Kinematics and sensorimotor control of walking in fruit flies

Lili Karashchuk

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2023

Reading Committee:
Bingni Brunton, Chair
John Tuthill, Chair
Katherine Steele
Thomas Daniel

Program Authorized to Offer Degree:
Neuroscience

University of Washington

Abstract

Kinematics and sensorimotor control of walking in fruit flies

Lili Karashchuk

Co-Chairs of the Supervisory Committee:
Professor Bingni Brunton
Biology

Associate Professor John Tuthill
Physiology and Biophysics

Animals move with remarkable agility, fluidly adjusting their limbs as they seamlessly move from one behavior to another. Although the movements may look effortless, they are generated by complex feedback circuits which integrate proprioceptive and exteroreceptive inputs with internal state to precisely move muscles. Understanding this circuitry would be not only scientifically satisfying, but also aid in the design of agile robots and medical interventions for motor impairements.

In this thesis, we provide some insight into this circuitry by proposing new frameworks for quantifying animal motion and modeling the sensorimotor loop. In Chapter 2, we introduce Anipose, an open-source toolkit for robust markerless 3D pose estimation. We apply Anipose to quantify the 3D kinematics of walking in *Drosophila melanogaster* and discover a hitherto undescribed degree of freedom: the rotation of the femur limb segment in each of the legs. In the course of applying Anipose to various datasets, we found that the manual annotations took up a disproportionate amount of time when setting up 3D tracking. Taking a step towards addressing this problem, in Chapter 3 we introduce BKinD-3D, a method for discovering 3D keypoints from video without requiring any annotations. Together, these contributions form a firm foundation for 3D animal pose estimation.

Next, we turn our attention to modeling the feedback circuitry underlying movement. In Chapter 4, we argue for building multiscale models of sensorimotor loops to elucidate the role of proprioceptive feedback during behavior. We follow this prescription in Chapter 5 as we build a model of the sensorimotor loop of walking in *Drosophila melanogaster*. Our model includes sensory and actuation delays as well realistic kinematics based on data obtained from tracking with Anipose. When we combine these two features, we discover that flies move as fast as their physiology permits for robust walking.

We conclude with future directions for improving frameworks for 3D pose estimation and modeling sensorimotor loops in Chapter 6.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

Reflecting on my whole graduate experience, it is hard to overstate how much I have changed. I have met so many inspiring people and through them learned different ways to understand the nervous system and to experience the greater world. I have grown as a person through conversations with new and old friends, experimented with different interests, experienced the joys of another gender, and conducted state of the art research. I have been incredibly lucky to be surrounded by many many supportive people that made all of this possible. As far as I can tell, there is no limit to the acknowledgement section, so I figured I would just thank everyone here.

Advisors

Above all, I would like to thank my co-advisors Bing Brunton and John Tuthill.

My initial conversations with Bing Brunton brought me to the University of Washington. I felt that she would support me throughout grad school and indeed she did. She has taught me so much about academia and expanded my conception of what a lab environment could be like. Our later conversations helped me refine my opinions on the unique value of systems integration in neuroscience (and especially sensorimotor control), a perspective which I plan to carry into my next adventure. Being part of the Brunton lab meetings, I feel I have gained a tremendous breadth in modeling techniques, all with an eye towards the magic of natural behavior. Bing pushed me to focus my modeling on reducing the complex patterns in existing data rather than trying to build an abstract model from the ground up, which has been a fruitful perspective. Bing has also introduced me to Lisa Li and Jen Sun, who were my collaborators for the later part of my PhD.

When I watched John Tuthill's talk at the physiology and biophysics departmental retreat, it felt like a new world was opened to me. I had never thought about fly sensorimotor control in this way and it was fascinating, allowing me to see a rather different perspective on neuroscience. This pattern has continued as John has gently challenged me to reach beyond my comfort zone and thus I learned about light field microscopy, fly husbandry and genetics, how to assemble computers, how to record from various high-end cameras, and somehow a lot about fly courtship through journal club as well. John also introduced me to the greater fly neuroscience community, which has been particularly interesting as it underwent a connectomics revolution over the course of my PhD. He provided the necessary biological grounding to keep my modeling work relevant to the neuroscience community throughout graduate school and beyond.

Collaborators

Next, I would like to thank all of my collaborators over the years.

Working with Lisa Li has been an absolute pleasure. Although we were over 1,000 miles apart, it often felt we were in the same room when debugging tricky controls code. Our collaboration led me to a new understanding of how control theory can be applied for studying biological system, largely due to Lisa's expertise on control theory.

It has been really fun to work with Jen Sun, Amil Dravid, and Serim Ryou. Although I had done research and read a lot of papers on computer vision, I had not previously published in an actual computer vision conference. Going through this process helped me gain a deeper perspective on the different motivations for computer science and neuroscience researchers. I'd like to thank Jen for welcoming me into this collaboration.

I have worked with multiple undergraduate students and working with each one has been a rewarding experience. First and foremost, I would like to thank Katie Rupp, who truly amazed me with all of her contributions over the 4 years that we worked together. She tested out some

fly walking models, helped annotate fly data, contributed a new training augmentation to DeepLabCut, contributed to Anipose documentation, added behavioral annotation capability to the Anipose visualizer, and explored some analyses on fly grooming kinematics. Her enthusiasm has been contagious and helped me maintain excitement throughout my PhD. Second, I would like to thank Sonia Fereidooni and Kaan Ingec, who worked with me through some dead-ends in unsupervised pose estimation, eventually leading to BKinD-3D. Thank you to Khanh Bui as well, who helped me think through what's next for semi-supervised pose estimation.

On the experimental side, I would like to thank Evyn Dickinson, Sarah Walling-Bell, and Elischa Sanders. Evyn worked with me through many early iterations of Anipose to make 3D tracking work for his setup. Sarah took over Evyn's setup and upgraded it further to record a lot more data, which ended up forming the dataset for the first analysis of 3D walking kinematics in *D. melanogaster*. Elischa trusted me with his data of mouse reaching and worked with me to make Anipose work on his setup. Thank you also to Brandon Pratt, Raveena Chhibber, and Chris Dallmann who trusted my code enough to use Anipose to analyze their experiments.

To track fly legs precisely requires a lot of manual annotations. Thank you to Hibah Javed, John So, Cindy Kibue, Sam Mak, Marche Cantiberos, Katie Rupp, and Evyn Dickinson for contributing manual annotations of calibration boards, fly legs and body, and mouse hands. Together, we labeled 15,032 frames of fly legs! Having this much help has made this venture manageable and also dramatically reduced the number of fly leg fueled nightmares I experienced.

Early in my PhD, I spent 6 months in a dark place trying all kinds of techniques to precisely calibrate cameras, going deep into printer settings to print small and precise calibration boards. None of it worked. Stephen Huston finally showed me the way by lending his fly-sized calibration board. Darren Dixon at Applied Image worked with me to manufacture similar

ones for our lab. Thank you to both for actually making my PhD feasible!

Thank you to Tony Azevedo for teaching me how to attach flies to image their muscle activity. His videos were so cool and it was so satisfying to be able to reproduce his setup! Thank you to Chris Dallmann for running "light field club" with me,[1] and also setting up the initial light field rig to image muscles. Thanks to Raveena Chhibber for taking on this project and getting ever closer to achieving my dream of obtaining simultaneous 3D kinematics and muscle imaging of a whole fly leg. I also cherish the many conversations I had with Chris and Raveena in that small and windowless room that currently houses the light field rig.

Thank you to Steve Brunton for patiently explaining Runge-Kutta methods to me at a cafe outside on a sunny afternoon. I had spent months trying everything else to stabilize my numerical integration and his patient explanation helped me get unstuck.

Aditya Nair gave me in-depth tutorials on the math of coupled oscillators over the course of a few months, to the point that fitting them and simulating them now feels like second nature. Thank you for passing on this super power.

Thank you to my commitee members: Tom Daniel, Kat Steele, and Sasha Aravkin. I had wonderful discussions with each of them about different aspects of the project. These helped me reframe the way I describe my projects and often also suggested new directions for ongoing work. I have grown to enjoy presenting for them at the yearly committee meetings, which they graciously made time for in their exceedingly busy lives.

Thank you to everyone with whom I've interacted with through the Brunton lab: Aaron Garcia, Aditya Nair, Ali Weber, Gabrielle Strandquist, Elliott Abe, Harsha Gurnani, Kaan Ingec, Katie Stanchak, Maryam Bahadori, Michelle Hickner, Nancy Wang, Raveena Chibber, Samantha Sun, Sara Ichinaga, Sarah Pugliese, Satpreet Singh, Seth Hirsh, Sonia Fereidooni, Tanvi Deora, Zeynep Toprakbasti, Zoe Steine-Hanson.

Thank you also to everyone with who I've interacted at the Tuthill lab: Akira Mamiya,

[1]Light field club had just two members: Chris and me. We would read light field papers and discuss them every other week until we finally understood what light field imaging meant and how to do it.

Anne Sustar, Tony Azevedo, Anthony Moussa, Brandon Mark, Brandon Pratt, Chenghao Chen, Chris Dallmann, Dominic Golding, Evyn Dickinson, Ellen Lesser, Grant Chou, Jessica Jones, Katie Rupp, Kiet Tran, Leila Elabbady, Lylah Deady, Pralaksha Gurung, Raveena Chhibber, Sara Pierce, Sarah Pugliese, Sarah Walling-Bell, Srinidhi Naidu, Su-Yee Lee, Sweta Agrawal, Sydney Brannoch, Urvashi Jha, Vanshika Balaji, Yichen Luo

Friends

I would also like to thank all the people who helped me finish this thesis, but with whom I primarily interacted outside of the lab.

I had many lovely conversations with David Holz over the years which reframed my view of how the world worked. In the context of this thesis, he reminded me of mode-adaptive networks for modeling movement data, which ended up working much better than I expected. I remember fondly the week we walked around San Francisco and talked at length on how to do pose estimation effectively with little manually annotated data. This helped me focus on unsupervised pose estimation over synthetic data generation.

I had some really thoughtful discussions with Moya Chen on the relationship between artificial neural networks and biological neural networks, which helped refine my thoughts on using one to model the other. This led me on a path to a much more satisfying story for the layered walking model.

Moya also organized the fateful wine and cheese party where I met a computer vision researcher, who at 2 in the morning after multiple drinks told me that a reprojection error of 30 pixels is shit and really I want to aim for 3 pixels or less, thus making me revisit my calibration algorithm. I am thankful that I got this computer vision lore.

Thank you to the entire Evolution Devices team: Elie Celnikier, Pier Mantovani, Juan Rodriguez, Andrew Ekelem, Aashyk Mohaiteen, Lisa Donahue. Working with patients with sensorimotor deficits has given me a greater perspective on the relevance of my work. My

first experience with setting up a 3D setup occurred in Berkeley with the Evolution Devices team for humans, thanks to a grant from the Toyota Mobility Foundation. This gave me the confidence to tackle it for flies.[2] In addition, the experience of planning, applying for grants, mentoring interns, and overall running a startup together has been invaluable and has provided me with a sense for different possible lives, a foundation for when my research hit road blocks. I am thankful that everyone involved has tolerated me living a double life for the past 6 years.

Thank you to all the co-conspirators at the UW Synaptech club (listed in roughly chronological order): Manjari Anant, Devyansh Gupta, Morgan Graham, Kirsten Graham, Chetana Iyer, Toma Itagaki, Alejandro Striefel. It has been amazing to see the club grow from 4 people meeting on Saturday afternoons to hackathons that bring in over 50 people. I'm really thankful to all these people who shared their passion for neurotechnology and neuroscience with me, and reminded me each week how interesting and applicable these can be. Thank you to Eric Chudler for all his support for the club, especially in advising on getting financial support and in procuring us a room.

Thank you to my neuroscience cohort, who have taught me so much about neuroscience and made moving to Seattle feel so much less lonely: Alainna Brown, Arena Manning, Brit Baskin, Devin Wehle, Ellen Lesser, Jordan Elum, Sneha Ray, Jesse Miles, Su-Yee Lee, Todd Appleby, Tony Bigelow.

Thank you especially to Tony Bigelow, who has helped me move across so many different places and with whom I've probably talked the most about the struggles of being a graduate student over so many different nights.

I'd like to thank all my friends who I didn't get a chance to thank above, especially Ena Hariyoshi, Herman Chau, Jon Scott, Jonathan Murphy, Jordan Brown, Dennis Xing, Zoe Tjoelker, Doron Reuven, Corry Lee, and Tomás Vega. Each of them have taught me so much

[2]It turned to be much harder than I anticipated though...

about myself and the world. It's hard to imagine who I would be if not for all the time with them.

Thank you to my cat Maxwell, whose affection has kept me sane. During the pandemic, there were stretches of months where he was the only one that I physically interacted with. I'm not sure where I would be without him.

Thank you to my brother Valentino Vargas, who ended up going to UW for undergrad because I was there. His love and openness have helped me maintain some connection to my family through everything.

Lastly, I would like to thank my fiancée Melissa Shemwell. She has been my partner in so many wonderful adventures, conversations, and sometimes even adventurous conversations. She has taught me how to relax and has helped me discover myself. The last year of my PhD was particularly emotionally strenuous, combining anxieties from both my gender transition and career trajectory. Melissa supported me through that year and gave me the mental strength to finish this thesis.

DEDICATION

To all my friends

Chapter 1

INTRODUCTION

Animals must move to navigate the world. To do this effectively, they often rely on sensory inputs to gauge the position of each of their limbs: proprioceptive signals. Although proprioception is crucial for executing fluid movements, how the nervous system encodes and integrates this sense for movement is unclear. This is troubling for two reasons. The first is that many neurological disorders (e.g. MS, ALS) affect proprioceptive senses in addition to motor pathways, so a proper plan for rehabilitation must consider the interplay between these in order to be effective [91][281]. However, how the proprioceptive and motor pathways integrate is unclear, so the design of new interventions is currently guided primarily by theories of motor control [156]. This lack of understanding similarly impedes the design of moving artificial systems. As a result, most systems today rely heavily on visual rather than proprioceptive guidance [194][147], which makes them bulky, expensive, and fragile. In this thesis, we provide some insight into this circuitry by proposing new frameworks for quantifying animal motion and modeling the sensorimotor loop in fruit flies. In this section, we review past work in order to motivate our approach.

1.1 Advantage of studying sensorimotor control in fruit flies

If the processing of proprioception is so important, why hasn't it already been characterized? Integration of proprioceptive signals with motor pathways occurs starting from the peripheral nervous system. Unfortunately, this system has been historically hard to record from and manipulate in behaving animals, due to its enclosure by bone and sensitive tissue in vertebrates. Furthermore, the cells contributing to proprioception and motor control do not have a straightforward topology, making it hard to untangle the contributions of various

circuits based purely on electrophysiological recordings.

