ARTICLES
https://doi.org/10.1038/s41592-022-01466-7

nature methods

Check for updates

NeuroMechFly, a neuromechanical model of adult *Drosophila melanogaster*

Victor Lobato-Rios[1], Shravan Tata Ramalingasetty[2,3], Pembe Gizem Özdil[1,2,3], Jonathan Arreguit[2], Auke Jan Ijspeert[2] and Pavan Ramdya[1]✉

Animal behavior emerges from an interaction between neural network dynamics, musculoskeletal properties and the physical environment. Accessing and understanding the interplay between these elements requires the development of integrative and morphologically realistic neuromechanical simulations. Here we present NeuroMechFly, a data-driven model of the widely studied organism, *Drosophila melanogaster*. NeuroMechFly combines four independent computational modules: a physics-based simulation environment, a biomechanical exoskeleton, muscle models and neural network controllers. To enable use cases, we first define the minimum degrees of freedom of the leg from real three-dimensional kinematic measurements during walking and grooming. Then, we show how, by replaying these behaviors in the simulator, one can predict otherwise unmeasured torques and contact forces. Finally, we leverage NeuroMechFly's full neuromechanical capacity to discover neural networks and muscle parameters that drive locomotor gaits optimized for speed and stability. Thus, NeuroMechFly can increase our understanding of how behaviors emerge from interactions between complex neuromechanical systems and their physical surroundings.

It is daunting to uncouple the contributions of many neuronal and biomechanical elements to animal behavior. Systems-level numerical simulations can assist in this goal by consolidating data into a dynamic framework, generating predictions that can be tested, and probing the extent to which theories can account for experimental observations[1–6]. In particular, computational models have long played an important role in the study of movement control in vertebrates[7–10] and invertebrates[11–18].

For animals with a relatively small number of identifiable and genetically accessible neurons, a dialog between experimental results and computational predictions represents an exciting but largely unrealized opportunity. Neuromechanical models have already been developed for a number of model organisms[19–22]. The adult fly *Drosophila melanogaster* is an ideal organism for establishing a dialog between experimental and computational neuroscience. Flies generate a large repertoire of complex behaviors[23–28], the kinematics of which can now be precisely quantified[29–32]. Fly neurons can also be genetically targeted[33] for recordings, or perturbations during behavior[34–37]. These neurons can also be placed within their circuit context using connectomics data[38,39]. However, for the adult fly, existing models[40,41] have until now lacked the morphological accuracy needed to simulate mass distributions, compliance and physical constraints; the muscle models and their associated passive dynamical properties; and the neural networks or other control architectures.

Here we describe NeuroMechFly, a neuromechanical model of adult *Drosophila* that fills this methodological gap. NeuroMechFly is an open-source computational framework consisting of exchangeable modules that provide access to biomechanics, neuromuscular control, and parameter optimization approaches. These modules enable whole-organism simulations while also facilitating extensions and improvements by the scientific community. We obtained the model's biomechanical exoskeleton and defined the degrees of freedom (DoFs) of the leg by analyzing real three-dimensional (3D) leg kinematics (Fig. 1a), and thus identified a previously unreported DoF of the leg. We then inferred unmeasured ground reaction forces, joint torques and tactile contacts by replaying measured leg kinematics in this biomechanical simulation (Fig. 1b)[42]. Subsequently, we leveraged NeuroMechFly's full neuromechanical capacity by implementing a central pattern generator-inspired coupled-oscillator network and torsional spring and damper muscle model to discover controllers for fast and stable walking (Fig. 1c). These use cases illustrate how NeuroMechFly's modules (Fig. 1d) can be used to accelerate our understanding of how behaviors emerge from an interplay between neural dynamics, musculoskeletal biomechanics and physical interactions with the environment.

Results

Constructing a biomechanical model of adult *Drosophila*. To achieve a high level of morphological realism in our model, we performed an X-ray microtomography scan of an adult female fly (Supplementary Video 1). First, we embedded this animal in resin to reduce blurring associated with scanner movements (Fig. 2a). Then we processed the microtomography data (Fig. 2b), discriminating foreground (fly) from background (Fig. 2c). Finally, we generated a polygon mesh 3D reconstruction of the animal's exoskeleton (Fig. 2d).

We separated the body into 65 segments (Fig. 2e and Supplementary Table 1) and reassembled them into a natural resting pose. Joints were added to actuate the antennae, proboscis, head, wings, halteres, abdominal segments and leg segments. Leg articulation points were based on observations[31] and reported leg DoFs[43–45] (Fig. 2f and Supplementary Table 1). We confirmed that the model's legs are within the range of natural size variation (Extended Data Fig. 1). We then used hinge-type joints to connect body parts (Extended Data Fig. 2). Finally, we textured the model for visualization purposes (Fig. 2g). This entire process yielded a rigged model of adult *Drosophila* with the morphological accuracy required for

[1]Neuroengineering Laboratory, Brain Mind Institute and Institute of Bioengineering, EPFL, Lausanne, Switzerland. [2]Biorobotics Laboratory, EPFL, Lausanne, Switzerland. [3]These authors contributed equally: Shravan Tata Ramalingasetty, Pembe Gizem Özdil. ✉e-mail: pavan.ramdya@epfl.ch

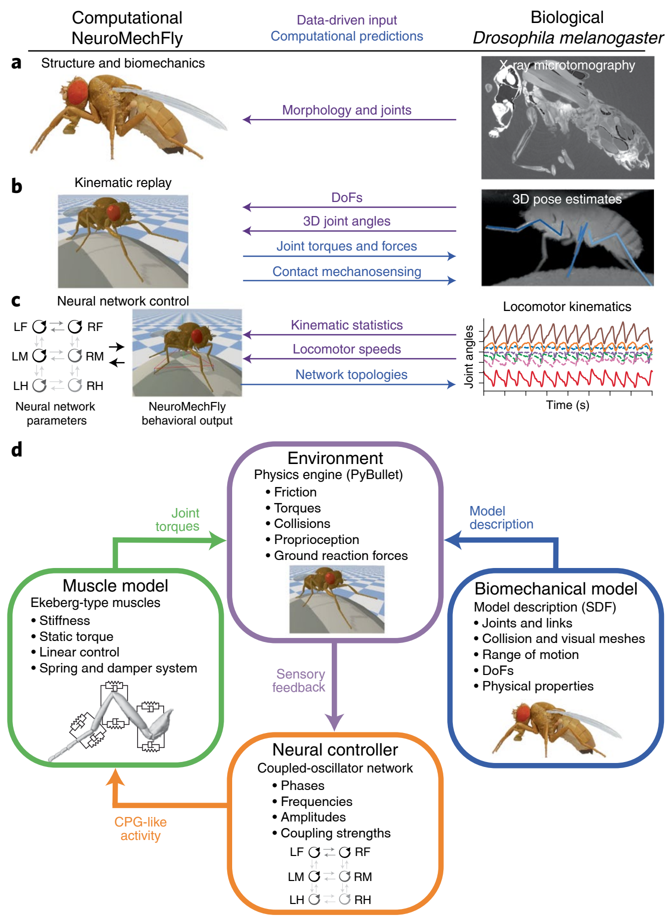


Fig. 1 | Data-driven development, applications, and modularity of NeuroMechFly. a, Body structures (that is, morphology and joint locations) were defined using X-ray microtomography and kinematic measurements. **b**, Real 3D poses were used to define the DoFs and replay kinematics in the model, permitting the prediction of unmeasured contact reaction forces and joint torques. **c**, Real limb kinematics were used to constrain the evolutionary optimization of neuromuscular parameters aiming to satisfy high-level objectives for fast and statically stable walking. The properties of optimized networks could then be more deeply analyzed. **d**, Modules that can be independently modified or replaced within NeuroMechFly. The controller generates neural-like activity to drive muscles. These muscles produce torques to operate a biomechanical model embedded in PyBullet's physics-based environment. When replacing any module it is necessary to preserve only the inputs and outputs (colored arrows). CPG, central pattern generator; LF, left front leg; LH, left hind leg; LM, left middle leg; RF, right front leg; RH, right hind leg; RM, right middle leg.

biomechanical studies and model-based computer vision tasks including pose estimation[46–50].

Identification of the minimum DoFs required to replay leg kinematics. We next asked whether the six reported and implemented leg DoFs, that is, thorax–coxa (ThC) elevation–depression, protraction–retraction, and rotation, coxa–trochanter (CTr) flexion–extension, femur–tibia (FTi) flexion–extension, and tibia–tarsus (TiTa) flexion–extension[43,44], would be sufficient to accurately replay experimentally measured 3D leg kinematics. We did not add a trochanter–femur (TrF) joint because, in *Drosophila*, the trochanter is thought to be fused to the femur[44]. For the middle and hind legs, ThC protraction–retraction occurs along a different axis from similarly named movements of the front legs. Therefore, we instead use 'roll', 'pitch' and 'yaw' to refer to rotations around the anterior–posterior, medial–lateral, and dorsal–ventral axes of articulated segments, respectively (Supplementary Video 2).

For our studies of leg kinematics, we focused on forward walking and grooming, two common *Drosophila* behaviors. First, we acquired 3D poses from recordings of tethered flies. Due to 3D pose estimation-related noise and some inter-animal morphological variability (Extended Data Fig. 1), directly actuating NeuroMechFly using raw 3D poses was impossible. To overcome this issue, we fixed the positions of base ThC joints and set each body part's length to its mean length for a given experiment. Then, we scaled relative ThC positions and body part lengths using our biomechanical model as a template. Thus, instead of using 3D Cartesian coordinates we could now calculate invariant joint angles that matched the DoFs used by NeuroMechFly.

When only these six DoFs were used to replay walking and grooming, we observed a large discrepancy between 3D pose-derived Cartesian joint locations and those computed from joint angles via forward kinematics (Fig. 3, base DoF dot product), including out-of-plane movements of the tibia and tarsus (Supplementary Video 3). Therefore, we looked for alternative leg configurations that would better match the 3D poses. First, we performed an inverse kinematics optimization of joint angles rather than dot product operations. This allowed us to identify angle configurations that minimize error at the distal tip of the kinematic chain: the pretarsus. Although inverse kinematics yielded a lower discrepancy (Fig. 3, base DoF inverse kinematics), we still observed consistent out-of-plane leg movements (Supplementary Video 3).

We next examined whether an extra DoF might be needed at the CTr joint to accurately replicate real fly leg movements. This was motivated by the fact that other insects use stabilizing rotations at or near the TrF joint[51–54], and *Drosophila* hosts reductor muscles of unknown function near the CTr joint[43]. To ensure that any improvements did not result simply from overfitting, we also tested the effect of adding one roll or yaw DoF to each of the more distal joints (CTr, FTi and TiTa) (Supplementary Video 2). Indeed, for both walking (Supplementary Video 3) and foreleg–antennal grooming (Supplementary Video 4), adding a CTr roll DoF to the six previously reported ('base') DoFs significantly and uniquely reduced the discrepancy between 3D pose-derived and forward kinematics-derived joint positions, even when compared with inverse kinematics (Fig. 3, base DoF and CTr roll; for statistical analysis see Supplementary Tables 2 and 3). This improvement was also evident on a joint-by-joint basis for walking (Extended Data Fig. 3a–d) and grooming (Extended Data Fig. 3e–h), and it was not achieved by any other kinematic chain tested, thereby arguing against overfitting (Fig. 3, base DoF and CTr yaw, base DoF and FTi roll, base DoF and FTi yaw, base DoF and TiTa roll, base DoF and TiTa yaw). These findings demonstrate that accurate kinematic replay of *Drosophila* leg movements requires seven DoFs per leg: the previously reported six DoFs[43,44] as well as a roll DoF near the CTr joint. Thus, by default, NeuroMechFly's biomechanical exoskeleton incorporates this additional DoF for each leg (Supplementary Table 1).

Estimation of joint torques and contact forces from kinematics. Having identified a suitable set of leg DoFs, we next tested the extent to which kinematic replay of real behaviors could be used to infer torques and contact forces such as body part collisions and ground reaction forces, that is, quantities that are technically challenging to measure in small insects[18,55]. We explored this possibility by using a proportional–derivative controller to actuate the model's leg joints, replaying measured leg kinematics during forward walking and foreleg–antennal grooming. Given that, when applying this kind of controller, there is no unique set of contact solutions that match forces and torques to prescribed kinematics, we first quantified how sensitive torque and force estimates were to changes in proportional–derivative

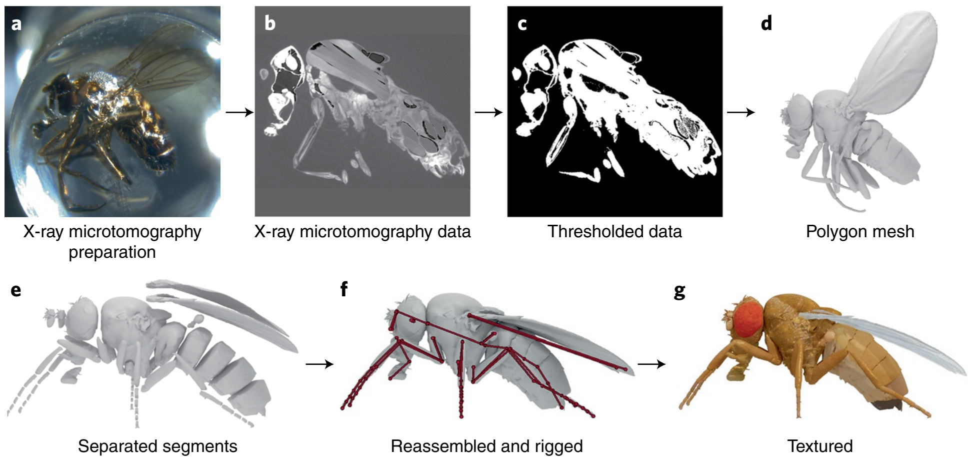


Fig. 2 | Constructing a data-driven biomechanical model of adult *Drosophila*. a, An adult female fly encased in resin for X-ray microtomography. **b**, Cross-section of the resulting X-ray scan. Cuticle, muscles, nervous tissues and internal organs are visible. **c**, Thresholded data separating the foreground (white) from the background (black). **d**, 3D polygon mesh of the exoskeleton and wings. **e**, Articulated body parts after separation from one another. **f**, Body parts after reassembly into a natural resting pose and overlaid with a rigged skeleton in dark red. **g**, Fly model after the addition of texture.

controller gains. We selected gain values that optimized the precision of kinematic replay (Extended Data Fig. 4a,b) and for which small deviations did not result in large variations in inferred physical quantities (Extended Data Fig. 4c–f). The model's 'zero pose' was selected to make joint angles intuitive (Extended Data Fig. 2a,b) and we included all seven leg DoFs (Extended Data Fig. 2c,d). We also fixed the orientation of abdominal segments, wings, halteres, head, proboscis and antennae to replicate a natural pose (Supplementary Table 4).

When we replayed walking (Fig. 4 and Supplementary Video 5) and grooming (Fig. 5 and Supplementary Video 6) we observed that the model's leg movements were largely similar to those that were experimentally measured. By quantifying real spherical treadmill rotations[56] and comparing them with simulated treadmill rotations for a range of soft constraint parameters (Extended Data Fig. 5), we observed high similarity between real and simulated spherical treadmill forward velocities and, to some extent, yaw velocities (Extended Data Fig. 6). This was notable given that the ball's rotations were not explicitly controlled but emerged from tarsal contacts and forces in the simulation. These observations support the accuracy of our computational pipeline in processing and replaying recorded joint positions.

