estimates with the addition of each DoF (see previous section). Next, we formed a new kinematic chain including the new DoF. This kinematic chain allowed us to compute forward kinematics from joint angles, which were then compared with 3D pose estimates to calculate an error. We performed an exhaustive search to find angles that minimize the overall distance between each 3D pose joint position and that joint's position as reconstructed using forward kinematics. The search spanned between $-90°$ to $90°$ from the 'zero pose' in $0.5°$ increments.

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

We ran a Kruskal-Wallis statistical test to compare the kinematic errors across the eight methods used. We then applied a posthoc Conover's test to perform a pairwise comparison. We used the Holm method to control for multiple comparisons. The resulting p-value matrices for walking and foreleg/antennal grooming behaviors are shown in Table 1 and Table 2, respectively. Our statistical tests suggested that adding a CTr roll DoF uniquely improved kinematic replay in comparison to all other methods.

4.2.5 Transferring 3D pose data into the NeuroMechFly reference frame

To incorporate the additional CTr roll DoF into NeuroMechFly, we enabled rotations along the Z axis of the CTr joints. Then, we recreated the SDF configuration files using custom Python scripts such that the resulting files would contain a CTr roll DoF for 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 equal to that of our real foam sphere ($10\,mm$). The simulated spherical treadmill has three joints along the x, y, and z axes, that allow the ball to rotate freely. The mass of each link was empirically set to $50\,\mu g$ which permitted the simulated fly to control the ball with its legs. We did not attempt to incorporate the mass of the real spherical treadmill because it is floating on an aerodynamically complex, stream of air.

We used position control for each joint in the model. We fixed the position of non-actuated joints to the values shown in Table 4. The actuated joints, i.e. the leg joints, were controlled to achieve the angles calculated from 3D pose data. The simulation was run with a time step of 1 ms, allowing PyBullet to accurately perform numerical calculations. Since the fly recordings were only captured at 100 fps, we up-sampled and interpolated pose estimates to match simulation time steps before calculating joint angles.

4.2.6 Application of external perturbations

To test the stability of the untethered model walking over flat ground, we introduced external perturbations. Specifically, we propelled solid spheres at the model according to 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 initial 3D position of the sphere, $\vec{u}$ is the initial velocity vector, $\vec{a}$ is the external acceleration vector 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 3 mg. Spheres were placed at a distance of 2 mm from the fly's center of mass in the y-direction. With t set to 20 ms, the initial velocity of the projectile was computed using Equation 1. The spheres were propelled at the model every 0.5 s. Finally, at 3 s into the simulation, a sphere with $3g$ was propelled 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 calculated from PyBullet. To infer ground reaction forces (GRFs), we computed and summed the magnitude of normal forces resulting from contact of each tarsal segment with the ball. Gait diagrams were generated by thresholding GRFs; a leg was considered 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 tibia and tarsal leg segments, as well as the antennae. We recorded normal forces generated by collisions between (i) the right and left front leg, (ii) the left front leg and left antenna, and (iii) the right front leg and right antenna. Grooming diagrams were calculated using the same rationale as for gait diagrams: a segment experienced a contact/collision if it reported a normal force greater than zero.

4.2.8 Controller gain sensitivity analysis

We achieved kinematic replay using a built-in PD position controller in PyBullet [53] that minimizes the error:

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

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—dynamics of motion—depend on the 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 numerous K_p and K_v pairs, ranging from 0.1 to 1.0 with a step size of 0.1 (i.e., 100 simulations in total). Target position and velocity signals for the controller were set as the calculated joint angles and the joint angular velocities, respectively. To compute the joint angular velocities, we used a Savitzky–Golay filter with a first-order derivative and a time-step of 1 ms on the joint angles. Feeding the controller with only the joint angles could also achieve the desired movements of the model. However, including the velocity signal ensured that the joint angular velocities of the fly and the simulation were properly matched. We then calculated the mean squared error (MSE) between the ground truth—joint angles obtained by running our kinematic replay pipeline on the pose estimates from DeepFly3D [31]—and joint angles obtained in PyBullet. Then, we averaged the MSE values across the joints in one leg, and summed up the mean MSEs from each of 6 legs to obtain a total error. We made the same calculations for the joint angular velocities as well. Our results (Figure S1) show that our biomechanical model can replicate real 3D poses while also closely

matching real measured velocities. In particular, an MSE of 18.7 rad^2/sec corresponds approximately to 3.1 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 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).

As shown in Figure S1, our model can match the real kinematics closely for almost every controller gain combination except for the low K_v band. By contrast, varying the gains proportionally increased the torque and force readings. Because there are no experimental data to validate these physical quantities, we selected gain values corresponding to intermediate joint torques and ground contact forces (Figure S2). Specifically, we chose 0.4 and 0.9 for K_p and K_v, respectively. These values were high enough to generate smooth movements, and low enough to reduce movement stiffness.