Genetic tools have recently been developed to make this problem tractable in the fruit fly, Drosophila melanogaster. Specifically, GAL4-UAS [77] and LexA-lexAop [157] expression systems allow us to target specific cells for recording and manipulations, thus making it easier to perform replicable and targeted experiments. Furthermore, these experiments can now yield even more information, as new fluorescent indicators may now be used to image different aspects of neuronal activity at scale, ranging from calcium levels [67] to voltage activity [2]. Various labs have also engineered new proteins to activate or inhibit neurons based on light activity [29], or to kill off cells altogether [291]. The combination of all of these recent tools, along with the easier access of the peripheral nervous system in invertebrates, make for an exciting opportunity to unravel the computations underlying proprioception and motor control in Drosophila melanogaster.

1.2 Models of walking

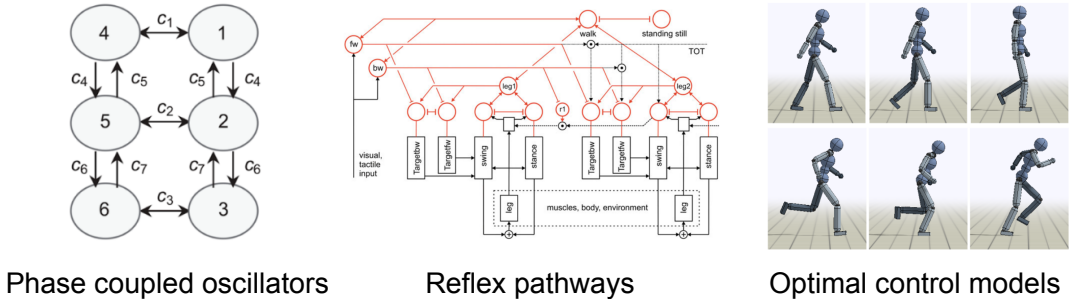

Figure 1.1: Schematic of different models of walking. Models are primarily based on phase coupled oscillators [63], reflex pathways [242], or optimal control [211].

As each experiment can only provide a partial view of the whole sensorimotor system, scientists have proposed various models to integrate observations and compare mechanisms across animals. Currently, these models primarily focus on walking, as it is a relatively simple but important behavior. These may be broadly broken down into three categories (see

Figure 1.1): (1) rule-based models based on mechanical considerations and past literature, (2) weakly coupled phase oscillators models, (3) control-based models which guide simulations to maximize some property of locomotion.

The rule-based models integrate the past literature on motor control to generate a few simple rules which are presumed to be responsible for walking (e.g. "loading of the hind left leg triggers unloading of front right leg") [242]. Such models are generally most strictly based on the past literature and when they are based on a few rules can be easy to interpret. However, the influence of each rule can be hard to interpret when these models become more complex. Furthermore, the tuning of parameters is often done in an ad hoc way, revealing little intuition about why the system may be organized in this way. Finally, such models are typically based fully on sensory feedback, ignoring the effect of descending and rhythm generating neurons, which have been shown to be crucial in coordinating proper walking.

Dynamical system theory shows that any network with cyclical dynamics may be modeled as a set of weakly coupled phase oscillators, so another line of research has been exploring this view of walking [63]. Each oscillator may model a muscle, joint angle, or even a whole leg, and is coupled to some set of other oscillators. These models have rich theory behind them, allow for some extensions based on sensory feedback, and can be fit to kinematics [63]. However, they need to be heavily constrained to fit to data and by themselves make no claims on which gaits allow the animal to move faster, more robustly, or more efficiently.

Control-based models start with a simulation based some description of the animal's anatomy and optimize control parameters to generate some behavior. These may be based on a musculoskeletal model [99], skeletal model [120] [211], or even simply a ball and stick model [267]. As these models generate locomotion by optimizing some metric, they can be used to explore the space of optimal locomotion behaviors under some assumptions. However, they typically do not generate plausible hypotheses on the neural coding underlying the behaviors and are typically sensitive to modeling assumptions.

1.3 Brief history of capturing walking in fruit flies

To validate each of these models, animal gait must be characterized. Motivated by clinical and industrial applications, scientists have been tracking and analyzing human kinematics for decades [198]. This sprouted a vast research literature on the subject, with now models of human gait based on stability and energy usage [156] and models to recover neuromuscular control based on kinematics [250]. However, quantification of fly and in general invertebrate kinematics has been comparatively underexplored. Due to the historical difficulty in setting up markers on insects and cost in developing a 3D tracking system, researchers have primarily focused on tracking foot falls [190], [260] or gross kinematic parameters (e.g. walking or turning speed) [196][164][36]. More recently, researchers have also started to quantify the movement of all the joints in the fly in two dimensions [212]. However, the 3D kinematics of a walking fruit fly remain unknown. Beyond the fly, as far as we are aware, only two studies have been conducted on 3D kinematics in insects: one for stick insects [65] and another for cockroaches [25]. Together, these provide some guidance on what may be expected in fruit flies. Still, the mechanics of walking may be different in flies due to their small size [267], so it is unclear how similar the statistics of the gait are.

1.4 Layout of thesis

In this thesis, we provide some insight into this circuitry by proposing new frameworks for quantifying animal motion and modeling the sensorimotor loop. In Chapter 2, we introduce Anipose, an open-source toolkit for robust markerless 3D pose estimation. We apply Anipose to quantify the 3D kinematics of walking in *Drosophila melanogaster* and discover a hitherto undescribed degree of freedom: the rotation of the femur limb segment in each of the legs. In the course of applying Anipose to various datasets, we found that the manual annotations took up a disproportionate amount of time when setting up 3D tracking. Taking a step towards addressing this problem, in Chapter 3 we introduce BKinD-3D, a method for discovering 3D keypoints from video without requiring any annotations. Together, these contributions form

a firm foundation for 3D animal pose estimation.

Next, we turn our attention to modeling the feedback circuitry underlying movement. In Chapter 4, we argue for building multiscale models of sensorimotor loops to elucidate the role of proprioceptive feedback during behavior. We follow this prescription in Chapter 5 as we build a model of the sensorimotor loop of walking in *Drosophila melanogaster*. Our model includes sensory and actuation delays as well realistic kinematics based on data obtained from tracking with Anipose. When we combine these two features, we discover that flies move as fast as their physiology permits for robust walking.

We conclude with future directions for improving frameworks for 3D pose estimation and modeling sensorimotor loops in Chapter 6.

Chapter 2

ANIPOSE: A TOOLKIT FOR ROBUST MARKERLESS 3D POSE ESTIMATION

This chapter was published in *Cell Reports* in 2021 [146].

Authors: Lili Karashchuk, Katie L. Rupp, Evyn S. Dickinson, Sarah Walling-Bell, Elischa Sanders, Eiman Azim, Bingni W. Brunton, and John C. Tuthill.

2.1 Abstract

Quantifying movement is critical for understanding animal behavior. Advances in computer vision now enable markerless tracking from 2D video, but most animals move in 3D. Here, we introduce Anipose, an open-source toolkit for robust markerless 3D pose estimation. Anipose is built on the 2D tracking method DeepLabCut, so users can expand their existing experimental setups to obtain accurate 3D tracking. It consists of four components: (1) a 3D calibration module, (2) filters to resolve 2D tracking errors, (3) a triangulation module that integrates temporal and spatial regularization, and (4) a pipeline to structure processing of large numbers of videos. We evaluate Anipose on a calibration board as well as mice, flies, and humans. By analyzing 3D leg kinematics tracked with Anipose, we identify a key role for joint rotation in motor control of fly walking. To help users get started with 3D tracking, we provide tutorials and documentation at www.anipose.org.

2.2 Introduction

Tracking body kinematics is key to answering questions in many scientific disciplines. For example, neuroscientists quantify animal movement to relate it to brain dynamics [187, 248], biomechanists quantify the movement of specific body structures to understand their

mechanical properties [8, 25], social scientists quantify the motion of multiple individuals to understand their interactions [246, 114], and rehabilitation scientists quantify body movement to diagnose and treat disorders [255, 56, 229]. In all of these disciplines, achieving rapid and accurate quantification of animal pose is a major bottleneck to scientific progress.

While it is possible for human observers to recognize body movements, scoring behaviors by eye is laborious and often fails to detect differences in the rapid, fine-scale movements that characterize many behaviors. Methods for automated tracking of body kinematics from video have existed for many years, but they typically rely on the addition of markers to identify and disambiguate body parts. Although such methods can achieve very precise pose estimation [177], the use of markers is often impractical, particularly when studying natural behaviors in complex environments, tracking multiple body parts, or studying small animals. Thus, there is a pressing need for methods that perform automated, markerless tracking of body kinematics.

Recent advances in computer vision and machine learning have dramatically improved the speed and accuracy of markerless body pose estimation [187]. There are now a number of tools that apply these methods to track animal movement from 2D videos, such as DeepLabCut [185], SLEAP [214], DeepPoseKit [109], among others [47, 169, 35]. These software packages allow users to label keypoints, train convolutional neural networks, and apply them to identify keypoints from videos; several toolkits also include auxiliary tools, such as visualizing and filtering the tracked keypoints. Among them, DeepLabCut is the most widely used [186].

While tracking of animal movement from 2D video is useful for monitoring specific body parts, full body pose estimation and measurement of complex or subtle behaviors require tracking in three dimensions. Multiple tools have emerged for 3D tracking and body pose estimation, including DANNCE [78], FreiPose [312], DeepFly3D [113], and OpenMonkeyStudio [19]. However, these tools use fundamentally distinct network architectures, workflows, and user interfaces from popular 2D tracking methods. Out of the existing 2D tracking tools, only DeepLabCut [201] supports triangulation with up to 2 cameras. However, three or more

cameras are often required to resolve pose ambiguities, such as when one body part occludes another. Thus, there is a need for additional tools that allow users to extend their existing 2D tracking setups to achieve robust 3D pose estimation while preserving their established workflows.

Here, we introduce Anipose (a portmanteau of "animal" and "pose"), a toolkit to quantify 3D body kinematics by integrating DeepLabCut tracking from multiple camera views. Anipose consists of a robust calibration module, filters to further refine 2D and 3D tracking, and an interface to visualize and annotate tracked videos. These features allow users to analyze 3D animal movement by extracting behavior and kinematics from videos in a unified software framework. Below, we demonstrate the value of 3D tracking with Anipose for analysis of mice, fly, and human body kinematics (Figure 2.1). Applying 3D tracking to estimate joint angles of walking *Drosophila*, we find that flies move their middle legs primarily by rotating their coxa and femur, whereas the front and rear legs are driven primarily by femur-tibia flexion. We then show how Anipose can be used to quantify differences between successful and unsuccessful trajectories in a mouse reaching task. Finally, we visualize how specific leg joint angles map onto a manifold of human walking.

We designed Anipose to make 3D tracking accessible for a broad community of scientists. Because it is built on DeepLabCut, Anipose allows users to easily upgrade from 2D to 3D tracking, as well as take advantage of the DeepLabCut community, documentation, and continued support. To help users get started, we provide in-depth tutorials and documentation at `http://anipose.org`. The release of Anipose as free and open-source Python software facilitates adoption, promotes ongoing contributions by community developers, and supports open science.

2.3 Results

We implement 3D tracking in a series of steps: estimation of calibration parameters from calibration videos, detection and refinement of 2D joint keypoints, triangulation and refinement of keypoints to obtain 3D joint positions, and computation of joint angles (Figure 2.2).

In addition to the processing pipeline, the key innovations of Anipose are a robust 3D calibration module, spatiotemporal filters that refine pose estimation in both 2D and 3D, and a visualization incorporating videos, tracked keypoints, and behavioral annotations in one interface. We evaluated the calibration and triangulation modules without filters by testing their ability to accurately estimate lengths and angles of a calibration board with known dimensions (Figure 2.1A) and to track the hand of a mouse reaching for a food pellet (Figure 2.1B). We then evaluated how filtering improves estimation in 3D of position and time derivative of walking flies (Figure 2.1C) and humans (Figure 2.1D).

2.3.1 Robust calibration of multiple camera views

An essential step in accurate 3D pose estimation is precise camera calibration, which determines the relative location and parameters of each camera (i.e., the focal length and distortions). We implemented an automated procedure that calibrates the cameras from simultaneously acquired videos of a standard calibration board (e.g., checkerboard or ChArUco board) moved by hand through the cameras' fields of view (Figure 2.2A). We recommend the ChArUco board because its keypoints may be detected even with partial occlusion and its rotation can be determined uniquely from multiple views. The pipeline starts by detecting keypoints on the calibration board automatically using OpenCV [33], based on the board's geometric regularities (e.g., checkerboard grid pattern, specific black and white markers). These board detections are used first to initialize camera calibration parameters from arbitrary positions through a greedy algorithm which adds edges between cameras one by one until it reaches a fully connected tree (Figure S2.1A).

Although some tracking tools (e.g., [47, 78]) stop at the initial estimate of camera parameters based on estimated calibration board orientation from different cameras, we found that this is often not sufficient to obtain accurate camera calibrations, especially when there are few frames with a detected board. To resolve this issue, we implemented procedures that optimize the camera calibration parameters to minimize the reprojection error of the calibration board keypoints, referred to as bundle adjustment in the camera registration

literature [274]. We implemented bundle adjustment with standard (least-squares) as well as robust losses (Huber and soft L1). Furthermore, we developed an iterative procedure we term "iterative bundle adjustment", which performs bundle adjustment in multiple stages, using only a random subsample of detected keypoints points in each stage (see Methods for a detailed description). This procedure automatically tunes the outlier thresholds and minimizes the impact of erroneous keypoint detections and bad camera initialization. Each of these bundle adjustment procedures improves the reprojection error from the initial estimate (Figure S2.1B). Iterative bundle adjustment produced marginally better results, but with no parameter tuning, so we use this as the default in Anipose.

2.3.2 Accurate reconstruction of physical lengths and angles in 3D

An important test of any calibration method is whether it can accurately reconstruct an object with known dimensions. We evaluated the Anipose calibration and triangulation toolkit by asking whether it could estimate the lengths and angles of a precisely manufactured ChArUco board [98].