Next, we more directly validated collisions and forces computed within the PyBullet physics-based simulation environment. We superimposed ground reaction force predictions on raw video recordings to visualize expected tarsal forces (Fig. 4d and Supplementary Video 5). These ground reaction forces, estimated from kinematic replay of joint angles during walking, closely tracked subtle differences in leg placement across walking cycles (Fig. 4e). Kinematic replay also allowed us to measure rich, periodic torque dynamics (Fig. 4e). Ground reaction force estimations could also be used to generate predicted gait diagrams during tethered walking (Fig. 4f). These predictions were accurate (83.5–87.3% overlap)

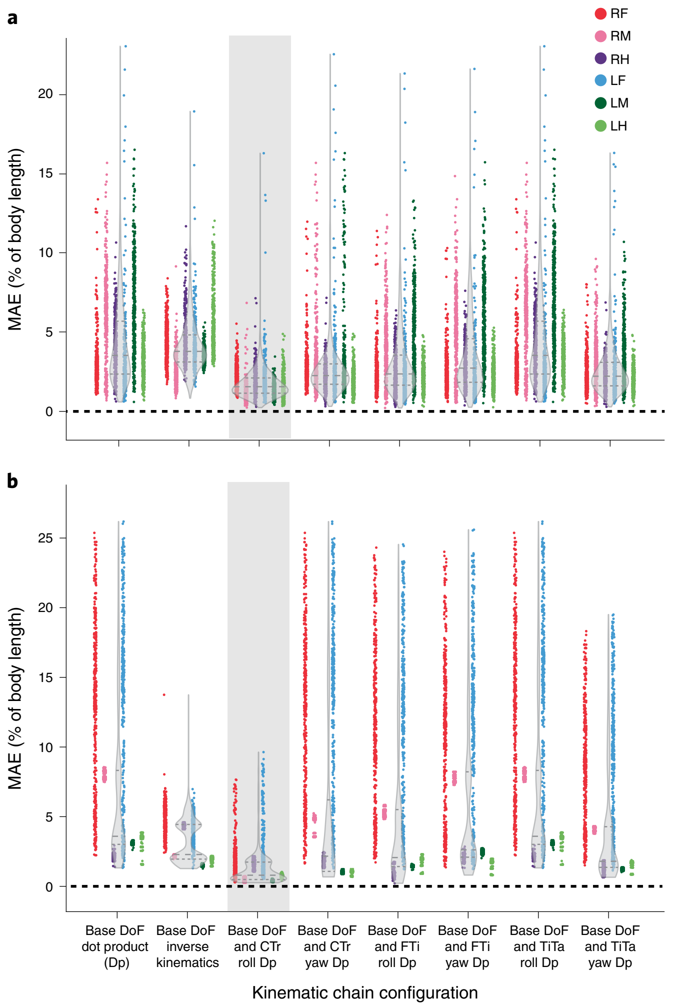


Fig. 3 | Adding a CTr roll DoF to base DoFs enables the most accurate kinematic replay of real walking and grooming. Body-length normalized mean absolute errors (MAE) comparing measured 3D poses and angle-derived joint positions for various DoF configurations in representative examples of forward walking (**a**) or foreleg–antennal grooming (**b**). For each condition, $n = 2,400$ samples were computed for all six legs (RF, RM, RH, LF, LM, LH) across 4 s of 100 Hz video data. Violin plots indicate the median (long dashed lines) and the upper and lower quartiles (short dashed lines). Results from adding a CTr roll DoF to base DoFs are highlighted in light gray. A Kruskal–Wallis statistical test with post-hoc Conover's test was used to compare the eight methods. The Holm method was used to control for multiple comparisons. The resulting *P* value matrices for walking and foreleg–antennal grooming behaviors are given in Supplementary Tables 2 and 3, respectively.

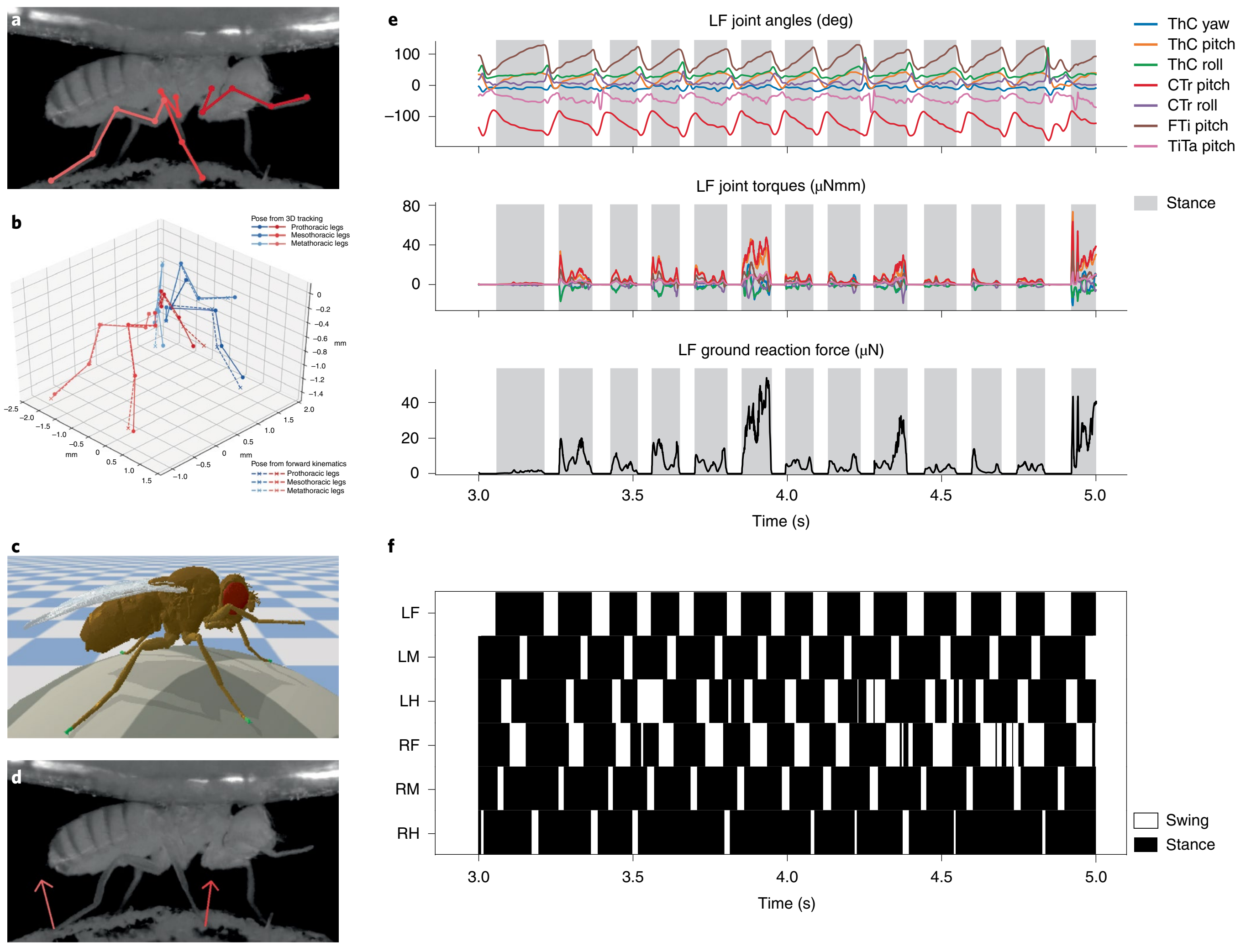


Fig. 4 | Kinematic replay of forward walking enables the estimation of ground contacts and reaction forces. **a**, Two-dimensional pose estimation of each leg joint for a tethered fly walking on a spherical treadmill. **b**, Comparison of tracked 3D poses (solid lines) and reconstructed poses derived from forward kinematics (dashed lines). **c**, Kinematic replay of measured joint angles during walking. Body segments in contact with the ground are green. **d**, Estimated ground reaction force vectors (red arrows) superimposed on original video data. **e**, Experimentally measured joint angles, estimated joint torques, and estimated ground reaction forces. Data for the left front leg (LF) are shown. Gray bars indicate stance phases when the leg is in contact with the ground. The joint DoFs are color-coded. **f**, Estimated gait diagram illustrating stance and swing phases for each leg.

when compared with manually labeled gait diagrams (Extended Data Fig. 7). We also used manual annotations to confirm that fly gaits were symmetric. This implies that animals were correctly tethered (Extended Data Fig. 7) and that errors in stance–swing classification, such as the 'dragging' of the right hind leg in Fig. 4f, probably result from 3D pose estimation noise.

Moreover, for foreleg–antennal grooming (Fig. 5a–c) we found that leg and antennal contact forces (Fig. 5d,e) reached magnitudes approximately threefold that of the fly's weight. This is within the range of previously measured maximum forces at the tip of the tibia (~100 µN) during ballistic movements[57]. Furthermore, measured joint angles gave rise to complex torque dynamics (Fig. 5e). Leg and antennal contact forces were used to generate grooming diagrams (akin to locomotor gait diagrams) that illustrate predicted contacts between distal leg segments and the antennae (Fig. 5f). Thus, collision data provide a rich description of grooming and can enable a precise physical quantification of many other behaviors. This approach emphasized the importance of having a morphologically accurate biomechanical model. When we replaced our computed tomography-based leg segments and antennae with more conventional stick segments, we observed less rich collision dynamics, such as the elimination of interactions between the tarsi and antennae (Extended Data Fig. 9 and Supplementary Video 7).

Finally, we asked to what extent our model might be able to walk without body support. We replayed 3D kinematics from tethered walking (Fig. 4 and Supplementary Video 5) freely on flat terrain. Our model walked stably on the ground (Supplementary Video 8) even in the presence of external perturbations (Supplementary Video 9). As expected, flat ground locomotion matched the velocities of tethered walking (Extended Data Fig. 8) better than walking trajectories (Supplementary Video 8), given that small deviations in heading direction yield large changes in trajectories.

In summary, we have shown how NeuroMechFly's biomechanical exoskeleton (without muscle or neuron models) can be used to replay real 3D poses and estimate otherwise inaccessible physical quantities such as joint torques, collisions and reaction forces.

Exploring locomotor control using neural and muscle models. Next, we used NeuroMechFly to discover neuromuscular controllers that optimize fast and statically stable tethered walking. Insect walking

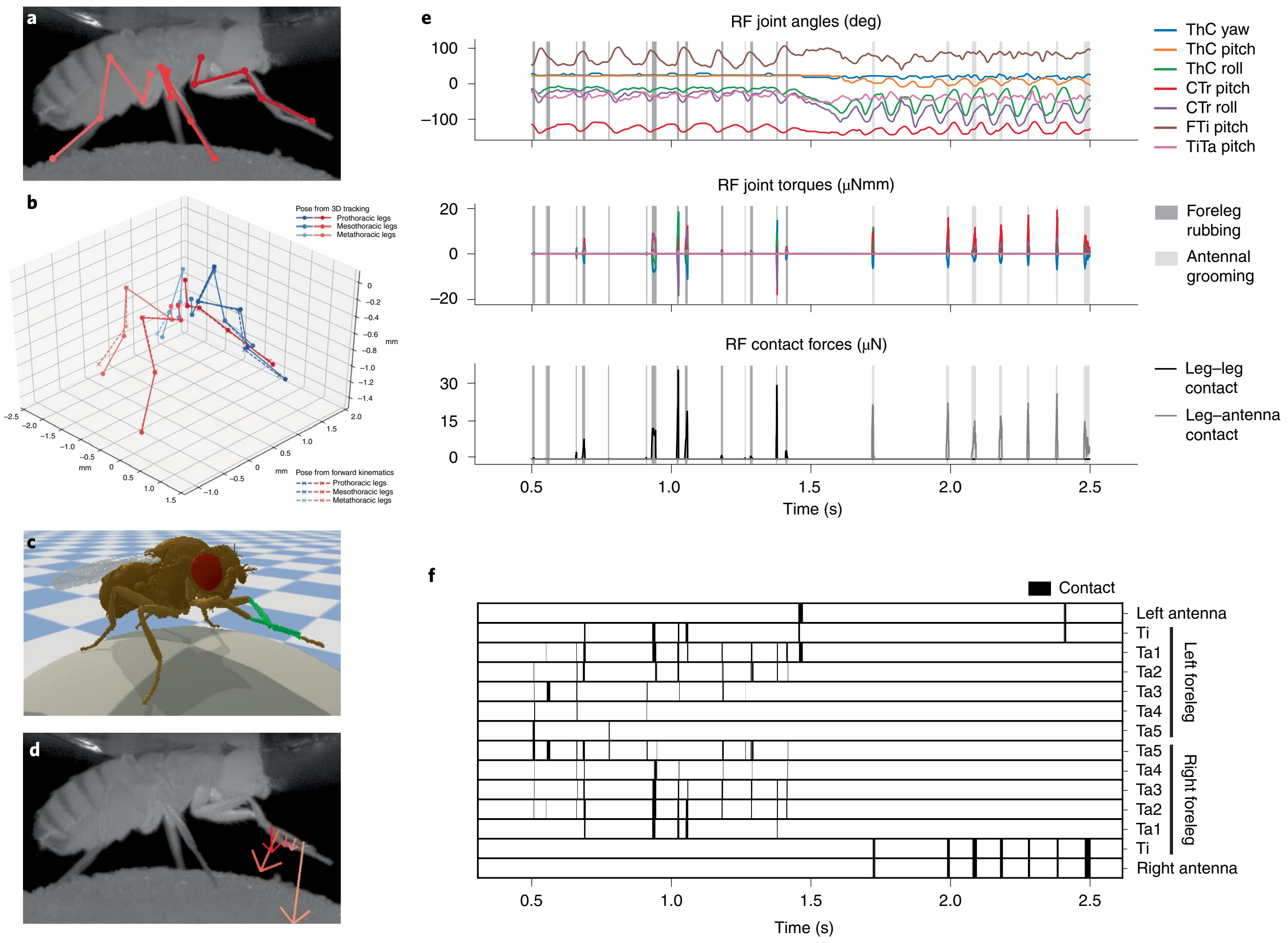


Fig. 5 | Kinematic replay enables the estimation of self-collisions and reaction forces during foreleg–antennal grooming. a, Two-dimensional pose estimation of each leg joint for a tethered fly grooming its forelegs and antennae. **b**, Comparison of tracked 3D poses (solid lines) and reconstructed poses derived from forward kinematics (dashed lines). **c**, Kinematic replay of measured joint angles during foreleg–antennal grooming. Body segments undergoing collisions are indicated in green. **d**, Estimated leg–leg and leg–antennae contact forces (red arrows) superimposed on original video data. **e**, Experimentally measured joint angles, estimated joint torques and estimated contact forces. Data for the right front leg (RF) are shown. The joint DoFs are color-coded. **f**, Estimated grooming diagram illustrating contacts made by both front legs' five tarsal segments (Ta1 and Ta5 being the most proximal and the most distal, respectively), tibia (Ti), and both antennae.

is thought to be controlled by networks of central pattern generators within the ventral nerve cord[15,16,58,59] although alternative, decentralized approaches have also been proposed[14,60]. Therefore, we designed a neural network controller consisting of a central pattern generator-like coupled oscillator[61] for each joint (Fig. 6a). For simplicity, we denote the output of each coupled oscillator as the activity of a central pattern generator. These central pattern generators, in turn, were connected to spring and damper muscles[62]. This simple muscle model has been used previously to effectively simulate locomotion[9,11,62].

We reduced the number of parameters and the search space by limiting neural control to leg DoFs that were sufficient to generate walking in other insect simulations[63], and that had the most pronounced effect on overall leg trajectories in our kinematic analysis of real flies (Extended Data Fig. 10a,b). Thus, we used CTr pitch and FTi pitch for all legs, as well as ThC pitch for the forelegs and ThC roll for the middle and hind legs. Coupled central pattern generators driving extensor and antagonistic flexor muscles controlled each DoF. We also mirrored optimization across the body to remove redundancy and reduce the optimization search space (Fig. 6a). To permit a wide range of joint movements, we set each central pattern generator's intrinsic frequency as an open parameter, the limits of which were constrained to frequencies measured during real walking[26,64] (Extended Data Fig. 10a,b).

We performed multiobjective optimization[65] using the NSGA-II (Non-dominated Sorting Genetic Algorithm II)[66] to identify neuromuscular parameters that drive walking gaits satisfying two high-level objectives: fast forward speed and high static stability. Forward speed was defined as the number of backward ball rotations within a fixed period of time (Fig. 6b). Static stability (that is, the stability of an animal's pose if, hypothetically, tested while immobile) was quantified as the minimum distance between the model's center of mass and the closest edge of the support polygon formed by the legs in stance phase (that is, in contact with the ground) (Fig. 6b). During optimization the Pareto front of best solutions evolved toward more negative values (Extended Data Fig. 10c), reflecting the increase in forward walking speed and stability over generations (Fig. 6c and Supplementary Video 10).