4.3 Neural network parameter optimization

4.3.1 CPG network architecture

To discover hexapodal locomotor gaits in our neuromechanical model, we designed a CPG-based controller composed of 36 nonlinear oscillators (Figure 6). A similar approach was previously applied to study salamander locomotion [59]. The CPG model is governed by the following system of differential equations:

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

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

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

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 faster convergence to a phase-locked regime between oscillators, we set coupling strengths to 1000 [82]. 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 three pairs of oscillators per leg, summing up to 36 in total. We only considered three DoFs per leg because this has been shown to be sufficient for locomotion in previous hexapod models [62]. Therefore, we selected the three joint DoFs that have the most pronounced joint angles (Figure S5). These DoFs included (i) ThC pitch for the front legs, (ii) ThC roll for the middle and hind legs, as well as (iii) CTr pitch and FTi pitch for all legs. 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), (iii) both front legs to each other, and (iv) coxa extensor and flexor oscillators. We visualized the resulting network architecture in an XML-based file format (graphml). Intraleg coordination is equally important to generate a fly-like gait since stance and swing periods depend on the intrasegmental phase relationships. For this reason, both interleg (phase relationships between ThC joints) and intraleg (phase relationships within each leg) couplings were determined by the optimizer for one side of the body and replicated on the other side.

4.3.2 Muscle model

We adapted the Ekeberg muscle model [60] to generate torques on the joints. This model simulates muscles as a torsional 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 [61]. $\Delta\varphi$ 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—$\alpha, \beta, \gamma, \delta$, and $\Delta\varphi$—which are determined by optimization.

4.3.3 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 *Drosophila melanogaster*. However, we did define joint limits during optimization to reduce search space dimensionality and to facilitate optimizer convergence. 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 mean angle of that joint $\pm$ one standard deviation. Using these constraints on joint angles, we created a new pose configuration (SDF) file. In addition, we fixed the position of the non-actuated joint DoFs as their mean joint angle during walking (Table 5). We used the same angles for kinematic replay of the remaining non-actuated joints (Table 4). Furthermore, we set the same limits and fixed positions for both sides of the body because the network is symmetrical: parameters for one side of the body are mirrored on the other side.

4.3.4 CPG network and muscle parameter optimization

To identify neuromuscular networks that could coordinate fast and statically stable locomotion, we performed optimization of the phase differences for each network connection, and five parameters controlling the gains and resting position of each spring and damper muscle (i.e., $\alpha, \beta, \gamma, \delta$, and $\Delta\varphi$). 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 parameters from the right joints to the left joints, and (iv) mirrored phase differences from ThC-ThC flexors to ThC-ThC extensors. Thus, 62 parameters remained flexible during optimization: 5 phases between ThC CPGs, 12 phases between intraleg CPGs (i.e. ThC-FTi extensor/flexor, FTi-TiTa extensor/flexor per leg), and 45 muscle parameters (five per joint). We empirically set the lower and upper bounds for the parameters so that leg movements would stay stable along the boundaries (Table 6).

For parameter optimization, we used an NSGA-II optimizer [83], a multi-objective genetic algorithm implemented in Python using the jMetalPy package [84]. We defined two objective functions. First, we aimed to maximize locomotor speed, as quantified by the distance traveled by the fly in a specific period of time. This, in turn, was quantified through ball rotations (Equation 7) along the Y axis. Second, we aimed to maximize the static stability of locomotor gaits. In small animals like *Drosophila*, static stability is a better approximation for overall stability than dynamic stability [64]. We measured static stability as follows. First, we identified a convex hull formed by the legs in stance phase. If there were less than 3 legs in stance and a convex hull could not be formed, the algorithm returned -1, indicating static instability. Then, we measured the closest distance between the fly's center of mass and the edges of the convex hull. Finally, we obtained the minimum of all measured distances at that time step. If the center of mass was outside the convex hull, we reversed the sign of the minimum distance to indicate instability. Because the optimizer works by minimizing objective functions, we inverted the sign of output distance and stability values.

Three penalties were added to the objective functions. First, to make sure the model was always moving, we set a threshold for the angular rotation of the ball, increasing from $-0.4\,rad$ to at least $4.0\,rad$ in 3 seconds, a value that will guarantee almost one revolution of the ball at the end of a simulation run. Second, to avoid high torque and velocities at each joint, we set joint angular

velocities to an upper limit of $100\,rad/sec$, a value measured from kinematic replay experiments. Third, because the most stable position for the model would be to have all of its legs on the ball, we penalized the amount of time spent by the legs in stance phase. If the average number of the legs in stance from the start of the simulation until that time step was less than 3 or larger than 3.6, we set the penalty function as the absolute difference between the average number of legs in stance and the minimum number of legs, i.e. $|\overline{n_{ms}} - 3|$.