We first compared the accuracy of tracking the 9 corners of the ChArUco board (Figure 2.3A) with three methods: manual annotation, neural network detections, and OpenCV detections (example detections in Figure 2.3B). Although manual annotations are typically assumed to be the ground truth in tracking animal kinematics, we started by assessing the reliability of manual annotations relative to high-precision, sub-pixel resolution keypoint detection based on the geometry of the ChArUco board with OpenCV [33, 98]. Relative to the OpenCV points, the manual keypoint annotations had a mean error of $(0.52, -0.75)$ pixels and standard deviation of (2.57, 2.39) pixels, in the (x, y) directions, respectively (Figure 2.3C). These observations provide a useful baseline of manual annotation accuracy.

We evaluated the accuracy of reconstructing ChArUco board lengths and angles as estimated by three methods: manual keypoint annotations, OpenCV keypoint detections, and neural network keypoint detections (see Methods for detailed descriptions). As our ground-truth dataset, we chose the known physical lengths and angles between all pairs of 9

corners on the ChArUco board. The ChArUco board was manufactured with precise tolerance ($< 2\,\mu m$), which allowed us to evaluate the accuracy of lengths and angles from manual keypoint annotations and OpenCV keypoint detections, which are commonly taken to be the ground truth. As expected, OpenCV detections had the lowest error in length and angle, as they leveraged prior knowledge of the ChArUco board geometry to make high-precision corner estimates (Figure 2.3D). Surprisingly, neural network (trained with DeepLabCut) predictions had a lower error than manual annotations, despite the network itself being trained on manual annotations. More than 90% of poses estimated by Anipose had an error of less than 20 μm in length and 1 degree in angle, relative to the true dimensions of the ChArUco board (Figure 2.3D). These results demonstrate the efficacy of camera calibration with Anipose and serve as useful bounds of expected performance.

2.3.3 Animal tracking in 3D

We evaluated the triangulation of markerless tracking on three different animal datasets (Figure 2.3E-G). For each dataset, we computed the error of estimated joint positions and angles on labeled animals withheld from the training data. The error in estimated joint angles was $<16^\circ$ in over 90% of frames, and $<10^\circ$ in over 75% of frames. Furthermore, the error in the estimated joint position was <18 pixels (approximately 1.6mm, 0.14mm, 86mm for mouse, fly, and human datasets respectively) in over 90% of frames and <12 pixels (approximately 1mm, 0.09mm, 57mm for mouse, fly, and human datasets respectively) in over 75% of frames. Importantly, the position error in units of camera pixels is roughly comparable across these three datasets, spanning more than 3 orders-of-magnitude in spatial scale. Therefore, we believe these errors are representative of what can currently be expected for accuracy of 3D markerless tracking.

Although triangulation usually resulted in accurate estimates of joint positions and angles, there were still some frames where it failed due to missing keypoint detections (as in Figure 2.3E). In other cases, incorrect keypoint detections led to erroneous 3D joint position estimates (as in Figure 2.3F). Even though these issues occurred in a small minority of frames, tracking

errors are especially problematic for analyzing movement trajectories. For instance, missing estimates complicate the estimation of derivatives, whereas erroneous estimates bias the distribution of summary statistics. To minimize these issues, we leveraged complementary temporal and spatial information within each dataset to refine tracking performance in 3D.

2.3.4 Addition of filters to improve tracking accuracy

Naturally behaving animals present unique challenges for 3D pose estimation. Animals can contort their bodies into many different configurations, which means that each behavioral session may include unique poses that have not been previously encountered, even across multiple animals. Our approach to tackling these challenges is to leverage prior knowledge that animal movements are usually smooth and continuous, and that rigid limbs do not change in length over short timescales. In particular, we developed and implemented a set of 2D and 3D filters that refine keypoints, remove errors in keypoint detections, and constrain the set of reconstructed kinematic trajectories. We demonstrate that both sets of filters work together to significantly improve pose estimation. Here we focus on detailed quantification of these filters in tracking flies and humans, where our datasets included keypoints at every limb joint tracked with at least 4 camera views.

Refining keypoints in 2D

We implemented three distinct algorithms to remove or correct errors in 2D keypoint detection: a median filter, a Viterbi filter, and an autoencoder filter. The median and Viterbi filters operate on each tracked joint across frames, and the autoencoder filter refines keypoints using learned correlates among all joints. The median filter removes any point that deviates from a median filtered trajectory of user-specified length, then interpolates the missing data. The Viterbi filter finds the most likely path of keypoint detections for each joint across frames from a set of top (e.g., 20) detections per frame, given the expected standard deviation of joint movement in pixels as a prior. Finally, the autoencoder filter corrects the estimated score of each joint based on the scores of the other joints, with no parameters set by the user.

Where errors in tracking cannot be corrected by filtering, the keypoint is removed altogether, since the missing joint can be inferred from other camera views, but an erroneous keypoint can produce large discrepancies in triangulation. We document the parameters we used to produce results across the paper in Table S2.1. Anipose users are encouraged to evaluate the effect these filtering parameters may have on their analyses. Depending on the particulars of the experimental setup, including the spatial and temporal resolution of the videos, the parameters may need to be adjusted.

The addition of each filtering step noticeably improved the tracking of fly leg joints (Figure 2.4A). The median and Viterbi filters both reduced spurious jumps in keypoint position, which may occur if the neural network detects a similar keypoint on a different limb or at another location in the frame. The Viterbi filter is able to remove small erroneous jumps in detected keypoint trajectories while also preserving high frequency dynamics, whereas the median filter may mistakenly identify fast movements as an error and remove them. The autoencoder filter removed detections for keypoints which were typically not visible from a given view, which improved 3D position estimates after triangulation (Figure S2.2).

For each of the 2D filters, we quantified the performance improvement of estimating the joint position and angle on manually annotated validation datasets. The 2D median filter significantly reduced error in joint position and angle estimation on the human dataset ($t = -14.8, p < 0.001$ for position, $t = -7.7, p < 0.001$, paired t-test) but not on the fly dataset ($t = -1.2, p = 0.2$ for position, $t = -0.98, p = 0.3$, paired t-test). The Viterbi filter reduced error on both fly and human datasets ($t = -4.4$ and $t = -4.1$ for fly position and angle, $t = -10.9$ and $t = -8.7$ for human position, with $p < 0.001$ for all, paired t-test). The autoencoder filter also reduced error in joint positions and angles on the fly dataset ($t = -5.4, p < 0.001$ for positions, $t = -2.16, p = 0.03$ for angles, paired t-test). We did not apply the autoencoder filter to human tracking, since all occluded points are annotated in the training dataset. In the fly dataset, applying the autoencoder filter after the Viterbi filter further improved the joint position and angle estimates above the autoencoder ($t = -3.97, p < 0.001$ for positions, $t = -3.44, p < 0.001$ for angles, paired t-test). In

summary, we found the addition of these three filters improved the ability of Anipose to accurately estimate joint positions and angles.

Refining poses and trajectories in 3D

To further refine joint position and angle estimates in 3D, we developed a triangulation optimization that takes advantage of the spatiotemporal structure of animal pose and behavior. Specifically, our optimization produces pose estimates that are smooth in time using temporal regularization, and limbs demarcated by adjacent keypoints that are constant in length with spatial regularization. The length for each limb is automatically estimated in the optimization. The relative strengths of the temporal and spatial regularization terms may be balanced and tuned independently. As with the 2D filters, we empirically determined default strengths that worked across multiple datasets. A complete description of each filter, along with all the parameters, is detailed in the Methods. For illustration, we compared the performance of these filters (Figure 2.5A) to other commonly used methods from the literature (Random sample consensus, or RANSAC, triangulation and 3D median filter) on the walking fly dataset. We applied the 3D filters on kinematic trajectories partially corrected with 2D filtering (Viterbi then autoencoder filters for the fly dataset, and Viterbi filter only for the human dataset), to evaluate how much the 3D filters improved the accuracy. Spatiotemporal regularization substantially improved pose estimation. The temporal regularization noticeably reduced jitter in the trajectory (Figure 2.5A), while the spatial regularization stabilized the estimate of limb length (Figure S2.3B). These improvements are also obvious in example videos of reconstructed pose before and after filtering (Video 2).

For each of the 3D filters, we quantified the improvement in position and angle error relative to tracking with 2D filters alone (Figure 2.5C and S2.3C). We found that RANSAC triangulation did not improve position and angle error. The 3D median filter significantly reduced position and angle errors relative to only 2D filters for the human dataset ($t = -11.8$ for position, $t = -7.3$ for angle, $p < 0.001$ for both, paired t-test), but not for the fly dataset. Spatial and temporal regularization applied together provided the largest reduction in tracking

error ($t = -18.7$ and $t = -6.1$ for human positions and angles, $t = -10.8$ and $t = 5.8$ for fly positions and angles, $p < 0.001$ for all, paired t-test). Overall, we find that the 3D filters implemented in Anipose significantly improve pose estimation.

Improving estimation of derivatives

In addition to tracking body pose, it is often valuable to track the speed of body movements. We compared the temporal derivative of 3D joint positions estimated with Anipose to the derivative computed from manual annotations (Figure 2.5B and D) and found both qualitative and quantitative improvements to estimation of body movement speed.

Filtered trajectories produced smoother derivatives, due to the fact that tracking errors are corrected through 2D and 3D filtering, and the temporal regularization explicitly penalizes deviations from smoothness (Figure 2.5B). It is challenging to evaluate the accuracy of Anipose derivative estimates because computing finite difference derivatives of manual annotations amplifies known errors in these annotations. Given that manual annotations deviate from the ground truth tracking with a standard deviation of at most 3.5 pixels in distance (Figure 2.3C), we expect computing the finite difference derivative of such annotations to produce derivatives with error of 4.95 pixels (about 0.037 mm corresponding to 11.1 mm/s over one frame in the fly dataset). Therefore, the manual annotations (dark green trace in Figure 2.5B) do not represent the true derivative, but rather a noisy approximation of the true derivative. The temporally regularized trajectory resembles this estimate of the derivative but is more smooth because of temporal regularization. The strength of this regularization, and the subsequent smoothness of the tracked keypoints, is a parameter that users may fine-tune (see [37] for a systematic way to tune this parameter). We suggest some default values and provide guidance on choosing parameters in the Discussion.

We found that the 2D filters (Viterbi and autoencoder in fly, only Viterbi in human) improved the error in derivative by 2.78 mm/s for the fly dataset ($t = -9.4, p < 0.001$, paired t-test) and by 30.0 mm/s on the human dataset ($t = -28.0, p < 0.001$, paired t-test) relative to no filters. The 3D median filter improved the error in derivative by 1.65 mm/s in the

fly dataset ($t = -4.8, p < 0.001$, paired t-test) and by 177.3 mm/s in the human dataset ($t = -324, p \ll 0.001$, paired t-test) RANSAC improved error in the derivative estimate by 2.16 mm/s in the fly dataset ($t = -7.07, p < 0.001$, paired t-test) but did not improve the error in the human dataset. The spatiotemporal regularization improved the error in derivative by an additional 0.67 mm/s for the fly dataset ($t = -4.10, p < 0.001$, paired t-test) and by 217.7 mm/s on the human dataset ($t = -213, p \ll 0.001$, paired t-test) relative to the 2D filters. Overall, we found that the filters implemented in Anipose significantly improved the estimation of body movement in the fly and human datasets.

2.3.5 Structured processing of videos

Animal behavior experiments are often high-throughput, meaning that large numbers of videos are recorded over many repeated sessions with different experimental conditions. To make the process of 3D tracking scalable to large datasets, we designed a specific file structure (Figure S2.5) to organize and process behavior videos, configuration files, and calibration data. This file structure also facilitates scalable analysis of body kinematics across individual animals and experimental conditions. For example, the command `anipose analyze` detects keypoints for each video in the project folder, and `anipose calibrate` obtains calibration parameters for all the cameras in all calibration folders. Each command operates on all videos in the project, circumventing the need to process each video individually. In addition, this design allows the user to easily reanalyze the same dataset using different filtering parameters or with different 2D tracking libraries (e.g., to compare DeepLabCut and SLEAP). For the users that prefer to set up their own pipelines, we also package the calibration, triangulation, and filtering functions in a separate library called aniposelib.

2.3.6 Visualization of tracking

The large number of videos and keypoints tracked in many behavior experiments make it challenging to visualize the resulting data. In addition, the large files created with high-speed video often make it impractical to store and visualize an entire dataset on a laptop.

To facilitate evaluation and interpretation of data tracked with Anipose, we developed a web-based visualization tool (Figure 2.6). The tool shows, for a given trial, each camera view, 3D tracking, and 2D projections of the tracked keypoints. The user can speed up and slow down the speed at which the videos play and rotate the tracked keypoints in 3D. By taking advantage of the standardized file structure, the interface provides a dropdown menu to navigate between trials and sessions. The interface also allows the user to annotate the behaviors in each video, which is particularly useful for isolating specific behaviors for further analysis. As this tool is web-based, it may be run on a server, allowing users to preview videos and inspect tracking from any computer. Furthermore, if the server is public, users may easily share links to particular trials with collaborators to point out specific behaviors (example here) .

2.3.7 3D tracking with Anipose provides insights into motor control of Drosophila walking

We first used 3D tracking with Anipose to analyze the leg joint kinematics of fruit flies walking on a spherical treadmill. Although fly walking has been studied in great detail from a 2D perspective [69, 190, 27], 3D joint kinematics of walking flies have not previously been analyzed. Thus, it was not clear how fly leg joints move during walking. Specifically, we sought to understand the relative contributions of leg joint flexion and rotation.

Some limb joints are not restricted to movement in a single plane, but can also rotate around the long axis of a limb segment. Whereas the importance of rotation angles has long been recognized for human gait analysis [230], rotation angles have been comparatively understudied in other animals. This gap exists largely because estimating rotation angles requires precise tracking of joint kinematics in 3D.

The fly leg consists of five segments, whose movements are defined by 8 angles (1 abduction, 3 rotation, 4 flexion). We observed significant rotations between the coxa and femur segments during walking. Figure 2.7A shows trajectories of coxa rotation, femur rotation, and femur-tibia flexion angles for one walking bout.

Interestingly, the magnitude of joint rotation varied across different legs. Although the

femur-tibia flexion angle has a high range of motion in the front and back legs, the femur-tibia flexion angle has a comparatively smaller range of motion in the middle legs (Figure 2.7B). In contrast, the middle legs are primarily driven by coxa and femur rotation. Furthermore, the coxa joints of contralateral legs rotate in opposing directions. These results suggest that the circuitry that coordinates walking (e.g., the central pattern generator) cannot be the same for all six legs. Rather, walking circuits must control different motor neurons and muscles to generate unique joint kinematics for each leg.