Next, we more deeply investigated three individual solutions from the final optimization generation: the fastest; the most stable; and a trade-off that was the best compromise between speed and static

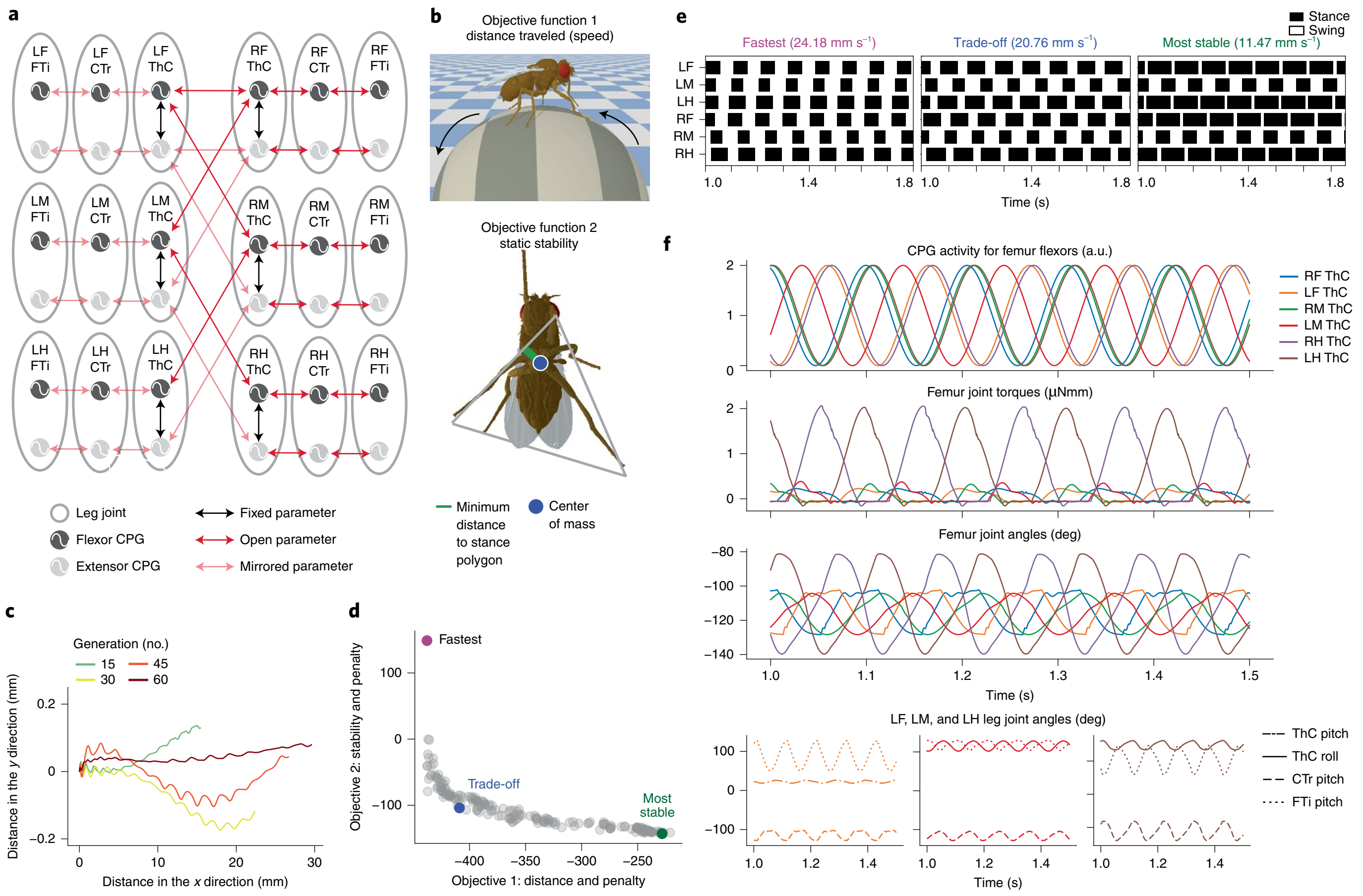


Fig. 6 | Evolutionary optimization identifies oscillator network and muscle parameters for fast and stable locomotion. a, Network of coupled oscillators modeling central pattern generator-based intra- and interleg circuits in the ventral nerve cord of *Drosophila*. Oscillator pairs control specific antagonistic leg DoFs (gray). Network parameter values are either fixed, modified during optimization or mirrored from oscillators on the other side of the body. **b**, Objective functions for the optimization of network and muscle parameters. Shown are the ball rotation during forward walking (objective function 1) and the support polygon formed by the legs in stance phase and the minimum distance between the fly's center of mass and its support polygon (objective function 2). **c**, Trajectories over 2 s from four trade-off solutions sampled across 60 optimization generations. **d**, The Pareto front of best solutions from generation 60. The colored dots represent the longest distance traveled ('fastest'), the solution having the smallest 2-norm of both objective functions after normalization ('trade-off'), and the most statically stable solution ('most stable'). **e**, Gait diagrams for selected solutions from generation 60 representing the stance and swing phases. **f**, Central pattern generator (CPG) outputs, joint torques and joint angles of each leg's femur for the trade-off solution. Intraleg joint angles for the LF, LM and LH are also shown.

stability (Fig. 6d). Although non-tripod gaits were observed across all generations (Extended Data Fig. 10d) even after the objectives were maximized and the penalties minimized at generation 60 (Extended Data Fig. 10e), the trade-off solution closely resembled the common insect tripod gait[26,67](Fig. 6e). This supports the notion that tripod locomotion satisfies a need for stability during fast insect walking[41,68].

We further analyzed how underlying neuromuscular quantities give rise to optimized locomotor gaits by focusing on the trade-off solution (Fig. 6f). As expected for a tripod gait, stance and swing phases of the left front and left hind legs were coordinated with those of the right middle leg. This coordination implies that the activities of the central pattern generator of the middle and hind legs are in phase with each other and phase-shifted by 180° with respect to the front leg (Fig. 6f). This is because, during stance phases, the front legs flex while the middle and hind legs extend. However, for the tripod generated by the other three legs, the central pattern generator activity of the left, middle femur was phase-shifted with respect to the right front and right hind legs (Fig. 6f). Torques were highest for the hind legs, suggesting an important role for driving ball rotations (Fig. 6f). Finally, we confirmed that the increase in hind leg torque was associated with a larger range of motion as measured by joint angles (Fig. 6f).

Discussion

Here, we have introduced NeuroMechFly, a computational model of adult *Drosophila* that can be used for biomechanical and neuromechanical studies. The biomechanical exoskeleton of NeuroMechFly can benefit from several future extensions. These include actuation of currently fixed body parts, increased joint compliance with stiffness and damping based on real measurements, cuticle structures with soft-bodied elements informed by measured responses to mechanical stresses and strains (that is Young's modulus)[69,70], and forces that are observed at small scales, including Van der Waals and attractive capillary forces of footpad hairs[71]. These and other physical estimates might be directly validated through force measurements[72,73], or indirectly by recordings of proprioceptive and tactile neurons[36,74].

In addition to its biomechanical exoskeleton, NeuroMechFly includes modules for neural controllers, muscle models and the physical environment (Fig. 1d). The biorealism of each of these modules can be independently improved (a tutorial can be found at https://nely-epfl.github.io/NeuroMechFly). First, more detailed neural controllers can be implemented (for example, Integrate-and-Fire, or Hodgkin–Huxley-type neurons) to facilitate the mapping of artificial neurons with their biological counterparts in connectomics[38,39] and functional[36] datasets. Second, to increase the realism of movement control, Hill-type muscle models that have non-linear force generation properties could be implemented based on species-specific muscle properties such as slack tendon lengths, attachment points, maximum forces and pennation angles[57,75]. Third, in our PyBullet framework[42] one might study behavior in more complex environments, for example, by introducing obstacles, or heightfield terrains.

In summary, NeuroMechFly can accelerate the investigation of how passive biomechanics and active neuromuscular control orchestrate animal behavior, and also serve as a bridge linking fundamental biological discoveries to applications in artificial intelligence and robotics.

Online content

Any methods, additional references, Nature Research reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at https://doi.org/10.1038/s41592-022-01466-7.

Received: 17 May 2021; Accepted: 23 March 2022;
Published online: 11 May 2022

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 Neurosci.* **20**, 553–557 (1997).
2. Webb, B. A framework for models of biological behaviour. *Int. J. Neural Syst.* **9**, 375–381 (1999).
3. Pearson, K., Ekeberg, Ö. & Büschges, A. Assessing sensory function in locomotor systems using neuro-mechanical simulations. *Trends Neurosci.* **29**, 625–631 (2006).
4. Prilutsky, B. I. & Edwards, D. H. (eds) *Neuromechanical Modeling of Posture and Locomotion* (Springer, 2015).
5. Seth, A. et al. OpenSim: simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement. *PLoS Comput. Biol.* **14**, e1006223 (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. *Ann. NY Acad. Sci.* **860**, 250–265 (1998).
8. Lansner, A., Hellgren Kotaleski, J. & Grillner, S. Modeling of the spinal neuronal circuitry underlying locomotion in a lower vertebrate. *Ann. NY Acad. Sci.* **860**, 239–249 (1998).
9. Ijspeert, A. J. A connectionist central pattern generator for the aquatic and terrestrial gaits of a simulated salamander. *Biol. Cybern.* **84**, 331–348 (2001).
10. 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 (1,2,3). *eNeuro* **2**, ENEURO.0069-15.2015 (2015).
11. Ekeberg, Ö., Blümel, M. & Büschges, A. Dynamic simulation of insect walking. *Arthropod Struct. Dev.* **33**, 287–300 (2004).
12. Toth, T. I., Schmidt, J., Büschges, A. & Daun-Gruhn, S. A neuro-mechanical model of a single leg joint highlighting the basic physiological role of fast and slow muscle fibres of an insect muscle system. *PLoS ONE* **8**, e78247 (2013).
13. Toth, T. I., Grabowska, M., Schmidt, J., Büschges, A. & Daun-Gruhn, S. A neuro-mechanical model explaining the physiological role of fast and slow muscle fibres at stop and start of stepping of an insect leg. *PLoS ONE* **8**, e78246 (2013).
14. Schilling, M., Hoinville, T., Schmitz, J. & Cruse, H. Walknet, a bio-inspired controller for hexapod walking. *Biol. Cybern.* **107**, 397–419 (2013).
15. 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*. *Biol. Cybern.* **108**, 1–21 (2014).
16. Proctor, J., Kukillaya, R. & Holmes, P. A phase-reduced neuro-mechanical model for insect locomotion: feed-forward stability and proprioceptive feedback. *Philos. Trans. A Math. Phys. Eng. Sci.* **368**, 5087–5104 (2010).
17. 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. *Bioinspir. Biomim.* **10**, 065005 (2015).
18. Guo, S., Lin, J., Wöhrl, T. & Liao, M. A neuro-musculo-skeletal model for insects with data-driven optimization. *Sci. Rep.* **8**, 2129 (2018).
19. Szigeti, B. et al. Openworm: an open-science approach to modeling *Caenorhabditis elegans*. *Front. Comput. Neurosci.* **8**, 137 (2014).
20. Izquierdo, E. J. & Beer, R. D. From head to tail: a neuromechanical model of forward locomotion in *Caenorhabditis elegans*. *Philos. Trans. R. Soc. Lond. B Biol. Sci.* **373**, 1758 (2018).
21. Loveless, J., Lagogiannis, K. & Webb, B. Modelling the mechanics of exploration in larval *Drosophila*. *PLoS Comput. Biol.* **15**, e1006635 (2019).
22. Merel, J. et al. Deep neuroethology of a virtual rodent. In *Proceedings of the International Conference on Learning Representations* https://openreview.net/forum?id=SyxrxR4KPS (2020).
23. Seeds, A. M. et al. A suppression hierarchy among competing motor programs drives sequential grooming in *Drosophila*. *eLife* **3**, e02951 (2014).
24. Pavlou, H. J. & Goodwin, S. F. Courtship behavior in *Drosophila melanogaster*: towards a 'courtship connectome'. *Curr. Opin. Neurobiol.* **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. et al. Quantification of gait parameters in freely walking wild type and sensory deprived *Drosophila melanogaster*. *eLife* **2**, e00231 (2013).
27. Wosnitza, A. et al. Inter-leg coordination in the control of walking speed in *Drosophila*. *J. Exp. Biol.* **216**, 480–491 (2013).
28. Pick, S. & Strauss, R. Goal-driven behavioral adaptations in gap-climbing *Drosophila*. *Curr. Biol.* **15**, 1473–1478 (2005).
29. Pereira, T. D. et al. Fast animal pose estimation using deep neural networks. *Nat. Methods* **16**, 117–125 (2019).
30. Mathis, A. et al. DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. *Nat. Neurosci.* **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**, e48571 (2019).
32. Gosztolai, A. et al. LiftPose3D, a deep learning-based approach for transforming two-dimensional to three-dimensional poses in laboratory animals. *Nat. Methods* **18**, 975–981 (2021).
33. Jenett, A. et al. A GAL4-driver line resource for *Drosophila* neurobiology. *Cell Rep.* **2**, 991–1001 (2012).
34. Seelig, J. D. et al. Two-photon calcium imaging from head-fixed *Drosophila* during optomotor walking behavior. *Nat. Methods* **7**, 535–540 (2010).
35. Maimon, G., Straw, A. D. & Dickinson, M. H. Active flight increases the gain of visual motion processing in *Drosophila*. *Nat. Neurosci.* **13**, 393–399 (2010).
36. Chen, C. et al. Imaging neural activity in the ventral nerve cord of behaving adult *Drosophila*. *Nat. Commun.* **9**, 4390 (2018).
37. Hermans, L. et al. Long-term imaging of the ventral nerve cord in behaving adult *Drosophila*. Preprint at https://doi.org/10.1101/2021.10.15.463778v1 (2021).
38. Phelps, J. S. et al. Reconstruction of motor control circuits in adult *Drosophila* using automated transmission electron microscopy. *Cell* **184**, 759–774 (2021).
39. Scheffer, L. K. et al. A connectome and analysis of the adult *Drosophila* central brain. *eLife* **9**, e57443 (2020).
40. Isakov, A. et al. Recovery of locomotion after injury in *Drosophila melanogaster* depends on proprioception. *J. Exp. Biol.* **219**, 1760–1771 (2016).
41. Ramdya, P. et al. Climbing favours the tripod gait over alternative faster insect gaits. *Nat. Commun.* **8**, 14494 (2017).
42. Coumans, E. Bullet physics simulation. In *ACM SIGGRAPH 2015 Courses* https://doi.org/10.1145/2776880.2792704 (2015).
43. 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).
44. Sink, H. *Muscle Development in* Drosophila (Springer, 2006).
45. Cruse, H., Dürr, V. & Schmitz, J. Insect walking is based on a decentralized architecture revealing a simple and robust controller. *Philos. Trans. A Math. Phys. Eng. Sci.* **365**, 221–250 (2007).
46. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G. & Black, M. J. SMPL: a skinned multi-person linear model. *ACM Trans. Graphics* **34**, 248 (2015).
47. 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).
48. Li, S. et al. Deformation-aware unpaired image translation for pose estimation on laboratory animals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* 13158–13168 (2020).
49. 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).