The optimization was formulated as follows

$$\min \quad -10 \cdot R_b \cdot \theta_{b,\|} + 0.1 \cdot p_v + 100 \cdot p_s + 0.01 \cdot p_m \quad \text{(Distance \& penalties objective)} \tag{7}$$

$$\min \quad -0.1 \cdot s + 0.1 \cdot p_v + 100 \cdot p_s + 0.01 \cdot p_m \quad \text{(Stability \& penalties objective)} \tag{8}$$

with the following penalty terms

$$p_m^i = \begin{cases} p_t^{i-1} + 1 & \text{if} \quad \theta_{b,\|} \leq (\frac{t}{t_{total}} \cdot 4.40 - 0.40) \\ p_t^{i-1} & \text{otherwise} \end{cases} \quad \text{(Movement penalty)} \tag{9}$$

$$p_v^i = \begin{cases} p_v^{i-1} + 1 & \text{if} \quad \omega > 100 \quad rad/sec \\ p_v^{i-1} & \text{otherwise} \end{cases} \quad \text{(Angular velocity penalty)} \tag{10}$$

$$p_s^i = \begin{cases} 0 & \text{if} \quad 3 \leq \overline{n_{ms}} \leq 3.6 \\ |\overline{n_{ms}} - 3| & \text{otherwise} \end{cases} \quad \text{(Stance penalty)} \tag{11}$$

where R_b is the ball radius, $\theta_{b,\|}$ is the angle of the ball in the direction of walking, t_{tot} is the maximum simulation duration, and $\overline{n_{ms}}$ is the mean number of legs in stance phase. All the penalties are multiplied with their corresponding weights and added to the cost function. Cost functions were evaluated for 3 (t_{total}) seconds, a time that was sufficiently long for the model to generate locomotion.

Each optimization generation consisted of a population of 30 individuals. Optimization runs lasted for 60 generations, an acceptable computing time of approximately 2.5 hours per run on an Intel(R) Core(TM) i9-9900K CPU at 3.60GHz. Mutations occurred with a probability of 1.0, and a distribution index of 1.0. We set the cross-over probability to 1.0 and the distribution index to 15 (for more details see [84]). These values allowed variations within populations and more exploration of possible gaits while also not diverging too much from parent solutions.

4.3.5 Analysis of optimization results

Following optimization, we selected four individual solutions from the last generation for further analysis. First, the objective functions were normalized with respect to their maximum and minimum values. Then, solutions were selected as follows

$$\begin{aligned} \text{Longest distance traveled (fastest):} \quad & i = argmin(d_g) \\ \text{Shortest distance traveled (slowest):} \quad & i = argmax(d_g) \\ \text{Closest to the plot's center (intermediate):} \quad & i = argmin\Big(\sqrt{(d_g - 0.5)^2 + (s_g - 0.5)^2}\Big) \\ \text{Distance-Stability trade-off (trade-off):} \quad & i = argmin\Big(\sqrt{d_g^2 + s_g^2}\Big) \end{aligned}$$

Where d_g and s_s are the vectors containing the distance and stability values, respectively, from all individuals in a given generation g.

We read-out and plotted motor neuron activation patterns (as represented by the couple oscillators' outputs), joint torques, joint angles, GRFs, and ball rotations from this final generation of solutions. GRFs were used to generate gait diagrams as described in the previous sections. Ball rotations were used to reconstruct the models' paths across optimization generations. Briefly, the distances travelled along the longitudinal (x) and transverse (y) axes were calculated from the angular displacement of the ball according to the following formula

$$\Delta x = \Delta\theta_t r \qquad\qquad \Delta y = \Delta\theta_l r$$

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

5 Supplementary Tables

Table 1: Matrix of p-values from pairwise comparisons of position errors after calculating forward kinematics for walking. Numbers in bold (except in the case of identity) indicate p-value > 0.001 or no statistical difference.