In addition to comparing joint angle distributions across legs, we analyzed trajectories of 3D leg kinematics across flies. We used the UMAP nonlinear embedding method [188] to embed coxa rotation, femur rotation, and femur-tibia flexion angles and their derivatives of all legs (Figure 2.7C). The three-dimensional embedding of joint kinematics formed a mushroom-shaped manifold. Individual flies reside at specific regions of the manifold, but for all flies, step phase is distributed along the circumference of the cap (Figure 2.7D). These results are consistent with the existence of a continuum of walking gaits across flies [69], but also suggest that different flies have slightly distinct walking kinematics. This analysis also demonstrates how 3D tracking can be used to dissect the contributions of specific joints to complex motor behaviors. Visualizing a manifold of 3D joint kinematics provides a means to understand how joint kinematics vary within the high-dimensional space of a motor control task (Figure 2.7E, Figure S2.6B).

2.3.8 Analysis of 3D mouse reaching and human walking kinematics

To illustrate the value of 3D tracking with Anipose for studying other animal species, we analyzed data from reaching mice and walking humans. Joint positions and angles have long been used to quantify movement in both healthy and impaired animals [152, 20, 97]. However, previous quantification has relied primarily on laborious manual tracking or marker-based tracking with extensive manual corrections. Here we demonstrate analysis of mouse and human behavior using fully automated 3D tracking with the Anipose toolkit.

We first analyzed 3D hand trajectories from mice trained to reach for and grasp a pellet.

This task has been extensively used to study neural circuits for sensorimotor control underlying skilled limb movements [17, 23, 111, 165, 89, 86]. Using the Anipose visualization tool, we labeled the reach outcome and start/end frame for each trial. We labeled the trial a "hit" if the mouse successfully grasped the pellet, a "miss" if the mouse missed the pellet holder, and a "bump" if the mouse bumped into the pellet holder or the pellet but failed to grasp the pellet. Each of the four mice in the dataset had multiple instances of each outcome. Figure 2.7F shows example 3D reaching trajectories, which demonstrate that reaching movements vary significantly from trial to trial (see also Figure S2.7A). Although reaching is a challenging behavior to track, due to its speed and variability, Anipose was able to accurately reconstruct forelimb reaching trajectories. The trajectory of each movement was variable, but plotting the distance to the pellet holder as a function of time to contact revealed that each reach type has a stereotyped trajectory (Figures 2.7G and S2.7B). Interestingly, the hit/bump and miss trajectories diverged around 50 ms prior to pellet contact, suggesting that mice are unable to correct their reaching trajectories in this period.

We next analyzed 3D walking kinematics reconstructed from the human dataset using methods similar to our analysis of fly walking. We extracted knee flexion, hip rotation, and hip flexion angles from 3D joint positions tracked with Anipose (Figures 2.7H and S2.7C). The distributions of these joint angles are symmetric across the two legs (Figures 2.7I and S2.7D) and match previous characterizations of human gait [97]. To characterize the structure of walking across the subjects, we used the UMAP nonlinear embedding method [188] to embed knee flexion, hip rotation, hip flexion, and their derivatives into a 3D space, as for the fly dataset above. The UMAP embedding reveals a manifold of angle coordination across subjects (Figure 2.7J). The manifold forms a cylindrical structure with the knee flexion angle mapping circularly along the cylinder (Figure 2.7K). The two trials that are to the left outside the main cylinder have lower variation of left leg hip rotation (Figure S2.7E). These examples illustrate the ease and utility of tracking and analyzing human walking behavior with Anipose. In the future, this approach could be used to automatically identify individuals with distinct walking gaits or other motor patterns.

2.4 Discussion

In this paper, we introduce Anipose, an open-source toolkit to accurately track animal movement in 3D. Anipose is designed to augment DeepLabCut, a toolkit for 2D markerless tracking [185], with calibration, filters, and a visualization tool to facilitate robust 3D tracking and analysis. Current users of DeepLabCut can easily upgrade to 3D tracking with Anipose by adding and calibrating additional cameras to an existing behavioral setup. We validated each optimization module and the full pipeline against ground truth data from four different experimental datasets and three organisms, demonstrating accurate reconstruction of 3D joint positions and angles. To help users get started, we developed detailed tutorials for both the Anipose pipeline and aniposelib at anipose.org.

The Anipose tracking pipeline is designed to streamline structured processing of videos recorded in high-throughput experiments. Users do not need to know Python to use the Anipose pipeline. All that is required to get started is editing a small configuration file and running the provided commands from a terminal. Although we designed Anipose to leverage 2D tracking with DeepLabCut [185], it can be made compatible with other 2D markerless tracking methods, including SLEAP [214] and DeepPoseKit [109] by modifying a single file. Users with programming experience can convert their 2D tracked data to the Anipose structure (see Figure S2.5) to take advantage of the calibration, filters, and visualization tools. We also provide access to individual functions via a separate library, aniposelib.

2.4.1 Impact of robust markerless 3D tracking

A key technical advantage of tracking with Anipose is the ability to interpret and analyze movement speed from 3D pose trajectories that are smooth in space and time, due to filtering and interpolation from multiple camera views. The resulting improvements in tracking smoothness make it easier to analyze pose and movement dynamics. Specifically, interpolated data enables the user to obtain better estimates of behavior statistics, such as mean and variance, and to perform dimensionality reduction techniques, such as principal component

analysis (PCA). Additionally, temporal regularization reduces noise in the first derivative and thus enables the user to obtain more precise estimates of movement speed (Figures 2.5D and S2.4).

This ability to analyze 3D pose trajectories may open up opportunities for behavioral neuroscience, where key insights have been gained through carefully controlled behavioral paradigms. In particular, experiments are often designed to accommodate the practical limitations of movement tracking, recording neural activity, and perturbing the animal in real time (e.g., [278, 82, 207, 36, 28]). Recent advances in experimental technologies (e.g., high-density extracellular recording probes [142], optical imaging of fluorescent reporters [67, 2], and optogenetics [29]) have made it feasible to precisely record and perturb neural activity from animals behaving freely in three dimensions. Complementing these technologies, a comprehensive toolbox for high-throughput 3D tracking will not only enable deeper analysis of current experiments, but also make it possible to study more natural behaviors.

A robust 3D markerless tracking solution could also greatly expand the accessibility of quantitative movement analysis in humans. Many neurological disorders, including some commonly thought of as cognitive disorders, affect walking gait [259, 294] and upper limb coordination [252, 271]. Many clinicians and basic researchers currently rely on qualitative evaluations or expensive clinical systems to diagnose motor disorders and assess recovery after treatment. While clinical approaches are commercially available [293], they are costly, require proprietary hardware, rely on the addition of markers to the patient, and cannot assess walking gait in natural contexts such as a patient's home. Anipose could be used as a tool in the diagnosis, assessment, and rehabilitative treatment of movement and neurodegenerative disorders.

2.4.2 Insights into the motor control of Drosophila walking

By analyzing 3D joint kinematics of tethered walking *Drosophila*, we found that each leg has a unique set of joint angle distributions. One valuable insight, which was not evident from 2D tracking alone, is that the movement of the middle legs is driven primarily by femur rotation,

in contrast to the front and hind legs, which are driven primarily by femur-tibia flexion. We also observed small differences in femur-tibia flexion and femur rotation distributions between front and hind legs (Figure 2.7B). Thus, the neural circuits that move each leg during walking must be specialized for controlling joints with distinct forces and dynamics within each leg. Previous models of *Drosophila* walking have used an identical control architecture for intra-leg joint coordination for all six legs [11, 101]. Our results provide a framework for constructing more biologically plausible neuromechanical models using distinct architectures for controlling different joints within each leg.

Inter-leg differences in joint kinematics also raise questions about limb proprioception. Proprioceptors in the fly femoral chordotonal organ (FeCO) encode femur-tibia flexion and movement [173]. Does the role of the FeCO differ for the middle legs, for which the femur-tibia generally does not flex in a rhythmic pattern during walking? Which proprioceptors, if any, are used to sense femur and coxa rotation of the middle legs? Answering these questions will be facilitated by combining Anipose with *in vivo* measurements and perturbations of proprioceptive neural circuits [66].

Rythmic motor behaviors, such as walking, are thought to be controlled by central pattern generators (CPGs): neural circuits that generate intrinsic rhythmic activity [30]. If fly walking is controlled by CPGs, our results suggest that the CPG for each leg must control different muscles. For example, we would predict that a walking CPG for the front legs would connect to motor neurons that control the tibia flexor and extensor muscles in the femur [14]. In contrast, a CPG for the middle legs might connect to motor neurons innervating muscles in the trochanter that control femur rotation. These insights will be useful in guiding ongoing efforts to trace motor control circuits using connectomic reconstruction of the *Drosophila* ventral nerve cord [175] and leg [153].

Femur rotation is also likely to be important for walking in other insect species. Fransevich and Wang tested the passive rotation of the trochanter-femur articulation in 23 insect species and found rotation ranges from 10°to 120°, depending on the species [93]. Our estimate for the physiological range for walking *Drosophila* is about 70°(Figure 2.7B), which falls within

the trochanter-femur articulation range observed in other insects. Thus, it is plausible that articulation of the trochanter-femur joint is sufficient to account for the femur rotation we measured during walking, and that other insects rely on femur rotation during walking as well. As an example, Bender et al. reported different kinematics across legs in walking cockroaches, with larger femur rotation and smaller femur-tibia flexion in the middle legs relative to the hind legs [25]. The application of Anipose to track 3D joint kinematics in other species will enable further comparative studies of the biomechanics and neural control of walking.

2.4.3 Potential for future improvement based on related work

Camera calibration has long been a rich topic in computer vision research. The most commonly used calibration code, based on Zhang's work [308] and part of OpenCV [33], can calibrate up to 2 cameras using images of checkerboards from multiple angles. Although this method can be used to calibrate 3 or more cameras by calibrating pairs of cameras, in practice, precise calibration requires an additional optimization step called bundle adjustment [274]. Bundle adjustment has been a key part of structure from motion toolkits [5, 245], but the method has received comparatively little attention as a solution to camera calibration for markerless tracking. An exception is DeepFly3D, which supports calibration based on animal keypoints but not based on a calibration board, which hinders its ability to handle setups with arbitrary camera positions [113]. Our key innovation is to provide an open source implementation of sparse bundle adjustment targeted for camera calibration for motion tracking. Our current implementation could eventually benefit from incorporating other methods from the literature. For instance, using a neural network to detect the calibration board may yield more detected keypoints and lead to more robust calibration under difficult conditions [127]. Currently, Anipose requires a calibration board to initialize camera parameters (even with animal calibration), but it may be possible to initialize camera parameters based on commonly detected points, as is commonly done in the structure from motion literature [5, 245], or perhaps by using a neural network directly [279]. Bundle adjustment itself may be made more robust by incorporating gauge constraints in the optimization function, further reducing

the number of parameters [274]. Finally, the calibration process itself may be streamlined if it were made interactive [228].

There has been extensive recent work to improve markerless tracking based on deep learning approaches. One common approach has been to improve the neural network architecture for training. For instance, this approach has been used to induce priors in the neural network based on occlusions [238, 55], multi-view geometry [134, 312, 78, 302], limb lengths [310], or time [205]. We note that this approach is complementary to our work, as the Anipose filters could be used with keypoint detection by any neural network. Another approach is to resolve tracking by using pictorial structures to add priors on limb lengths [301, 10, 113] or motion [297] or both [306]. The Viterbi filter used in Anipose is analogous to the motion based pictorial structures and could be further extended to handle priors on limb lengths based on insights from these papers. Beyond tracking single animals, toolboxes like SLEAP [212], OpenPose [47], and DeepLabCut [201] have some support for multi-animal pose estimation in 2D. For tracking multiple animals in 3D, a promising approach is to build correspondences based on geometry and appearance [76] across multiple views. As automated, high-throughput tracking of animal behavior grows in scale, new methods for data analysis, visualization, and modeling will also be needed to gain insight into the neural control of dynamic behavior [303, 177, 28, 66].

2.4.4 Limitations and practical recommendations

There are several common scenarios under which Anipose may fail to produce accurate 3D tracking. Below, we enumerate some of the scenarios we have encountered in applying Anipose on different datasets and suggest practical strategies for troubleshooting.

As is the case for any tracking system, the ability of Anipose to track and estimate body pose is fundamentally limited by the quality of the underlying data. High quality videos are well illuminated, contain minimal motion blur, and provide coverage of each keypoint from different views. A common failure mode we encountered was when the neural network misplaced 2D keypoints in some frames. If the errors are uncorrelated across camera views,

then the Anipose filters can compensate and still produce accurate tracking in 3D. But in some cases, multiple views have correlated errors or these errors persist in time. These type of errors most commonly arise when the neural network has not been trained on a subset of rare behaviors, so that the animal adopts poses unseen by the trained network. One solution to reducing the frequency of such errors involves systematically identifying outlier frames, manually relabeling them, then retraining the network. Anipose supports this functionality, as do other tracking toolboxes [185, 212, 109, 113].

Poor multi-camera calibration also results in tracking errors. A good calibration should have an average reprojection error of less than 3 pixels, and ideally less than 1 pixel. To obtain a quality calibration, the calibration videos should be recorded so that the board is clearly visible from multiple angles and locations on each camera. If it is not possible to achieve this, we suggest exploring a preliminary calibration module in Anipose that refines an initial calibration based on the detected points on the animal itself. This module was inspired by the animal based calibration in DeepFly3D [113], but our implementation uses the initial calibration from a calibration board as a starting guess, permitting generalization in different setups. It also takes advantage of our iterative calibration procedure to yield robust calibration even with errors in tracking.

An effective experimental setup needs to have an appropriate number of cameras to track all keypoints across possible pose configurations. In particular, each joint must be visible from at least 2 cameras at all times. Thus, for tracking multiple limbs or body parts, we recommend at least 3 equally spaced cameras, so that half of the body is visible from any single camera. We evaluated this quantitatively in the human dataset (Table S2.2), where there is a dramatic reduction in error from 2 to 3 cameras.

The mouse reaching dataset is one example where tracking was reasonably accurate without filters, but filters did not further improve tracking accuracy. There are several potential explanations for this result. The reaches are very short (about 40-100 frames or 200-500ms) and the hand is hard to see when it is on the ground, so temporal filters such as the Viterbi filter or temporal regularization lack the information to resolve tracking errors.

There are very few keypoints (only 3 per hand) and these can change in distance relative to each other, so the spatial regularization cannot impose strong constraints. With only 2 cameras, the spatiotemporal regularization cannot fully leverage multiple views to remove outliers (Table S2.2) and the autoencoder has limited utility. In this situation, using basic linear least-squares triangulation works well enough for analysis (Figure 2.7F and G). The accuracy of tracking mouse reaching might be improved by labeling more keypoints on each hand, increasing the camera frame rate, and adding more cameras.