50. Bolaños, L. A. et al. A three-dimensional virtual mouse generates synthetic training data for behavioral analysis. *Nat. Methods* **18**, 378–381 (2021).
51. Watson, J. T., Ritzmann, R. E., Zill, S. N. & Pollack, A. J. Control of obstacle climbing in the cockroach, *Blaberus discoidalis*. I. Kinematics. *J. Comp. Physiol. A Neuroethol. Sens. Neural Behav. Physiol.* **188**, 39–53 (2002).
52. Frantsevich, L. & Wang, W. Gimbals in the insect leg. *Arthropod Struct. Dev.* **38**, 16–30 (2009).
53. Bender, J. A., Simpson, E. M. & Ritzmann, R. E. Computer-assisted 3D kinematic analysis of all leg joints in walking insects. *PLoS ONE* **5**, e13617 (2010).
54. Zill, S. N. et al. Effects of force detecting sense organs on muscle synergies are correlated with their response properties. *Arthropod Struct. Dev.* **46**, 564–578 (2017).
55. Cofer, D., Cymbalyuk, G., Heitler, W. J. & Edwards, D. H. Neuromechanical simulation of the locust jump. *J. Exp. Biol.* **213**, 1060–1068 (2010).
56. Moore, R. J. D. et al. Fictrac: a visual method for tracking spherical motion and generating fictive animal paths. *J. Neurosci. Methods* **225**, 106–119 (2014).
57. Azevedo, A. W. et al. A size principle for recruitment of *Drosophila* leg motor neurons. *eLife* **9**, e56754 (2020).
58. Fuchs, E., Holmes, P., Kiemel, T. & Ayali, A. Intersegmental coordination of cockroach locomotion: adaptive control of centrally coupled pattern generator circuits. *Front. Neural Circuits* **4**, 125 (2011).
59. Mantziaris, C. et al. Intra-and intersegmental influences among central pattern generating networks in the walking system of the stick insect. *J. Neurophysiol.* **118**, 2296–2310 (2017).
60. Schilling, M. & Cruse, H. Decentralized control of insect walking: a simple neural network explains a wide range of behavioral and neurophysiological results. *PLoS Comput. Biol.* **16**, e1007804 (2020).
61. Ijspeert, A. J., Crespi, A., Ryczko, D. & Cabelguen, J. From swimming to walking with a salamander robot driven by a spinal cord model. *Science* **315**, 1416–1420 (2007).
62. Ekeberg, Ö. A combined neuronal and mechanical model of fish swimming. *Biol. Cybern.* **69**, 363–374 (1993).
63. Daun-Gruhn, S. A mathematical modeling study of inter-segmental coordination during stick insect walking. *J. Comput. Neurosci.* **30**, 255–278 (2011).
64. DeAngelis, B. D., Zavatone-Veth, J. A. & Clark, D. A. The manifold structure of limb coordination in walking *Drosophila*. *eLife* **8**, e46409 (2019).
65. Oliveira, M. et al. 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).
66. Deb, K., Pratap, A., Agarwal, S. & Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evolutionary Computation* **6**, 182–197 (2002).
67. Strauss, R. & Heisenberg, M. Coordination of legs during straight walking and turning in *Drosophila melanogaster*. *J. Comp. Physiol. A* **167**, 403–412 (1990).
68. Szczecinski, N. S., Bockemühl, T., Chockley, A. S. & Büschges, A. Static stability predicts the continuum of interleg coordination patterns in *Drosophila*. *J. Exp. Biol.* **221**, jeb189142 (2018).
69. Vincent, J. F. & Wegst, U. G. Design and mechanical properties of insect cuticle. *Arthropod Struct. Dev.* **33**, 187–199 (2004).
70. Flynn, P. C. & Kaufman, W. R. Mechanical properties of the cuticle of the tick *Amblyomma hebraeum* (Acari: Ixodidae). *J. Exp. Biol.* **218**, 2806–2814 (2015).
71. Kimura, K., Minami, R., Yamahama, Y., Hariyama, T. & Hosoda, N. Framework with cytoskeletal actin filaments forming insect footpad hairs inspires biomimetic adhesive device design. *Commun. Biol.* **3**, 272 (2020).
72. 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).
73. Elliott, C. J. & Sparrow, J. C. In vivo measurement of muscle output in intact *Drosophila*. *Methods* **56**, 78–86 (2012).
74. Mamiya, A., Gurung, P. & Tuthill, J. C. Neural coding of leg proprioception in *Drosophila*. *Neuron* **100**, 636–650 (2018).
75. Kuan, A. T. et al. Dense neuronal reconstruction through X-ray holographic nano-tomography. *Nat. Neurosci.* **23**, 1637–1643 (2020).

Methods

Constructing an adult *Drosophila* biomechanical model. *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 although 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 anesthetized with CO_2 gas and then carefully immersed in a solution of 2% paraformaldehyde in phosphate buffer (0.1 M, 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 that the flies did not float on the surface but instead remained just below the meniscus. They were then washed in 0.1 M cacodylate buffer (two 3 min washes), and placed in 1% osmium tetroxide in 0.1 M 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) for 24 h at 21°C. This was polymerized for 24 h at 60 °C inside a closed gelatin capsule (size 1; Electron Microscopy Sciences) half-filled with previously hardened resin to ensure that the insect was situated in the center of the final resin block and away from the side.

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). The X-ray voltage was set to 40 kV and the current was set to 112 µA. The voxel size was 0.00327683 mm. To perform the reconstruction we used X-Act software from the microtomography system developer (RX-solutions) to obtain a stack of 982 tiff images of 1,046 × 1,636 pixels each.

Building a polygonal mesh 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 beginning at slice 300. The tiff stack with the region of interest 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[78] (implemented in the scikit image package[79]) to obtain a polygon mesh for each stack. Both meshes were then exported to a standard compressed mesh storage format.

Separating and reassembling articulated body parts. We used Blender (Foundation version 2.81, ref. [80]) 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. We obtained 65 body segments (Supplementary Table 1) based on ref. [81]. More recent literature corroborated these propositions for body morphology and joint DoFs. 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. We made some simplifications. Most notably, in the antennae, we considered only one segment instead of three because cutting this small element into a few pieces would alter its morphology.

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 division. 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 as 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 into 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 they had 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 resulting from the microtomography preparation or from separating body segments.

All segments were then copied into a single *.blend file and rearranged into a natural resting pose (Fig. 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 9 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 small bristles and cuticular textures.

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 Supplementary Table 1. The bones were then 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, for example, to 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 DoFs of each bone (segment) were based on previous studies[43,82,83] (Supplementary Table 1). Any bone can be rotated in Blender to observe the constraints imposed upon each axis of rotation. These axes are defined locally for each bone.

Finally, we defined a 'zero position' for our model. Most bones were positioned in the direction of an axis of rotation (Extended Data Fig. 2). 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 to standardize the direction of movements more effectively.

Exporting the Blender model into the 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 for each bone.

We used the Simulation Description Format (SDF, http://sdformat.org/) convention 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 and joints, or to modify features of existing segments and joints. To implement joint DoFs, we used hinge-type joints because they offer more freedom to control individual rotations. Therefore, for joints with more than one DoF, we positioned in a single location as many rotational joints as the number of DoFs needed to describe its movement. The parenting hierarchy for 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 the 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 mg, respectively. To make the center of mass and the rigid-body dynamics of the model more similar to a real fly, we used different masses (densities) for certain body parts instead of having a homogeneous mass distribution. Specifically, the mass of the head was 0.125 mg, the mass of the thorax was 0.31 mg, the mass of the abdomen was 0.45 mg, that of the wings was 0.005 mg and that of the legs was 0.11 mg (ref. [68]).

In PyBullet, contacts are modeled based on penetration depth between any two interacting bodies. The contact parameters are set to 0.02 units of length (1 unit = 1 m in SI units). It is preferable to have bodies larger than 0.02 units. Therefore, we performed dynamic scaling to rescale the model, the physical units, and quantities such as gravity while preserving the dynamics and improving the numerical stability of the model. Notably, we are not compromising the dynamics of the simulated behaviors. Specifically, 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 recording 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.

Comparing leg sizes between NeuroMechFly and real flies. We dissected the right legs from 10 wild-type female adult flies, 2–4 days after eclosion. Flies were cold anesthetized using ice. The legs were then removed using forceps from the sternal cuticle to avoid damaging the coxae. Dissected legs were straightened onto a glass slide and fixed with ultraviolet-curable glue (Extended Data Fig. 1a). We used a Leica M205 C stereo microscope to take images from the legs placed next to a 0.5 mm graduated ruler. Joints in the legs were manually annotated and then distances between them were measured in pixels and converted to mm using the ruler as a reference. Lengths between joints were compared with rigged bone lengths in NeuroMechFly.

Most leg segments in NeuroMechFly were within the range of natural size variations (Extended Data Fig. 1b). The middle and hind leg tarsus were the only segments out of range. However, the maximum length difference (middle leg tarsal segment data median versus NeuroMechFly datapoint) is 0.1 mm out of an approximate middle leg length of 2.3 mm, or ≈4% of the total leg length. This ≈4% had a negligible impact on kinematic replay, as confirmed by our validation experiments.

Kinematic replay and analysis. *Forward walking data.* We recorded spontaneous behaviors from wild-type female flies 3–4 days after eclosion. Flies were mounted on a custom stage and allowed to acclimate for 15 min on an air-supported spherical treadmill[36]. Experiments were conducted during the evening in Zeitgeber time. Flies were recorded five times for 30 s at 5 min intervals. Data were excluded if forward walking was not present for at least 5 continuous seconds in 10 s windows. To record data, we used a 7-camera system as in ref. [31]. However, we replaced the front camera's InfiniStix lens with a Computar MLM3X-MP lens at ×0.3 zoom to visualize the spherical treadmill. After the fifth trial of each experiment we recorded an extra 10 s trial, having replaced the lens from a lateral camera with another Computar MLM3X-MP lens. We used these images to calculate the longitudinal position of the spherical treadmill with respect to the fly for the preceding five trials.

Foreleg–antennal grooming data. Data for kinematic replay of foreleg–antennal grooming were obtained 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 frames per second (fps) (https://dataverse.harvard.edu/dataverse/DeepFly3D). Time axes (Fig. 5e,f) correspond to time points from the original, published videos. Data were specifically obtained from experiment 3, taken of an animal (no. 6) expressing aDN-GAL4 driving UAS-CsChrimson.

Processing 3D pose data. We used DeepFly3D v0.4 (ref. [31]) to obtain 3D poses from the images acquired for each behavior. Two-dimensional poses were examined using the graphical user interface to manually correct 10 frames during walking and 72 frames during grooming. DeepFly3D, like other pose estimation software, uses a local reference system based on the positions of the cameras to define the animal's pose. Therefore, we first defined a global reference system for NeuroMechFly from which we could compare data from experiments on different animals (Extended Data Fig. 2).

Aligning both reference systems consisted of six steps. First, we defined the mean position of each 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 segment independently and defined its origin as the position of the proximal joint. Fourth, we rotated all data points on each leg according to its side (that is, left or right) and previously obtained orientations. Fifth, we scaled the real fly's leg lengths for each experiment to fit NeuroMechFly's leg size: a scaling factor was calculated for each leg segment as the ratio of its mean length throughout the experiment and the template's segment length and then each datapoint was scaled using this factor. Finally, we used the NeuroMechFly exoskeleton as a template to position all coxae within our global reference system; the exoskeleton has global location information for each joint. Next, we translated each datapoint for each leg (that is, CTr, FTi and TiTa joints) with respect to the ThC position based on this template.

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 names of the angles correspond to the rotational axis of the movement, that is, roll, pitch or yaw, for rotations around the anterior–posterior, mediolateral and dorsoventral axes, respectively.

The ThC joint 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 longitudinal axes of the coxa and the femur. Subsequently, we discovered that a CTr roll DoF would be required to accurately match the kinematic chain. To calculate this angle, we rotated the TiTa joint 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 longitudinal axes of the femur and the tibia. The pitch angle for the TiTa was measured between the longitudinal axes of the tibia and the tarsus. 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.

Comparing original and reconstructed 3D poses. To quantify the contribution of each DoF to kinematic replay, we used the forward kinematics method to compare original and reconstructed poses. Given that 3D pose estimation noise 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 the previous section). 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 from −90° to 90° with respect to 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. We note that differences in errors can vary across legs and leg pairs because each joint's 3D pose estimate is independent and each leg acts as an independent kinematic chain, adopting its own pose. Thus, errors may also be asymmetric across the body halves. As well, errors integrate along the leg when using forward kinematics for walking (Extended Data Fig. 3a–d) and for grooming (Extended Data Fig. 3e–h). By contrast, inverse kinematics acts as an optimizer and minimizes the error at the end of the kinematic chain (that is, where the forward kinematics error is highest) for walking (Extended Data Fig. 3d) and for grooming (Extended Data Fig. 3h). This explains why errors using forward kinematics are generally higher than those using inverse kinematics, with the exception of adding a roll DoF at the CTr 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 from fly 1 and frames 0–399 for foreleg–antennal grooming.

We ran a Kruskal–Wallis statistical test to compare kinematic errors across the eight methods used. We then applied a post-hoc 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 Supplementary Tables 2 and 3, respectively. Our statistical tests suggested that adding a CTr roll DoF uniquely improved kinematic replay compared with all other methods.

Transferring real 3D poses into the NeuroMechFly environment. To incorporate the additional CTr roll DoF into NeuroMechFly, we enabled rotations along the z axis of CTr joints. Then, we created new SDF configuration files using custom Python scripts to include 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 hold our model in place. We removed these supports for ground walking experiments (Supplementary Videos 8 and 9).

We used position control for each joint in the model. We fixed the position of non-actuated joints to the values shown in Supplementary Table 4. The actuated joints (that is, the leg joints) were controlled to achieve the angles calculated from 3D pose data. The simulation was run with a time step of 0.5 ms, allowing PyBullet to accurately perform numerical calculations. Given that the fly recordings were captured at only 100 fps, we up-sampled and interpolated pose estimates to match the simulation time steps before calculating joint angles.

Comparing real and simulated spherical treadmill rotations. We obtained spherical treadmill rotational velocities from real experiments using Fictrac[56]. We also obtained the relative inclination of each tethered fly (Φ) (Extended Data Fig. 6a) as the angle between the ground plane and the axis between the hind leg ThC joint and the dorsal part of the neck. Finally, we estimated the position of the ball with respect to the fly from both front and lateral views (Extended Data Fig. 6b,c) by identifying the ball and fly using a Hough transform and standard thresholding, respectively. For axes observed from both views, we averaged the expected position.

For the simulated environment we created a spherical body in PyBullet with three hinge joints along the *x*, *y* and *z* axes, allowing our sphere to rotate in each direction like a real spherical treadmill. Rolling and spinning frictions were set to zero to obtain virtually frictionless conditions similar to a real treadmill floating on air. The mass of the simulated spherical treadmill was set to 54.6 mg: the measured mass of the real foam sphere. Finally, the sphere's diameter was measured and set in the simulation as 9.96 mm.

We ran a kinematic replay of walking by setting the simulated spherical treadmill position and fly inclination based on measurements from experimental images. We used predefined values for kinematic replay of grooming. Then, we empirically determined the following parameters: global ERP (error reduction parameter) = 0.0, friction ERP = 0.0, solver iterations = 1,000 and treadmill lateral friction = 1.3.

After running the simulation we compared the rotational velocities estimated for each axis with the real velocities obtained with Fictrac. First, we smoothed both Fictrac and estimated signals using a median filter with a window size of 0.1 s. Second, we interpolated Fictrac data from time steps of 0.1 s (100 fps) to the simulation time step. We then established each signal's baseline as the mean of the first 0.2 s of data. Finally, we computed the Spearman correlation coefficient (ρ) to assess correlations of forward, lateral and heading (yaw) velocities for both signals.

Constraint parameter sensitivity analysis. Simulated spherical treadmill velocity estimates depend on constraint force mixing (CFM) and contact ERPs. These parameters change the 'softness' of joint and contact constraints in the physics engine. Therefore we performed a sensitivity analysis to determine the best combination of CFM and ERP. CFM values were swept from 0 to 10, and ERP from 0 to 1.0. Then, we ran a simulation for each of 121 combinations. We assessed their performance by calculating the Spearman correlation coefficient for each axis (Extended Data Fig. 5a–c).

Finally, to select optimal parameter values, we applied a weighted sum (WS_i) to the results as shown in equation (1):

$$\mathrm{WS}_i = \alpha * \mathrm{Fw}(\rho_i) + \beta * \mathrm{Lat}(\rho_i) + \gamma * \mathrm{Head}(\rho_i) \qquad (1)$$

where Fw, Lat, and Head are the rotational axes, ρ_i is the Spearman correlation coefficient obtained for each CFM–ERP combination, and α, β and γ are the standard deviation contributions for each axis calculated as shown in equations (2), (3) and (4), respectively. Therefore, we favored the axis with the largest amplitude of variation.

$$\alpha = \frac{\mathrm{s.d.(Fw)}}{\mathrm{s.d.(Fw)} + \mathrm{s.d.(Lat)} + \mathrm{s.d.(Head)}} \qquad (2)$$

$$\beta = \frac{\mathrm{s.d.(Lat)}}{\mathrm{s.d.(Fw)} + \mathrm{s.d.(Lat)} + \mathrm{s.d.(Head)}} \qquad (3)$$

$$\gamma = \frac{\mathrm{s.d.(Head)}}{\mathrm{s.d.(Fw)} + \mathrm{s.d.(Lat)} + \mathrm{s.d.(Head)}} \qquad (4)$$

Finally, we normalized WS (NWS) with respect to its maximum and minimum values (Extended Data Fig. 5d). Consequently, a combination with NWS equal to 1 was selected: CFM = 3 and ERP = 0.1.

Controller gain sensitivity analysis. We performed kinematic replay using a built-in proportional–derivative position controller in PyBullet[42]. A proportional–derivative controller was used rather than the more widely known proportional-integral-derivative controller because the integral component is mainly used to correct steady-state errors (for example, while maintaining a fixed posture). Thus, it is not used for time-varying postures such as those during locomotion. We used PyBullet's built-in position control method because it operates with proportional and derivative gains that are stable and efficient. This proportional–derivative controller minimizes the error:

$$\mathrm{error} = K_p(\theta_r - \theta_a) + K_d(\omega_r - \omega_a) \qquad (5)$$

where θ_r and θ_a denote reference and actual positions, ω_r and ω_a are reference and actual velocities, and K_p and K_d are proportional and derivative gains, respectively, which provides some compliance in the model.