	Base	**IK**	**Base & CTr roll**	**Base & CTr yaw**	**Base & FTi roll**	**Base & FTi yaw**	**Base & TiTa roll**	**Base & TiTa yaw**
Base	1.00	2.49e-24	0.00	1.32e-276	3.11e-190	5.24e-19	**9.95e-01**	2.05e-204
IK	2.49e-24	1.00	0.00	1.88e-143	1.66e-82	3.54e-19	2.49e-24	4.93e-92
Base & CTr roll	0.00	0.00	1.00	4.52e-229	0.00	0.00	0.00	8.37e-305
Base & CTr yaw	1.32e-276	1.88e-143	4.52e-229	1.00	7.86e-10	4.13e-61	1.54e-276	4.87e-07
Base & FTi roll	3.11e-190	1.66e-82	0.00	7.86e-10	1.00	3.82e-24	3.46e-190	**5.19e-01**
Base & FTi yaw	5.24e-83	3.54e-19	0.00	4.13e-61	3.82e-24	1.00	5.41e-83	2.59e-29
Base & TiTa roll	**9.95e-01**	2.49e-24	0.00	1.54e-276	3.46e-190	5.41e-83	1.00	2.31e-204
Base & TiTa yaw	2.05e-204	4.93e-92	8.37e-305	4.87e-07	**5.19e-01**	2.59e-29	2.31e-204	1.00

Table 2: Matrix of p-values matrix from pairwise comparisons of position errors after calculating forward kinematics for grooming. Numbers in bold (except in the case of identity) indicate p-value > 0.001 or no statistical difference.

	Base	**IK**	**Base & CTr roll**	**Base & CTr yaw**	**Base & FTi roll**	**Base & FTi yaw**	**Base & TiTa roll**	**Base & TiTa yaw**
Base	1.00	1.44e-152	0.00	6.93e-138	3.56e-153	3.59e-09	**1.00**	2.86e-119
IK	1.44e-152	1.00	0.00	**6.37e-01**	**1.00**	1.57e-190	6.01e-150	**9.36e-03**
Base & CTr roll	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00
Base & CTr yaw	6.93e-138	**6.37e-01**	0.00	1.00	**6.38e-01**	3.43e-79	2.26e-135	**3.49e-01**
Base & FTi roll	3.56e-153	**1.00**	0.00	**6.38e-01**	1.00	5.54e-91	1.51e-150	**8.99e-03**
Base & FTi yaw	3.59e-09	1.57e-190	0.00	3.43e-79	5.54e-91	1.00	1.43e-08	3.85e-65
Base & TiTa roll	**1.00**	6.01e-150	0.00	2.26e-135	1.51e-150	1.43e-08	1.00	6.22e-117
Base & TiTa yaw	2.86e-119	**9.36e-03**	0.00	**3.49e-01**	**8.99e-03**	3.85e-65	6.22e-117	1.00

Table 3: Model 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 4: Fixed angles for body joints during kinematic replay and optimization

Body part	**Joint**	**Fixed angle (deg)**
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 (deg)**
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 5: Fixed angles for leg joints during optimization (deg)

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 6: Lower and upper limits for the muscle parameters during optimization

Body part	**Joint**	α **[Lower limit, Upper limit]**	β **[Lower limit, Upper limit]**	γ **[Lower limit, Upper limit]**	δ **[Lower limit, Upper limit]**	$\Delta\varphi$ **[Lower limit, Upper limit]**
Front leg	ThC pitch	$[1 \times 10^{-3}, 7\times10^{-2}]$	$[1 \times 10^{-3}, 3\times10^{-3}]$	$[1 \times 10^{-3}, 1\times10^{-2}]$	$[1 \times 10^{-4}, 1\times10^{-3}]$	$[-0.22, 0.49]$
	CTr pitch					$[-2.50, -1.30]$
	FTi					$[0.76, 2.19]$
Middle leg	ThC pitch	$[1 \times 10^{-3}, 2\times10^{-1}]$	$[1 \times 10^{-3}, 3\times10^{-3}]$	$[1 \times 10^{-3}, 1\times10^{-2}]$	$[1 \times 10^{-4}, 3\times10^{-3}]$	$[-2.20, -1.75]$
	CTr pitch					$[-2.35, -1.84]$
	FTi					$[1.73, 2.63]$
Hind leg	ThC pitch	$[1 \times 10^{-3}, 3\times10^{-1}]$	$[1 \times 10^{-3}, 3\times10^{-3}]$	$[1 \times 10^{-3}, 1\times10^{-2}]$	$[1 \times 10^{-4}, 3\times10^{-3}]$	$[-2.78, -2.44]$
	CTr pitch	$[1 \times 10^{-3}, 2\times10^{-1}]$				$[-2.46, -1.31]$
	FTi					$[1.12, 2.79]$