As a practical starting point, we recommend users start with no filters to first evaluate the quality of the tracking. If outliers or missing data impede data analysis, then we recommend enabling the default filter parameters in Anipose, which we have found to produce good tracking results across multiple datasets. In some cases, some additional tuning of parameters may be required, especially on datasets with unique constraints or when studying behaviors with unusual dynamics. If any joints are not visible for an extended period of time in certain videos, we recommend disabling the spatiotemporal optimization, as it can hallucinate trajectories, increasing overall error (as in Table S2.2). We provide suggestions for tuning parameters in our documentation at anipose.org.

2.4.5 Outlook

We designed Anipose to make markerless 3D tracking simple and broadly accessible for the scientific community. With this goal in mind, we built Anipose on DeepLabCut, a widely used 2D tracking toolkit. As many labs develop machine learning tools for behavior tracking and analysis, we advocate for pooling efforts around common frameworks that emphasize usability [145, 239]. In particular, we suggest that tools be built in a modular way, so that code can be extended and reused in other frameworks. We hope that the Anipose toolkit contributes to these community efforts. We welcome contributions to improve and extend the Anipose toolkit and conversely are ready to contribute the ideas and code from Anipose to other toolkits.

2.5 Methods

2.5.1 Resource availability

Lead contact Further information and requests for resources should be directed to and will be fulfilled by the lead contact, John Tuthill (tuthill@uw.edu).

Materials availability This study did not generate new unique reagents.

Data and code availability

- Data has been deposited at https://doi.org/10.5061/dryad.nzs7h44s4 and are publicly available as of the date of publication. DOIs are listed in the key resources table.
- All original code has been deposited at `https://doi.org/10.5281/zenodo.5224213`. Documentation for the software is available at anipose.org. DOIs are listed in the key resources table.
- Any additional information required to reanalyze the data reported in this paper is available from the lead contact upon request.

2.5.2 Experimental model and subject details

Mouse Reaching data were obtained from four adult C57BL/6 mice (JAX:000664, ∼8-12 weeks old, two male and two female) trained to reach for a pellet. Procedures performed in this study were conducted according to US National Institutes of Health guidelines for animal research and were approved by the Institutional Animal Care and Use Committee of The Salk Institute for Biological Studies.

Fly Male and female Berlin wild type *Drosophila melanogaster* (RRID:BDSC_8522), 4 days post-eclosion, were used for all experiments. Flies were reared on standard cornmeal agar food on a 14 hr/10 hr light-dark cycle at 25 °C in 70% relative humidity.

Human We evaluated 3D tracking with Anipose on the Human 3.6M dataset [132, 49]. The Human 3.6M dataset contains data from 5 subjects as a training dataset (2 female and 3 male), 2 subjects as a validation dataset, and 4 subjects as a testing dataset (2 female and 2 male).

2.5.3 Method details

ChArUco dataset. To evaluate the performance of Anipose compared to physical ground truth, we collected videos of a precision-manufactured ChArUco board [98]. The ChArUco board was manufactured by Applied Image Inc (Rochester, NY) with a tolerance of 2 µm in length and 2° in angle. It is a 2 mm × 2 mm etching of opal and blue chrome, on a 5 mm × 5 mm board. The ChArUco pattern itself has 6 × 6 squares, with 4 bit markers and a dictionary size of 50 markers. With these parameters, the size of each marker is 0.375 mm and the size of each square is 0.5 mm. We filmed the ChArUco board from 6 cameras (Basler acA800-510µm) evenly distributed around the board (Figure 2.1A), at 30Hz and with a resolution of 832 x 632 pixels, for 2-3 minutes each day over 2 separate days. While filming, we manually rotated the ChArUco board within the field of view of the cameras. These videos were used as calibration videos for both the ChArUco dataset and the fly dataset detailed below.

We chose 9 of the corners as keypoints for manual annotation and detection (Figures 2.1A and 2.3A). We extracted and manually annotated 200 frames from each camera from day 1, and an additional 200 cameras per camera from day 2 (1200 frames per day, 2400 frames total). We used the frames for day 1 for training the neural network and the frames from day 2 for evaluation of all methods.

Mouse dataset. The reaching task is described in detail elsewhere [17]. Briefly, the training protocol consisted of placing the mouse in a 20 cm tall × 8.5 cm wide × 19.5 cm long clear acrylic box with an opening in the front of the box measuring 0.9 cm wide and 9 cm tall. A 3D-printed, 1.8 cm tall pedestal designed to hold a food pellet (20 mg, 3 mm diameter;

Bio-Serv) was placed 1 cm away from the front of the box opening and displaced to one side by 0.5 cm (to encourage mice to use their preferred forelimb), and food pellets were placed on top as the reaching target (Fig. 2.1B). Mice were food deprived to ∼85% of their original body weight and trained to reach for food pellets for either 20 minutes or until 20 successful reaches (defined as pellet retrieval) were accomplished. Mice were trained in this setup for 14 consecutive days before reaches were captured with 2 cameras (Sentech STC-MBS241U3V with Tamron M112FM16 16mm lens) placed in front and to the side of the mouse ($\sim 85^\circ$ apart). Videos were acquired at a frame rate of 200 Hz at a resolution of 1024×768 pixels.

We chose 6 points on the mouse hands as keypoints (Figure 2.1B). On each mouse hand, we labeled 3 points: the dorsal wrist, the base of digit 5, and the proximal end of digit 3. In total, we manually labeled 2200 frames (1100 frames per camera) for training the neural network from 2 mice. For test data to evaluate the post estimation performance, we labeled an additional 400 frames (200 frames per camera) taken from videos of 2 mice that were not in the training set.

Fly dataset. We next evaluated 3D tracking with Anipose on walking fruit flies. The flies' wings were clipped 24-48 hours prior to the experiment in order to increase walking and prevent visual obstruction of the legs and thorax. For all experiments, a tungsten wire was tethered to the dorsal thorax of a cold-anesthetized fly with UV cured glue. Flies were starved with access to water for 2–15 hours before they were tethered. After 20 minutes of recovery, tethered flies were positioned on a frictionless spherical treadmill [42, 112] (hand-milled foam ball, density: 7.3 mg/mm^3, diameter: 9.46 mm) suspended on a stream of compressed air (5 L/min). Six cameras (imaging at 300 Hz, Basler acA800-510µm with Computar zoom lens MLM3X-MP) were evenly distributed around the fly, providing full video coverage of all six legs (Figure 2.1C). Fly behavior was recorded in 2 second trials, capturing a range of behaviors such as walking, turning, grooming, and pushing against the ball. The recording region of each video was cropped slightly so that the fly filled the frame and the camera was able to acquire at 300Hz. For all training and test evaluation data, the interval between trials

was 25 seconds. For some of the flies in the larger walking dataset used in Figure 2.7, the interval between trials was set to 9 seconds.

We selected 30 points on the fly as keypoints (Figure 2.1C). On each fly leg, we labeled 5 points: the body-coxa, coxa-femur, femur-tibia, and tibia-tarsus joints, as well as the tip of the tarsus. In total, we manually labeled 6632 frames (about 1105 frames per camera) for training the neural network. For test data to evaluate the post estimation performance, we labeled an additional 1200 frames (200 frames per camera) taken from videos of 5 flies that were not in the training set. For analyzing flexion and rotation of angles during walking in Figure 2.7, we used a larger dataset of videos from 39 flies, all collected with the methods described above.

Human dataset. We evaluated 3D tracking with Anipose on the Human 3.6M dataset [132, 49]. Because this dataset has been used extensively for human pose estimation, it provides a useful comparison to existing computer vision methods. It consists of 11 professional actors performing a range of actions, including greeting, posing, sitting, and smoking. The actors were filmed in a 4m $\times$ 3m space with 4 video cameras (Basler piA1000) at a resolution of 1000 $\times$ 1000 pixels at 50Hz (Figure 2.1D). To gather ground-truth pose data, the actors were also outfitted with reflective body markers and tracked with a separate motion capture system, using 10 Vicon cameras at 200 Hz. Leveraging these recordings, the authors derived the precise 3D positions of 32 body joints and their 2D projections onto the videos. For camera calibration, we used the camera parameters from the Human 3.6M dataset, converted by Martinez et al. [182].

To compare the performance of Anipose against previous methods, we used a protocol from the literature [134]. We used frames from the training dataset to train the network and evaluated the predictions on the validation dataset. We also removed frames from the training dataset in which the subject did not move relative to the previous frame ($<$ 40mm movement of all joints from the previous frame). We evaluated the tracked human dataset on every 64th frame. We used 17 of the 32 provided joints as keypoints (Figure 2.1D). Iskakov

et al. [134] showed that some scenes from the S9 validation actor (parts of the Greeting, SittingDown, and Waiting actions) have ground-truth shifted in global coordinates compared to the actual position [134], so we exclude these scenes from the evaluation set. Furthermore, for subject S11, one of the videos is corrupted (part of the "Directions" action), so we exclude this from the dataset as well. In total, we obtained 636,724 frames (159,181 per camera) for training the neural network, and 8608 frames (2152 per camera) frames for evaluation.

Manual annotation of datasets. To produce neural network training data, we annotated the fly dataset using Fiji [244] and the VGG Image Annotator (VIA) [80, 81]. All the images in the fly test set were annotated with VIA. We annotated all the images in the ChArUco dataset and mouse dataset with VIA.

2.5.4 Quantification and statistical analysis

Neural network keypoint detections

Detection of keypoints in each of the datasets was performed with DeepLabCut 2.1.4 [201]. Briefly, to produce training data, we used k-means clustering to pick out unique frames from each of the views, then manually annotated the keypoints in each frame. We trained a single Resnet-50 [118] network for all camera views for the fly, mouse, and ChArUco datasets, starting from a network pretrained on Imagenet. For the human dataset, we started with a Resnet-101 network pretrained on the MPII human pose dataset [130]. During training, we augmented the training dataset with cropping, rotation, brightness, blur, and scaling augmentations using Tensorpack [299]. We then used the Anipose pipeline to run the network on each video. For each keypoint, the network produced a list of predicted positions, each associated with a confidence score (between 0 and 1). We saved the top-n most likely predictions of each joint location for each frame for use in Viterbi filtering of likely keypoints in 2D, as described below.

Filtering of 2D keypoint detections

The raw keypoint detections obtained with DeepLabCut were often noisy or erroneous (Figure 2.4). Thus, filtering the detections from each camera was necessary before triangulating the points. Anipose contains 3 main algorithms to filter keypoint detections; we elaborate on each algorithm below. Example applications of these filters and results are compared in Figure 2.4.

Median filter. The first algorithm identifies outlier keypoint detections by comparing the raw detected trajectories to median filtered trajectories for each joint. We started by computing a median filter on the detected trajectory for each joint's x and y positions, which smooths the trajectory estimate. We then compared the offset of each point in the raw trajectory to the median filtered trajectory. If a point deviated by some threshold number of pixels, then we denoted this point as an outlier and removed it from the data. The missing points were then interpolated by fitting a cubic spline to the neighboring points. The median filter is simple and intuitive, but it cannot correct errors spanning multiple frames.

Viterbi filter. To correct for errors that persist over multiple frames, we implemented the Viterbi algorithm to obtain a single most consistent path in time from the top-n predicted keypoints in each frame for each joint. To be specific, we expressed this problem as a hidden Markov model for each joint, wherein the possible values at each frame are the multiple possible detections of this keypoint. To obtain a cleaner model, we removed duplicate detections (within 7 pixels of each other) within each frame. To compensate for missed detected keypoints over many frames, we augmented the possible values at each frame with all detections up to F previous frames, weighted in time elapsed by multiplying their probability 2^{-F}. We then identified the best path through the hidden Markov model using the Viterbi algorithm [68]. This procedure estimates a consistent path, even with missed detections of up to F frames.

Autoencoder filter. We found that the network would often try to predict a joint location even when the joint was occluded in that view. This type of error is particularly problematic when used in subsequent 3D triangulation. The convolutional neural network confidence scores associated with these predictions can be high, making them difficult to distinguish from correct, high-confidence predictions. To remove these errors, inspired by [199], we implemented a neural network that takes in a set of confidence scores from all keypoints in one frame, and outputs a corrected set of confidence scores. To generate a training set, we made use of the fact that human annotators do not label occluded joints but label all of the visible joints in each frame. Thus, we generated artificial scores from biased distributions to mimic what the convolutional neural network might predict for each frame, with visible joints given a higher probability on average. Specifically, we sample the scores from a normal distribution, with standard deviation of 0.3 and mean 0 for invisible and 1 for visible joints, clipped to be between 0 and 1. To mimic false positive or false negative detections, we flip 5% of the scores ($x \to 1 - x$) at random. The task of the network is to predict a high score for each joint that is truly visible in that frame and a low score for any occluded joint. The network is a multilayer perceptron network with a single hidden layer and tanh activation units to perform this task. The size of the hidden layer is the number of joints (e.g. if there are 10 joint scores to predict, we set the hidden layer to 10 units). We trained the network using the Adam optimizer [149] implemented in the scikit-learn library [210]

Calibration of multiple cameras.

Camera model. A camera captures 2D images of light reflecting from 3D objects; thus, we can think of each camera as a projection, transforming 3D vectors to 2D vectors. To establish our notation, for a point $\boldsymbol{p} = (x, y, z)^T$ or $\boldsymbol{u} = (x, y)^T$, we use a tilde to denote that point in homogeneous coordinates (with a 1 at the end), so that $\tilde{\boldsymbol{p}} = (x, y, z, 1)^T$ or $\tilde{\boldsymbol{u}} = (x, y, 1)^T$.

A camera model specifies a transformation from a 3D point $\tilde{\boldsymbol{p}}$ to a 2D point $\tilde{\boldsymbol{u}}$. We use the camera model described by Zhang [308], which consists of a product of an intrinsics matrix $\mathbf{A}$, an extrinsics matrix $\mathbf{P}$, and a distortion function $\mathcal{D}$.

The extrinsics matrix $\mathbf{P} \in \mathbb{R}^{4\times3}$ describes how the camera is positioned relative to the world. We represent $\mathbf{P}$ as the product of a rotation matrix and a translation matrix. Both rotations and translations may be fully specified with 3 parameters each, for 6 parameters total in $\mathbf{P}$.

The intrinsics matrix $\mathbf{A} \in \mathbb{R}^{3\times3}$ describes the internal coordinate system of the camera. It is often modeled using 5 parameters: focal length terms f_x and f_y, offset terms c_x and c_y, and a skew parameter s:

$$A = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}.$$

In practice, we found that we obtain a more robust calibration by reducing the number of parameters, setting $f = f_x = f_y$, $s = 0$, and (c_x, c_y) to be at the center of the image, so that we need to estimate only the focal length parameter f for the intrinsics matrix.