Because the outputs of our model (that is, the dynamics of motion) depend on the controller gains K_p and K_d, we first systematically searched for optimal gain values. To do this, we ran the simulation's kinematic replay for numerous K_p and K_d pairs, ranging from 0.1 to 1.0 with a step size of 0.1 (that is, 100 simulations in total). Target position and velocity signals for the controller were set as the calculated joint angles and angular velocities, respectively. To compute joint angular velocities, we used a Savitzky–Golay filter with a first-order derivative and a time step of 0.5 ms on the joint angles. Feeding the controller with only the joint angles could also achieve the desired movements of the model. However, inclusion of 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 between the ground truth (that is, the joint angles obtained by running our kinematic replay pipeline on pose estimates from DeepFly3D[31]) and the joint angles obtained from PyBullet. Finally, we averaged the mean squared error across the joints in one leg, and summed the mean of the mean squared errors from each of the six legs to obtain a total error. We made the same calculations for the joint angular velocities as well. Our results (Extended Data Fig. 4a,b) show that our biomechanical model can replicate real 3D poses while also closely matching real measured velocities. In particular, a mean squared error of 360 $(\mathrm{rad\,s^{-1}})^2$ for the six legs corresponds to approximately 7.74 $\mathrm{rad\,s^{-1}}$ per leg, that is, 1.27 Hz. This is acceptable given the rapid, nearly 20 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) (Extended Data Fig. 4c–f) but are robust to variations in derivative gain (K_d). These results are expected given that high proportional gains cause 'stiffness' in the system whereas derivative gains affect the 'damping' in a system's response. We observed rapid changes in estimated torques and ground reaction forces at high K_p (Extended Data Fig. 4c,d). Notably, in principle there can also be internal forces affecting contact forces. For example, a fly's legs can squeeze the spherical treadmill with different internal forces but have identical postures.

As shown in Extended Data Fig. 4a,b our model can match the real kinematics closely for almost every controller gain combination except for the low K_p, K_d 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 (Extended Data Fig. 4c–f). Specifically, we chose 0.4 and 0.9 for K_p and K_d, respectively. These values were high enough to generate smooth movements and low enough to reduce movement stiffness.

Comparing tethered and flat ground walking. To test the ability to run NeuroMechFly in an untethered context, we replayed the kinematics of a tethered walking experiment (Fig. 4) but removed body supports and placed the model on the floor. To remove body supports, we deleted the corresponding links from the model's description (SDF configuration file). The physics engine parameters remained the same. The lateral friction for the floor was set to 0.1.

Application of external perturbations. To test the stability of the untethered model walking over flat ground, we set the floor's lateral friction to 0.5 and introduced external perturbations. Specifically, we propelled solid spheres at the model according to the following equation of motion,

$$\vec{p} = \vec{r_0} + \vec{u_0}t + \frac{1}{2}\vec{g}t^2 \qquad (6)$$

where $\vec{p}$ is the 3D target position (the fly's center of mass), $\vec{r_0}$ is the initial 3D position of the sphere, $\vec{u_0}$ is the initial velocity vector, $\vec{g}$ is the external acceleration vector due to gravity in the *z* direction, and *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 and its radius was 50 µm. 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 (6). The spheres were propelled at the model every 0.5 s. Finally, 3 s into the simulation, a 3 g sphere with a radius of 150 µm was propelled at the fly to topple it over (Supplementary Video 9).

Analyzing NeuroMechFly's contact and collision data. The PyBullet physics engine generates forward dynamics simulations and collision detections. We plotted joint torques as calculated from PyBullet. To infer ground reaction forces 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 ground reaction forces: a leg was considered to be in stance phase if its ground reaction force was greater than zero. These gait diagrams were compared with a ground truth (Extended Data Fig. 7) obtained by manually annotating when the legs were in contact with the ball for each video frame. Gait prediction accuracy was calculated by dividing the frames correctly predicted as being in stance or swing over the total number of frames.

Self-collisions are disabled by default in PyBullet. Therefore, for 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 the right and left front leg; the left front leg and left antenna; and the right front leg and right antenna. Grooming diagrams were calculated as for gait diagrams: a segment experienced a contact or collision if it reported a normal force greater than zero.

Assessing the importance of morphological accuracy. We replayed the foreleg–antennal grooming kinematics (Fig. 5) for three conditions to assess the degree to which biomechanical realism is important for collision estimation. We tested two experimental conditions: one in which both front legs were modeled as sticks, and one in which the front legs as well as the antennae were modeled as sticks. Notably, multisegmented tarsi are not found in other published insect stick models[63]. Thus, as for our previous model[41], each stick leg consisted of four segments: coxa, trochanter–femur, tibia and one tarsal segment. Each leg and antennal stick segment had a diameter equal to the average diameter of the corresponding segment in our more detailed NeuroMechFly model. These changes were accomplished by modifying the model's description (SDF configuration file) and by changing the collision and visual attributes for each segment of interest.

Neural network parameter optimization. *Central pattern generator network architecture.* For evolutionary optimization of neuromuscular parameters, we designed a central pattern generator-based controller composed of 36 non-linear oscillators (Fig. 6), as for a previous investigation of salamander locomotion[61]. These central pattern generators consisted of mathematical oscillators that represent neuronal ensembles firing rhythmically in the ventral nerve cord[84]. The central pattern generator model was governed by the following system of differential equations:

$$\dot{\theta}_i = 2\pi\nu_i + \sum_j r_j w_{ij} \sin(\theta_j - \theta_i - \phi_{ij}) \tag{7}$$

$$\dot{r}_i = a_i(R_i - r_i) \tag{8}$$

$$M_i = r_i(1 + \sin(\theta_i)) \tag{9}$$

where the state variables, that is, phase and amplitude of the oscillator i, are denoted θ_i and r_i, respectively; ν_i and R_i represent oscillator i's intrinsic frequency and amplitude, a_i is a constant and M_i represents the cyclical activity pattern of neural ensembles activating muscles. The coupling strength and phase bias between the oscillators i and j are denoted w_{ij} and ϕ_{ij}, respectively.

During optimization, for the entire network of coupled oscillators we set the intrinsic frequency ν as an open parameter ranging from 6 to 10 Hz, which corresponds to the stepping frequencies measured from our kinematic replay experiment and also previously reported stepping frequencies[64]. The intrinsic amplitude R was set to 1, and the constant a_i was set to 25. To ensure a faster convergence to a phase-locked regime between oscillators, we set coupling strengths to 1,000 (ref. [85]). We solved this system of differential equations using the explicit Runge–Kutta fifth order method with a time step of 0.1 ms.

Each oscillator pair sends cyclical bursts to flexor and extensor muscles, which apply antagonistic torques to the corresponding revolute joint. We considered three DoFs per leg that had been sufficient for locomotion in previous hexapod models[63] and that had the most pronounced joint angles (Extended Data Fig. 10a,b). These DoFs were ThC pitch for the front legs, ThC roll for the middle and hind legs, and CTr pitch and FTi pitch for all legs. Thus, there were three pairs of oscillators optimized per leg, for a total of 36. We coupled the intraleg oscillators in a proximal to distal chain, the interleg oscillators in a tripod-like fashion (the ipsilateral front and hind legs to the contralateral middle leg from anterior to posterior), both front legs to each other, and the coxa extensor and flexor oscillators to one another. Intraleg coordination is equally important to generate a fly-like gait because stance and swing phases depend on intrasegmental phase relationships. For this reason, both interleg couplings (phase relationships between ThC joints) and intraleg couplings (phase relationships within each leg) were optimized for one half of the body and mirrored on the other.

Muscle model. We adapted an Ekeberg-type muscle model[62] to generate torques on the joints. This model simulates muscles as a torsional spring and damper system, enabling torque control of any joint as a linear function of motor neuron (central pattern generator output) 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:

$$T = \alpha(M_F - M_E) + \beta(M_F + M_E + \gamma)\Delta\varphi + \delta\dot{\varphi} \tag{10}$$

where α, β, γ and δ represent the gain, stiffness gain, tonic stiffness and damping coefficient, respectively[9]. $\Delta\varphi$ is the difference between the current angle of the joint and its resting pose. $\dot{\varphi}$ 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 optimized muscle coefficients, that is, $\alpha, \beta, \gamma, \delta$ and $\Delta\varphi$.

Central pattern generator network and muscle parameter optimization. To identify neuromuscular network parameters that could coordinate fast and statically stable locomotion, we optimized the phase differences for each network connection, the intrinsic frequency of the oscillators, and five parameters controlling the gains and resting positions of each spring and damper muscle (that is, $\alpha, \beta, \gamma, \delta$ and $\Delta\varphi$). To simplify the problem for the optimizer, we first fixed ThC flexor–extensor phase differences to 180°, making them perfectly antagonistic, second, we mirrored the phase differences from the right leg oscillators to the left leg oscillators, third, we mirrored muscle parameters from the right joints to the left joints, and last, we mirrored phase differences from ThC–ThC flexors to ThC–ThC extensors. Thus, a total of 63 open parameters were set by optimization: five phases between ThC central pattern generators (Fig. 6a), 12 phases between intraleg central pattern generators (ThC–FTi extensor–flexor, FTi–TiTa extensor–flexor per leg), 45 muscle parameters (five per joint), and one parameter (ν) controlling the intrinsic frequency of the oscillators. We empirically set the lower and upper bounds for the parameters to ensure that leg movements would stay stable along the boundaries (Supplementary Table 6). Upper and lower bounds for the resting positions of the joints used in the muscle model were set as the first and third quartiles of measured locomotor angles. Finally, we optimized the intrinsic frequency of central pattern generators, denoted by ν in equation (7), to be between 6 and 10 Hz for the reasons described above.

For parameter optimization we used NSGA-II (ref. [66]), a multiobjective genetic algorithm implemented in Python using the jMetalPy library[86]. We defined two objective functions. First, we aimed to maximize locomotor speed, as quantified by the number of spherical treadmill rotations (equation (11)) along the y axis within a specific period of time. Second, we maximized static stability. In small animals such as *Drosophila*, static stability is a better approximation for overall stability than dynamic stability[68]. We measured static stability by first identifying a convex hull formed by the legs in stance phase. If there were less than three 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, dynamically calculated based on the fly's moving body parts, 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 the speed and stability values: the most negative values meant the fastest and most stable solutions, respectively.

Four penalties were added to the objective functions. First, to make sure that the model was always moving, we set moving lower and upper thresholds for the angular rotation of the ball, increasing from −0.2 rad to 1.0 rad and from 0 to 7.2 rad in 1 s, respectively. These values were determined such that the lower moving boundary was slower than the slowest reported walking speed of *Drosophila* ($10\,\mathrm{mm\,s^{-1}} = 2$ rad when the ball radius r is 5 mm) (ref. [64]), and the upper moving boundary would exceed the highest reported walking speed ($34\,\mathrm{mm\,s^{-1}} = 6.8$ rad) (ref. [26]). Second, to avoid high torque and velocities at each joint, we set joint angular velocities to have an upper limit of $250\,\mathrm{rad\,s^{-1}}$, a value measured from real fly experiments. Third, because we do not introduce physical joint limits in the model, we simulated these joint limits by setting a penalty on the difference between the joint angle range observed during kinematic replay of walking and the joint angles of individual solutions. We used this penalty to prevent joint angles from generating unrealistic movements (for example, one full rotation around a DoF). And last, because the optimizer can exploit the objective function by simply leaving all legs on the ground, that is, the highest possible stability, or can rotate the ball by using as few as two legs while the remaining legs are constantly on the ground, we introduced a penalty on duty factors. Specifically, we computed the ratio of stance phase duration to the entire epoch and penalized the solutions for which the duty factors for each leg were outside of the range 0.4–0.9, based on ref. [26].

The optimization was formulated as follows

$$\begin{aligned} &\min -10 \cdot R_b \cdot \theta_{b,\parallel} + 0.1 \cdot p_v + 0.05 \\ &\cdot p_{jl} + 0.1 \cdot p_m + 100 \cdot p_d \quad \text{(Distance and penalties)} \end{aligned} \tag{11}$$

$$\begin{aligned} &\min -0.01 \cdot s + 0.1 \cdot p_v + 0.05 \\ &\cdot p_{jl} + 0.1 \cdot p_m + 100 \cdot p_d \quad \text{(Stability and penalties)}, \end{aligned} \tag{12}$$

with the following penalty terms

$$p_m^i = \begin{cases} p_m^{i-1} + 1 & \text{if } \quad \theta_{b,\parallel} \leq \left(\frac{t}{t_{\text{total}}} \cdot 1.20 - 0.20\right) \text{ or } \theta_{b,\parallel} \geq \left(\frac{t}{t_{\text{total}}} \cdot 7.20\right) \\ p_m^{i-1} & \text{otherwise} \end{cases}$$
$$\text{(Moving boundary penalty)} \tag{13}$$

$$p_v^i = \begin{cases} p_v^{i-1} + 1 & \text{if } \omega > 250\,\mathrm{rad\,s^{-1}} \\ p_v^{i-1} & \text{otherwise} \end{cases} \quad \text{(Angular velocity penalty)} \tag{14}$$

$$p_{jl}^i = \begin{cases} p_{jl}^{i-1} + \sum_k \theta_k - \max(\text{joint limit}_k) & \text{if } \theta_k \geq \max(\text{joint limit}_k) \\ p_{jl}^{i-1} + \sum_k -\theta_k + \min(\text{joint limit}_k) & \text{if } \theta_k \leq \min(\text{joint limit}_k) \\ p_{jl}^{i-1} & \text{otherwise} \end{cases} \tag{15}$$
$$\text{(Joint limit penalty)}$$

$$p_d^i = \begin{cases} p_d^{i-1} + 1 & \text{if } \frac{t^k_{\text{stance}}}{t^k_{\text{bout}}} < 0.4 \text{ or } \frac{t^l_{\text{stance}}}{t^l_{\text{bout}}} > 0.9 \text{ for } l = 1, 2, .., 6 \\ p_d^{i-1} & \text{otherwise} \end{cases} \quad (16)$$

(Duty factor penalty)

where R_b is the ball radius (5 mm), $\theta_{b,\|}$ is the angle of the ball in the direction of walking, t_{total} is the maximum simulation duration, θ_k is the angular position of the joint k, and t^l_{stance} and t^l_{bout} are the total times spent in stance and the entire walking epoch duration of the leg l, respectively. Every penalty was multiplied by its corresponding weight and added to the objective function. Objective functions were evaluated for 2 s (t_{total}), a period that was sufficiently long for the model to generate locomotion. We ran 60 generations with the weights given in equation (11) and equation (12).

To avoid a high computational cost during optimization, we reduced the model's complexity by removing collision shapes (such as the wings and head) that were not required for locomotion, and by converting joints that are not used in the simulation (Supplementary Table 5) from revolute to fixed. This model was saved as a new SDF file. Thus, we could reduce the computation time, and the memory needed to check for collisions on unused body segments and for the position controller to set unused joints to fixed positions. This simplification increased the speed of the simulation, enabling us to reduce the time step to 0.1 ms and to run optimization with larger populations. In the simulation we used a spherical treadmill with a mass, radius and friction coefficient of 54.6 mg, 5 mm and 1.3, respectively. We additionally increased the friction coefficient of the leg segments from the default value of 0.5 to 1.0.

Each optimization generation had a population of 200 individuals. Optimization runs lasted for 60 generations, a computing time of approximately 20 h per run on an Intel Core i9-9900K central processing unit at 3.60 GHz. Mutations occurred with a probability of 1.0 divided by the number of parameters (63), and a distribution index of 20. We set the cross-over probability to 0.9 and the distribution index to 15 (for more details see ref. [86]).

Analysis of optimization results. After optimization we selected three individual solutions from the last generation for deeper analysis. First, the objective functions were normalized with respect to their maximum and minimum values. Note that the signs of the objective functions were inverted. Then, solutions were selected as follows:

Longest distance traveled (fastest): $i = \text{argmin}(d_g)$

Highest stability coefficient (most stable): $i = \text{argmin}(s_g)$

Distance–Stability minimum 2-norm (trade-off): $i = \text{argmin}\left(\sqrt{d_g^2 + s_g^2}\right)$,

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

We plotted central pattern generator activity patterns (as represented by the outputs of the coupled oscillators), joint torques, joint angles, ground reaction forces, and ball rotations from this final generation of solutions. Ground reaction forces were used to generate gait diagrams as previously described. Ball rotations were used to reconstruct the models' walking paths. The distances traveled along the longitudinal (x) and transverse (y) axes were calculated from the angular displacement of the ball according to the following formula:

$$\Delta x = \Delta\theta_t r \quad \Delta y = \Delta\theta_l r,$$

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.