6 Supplementary Figures

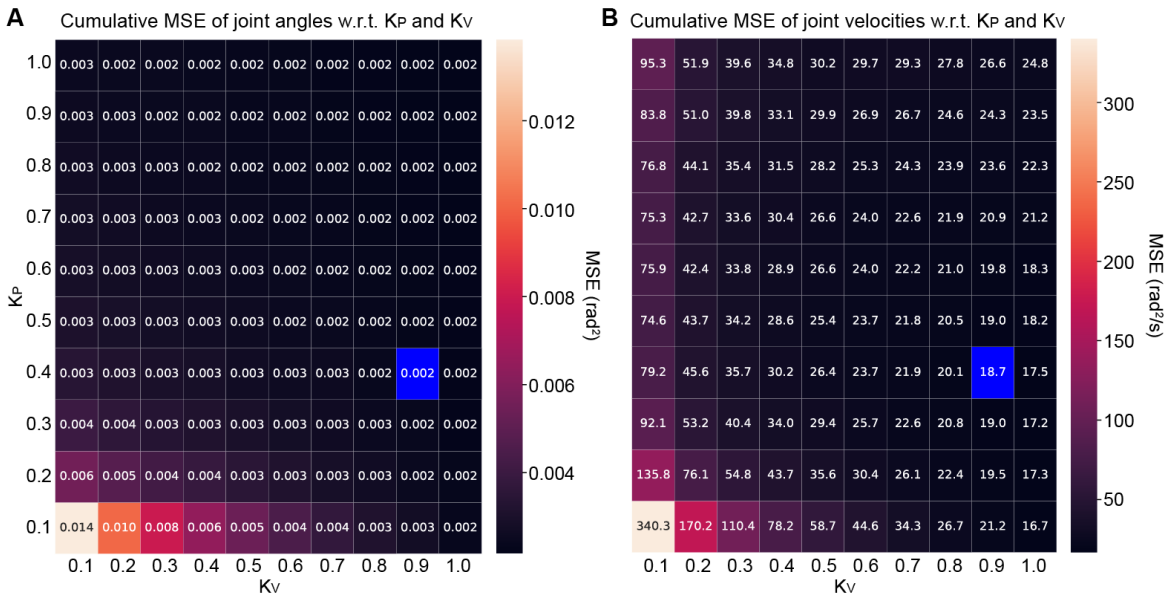

Figure S1: **Mean squared error between tracked and simulated joint positions and velocities as a function of position and velocity gain values.** Proportional gain (K_p) reduces the error in **(A)** joint angles and **(B)** joint velocities by acting on the error itself at the expense of increasing the stiffness of the system. Derivative gain (K_v) doesn't affect estimated errors as much as K_p. Chosen K_p and K_v values are indicated in blue. These values were high enough to keep the error sufficiently small and low enough to avoid stiffness and large variations in joint torques.

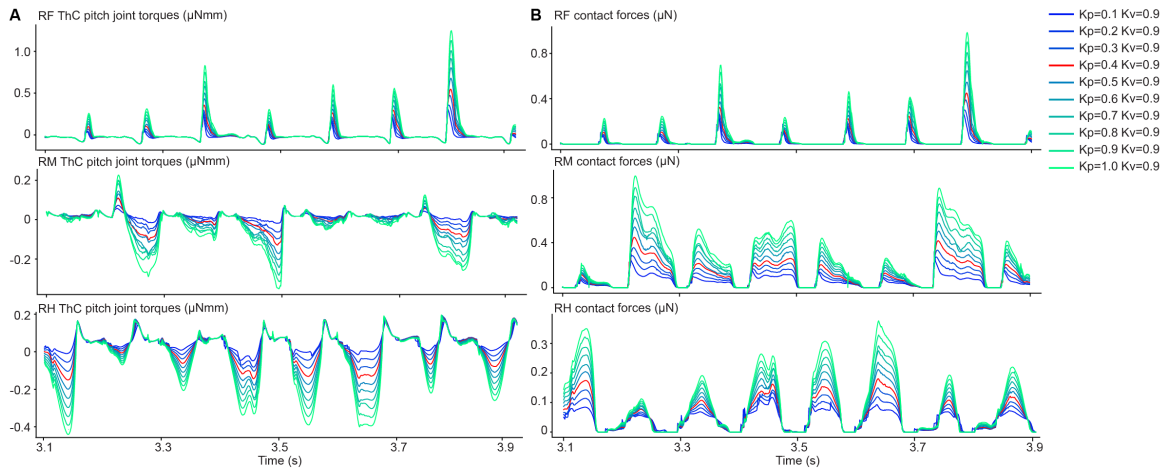

Figure S2: **Sensitivity of estimated torques and GRFs to proportional gain values. (A)** Estimated torque values during forward locomotion as a function of proportional gain values (K_p) while the velocity gain (K_v) is fixed at 0.9. Graphs show measurements for right leg ThC pitch DoFs. Measurements for the contralateral legs were nearly symmetrical. **(B)** Ground reaction force measurements of right side legs during forward walking as a function of varying K_p values. Results obtained from the selected K_p and K_v values are shown in red.