The distortion function models nonlinear distortions in the camera pixel grid. This distortion is typically modeled with 3 parameters as

$$\mathcal{D}([x, y]) = \begin{bmatrix} x + x\left(k_1(x^2+y^2) + k_2(x^2+y^2)^2 + k_3(x^2+y^2)^4\right) \\ y + y\left(k_1(x^2+y^2) + k_2(x^2+y^2)^2 + k_3(x^2+y^2)^4\right) \end{bmatrix}.$$

In practice, we found that the higher-order distortion terms k_2 and k_3 are often small for modern cameras, so we assume $k_2 = k_3 = 0$ and only estimate a single parameter k_1.

Thus, the full mapping may be written as

$$\tilde{\boldsymbol{u}} = \mathcal{D}(\mathbf{AP}\tilde{\boldsymbol{p}}).$$

In total, the camera model involves estimating 8 parameters per camera: 6 for extrinsics, 1 for intrinsics, and 1 for distortion.

For the camera calibration and triangulation methods described below, we define the projection $\mathcal{T}$ from $\tilde{\boldsymbol{p}}$ to $\tilde{\boldsymbol{u}}$ as

$$\mathcal{T}(\tilde{\boldsymbol{p}}, \boldsymbol{\theta}_c) = \tilde{\boldsymbol{u}} = \mathcal{D}(\mathbf{AP}\tilde{\boldsymbol{p}}),$$

where $\boldsymbol{\theta}_c$ are the 8 parameters for the camera model of camera c.

Initial estimate of camera parameters. In order to calibrate the cameras and estimate parameters of the camera models, we start by obtaining an initial estimate of the camera parameters. We detected calibration board keypoints in videos simultaneously captured from all cameras. We then initialized intrinsics based on these detections following the algorithm from Zhang [308]. We initialized the distortion coefficients to zero.

We developed the following method to initialize camera extrinsics from arbitrary locations. For each pair of cameras, the number of frames in which the board is seen simultaneously is counted and used to build a graph of cameras. To be specific, each node is a camera, and edges represent pairs of cameras whose relation we will use to seed the initialization.

The greedy graph construction algorithm is as follows. Starting with the pair of cameras for which the number of frames the board is simultaneously detected is the largest, connect the two camera nodes with an edge. Next, proceed with iterations in decreasing order of the number of boards simultaneously detected. At each iteration, if the two nodes (cameras) are not already connected through some path, connect them with an edge. Processing iteratively through all pairs of cameras in this manner, a graph of camera connectivity is produced. Full 3D calibration is possible if and only if the graph is fully connected.

To initialize the extrinsics using this graph, we start with any camera and set its rotation and translation to zero. Then, we initialize its neighbors from the estimated relative pose of the calibration board between them using the initial intrinsics. This procedure is continued recursively until all cameras are initialized. A diagram of the camera initialization for an example dataset is provided in Figure S2.1A.

Bundle adjustment. To refine the camera parameters from initial estimates, we performed a bundle adjustment by implementing a nonlinear least-squares optimization to minimize the reprojection error [274]. Given all $\tilde{\boldsymbol{u}}_{c,j,t}$, the detected j^{th} keypoints from the calibration board at cameras c in frames t, we solve for the best camera parameters $\boldsymbol{\theta}_c$ and 3D points

$\tilde{\boldsymbol{p}}_{j,t}$ such that the reprojection loss $\mathcal{L}$ is minimized:

$$\mathcal{L} = \sum_{c} \sum_{j} \sum_{t} E\left(\tilde{\boldsymbol{u}}_{c,j,t} - \mathcal{T}(\tilde{\boldsymbol{p}}_{j,t}, \boldsymbol{\theta}_c)\right).$$

Here, $E(\cdot)$ denotes the norm using which the error is computed. This norm may be the least squares norm, but in practice, we used a robust norm, such as the Huber or soft ℓ_1 norm, to minimize the influence of outliers.

This optimization is nonlinear because the camera projection function $\mathcal{T}$ is nonlinear. We recognized that it is a nonlinear least-squares problem with a sparse Jacobian and thus solved it efficiently using the Trust Region Reflective algorithm [45, 34], as implemented in SciPy [282].

Iterative bundle adjustment. When calibrating cameras, we found that outliers have an outsized impact on calibration results, even when using robust losses such as the Huber loss or soft ℓ_1 loss. Thus, we designed an iterative calibration algorithm, inspired by the fast global registration algorithm from Zhou et al. [309], which solves a minimization with a robust loss efficiently through an alternating optimization scheme.

We approximate this alternating optimization in the camera calibration setting through an iterative threshold scheme. In our algorithm, at each iteration, a reprojection error threshold is defined and the subset of points $\boldsymbol{u}_{c,i}$ with reprojection error below this threshold is chosen. Bundle adjustment is then performed on these points alone. The threshold decreases exponentially with each iteration, to refine the points to be calibrated. The pseudocode for the algorithm is listed in Algorithm 1.

Triangulation and 3D filtering

The 3D triangulation task seeks 3D points $\boldsymbol{p}_{j,t}$ for joint j at frame t, given a set of detected 2D points $\boldsymbol{u}_{c,j,t}$ from cameras c with camera parameters $\boldsymbol{\theta}_c$. There are several common methods for solving this triangulation task. Below, we describe 3 of these methods, then describe our

method for spatiotemporally constrained triangulation. For illustration, a comparison of the performance of these methods is shown on an example dataset in Figure 2.5.

Linear least-squares triangulation. The first method triangulates 3D points by using linear least-squares [116]. Linear least-squares is the fastest method for multi-camera triangulation, but it may lead to poor results when the 2D inputs contain noisy or inaccurate keypoint detections. To be specific, we start with a camera model with parameters estimated from the calibration procedure described above, so that the extrinsics matrix $\mathbf{P}_c$, intrinsics matrix $\mathbf{A}_c$, and distortion function $\mathcal{D}_c$ are known for each camera c. By rearranging the camera model, we may write the following relationship:

$$\mathcal{D}_c^{-1}(\tilde{\boldsymbol{u}}_{c,j,t}) = \mathbf{A}_c\mathbf{P}_c\tilde{\boldsymbol{p}}_{j,t}.$$

We solved this linear system of equations using the singular value decomposition (SVD) of the product $\mathbf{A}_c\mathbf{P}_c$ to approximate the solutions for the unknown $\tilde{\boldsymbol{p}}_{j,t}$ [116].

Median-filtered least-squares triangulation. As a simple extension of least-square triangulation to correct some of the noisy detections, we applied a median filter to the resulting 3D points tracked across frames. This filtering improves the tracking, but at the cost of losing high frequency dynamics. Furthermore, a median filter does not improve triangulation if the original tracking is consistently poor.

RANSAC triangulation. Random sample consensus (RANSAC) triangulation aims to reduce the influence of outlier 2D keypoint detections on the triangulated 3D point, by finding the subset of keypoint detections that minimizes the reprojection error. We implemented RANSAC triangulation by triangulating all possible pairs of keypoints detected from multiple views and picking the resulting 3D point with the smallest reprojection error.

Formally, let $\tilde{\boldsymbol{p}}_{j,t}^{a,b}$ be the triangulated 3D point for keypoint j at frame t computed using the 2D keypoint detections from cameras a and b, then our algorithm finds $\tilde{\boldsymbol{p}}_{jt}$ using the

following relation:

$$\tilde{\boldsymbol{p}}_{j,t} = \underset{\tilde{\boldsymbol{p}}_{j,t}^{a,b}}{\arg\min} \left\| \mathcal{T}\left(\tilde{\boldsymbol{p}}_{j,t}^{a,b}, \boldsymbol{\theta}_a\right) - \tilde{\boldsymbol{u}}_{a,j,t} \right\|_2 + \left\| \mathcal{T}\left(\tilde{\boldsymbol{p}}_{j,t}^{a,b}, \boldsymbol{\theta}_b\right) - \tilde{\boldsymbol{u}}_{b,j,t} \right\|_2 .$$

Spatiotemporally regularized triangulation. We formulated triangulation as an optimization problem, which allowed us to specify soft spatiotemporal constraints (i.e. regularization) on the triangulated points. We propose that the points must satisfy three soft constraints: (1) the projection of the 3D points onto each camera should be close to the tracked 2D points, (2) the 3D points should be smooth in time, and (3) the lengths of specified limbs in 3D should not vary too much. Each of these constraints may be formulated as a regularization in the full objective function.

First, the **reprojection loss** is written as

$$L_{\text{proj}} = \sum_c \sum_j \sum_t E\left(\mathcal{T}\left(\tilde{\boldsymbol{p}}_{j,t}, \boldsymbol{\theta}_c\right) - \tilde{\boldsymbol{u}}_{c,j,t}\right).$$

Here, $E(\cdot)$ is a robust norm function such as the Huber or soft-ℓ_1 norm, to minimize the influence of outlier detections.

Second, the **temporal loss** is formulated as follows:

$$L_{\text{time}} = \sum_j \sum_t \left\| \tilde{\boldsymbol{p}}_{j,t} - \tilde{\boldsymbol{p}}_{j,(t-1)} \right\|_2 .$$

We extend this penalty to minimize higher-order (e.g. 2nd or 3rd) finite-difference derivatives, which produces smoother trajectories but has less impact on important high frequency dynamics (see Figure S2.4).

Third, the **limb loss** may be formulated by adding an additional parameter d_l for each limb l, defined to consist of joints j_1 and j_2:

$$L_{\text{limb}} = \sum_{l,j_1,j_2 \in \text{limbs}} \sum_t \left(\frac{\left\| \tilde{\boldsymbol{p}}_{j_1,t} - \tilde{\boldsymbol{p}}_{j_2,t} \right\|_2 - d_l}{d_l} \right)^2 .$$

The limb error is normalized relative to the limb length so that each limb contributes equally to the error.

Given each of the losses above, the overall objective function to minimize may be written as:

$$\mathcal{L} = L_{\text{proj}} + \alpha_{\text{time}} L_{\text{time}} + \alpha_{\text{limb}} L_{\text{limb}}.$$

We solve this sparse nonlinear least-squares problem efficiently using the Trust Region Reflective algorithm [45, 34], as implemented in SciPy [282], similarly to the bundle adjustment optimization. To initialize the optimization, we use linear least-squares triangulation. When formulated as a sparse nonlinear least-squares problem, the time and memory requirements of the optimization scale linearly relative to the number of input time points.

The parameters α_{time} and α_{limb} may be tuned to adjust the strength of the temporal or limb loss, respectively. Note, however, that the temporal loss is in units of distance, which may vary substantially across datasets. Thus, to standardize these parameters, we break down the parameter α_{time} in terms of a user-tunable parameter β_{time} and an automatically computed scale γ such that

$$\alpha_{\text{time}} = \beta_{\text{time}} \gamma.$$

We compute the scale γ as

$$\gamma = \frac{N}{\sum_j \sum_t \left\| \tilde{\boldsymbol{p}}_{j,t} - \tilde{\boldsymbol{p}}_{j,(t-1)} \right\|_2},$$

where $\tilde{\boldsymbol{p}}_{j,t}$ is an initial estimate obtained from linear least-squares triangulation. We found that the parameters $\beta_{\text{time}} = 2$ and $\alpha_{\text{limb}} = 2$ work well across a variety of datasets, and we used these parameters for tracking all four datasets in this manuscript. The user may additionally specify weaker constraints for the lengths of certain limbs to allow for some flexibility, such as the shoulder length in humans or the length of the tarsus in flies.

Estimating joint angles. We estimated joint angles from the tracked 3D positions. To compute the flexion angle defined by the three 3D points surrounding the joint $(\boldsymbol{p}_i, \boldsymbol{p}_j, \boldsymbol{p}_k)$,

where point $\boldsymbol{p}_j$ lies at the joint, the angle ϕ_j is

$$\phi_j = \arccos\left((\boldsymbol{p}_i - \boldsymbol{p}_j) \cdot (\boldsymbol{p}_k - \boldsymbol{p}_j)\right).$$

To estimate rotation and abduction angles, we solve an inverse kinematics problem treating the set of limb joints as a kinematic chain. When estimating limb angles from 3D coordinates of joints, the rotation of a joint is indistinguishable from the abduction of the next joint in the chain. We observed that fly and human limbs can be approximated to only have abduction at the joint closest to the body, so we resolve this ambiguity by assuming that only the first (most proximal) joint may abduct and the last (most distal) joint may not rotate.

The solution proceeds in two stages. In the first stage, we estimate the absolute rotation of each joint based on its {x, y, z} coordinate axes. The axes of the first joint match the coordinate system for the body. For other joints, the z axis is in the direction of the limb segment pointing from that joint away from the body, the x axis is in direction of proximal limb segment (towards the body) orthogonalized to the z-axis, and the y-axis is the cross product of the z-axis with the x-axis. In the second stage, the relative rotation between joints is computed and transformed to an Euler angle with an order of {z, y, x} for axis rotations. The rotations about the {z, y, x} axis represent rotation, flexion, and abduction angles, respectively. For more details of the implementation, see the accompanying code.

Evaluation

Comparison of bundle adjustment algorithms To evaluate the different bundle adjustment algorithms (Figures S2.1B and C), we ran the algorithms with different parameters on the calibration videos from the fly setup. There were 4475 frames where the calibration board was detected in 2 or more cameras. To demonstrate the usefulness of our iterative bundle adjustment procedure with lower number of detections, we evaluated all bundle adjustment algorithms after subsampling the frames with board detections to 313 (7%) and 4475 (100%). At each of these frame counts, we initialized the camera parameters and then ran our iterative bundle adjustment procedure, as well as traditional bundle adjustment

with a linear least-squares loss, a Huber loss, and soft L1 loss. As the Huber and soft L1 losses are sensitive to the outlier threshold parameter, we evaluated them at multiple outlier thresholds on our dataset (Figure S2.1C). We picked the loss with the best outlier threshold, as evaluated by the reprojection error at the 75th percentile, to plot in the main calibration figure. The iterative bundle adjustment procedure was run with the default parameters in Anipose: $N_{iter} = 12, \mu_{\text{start}} = 15, \mu_{\text{end}} = 1$.