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

Data availability
Kinematic data and 3D pose estimations for walking behavior are available at https://doi.org/10.7910/DVN/Y3TAEC, and kinematic data and 3D pose estimations for grooming behavior are available at https://dataverse.harvard.edu/dataverse/DeepFly3D

Code availability
The code required to reproduce the experiments described here can be obtained as a Code Ocean capsule (https://codeocean.com/capsule/2418941/tree/v1, ref. [87]). Code and documentation for developers are available in GitHub under an Apache 2.0 license (https://github.com/NeLy-EPFL/NeuroMechFly and https://nely-epfl.github.io/NeuroMechFly).

References
76. Hayat, M.A. *Principles and Techniques of Electron Microscopy: Biological Applications* (Van Nostrand Reinhold, 1976).
77. Schindelin, J. et al. Fiji: an open-source platform for biological-image analysis. *Nat. Methods* **9**, 676–682 (2012).
78. Lewiner, T., Lopes, H., Vieira, A. W. & Tavares, G. Efficient implementation of marching cubes' cases with topological guarantees. *J. Graphics Tools* **8**, 1–15 (2003).
79. van der Walt, S. et al. Scikit-image: image processing in Python. *PeerJ* **2**, e453 (2014).
80. Blender Foundation. *Blender: a 3D Modelling and Rendering Package* http://www.blender.org (2012).
81. Ferris, G. External morphology of the adult. In *Biology of* Drosophila (ed. Demerec M) 368–419 (Wiley & Sons, 1950).
82. Dickson, W. B., Straw, A. D. & Dickinson, M. H. Integrative model of *Drosophila* flight. *AIAA J.* **46**, 2150–2164 (2008).
83. Geurten, B. R. H., Jähde, P., Corthals, K. & Göpfert, M. C. Saccadic body turns in walking *Drosophila*. *Front. Behav. Neurosci.* **8**, 365 (2014).
84. Mantziaris, C., Bockemühl, T. & Büschges, A. Central pattern generating networks in insect locomotion. *Dev. Neurobiol.* **80**, 16–30 (2020).
85. 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. *J. Math. Biol.* **13**, 345–369 (1982).
86. Benitez-Hidalgo, A., Nebro, A., Garcia-Nieto, J., Oregi, I. & Del Ser, J. jMetalPy: a Python framework for multi-objective optimization with metaheuristics. *Swarm Evol. Comput.* **51**, 100598 (2019).
87. Lobato-Rios, V. et al. NeuroMechFly, a neuromechanical model of adult *Drosophila melanogaster*: *Code Ocean* https://codeocean.com/capsule/2418941/tree/v1 (2022).

Acknowledgements
We thank S. Clerc Rosset and G. Knott (Biological Electron Microscopy Facility, EPFL, Lausanne, Switzerland) for preparing *Drosophila melanogaster* samples for X-ray microtomography. We thank H. Sigurthorsdottir for early work on limb degrees of freedom. Furthermore, P.R. acknowledges support from an SNSF Project Grant (175667) and an SNSF Eccellenza Grant (181239). V.L.-R. acknowledges support from the Mexican National Council for Science and Technology, CONACYT, under the grant number 709993. S.T.R. acknowledges support from the European Union's Horizon 2020 research and innovation program under grant agreement nos. 720270 (SGA1) and 785907 (SGA2). P.G.O. acknowledges support from the Swiss Government Excellence Scholarship for Doctoral Studies. J.A. acknowledges support from the Human Frontier Science Program (HFSP-RGP0027/2017).

Author contributions
V.L-R., conceptualization, methodology, software, validation, formal analysis, investigation, data curation, validation, writing (original draft preparation, and review and editing), visualization. S.T.R., conceptualization, methodology, software, validation, writing (review and editing), visualization. P.G.O., conceptualization, methodology, software, validation, formal analysis, investigation, data curation, writing (review and editing), visualization. J.A., conceptualization, methodology, software, validation, writing (review and editing). A.J.I., conceptualization, methodology, resources, writing (review and editing), supervision, project administration, funding acquisition. P.R., conceptualization, methodology, resources, writing (original draft preparation, and review and editing), supervision, project administration, funding acquisition.

Competing interests
The authors declare no competing interests.

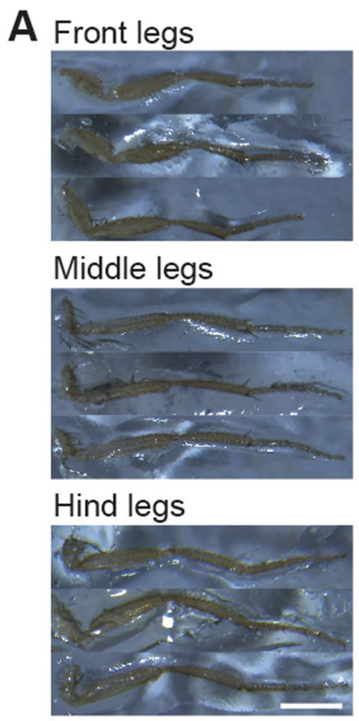
Additional information
Extended data are available for this paper at https://doi.org/10.1038/s41592-022-01466-7.

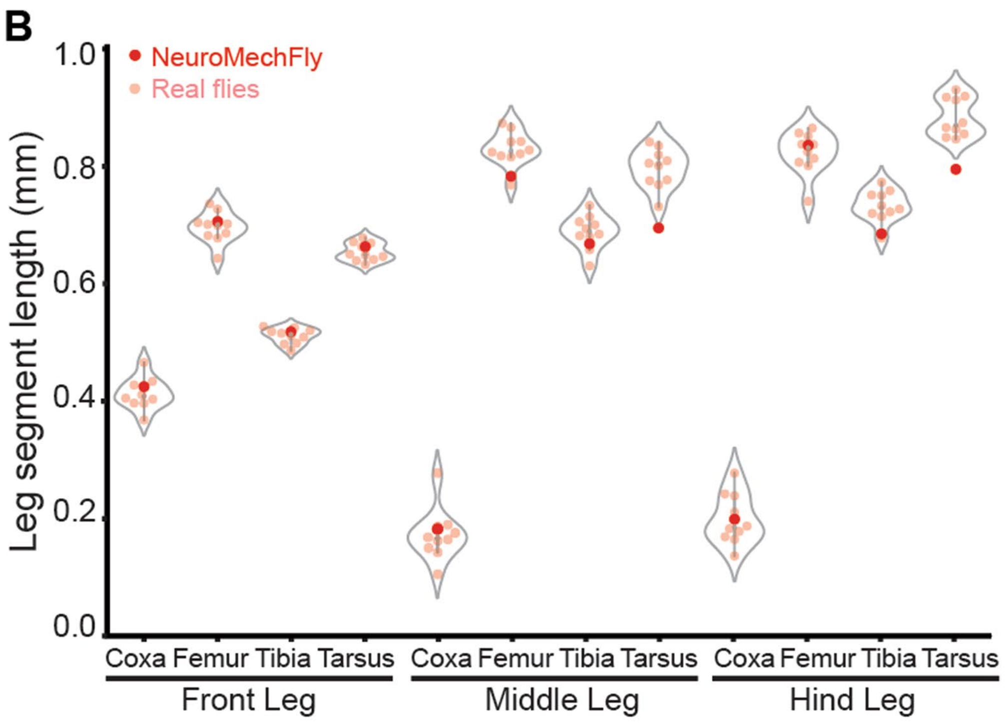
Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41592-022-01466-7.

Correspondence and requests for materials should be addressed to Pavan Ramdya.

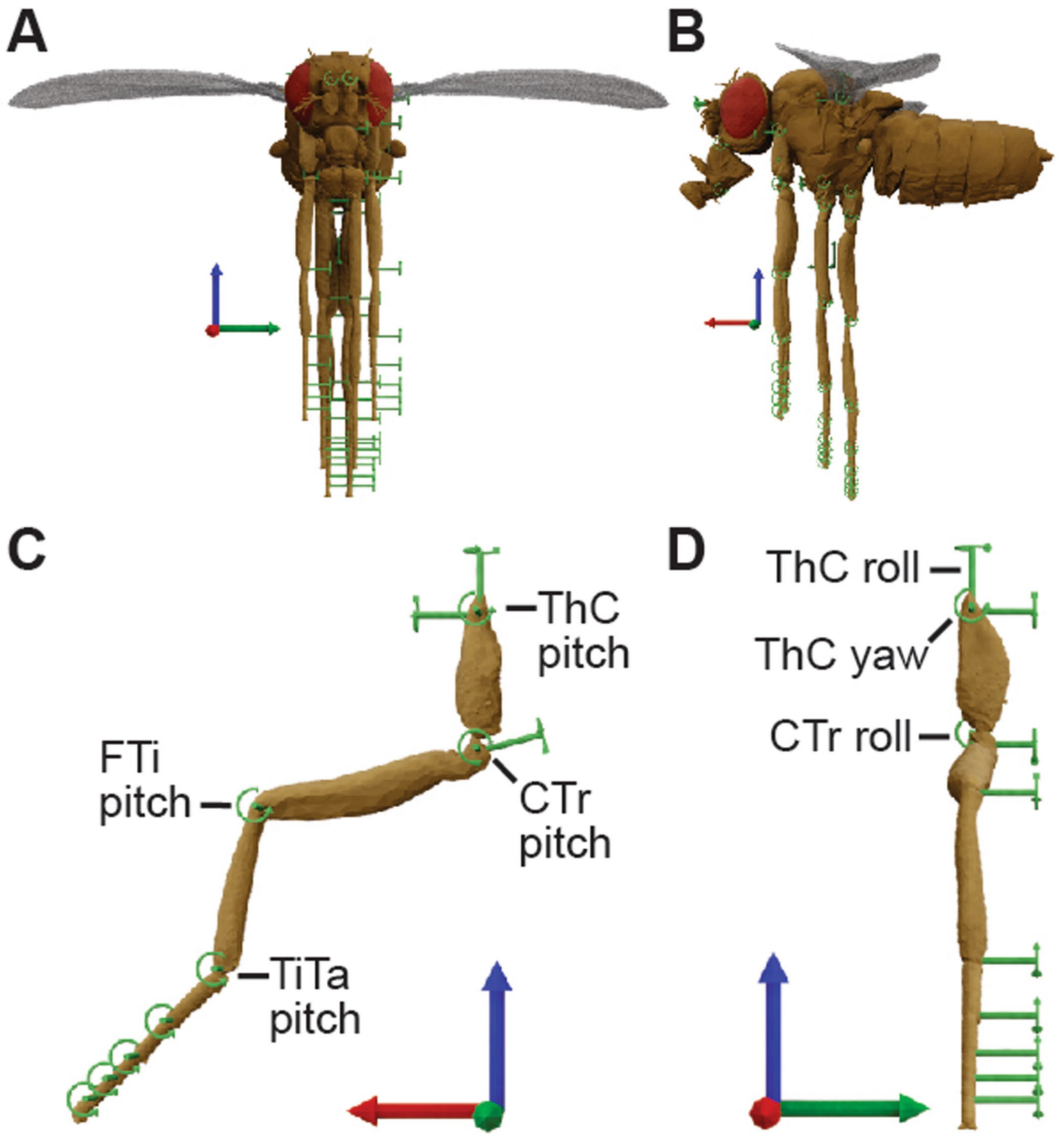
Peer review information *Nature Methods* thanks Benjamin de Bivort and the other, anonymous, reviewer(s) for their contribution to the peer review of this work. Nina Vogt was the primary editor on this article and managed its editorial process and peer review in collaboration with the rest of the editorial team.

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

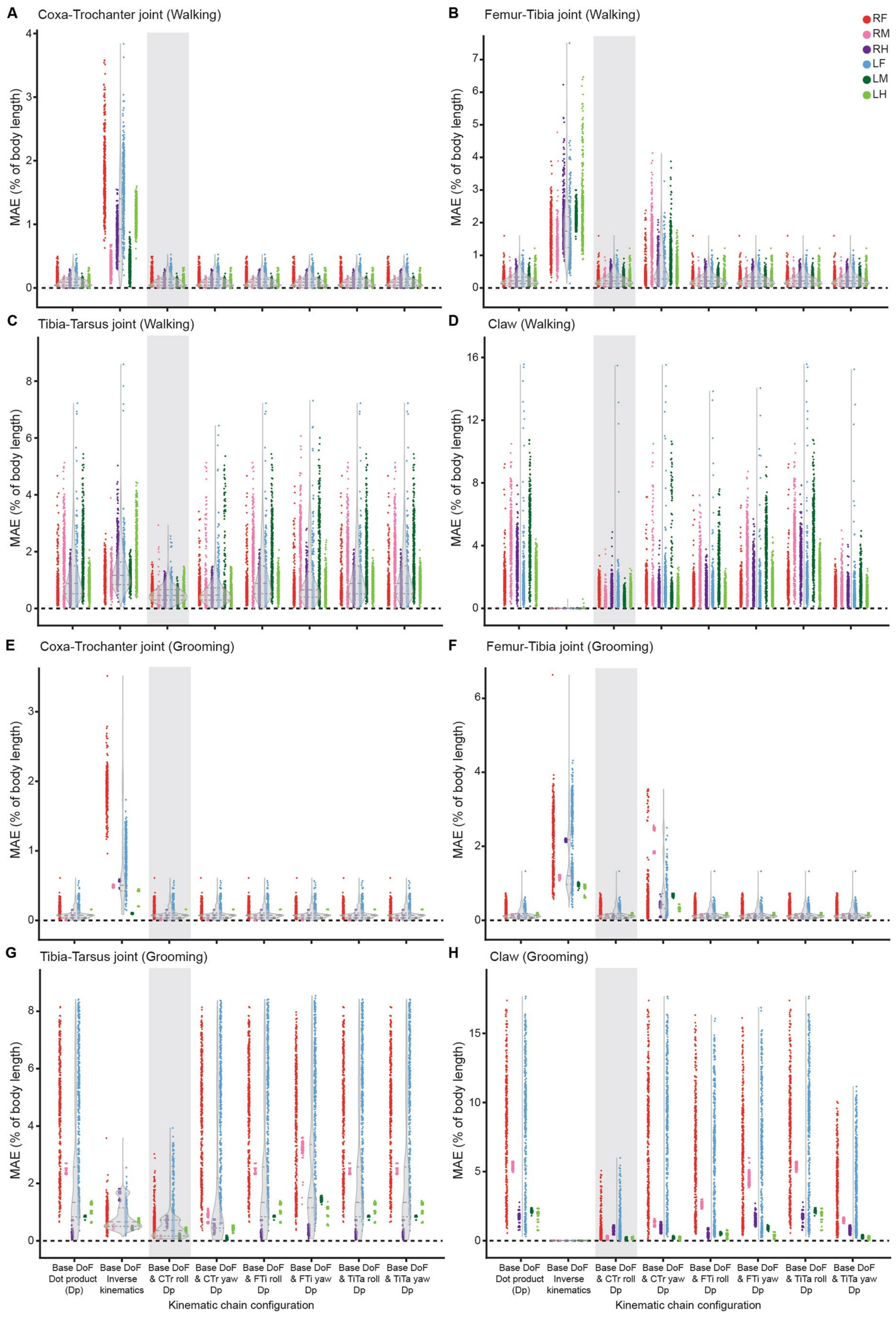




Extended Data Fig. 1 | Leg segment lengths for real female *Drosophila melanogaster* and NeuroMechFly. (A) Legs were dissected, straightened, and fixed onto a glass slide for measurements. Scale bar is 0.5mm. **(B)** The lengths of leg segments from 1-3 dpe animals (pink) and NeuroMechFly (red) are shown. Violin plots indicate median, upper, and lower quartiles.