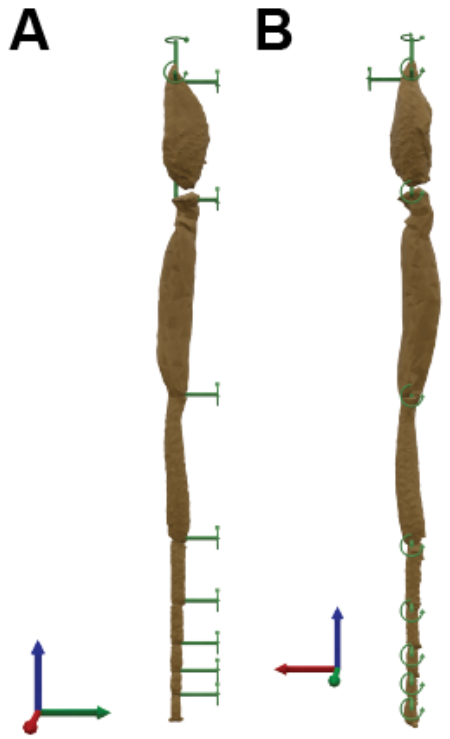

Figure S3: **Limb joint degrees-of-freedom and their rotational axes.** Each leg is composed of 11 hinge joints. Joints with more than one DoF were modeled as a union of multiple hinge joints. Front leg observed from **(A)** front and **(B)** side views. The global coordinate system's x, y, and z axes are red, green, and blue, respectively.

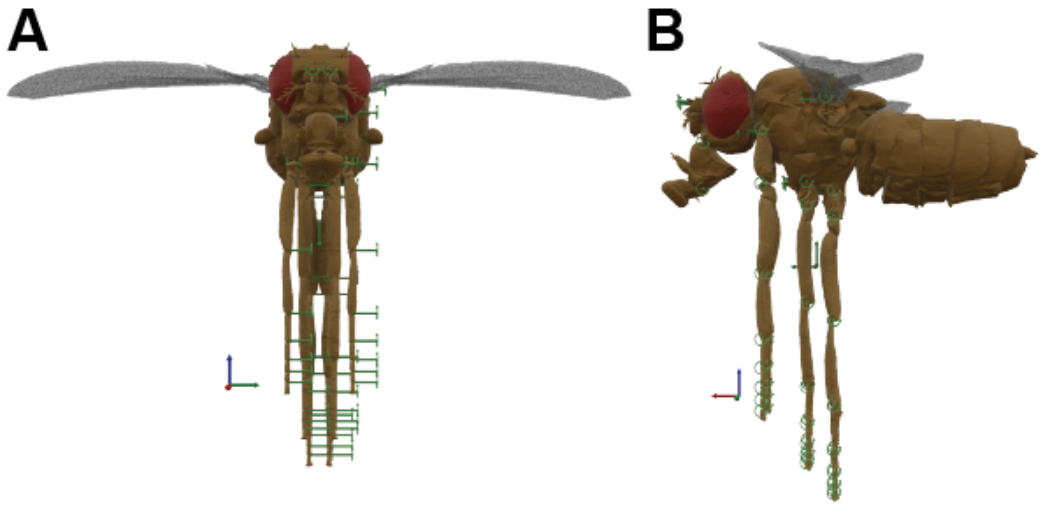

Figure S4: **The zero pose of NeuroMechFly.** Each body segment (Table 3) is aggregated using hinge joints. Rotational axes of joints and their global coordinate system are shown. **(A)** Zero pose from **(A)** front and **(B)** side views. The global coordinate system's x, y, and z axes are red, green, and blue, respectively.