Evaluation against physical ground truth. To evaluate the calibration and triangulation, we compared the accuracy of manual keypoint annotations, neural network keypoint detections, and OpenCV keypoint detections (Figure 2.3). The ground truth was considered to be known physical length and angles of the ChArUco board. The physical lengths were calculated between all pairs of keypoints by taking the length between the known positions of pairs of corners. Similarly, the physical angles were estimated between all triplets of non-collinear keypoints. The subpixel OpenCV detections were done using the Aruco module [98]. The manual annotation and neural network methods are detailed above. Given the keypoint detections from each method, we used linear least-squares triangulation to obtain 3D points and computed angles using the dot product method detailed above. If a keypoint was detected in fewer than 2 cameras at any time, we could not triangulate it and therefore did not estimate the error at that frame.

Evaluation of 3D tracking error for different filters. To evaluate the contribution of 2D and 3D filters, we applied each filter and measured the reduction in error. For the 2D filters, we applied each of the filters (2D median filter, Viterbi filter, and autoencoder filter) and computed the 3D position using linear least-squares triangulation. We could not train the autoencoder filter on the human dataset, as the filter relies on occluded keypoints not being present in the annotated dataset and, due to the nature of the human dataset, all keypoints are annotated from every view at every frame. When applying the spatiotemporal regularization, we assumed a low variance in length of the coxa, femur, and tibia in flies and

of the arm, the forearm, pelvis, femur, and tibia in the human. We assumed a slightly higher variance for the length of the tarsus in each fly and of the neck and shoulders in each human, because these body segments are more flexible. The parameters for each filter are listed in Table S2.1. We measured the error in joint positions and angles relative to those computed from manual annotations, using the ℓ_2 norm. To evaluate the effect of the filter addition, as there was a lot of variance in error across points, we computed the difference in error for each point tracked. We treated points with reprojection error above 20 pixels as missing. The procedure for evaluating the 3D filters was similar, except that we compared the error in joint position and angle relative to the error from 3D points obtained with a Viterbi filter and autoencoder filter with linear least-squares triangulation.

Evaluation of derivative error for different filters. To evaluate the contribution of different 2D and 3D filters to the error in derivative estimation, we applied each filter to the 3D trajectory of each joint and estimated the derivative by using the finite difference method. For each joint, each frame, and each filter, we obtain a 3D vector representing a derivative. We compare the error between this derivative vector and the true derivative vector from manual annotations by using the ℓ_2 norm, as in the previous section.

Evaluation of 3D tracking error for different number of cameras To evaluate how the number of cameras contributes to the estimate of error, we ran Anipose on all combinations of 2, 3, and 4 cameras for the human dataset. We measured the error in joint position and angles relative to manual annotations as described above. We plotted the mean error across all joint positions or angles and across all possible combinations of cameras (Table S2.2) at each number of cameras.

Evaluation of temporal regularization on synthetic dataset To evaluate how minimizing higher order derivatives affects tracking of high frequency movement dynamics, we evaluated the temporal regularization on a synthetic dataset (Figure S2.4). We synthesized

30 ground-truth keypoint trajectories, each of length 500, by applying a low-pass filter with a cutoff of 0.12 cycles/sample on white noise. We then corrupted these trajectories by adding white noise and removing 10% of the points, simulating observed triangulated points (for example, as in the "No filters" trace in Figure 2.5A). We reconstructed the signal using temporal regularization and minimizing the 1st, 2nd, or 3rd derivative across different levels of smoothing factor β_{time}. We estimated the power spectrum of the ground truth, corrupted, and reconstructed signals by taking the average power spectral density at each frequency across all 30 simulated trajectories. We estimated the power spectral density using the Welch's method as implemented in SciPy [282]. We computed the root mean squared error (RMSE) between the ground truth and reconstructed signals for each derivative minimized at different levels of smoothing. We evaluated the RMSE of median filters with window size of 3 to 25 samples on the same trajectories, and found the median filter with a window size of 9 samples to have the lowest RMSE, which we plot as a reference.

Analysis of kinematics

Analysis of fly walking kinematics For the analysis in Figure 2.7, we used data from 39 wild-type Berlin flies on a spherical treadmill (details of experimental setup above). We tracked the flies using Anipose with spatiotemporal regularization and Viterbi and autoencoder filters. We confirmed by visual inspection and by checking reprojection errors that all flies were well tracked.

To restrict the data to only walking, we manually labeled fly behavior for a random subset of videos using the VGG Image Annotation tool [81]. The categories of behaviors labeled were abdomen grooming, antennae grooming, ball push, ball tapping, eye grooming, head grooming, standing, t1 grooming, t3 grooming, walking. To detect walking behavior across the entire dataset, we fit a logistic classifier to predict the type of behavior. The input data to the classifier for each time point was a chunk of 24 samples around that time of 3D joint positions and angles and the Fourier transform of the 24 samples of each variable. The confusion matrix for the classifier on a test set is shown in Figure S2.6C. The false negative

rate was 0%, whereas the false positive rate was about 3%. To detect bouts of walking, we used the classifier to predict a walking probability for each sample in a video, applied a mean filter with a window of 16 samples to the probability, then kept bouts where the probability was above 0.5 for at least 40 consecutive samples. To further reduce spurious walking bout detections, we removed any bout where the femur-tibia flexion of the left front and hind legs varied less than 10 degrees over the full bout. We confirmed with visual inspection that all bouts removed in this way did not include walking.

To perform the UMAP embeddings, we followed a procedure inspired by DeAngelis et al [69], which mapped the manifold structure of *Drosophila* walking from 2D tracking data. We took chunks of 32 samples, advancing by 8 samples, of the coxa rotation, femur rotation, and femur-tibia flexion angles and their derivatives. Thus, we obtained a set of vectors of size 1152 (32 samples * 6 legs * 3 angles * 2 raw & derivatives), which we standardized by subtracting the mean and dividing by the standard deviation along each dimension. We embedded this set of vectors in 3 dimensions using the UMAP algorithm [188], with effective minimum distance of 0.4 and 30 neighbors as parameters. To compute the phase of the step cycle, we applied a band-pass filter (1st order Butterworth over 3–60Hz) to front left leg femur-tibia flexion and estimated the phase from the analytic signal obtained using the Hilbert transform.

Analysis of mouse reaching kinematics In Figures 2.7and S2.7, we analyzed videos from 4 mice recorded over 2 different days (details of experimental setup above). We tracked 3 keypoints on the hand for each mouse using Anipose with no filters. To obtain accurate 3D tracking for all trajectories, we removed all points with reprojection error above 10 pixels, then filled in missing data (about 11% of the data) using linear interpolation. We used the proximal end of digit 3 as a marker for the overall hand position. Mice 1 and 3 reached with their left hand, whereas mice 2 and 4 reached with their right hand. Accordingly, we quantified the movement of the hand each mouse reached with. We labeled the start and end of each reach, along with the reach type using the Anipose visualizer (Figure 2.6). To obtain

the 3D position of the pellet holder, we labeled the pellet holder for each mouse and day from both views using the VGG Image Annotation tool [81], then triangulated the labeled points for each pair of views using aniposelib. We measured the distance of the hand (proximal end of digit 3) to the pellet holder by using the ℓ_2 norm.

Analysis of human walking kinematics In Figure 2.7 and S2.7, we analyzed videos from all 7 publicly available subjects in the Human 3.6M dataset (dataset described above). We tracked 17 keypoints for each human using Anipose with spatiotemporal regularization and Viterbi filters.

To focus on walking, we restricted our analysis on the "Walking-1", "Walking-2", "WalkingTogether-1", and "WalkingTogether-2" actions in the dataset. We estimated the knee flexion, hip flexion, and hip rotation angles as described in the "Estimating joint angles" section above. For the UMAP embedding, we followed a procedure similar to our analysis of fly kinematics. Specifically, we took chunks of 24 samples, advancing by 8 samples, of the knee flexion, hip rotation, and hip flexion angles and their derivatives. Thus, we obtained a set of vectors of size 288 (24 samples * 2 legs * 3 angles * 2 raw & derivatives), which we standardized by subtracting the mean and dividing by the standard deviation along each dimension. We embedded this set of vectors in 3 dimensions using the UMAP algorithm [188], with effective minimum distance of 0.4 and 30 neighbors as parameters.

2.6 Acknowledgments

We thank Su-Yee Lee and Chris Dallmann for help with annotating keypoints on flies, John So for help with annotating keypoints on the ChArUco board, and Sam Mak for help with annotating mice keypoints and fly behavior. We thank Stephen Huston for loan of his calibration board and Julian Pitney for contributing code to check calibration board detections to Anipose. Finally, we thank the Tuthill and Brunton labs and Mackenzie and Alexander Mathis for support, suggestions, and feedback on the manuscript.

LK was supported by a National Science Foundation Graduate Research Fellowship. KLR

was supported by fellowships from the University of Washington's Institute for Neuroengineering (UWIN) and Center for Neurotechnology (CNT). ESD was supported by a fellowship from University of Washington's Institute for Neuroengineering. ES was supported by the National Institutes of Health (F31NS115477). EA was supported by the National Institutes of Health (R00 NS088193, DP2NS105555, R01NS111479, and U19NS112959), the Searle Scholars Program, The Pew Charitable Trusts, and the McKnight Foundation. BWB was supported by a Sloan Research Fellowship and the Washington Research Foundation. JCT was supported by the Searle Scholar Program, the Pew Biomedical Scholar Program, the McKnight Foundation, and National Institute of Health grants R01NS102333 and U19NS104655. JCT is a New York Stem Cell Foundation – Robertson Investigator.

Contributions

LK, BWB, and JCT conceived the project. LK designed, implemented, and evaluated the Anipose toolkit. KR wrote the Anipose documentation, contributed Tensorpack data augmentation to DeepLabCut, and wrote key parts of the Anipose visualizer. ESD and SWB collected the ChArUco and fly datasets. ES and EA collected the mouse dataset. LK, BWB, and JCT wrote the paper, with input from KR, ESD, ES, and EA.

Declaration of Interests

LK is a founder and the Chief Science Officer of Evolution Devices, a company that uses motion tracking to help people with walking disorders.

Inclusion and Diversity

One or more of the authors of this paper self-identifies as an underrepresented ethnic minority in science. One or more of the authors of this paper self-identifies as a member of the LGBTQ+ community.

	ChArUco	Mouse	Fly	Human
Training frames	1200	2200	6632	636724
Test frames	1200	400	1200	159181
Num cameras	6	2	6	4
Pixel scale (mm)	0.0075	0.0897	0.0075	4.79
2D filter				
score threshold			0.05	0.05
n_back			3	3
medfilt			13	13
offset_threshold			15	30
spline			true	true
3D filter				
score_threshold	0.3	0.3	0.3	0.3
reproj_error_threshold			5	5
scale_length			3	1.5
scale_length_weak			0.5	0.5
scale_smooth			2	4
n_deriv_smooth			3	2

Table S2.1: Anipose configuration parameters used in this paper. Related to Figures 2.4 and 2.5.

num cams	filter type	mean joint angle error (deg)	mean joint position error (mm)
2	No filters	11.5	84.3
	Viterbi filter	10.6	76.9
	Spatiotemporal reg. + Viterbi	12.0	98.8
3	No filters	7.5	42.6
	Viterbi filter	7.3	41.7
	Spatiotemporal reg. + Viterbi	7.1	40.2
4	No filters	6.9	37.0
	Viterbi filter	6.7	36.8
	Spatiotemporal reg. + Viterbi	6.4	33.1

Table S2.2: Estimates of error with different number of cameras for the human dataset. Related to Figure 2.5.

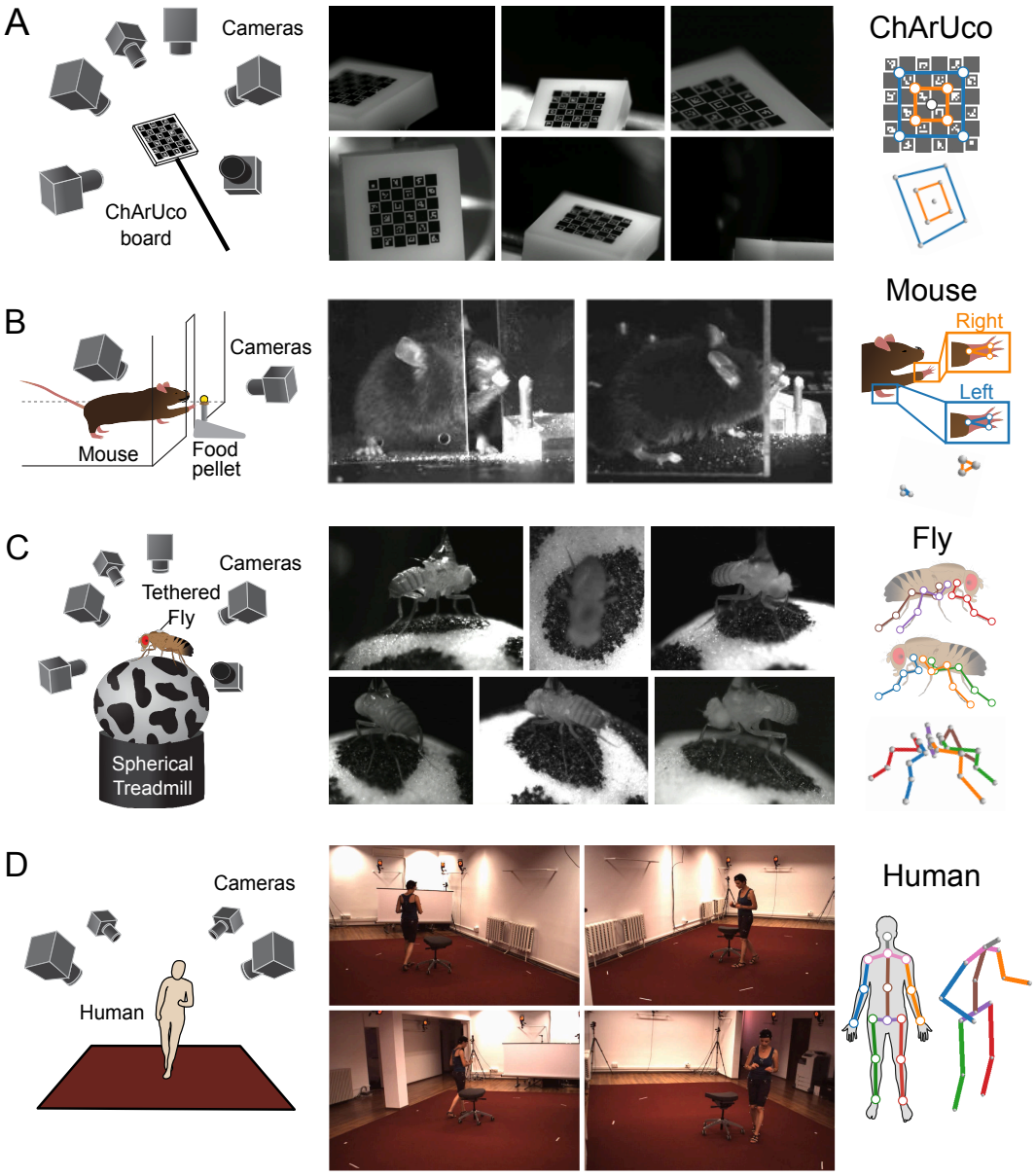

Figure 2.1: Four experimental datasets were used for evaluating 3D calibration and tracking with Anipose. (Full caption on following page)

Figure 2.1: Four experimental datasets were used for evaluating 3D calibration and tracking with Anipose. (A) To evaluate tracking errors, a 2×2mm precision manufactured ChArUco board was simultaneously filmed from 6 cameras focused on the same point in space. We manually annotated and tracked 9 keypoints on the ChArUco board, a subset of the points that can be detected automatically with OpenCV. (B) Adult mice were trained to reach for food pellets through an opening in a clear acrylic box. After training, reach attempts were captured from 2 cameras. To quantify reach kinematics, we labeled and tracked 3 keypoints on each hand. (C) Fruit flies were tethered and positioned on a spherical treadmill, where they were able to walk, groom, etc. Fly behavior was filmed from 6 cameras evenly distributed around the treadmill. We labeled and tracked 5 keypoints on each of the 6 legs, one keypoint for each of the major leg joints. (D) As part of the Human 3.6M dataset, professional actors performing a range of actions were filmed from 4 cameras. We tracked 17 joints on each human, covering the major joints of the human body.