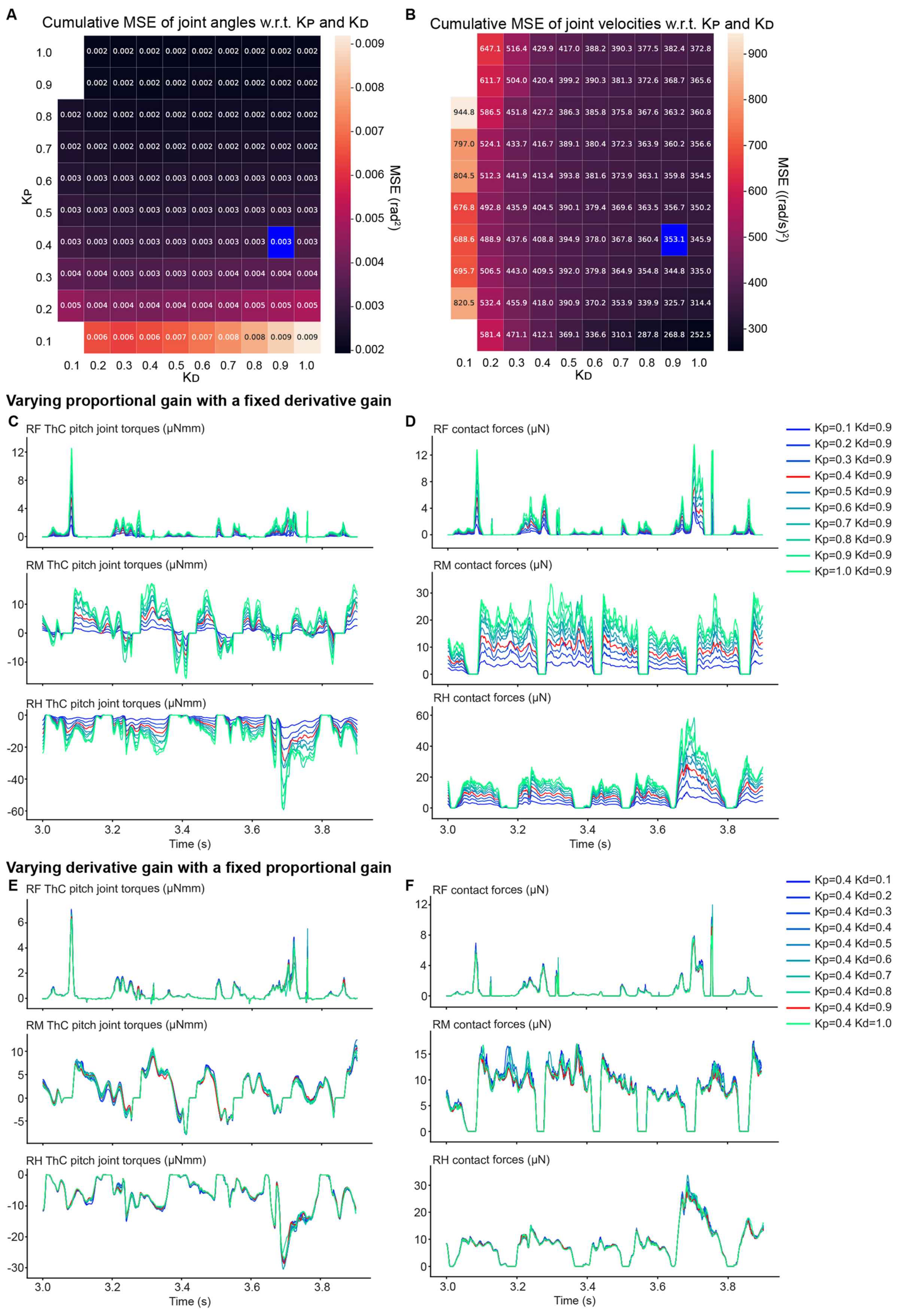


Extended Data Fig. 2 | The 'zero pose' of NeuroMechFly and its leg joint degrees-of-freedom. Zero pose of NeuroMechFly from **(A)** front and **(B)** side views. Each leg is composed of 11 hinge joints. Joints with more than one DoF were modeled as a union of multiple hinge joints. The left foreleg observed from the **(C)** side and **(D)** front views. Rotational axes of joints are shown in light green. The global coordinate system's x, y, and z axes are red, green, and blue, respectively.



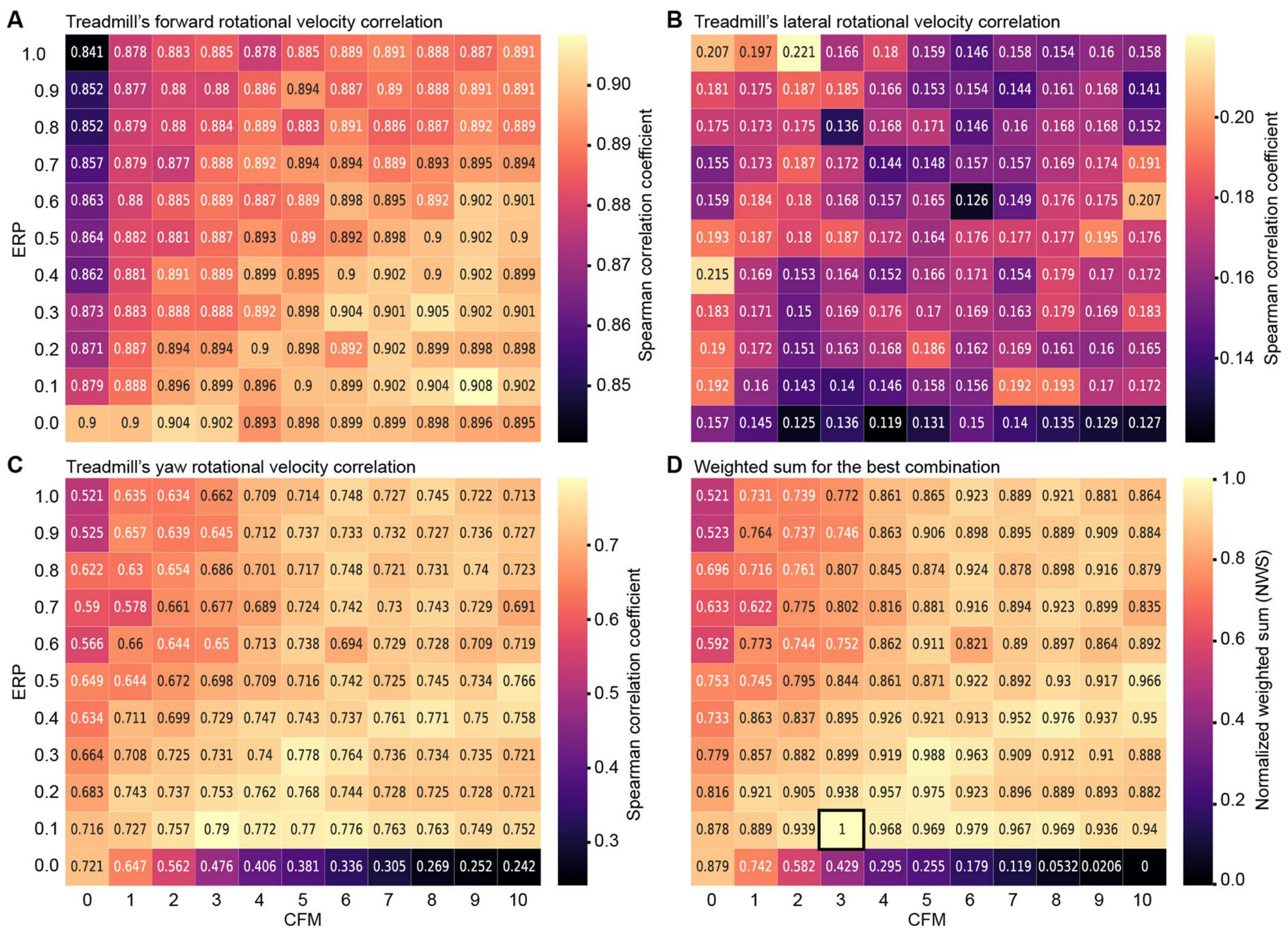
Extended Data Fig. 3 | See next page for caption.

Extended Data Fig. 3 | The position error for every joint in the distal leg during walking and grooming as a function of kinematic chain configuration. Body-length normalized mean absolute errors (MAE) comparing measured 3D poses and angle-derived joint positions during walking. Errors are compared among different DoF configurations for **(A)** Coxa–Trochanter joints, **(B)** Femur–Tibia joints, **(C)** Tibia–Tarsus joints, and **(D)** Claw positions during walking and **(E-H)** grooming. For each condition, n = 2400 samples were computed across all six legs from 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.

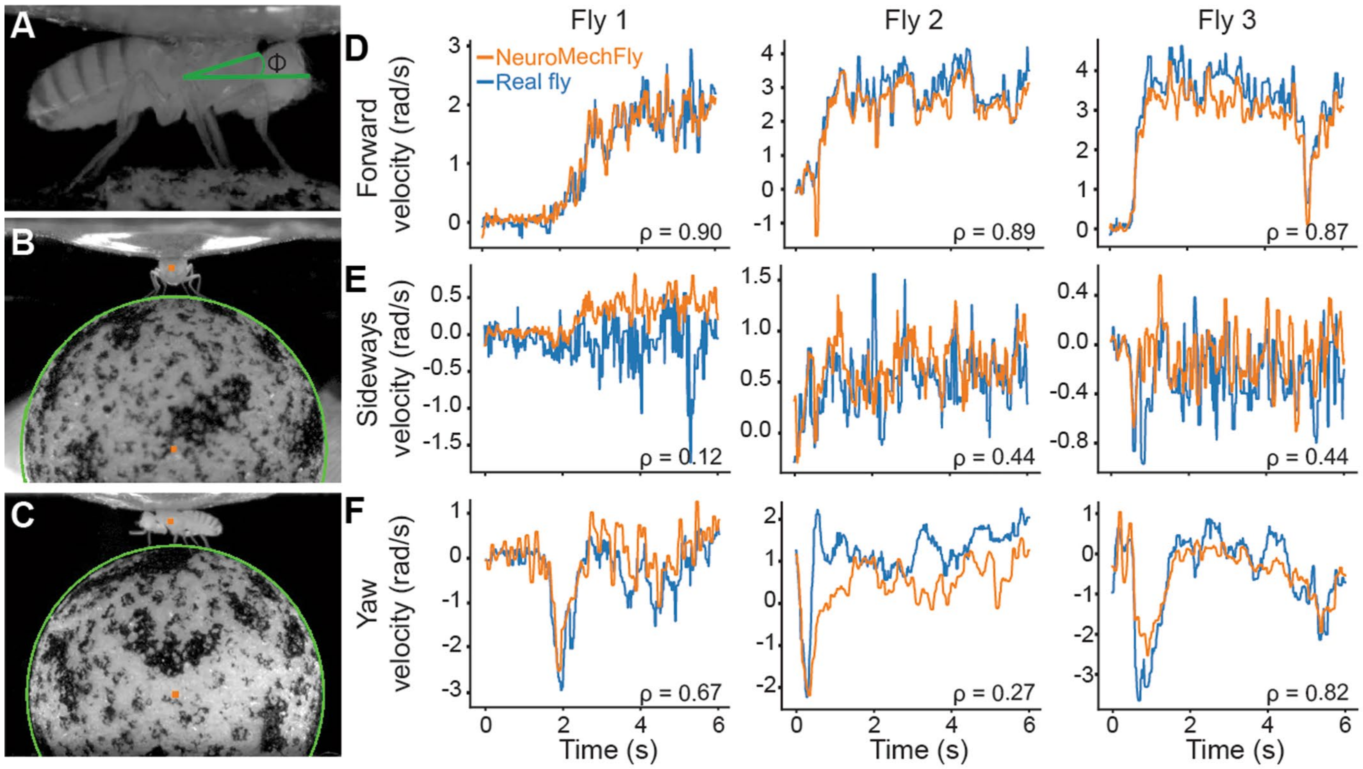


Extended Data Fig. 4 | See next page for caption.

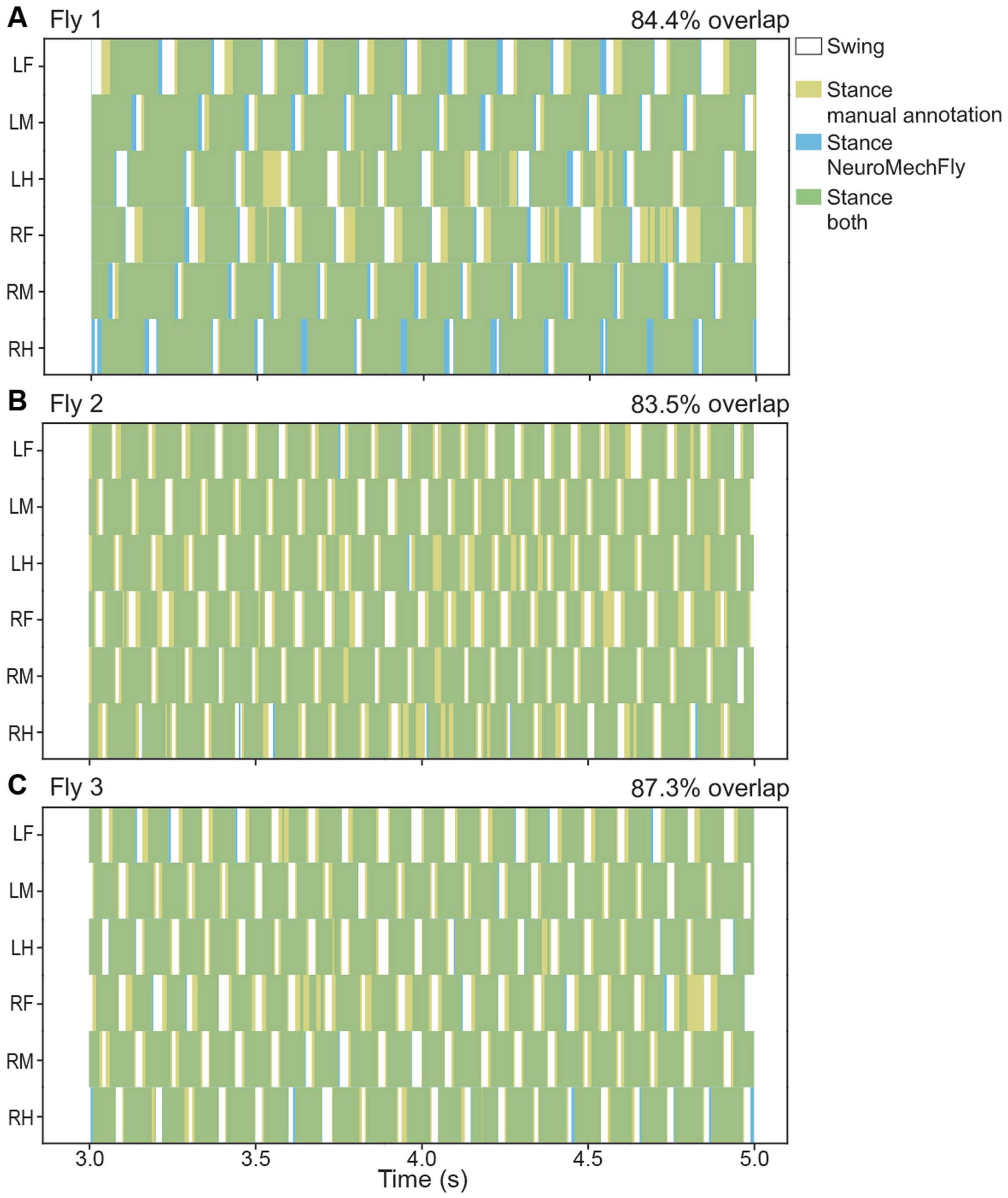
Extended Data Fig. 4 | Sensitivity to proportional and derivative gains of kinematic replay during walking. MSE of **(A)** joint angles and **(B)** joint velocities as a function of derivative (K_d) and positional gains (K_p). Selected K_p and K_d values are indicated (blue). K_p and K_d pairs rendering the simulation nonfunctional during kinematic replay are also indicated (white). **(C)** Estimated ThC pitch torques and **(D)** contact force measurements of the right legs during forward walking as a function of proportional gain (K_p) when the derivative gain (K_d) is fixed at 0.9. Measurements for the contralateral legs were nearly symmetrically identical and are not shown. Results from the selected K_p and K_d values are indicated (red). **(E)** Estimated ThC pitch torques and **(F)** Contact force measurements of the right legs during forward walking as a function of derivative gain (K_d) while holding proportional gain (K_p) fixed at 0.4. Measurements for the contralateral legs were nearly symmetrically identical and are not shown. Results from the selected K_p and K_d values are indicated (red).