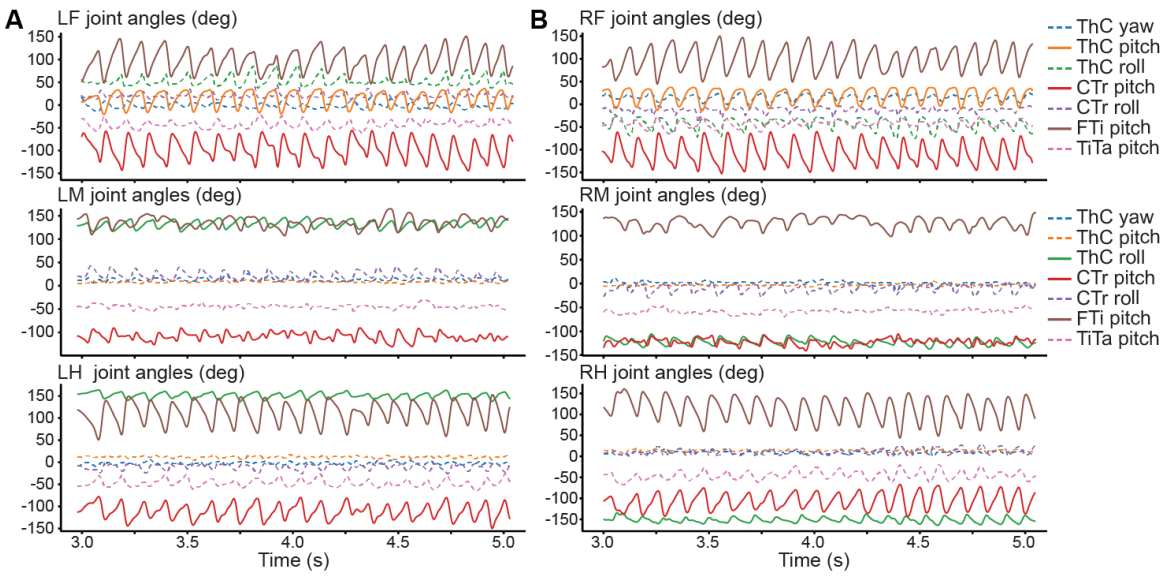

Figure S5: **Joint trajectories during forward locomotion.** Joint angles for the **(A)** left and **(B)** right legs measured from a real fly during forward walking. Only the three DoFs with the highest amplitudes (solid lines) were controlled during optimization—for the front legs: ThC pitch, CTr pitch, and FTi pitch; for the middle and hind legs: ThC roll, CTr pitch, and FTi pitch DoFs. The remaining four DoFs (dashed lines) for each leg did not have pronounced angular changes. Therefore, they were fixed to their mean values during optimization.

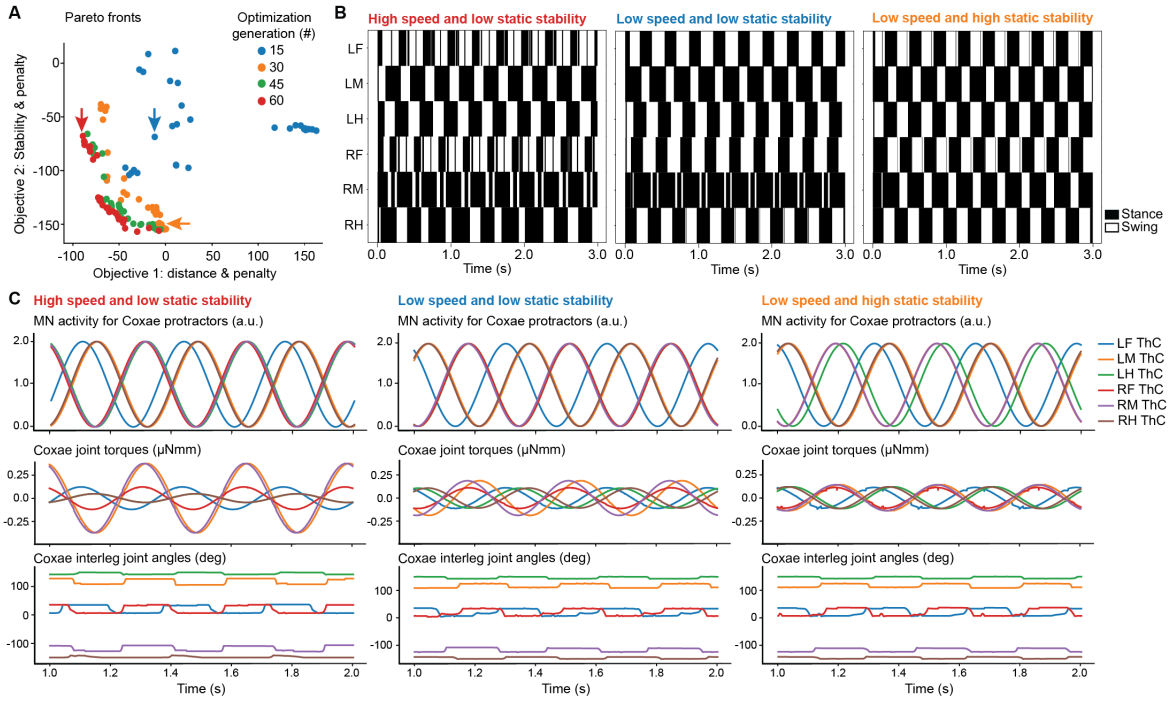

Figure S6: **Exploring the neuromuscular dynamics underlying fast and/or statically stable locomotor gaits resulting from optimization.** **(A)** The pareto front for solutions from four optimization generations. Later generations show more negative values because the optimizer aims to minimize the objective functions. Three individual solutions were selected for deeper analysis: two with similar stability values (indicated by red and blue arrows), and two with similar speed/distance values (indicated by blue and orange arrows). **(B)** Gait diagrams from selected individual solutions. Stance (black) and swing (white) phases were calculated by reading-out tarsus ground contacts for each leg. The more stable gaits exhibit tripod-like coordination. **(C)** Underlying neuromuscular dynamics for each individual solution. The fastest solution (left) and the more statically stable solution (middle) exhibit synchronized ThC joint torques (most leg pairs are either in phase or antiphase). Torque magnitudes varied considerably to achieve higher speeds (left) but were more constant for slower, statically stable gaits (right).

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 thresholded to separate the foreground from background. Then 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 DoFs that may be further be required to accurately replay real fly limb kinematics. The 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 the accuracy of replaying forward walking 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)** instead 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 row)** by adding a single new DoF (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 the accuracy of replaying foreleg/antennal grooming 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)** instead 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 row)** by adding a single new DoF (BaseDoF & 'joint' 'DoF'). Legs are color-coded.
https://www.dropbox.com/s/jq9g8tu6fm66eue/Video4.mp4?dl=0

Video 5: Kinematic replay of *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 0.2x 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 poses obtained by solving forward kinematics from joint angles (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 *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 0.2x 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 poses obtained by solving forward kinematics from joint angles (dashed lines). **(top-right, 'Kinematic replay')** These joint angles actuate NeuroMechFly limb movements as it grooms on a simulated spherical treadmill. Leg segments and antennal collisions are indicated (green). Estimated collision force vectors for the front legs and antennae 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 using NeuroMechFly behaving on flat terrain without body support. Pose estimates obtained from a real tethered fly walking on a spherical treadmill are replayed in NeuroMechFly as it walks untethered on flat terrain without body support. Simulated spheres are projected at the model to demonstrate its stability against perturbation.
https://www.dropbox.com/s/yug5bd0s22vpk13/Video7.mp4?dl=0

Video 8: Locomotor gaits improve across optimization generations. Forward walking gaits from four individual solutions shown across optimization generations 1, 15, 30, 45 and 60. Tarsal contacts with the ground are indicated (green). Videos are replayed at 0.3x real time. Solutions were selected from each generation according to the following criteria: the fastest / longest distance traveled (top-left), the closest to the center of the pareto front (top-right), the best trade-off to achieve both speed and static stability (bottom-left), and the slowest / shortest distance traveled (bottom-right).
https://www.dropbox.com/s/leutjkpj1xaey2b/Video8.mp4?dl=0

8 Code and data availability

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

9 Funding

PR acknowledges support from an SNSF Project grant (175667), and an SNSF Eccellenza grant (181239). 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 program under grant agreement nos. 720270 (SGA1), 785907 (SGA2). JA acknowledges support from the Human Frontier Science Program (HFSP-RGP0027/2017).

10 Acknowledgments

We thank Stéphanie Clerc Rosset and Graham Knott (Biological Electron Microscopy Facility, EPFL, Lausanne, Switzerland) for preparing *Drosophila melanogaster* samples for X-ray microtomography.

11 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.
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.

12 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* (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** (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** (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** (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** (2018).

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

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

[23] Seeds, A. M. *et al.* A suppression hierarchy among competing motor programs drives sequential grooming in Drosophila. *eLife* **3** (2014).

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

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

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

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

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

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

[32] Gosztolai, A. *et al.* LiftPose3D, a deep learning-based approach for transforming 2D to 3D pose in laboratory animals. *bioRxiv* (2020).

[33] Jenett, A. *et al.* A GAL4-driver line resource for Drosophila neurobiology. *Cell reports* **2**, 991–1001 (2012).

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

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

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

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

[38] Scheffer, L. K. *et al.* A Connectome and Analysis of the Adult Drosophila Central Brain. *bioRxiv* (2020).

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

[40] 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).

[41] 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).

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

[43] 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).

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

[45] 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).

[46] Li, S. *et al.* Deformation-aware Unpaired Image Translation for Pose Estimation on Laboratory Animals. *arXiv* (2020).

[47] 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).

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

[49] 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).

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

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

[52] 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).

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

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

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

[56] 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).

[57] 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).

[58] 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** (2020).

[59] 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).

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

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

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

[63] 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).

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

[66] 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).

[67] 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).

[68] Kuan, A. T. *et al.* Dense neuronal reconstruction through X-ray holographic nano-tomography. *Nature neuroscience* (2020).

[69] 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 (2019).

[70] Dickson, W. B., Straw, A. D. & Dickinson, M. H. Integrative Model of Drosophila Flight. *AIAA Journal* **46**, 2150–2164 (2008).

[71] 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).

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

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

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

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

[76] Chaffey, N. Principles and techniques of electron microscopy: biological applications (2001).

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

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

[79] Foundation, B. Blender-a 3D Modelling and Rendering Package (2012).

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

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

[82] 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).

[83] 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).

[84] 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** (2019).