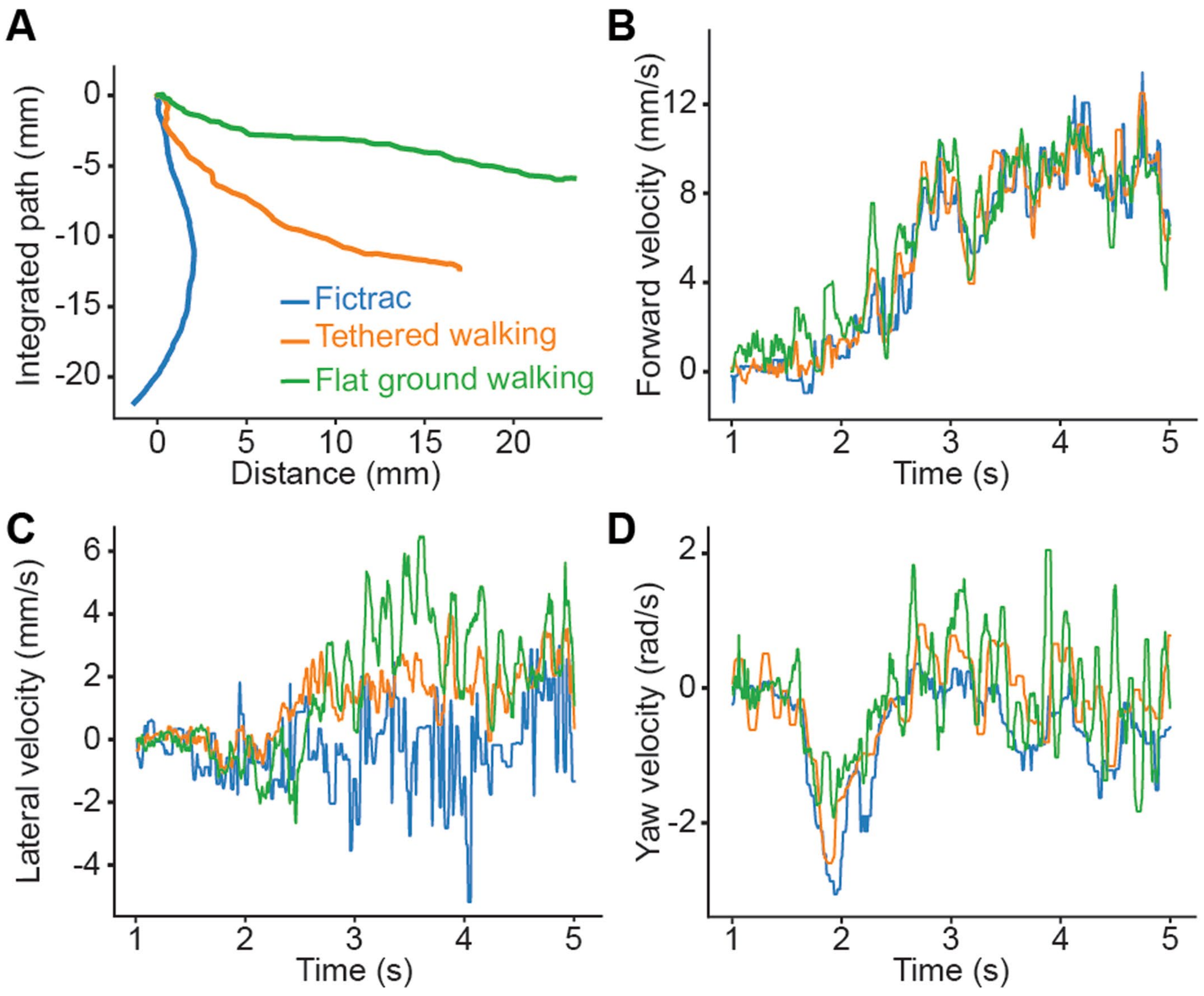
Extended Data Fig. 5 | Sensitivity of simulated spherical treadmill rotation prediction accuracy during tethered walking to ERP and CFM constraint parameters. Spherical treadmill rotational velocities resulting from Kinematic Replay of walking depend on simulation constraint parameters. Shown are Spearman correlation coefficients computed between measured and estimated treadmill rotational velocities for **(A)** forward, **(B)** lateral, and **(C)** yaw axes when varying the simulation's error reduction parameter (ERP), and the constraint force mixing (CFM). **(D)** The best combination of ERP and CFM—0.1 and 3, respectively (black outline)—was selected through a normalized weighted sum (NWS) of the correlation coefficients for each axis.



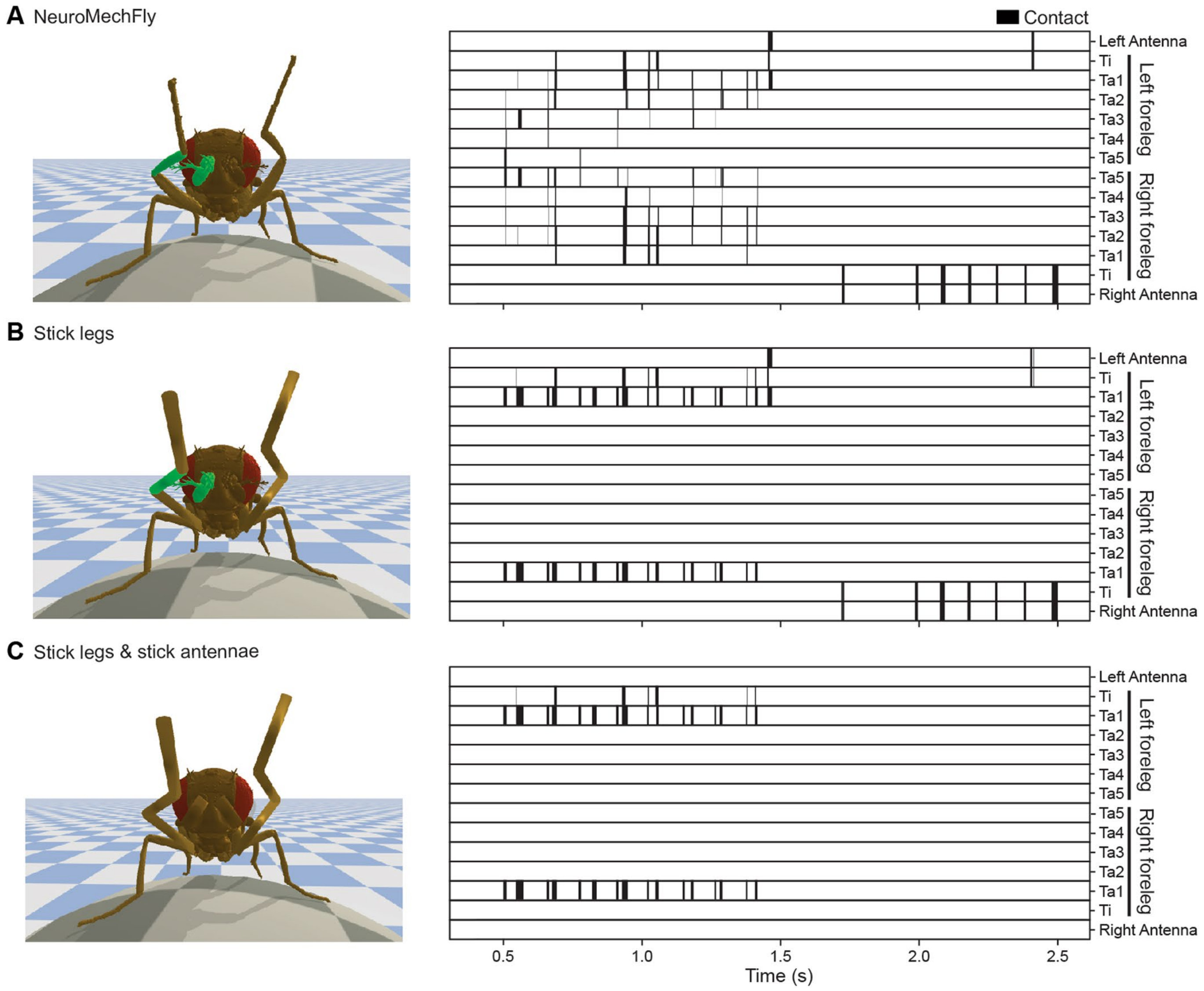
Extended Data Fig. 6 | Comparing real to simulated spherical treadmill rotational velocities during tethered walking. Spherical treadmill rotations depend on a tethered fly's **(A)** inclination (Φ, green), **(B)** lateral, and **(C)** longitudinal positions with respect to the ball (green outlines). These positions (orange dots) were automatically detected and recreated in the simulation. Rotational velocities of the spherical treadmill generated by three real flies (blue) were compared with those generated by NeuroMechFly (orange) for **(D)** forward, **(E)** lateral, and **(F)** yaw axes. Spearman correlation coefficients (ρ) comparing blue and orange traces are indicated.



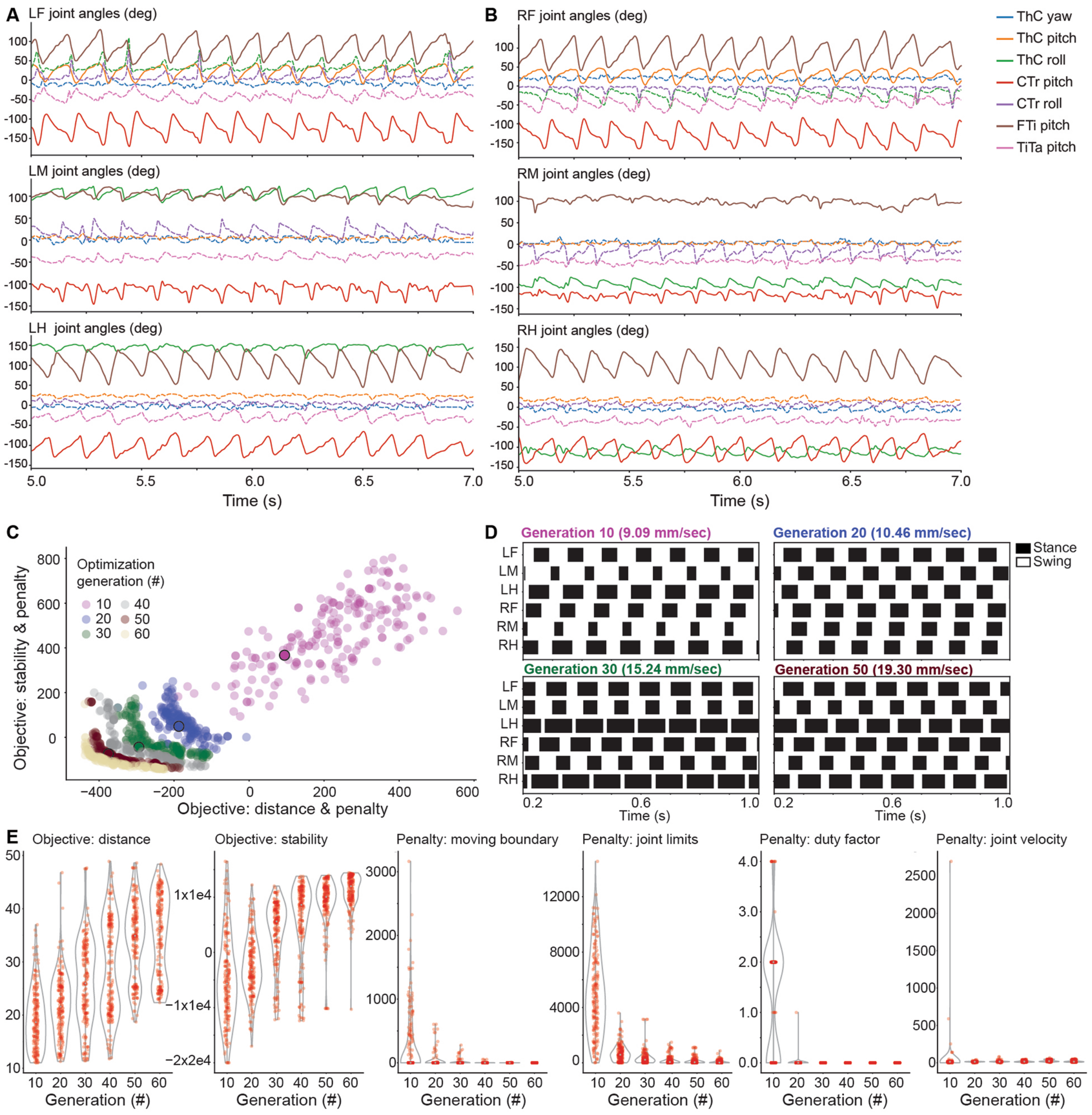
Extended Data Fig. 7 | Comparing real and simulation predictions for gait diagrams during tethered walking. Gait diagrams showing manually annotated stance phases for three real flies (**A-C**, gold) as well as those obtained from estimated ground reaction forces in NeuroMechFly (blue). Percentage of overlap in real and simulated stance phases (green) is quantified. 'R' and 'L' indicate right and left legs, respectively. 'F', 'M', and 'H' indicate front, middle, and hind legs, respectively.



Extended Data Fig. 8 | Comparison of walking paths and velocities for real tethered walking versus kinematic replay in a tethered or untethered model. Leg kinematics from a tethered walking experiment (blue) were used for kinematic replay in NeuroMechFly either tethered on a simulated spherical treadmill (orange) or freely walking on flat ground (green). Shown are resulting **(A)** integrated walking paths, as well as associated **(B)** forward, **(C)** lateral, and **(D)** yaw velocities.



Extended Data Fig. 9 | The impact of the morphological realism on estimates of leg–leg and leg-antenna contact during grooming. Collision diagrams from kinematic replay of foreleg–antennal grooming when using either **(A)** NeuroMechFly's morphologically detailed legs and antennae, or after replacing its **(B)** forelegs, or **(C)** forelegs and antennae with simple cylinders, as in a conventional stick skeletal model.



Extended Data Fig. 10 | Joints controlled and comparison over generations when optimizing for fast and statically stable tethered walking. 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. The remaining four DoFs per leg (dashed lines) were fixed during optimization because they did not exhibit pronounced angular changes. **(C)** Pareto front approximations for six optimization generations. The more negative values indicate the more optimal objective functions. Four individual solutions dominated by the pareto optimal solutions were selected for more in-depth analysis (10th (purple), 20th (blue), 30th (green), and 50th (dark red); all are outlined in black). **(D)** Gait diagrams from selected solutions. Stance (black) and swing (white) phases were calculated by reading-out tarsal ground contacts for each leg. Indicated are the velocities of each solution as calculated by averaging the spherical treadmill forward velocity. **(E)** Progression of weighted objective values (shown without sign inversion) and penalties over the course of 60 generations. Objectives (distance and stability coefficients) increase across generations, while penalties decrease or converge to, or near, zero. The objective distance (mm) is the distance traveled in 2 s. The penalty duty factor is the number of legs violating the duty factor constraint. The remaining penalties are shown in Arbitrary Units.

nature research

Corresponding author(s): Pavan Ramdya

Last updated by author(s): Jan 21, 2022

Reporting Summary

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

Statistics

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

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

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

Software and code

Policy information about availability of computer code

Data collection	The code used for recording the experiments regarding kinematic replay is available in a public GitHub repository: https://github.com/NeLy-EPFL/SeptaCam.git
Data analysis	The code for all analysis was written in Python 3 and is available for editors and reviewers as a CodeOcean capsule. Furthermore, it is available in a public GitHub repository along with a full description of use. https://github.com/NeLy-EPFL/NeuroMechFly Documentation of the code can be also found at https://nely-epfl.github.io/NeuroMechFly

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

Data

Policy information about availability of data

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

- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

Data for tethered walking is available at https://doi.org/10.7910/DVN/Y3TAEC. Data for tethered grooming is available at https://dataverse.harvard.edu/dataverse/DeepFly3D

Field-specific reporting

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

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

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

Life sciences study design

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

Sample size	No sample size calculation was performed. The estimations obtained from the physics engine in our simulations don't depend on the number of examples. We run several experiments to verify the consistency of the results. Population sizes for optimization were determined empirically and chosen to have a reasonable computation time (maximum 24 h per run).
Data exclusions	Unclassifiable behaviors were excluded for error analysis. We focused on walking or grooming.
Replication	The simulation has been run numerous times on different computers and on different operating systems. We consistently replicated our results.
Randomization	Randomization was not relevant for our study since we do not have experimental groups. We describe a neuromechanical framework and its usability. However, the wild-type animals used in our kinematic replay experiments were randomly chosen.
Blinding	Blinding was not relevant for our study because we are presenting a neuromechanical modeling framework. We only have wild-type animals which are, in principle, indistinguishable from each other.

Reporting for specific materials, systems and methods

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

Materials & experimental systems

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

Methods

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

Animals and other organisms

Policy information about studies involving animals; ARRIVE guidelines recommended for reporting animal research

Laboratory animals	Wild-type Drosophila melanogaster female flies were used. They were raised at 25°C on a 12 h light/dark cycle and tested at 2-4 days post-eclosion.
Wild animals	No wild animals were used in this study.
Field-collected samples	No field-collected samples were used in this study.
Ethics oversight	Drosophila melanogaster studies were performed in accordance with an institutional authorization by Swiss national authorities (A120851-21)

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