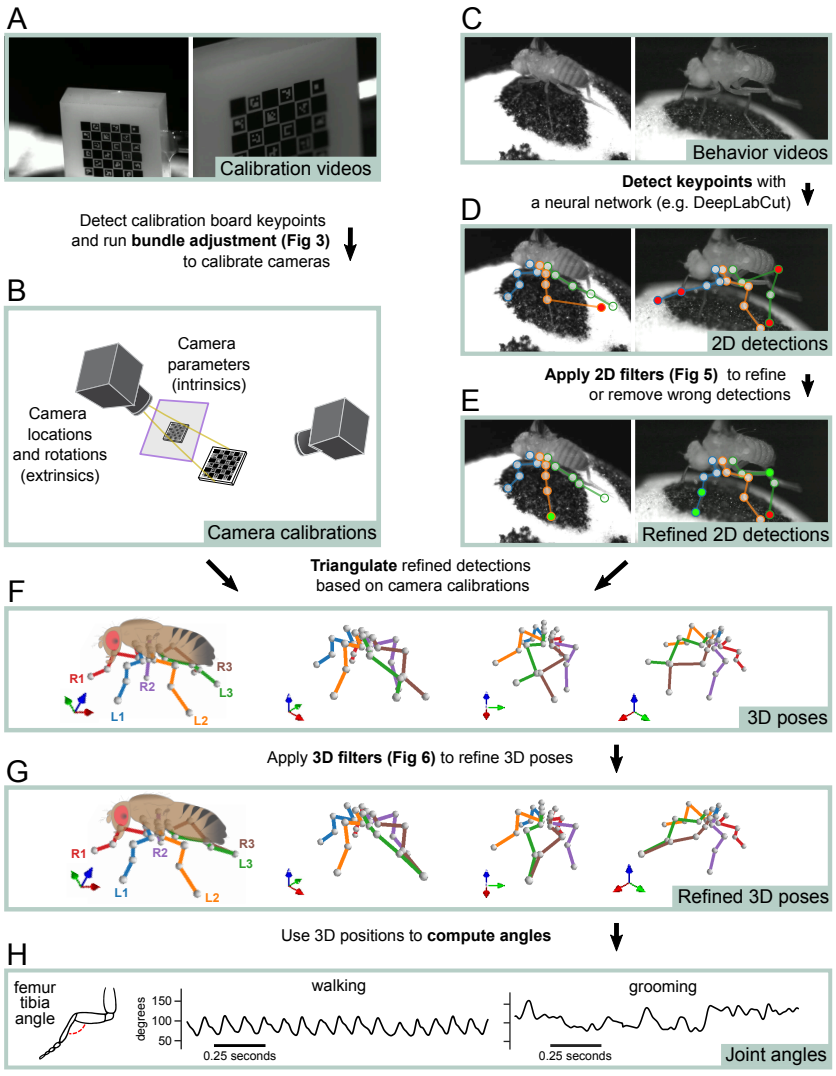

Figure 2.2: Overview of the Anipose 3D tracking pipeline. (A) The user collects simultaneous video of a calibration board from multiple cameras. (B) Calibration board keypoints are detected from calibration videos and processed to calculate intrinsic and extrinsic parameters for each camera using iterative bundle adjustment (See Figure S2.1). (C) With the same hardware setup as in A, the user collects behavior videos. (D) Behavior videos are processed by a neural network (e.g., DeepLabCut) to detect 2D keypoints. (E) 2D keypoints are refined with 2D filters to obtain refined 2D detections (Figure 2.4). (F) The filtered 2D keypoints are triangulated to estimate 3D poses. (G) The estimated 3D poses are passed through an additional spatiotemporal filtering step to obtain refined 3D poses (Figure 2.5). (H) Joint angles are extracted from the refined 3D poses for further analysis.

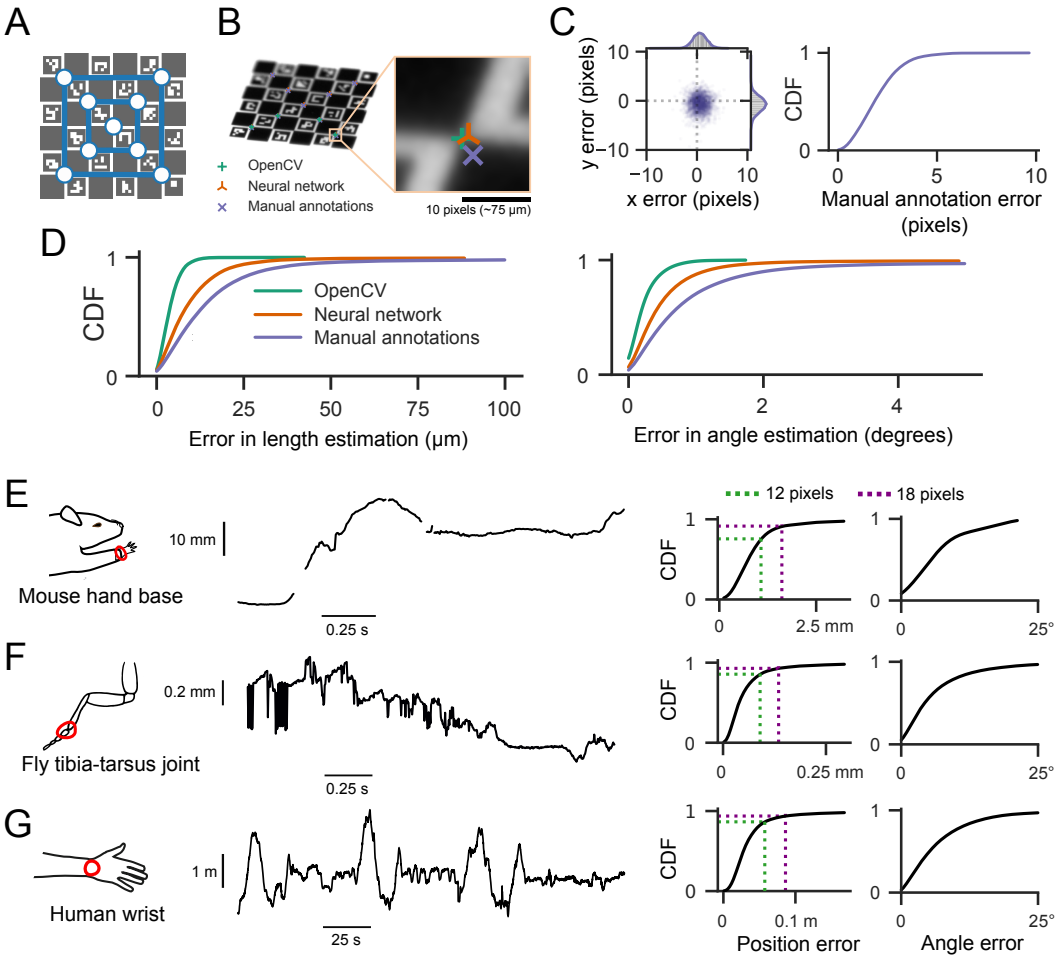

Figure 2.3: Anipose can consistently estimate positions and angles of keypoints across four different datasets. (Full caption on following page)

Figure 2.3: Anipose can consistently estimate positions and angles of keypoints across four different datasets. (A) We identified 9 corners as keypoints on the ChArUco board in 200 frames from each of 6 cameras. (B) For comparison, we used manual annotation of the same ChArUco board dataset to train a neural network. We then compared tracking errors of the manual annotations, the neural network, and OpenCV. (C) Error in manually annotated keypoints relative to the sub-pixel precision of OpenCV detections. Manually annotated keypoints had a mean error of (0.52, -0.75) pixels and standard deviation of (2.57, 2.39) pixels. (D) Lengths between all possible pairs of keypoints were computed and compared to the physical lengths. Similarly, all possible angles between triplets of keypoints were computed and compared to known physical angles. OpenCV keypoints provided the most reliable estimates, followed by neural network predictions, then manual annotations. Note that OpenCV generally detected only a small fraction of the keypoints detected by the neural network or through manual annotation (19.3% of keypoints detected by OpenCV, compared to 78.1% by the neural network and 75% by manual annotations). (E) At this stage, prior to filtering, outlier and missing keypoint detections are apparent. Shown at left is an example trace of the tracked 3D position of the base of the mouse hand, projected onto the direction of the reach. On the right, we quantified the distribution of errors when estimating all joint positions and angles, relative to manual annotations. For the mouse dataset, 1 pixel corresponds to approximately 0.09 mm. (F) Same layout as A, but for 3D position of the fly hind-leg tibia-tarsus joint, projected onto the longitudinal axis of the fruit fly. For the fly dataset, 1 pixel $\approx$.0075 mm. (G) Same layout as A, but for tracked 3D position of a human wrist, projected onto an arbitrary axis. Note that the human (and their wrist) is moving throughout the room. For the human dataset, 1 pixel $\approx$4.8 mm.

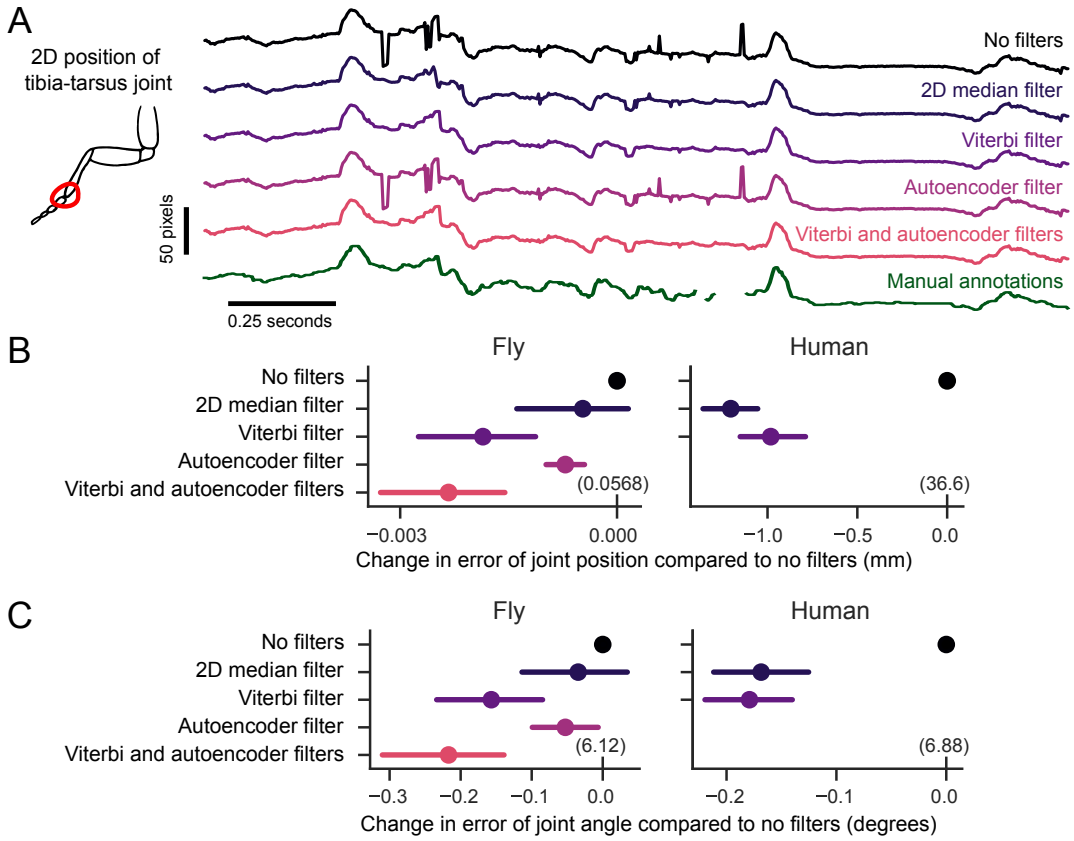

Figure 2.4: 2D filters improve accuracy of 2D pose estimation by taking advantage of the temporal structure of animal behavior. (A) An example trace of the x-coordinate of the 2D position of a fly's tibia-tarsus joint before and after each step in filtering. Filtering reduces spurious jumps while preserving correct keypoint detections. See Figure S2.2 for a demonstration of the autoencoder filter. (B) Comparison of error in joint position before and after filtering. The mean difference in error for the same tracked points is plotted, along with the 95% confidence interval. Viterbi and autoencoder filters significantly improved the estimation of joint position in flies ($p < 0.001$, paired t-test). The Viterbi filter significantly improved estimation of joint position in humans ($p < 0.001$, paired t-test). For the fly dataset, 1 pixel ≈.0075 mm. For the human dataset, 1 pixel ≈4.8 mm. The absolute error values are indicated in parentheses above the 0 tick mark for each dataset. (C) Comparison of angle estimates before and after filtering. The mean difference is plotted as in B. Viterbi and autoencoder filters significantly improved the estimation of angles in flies and humans ($p < 0.001$, paired t-test). The results in (B) and (C) are evaluated on a validation dataset withheld from the training (1200 frames for the fly, 8608 frames for the humans). See Table S2.1 for filter parameters.

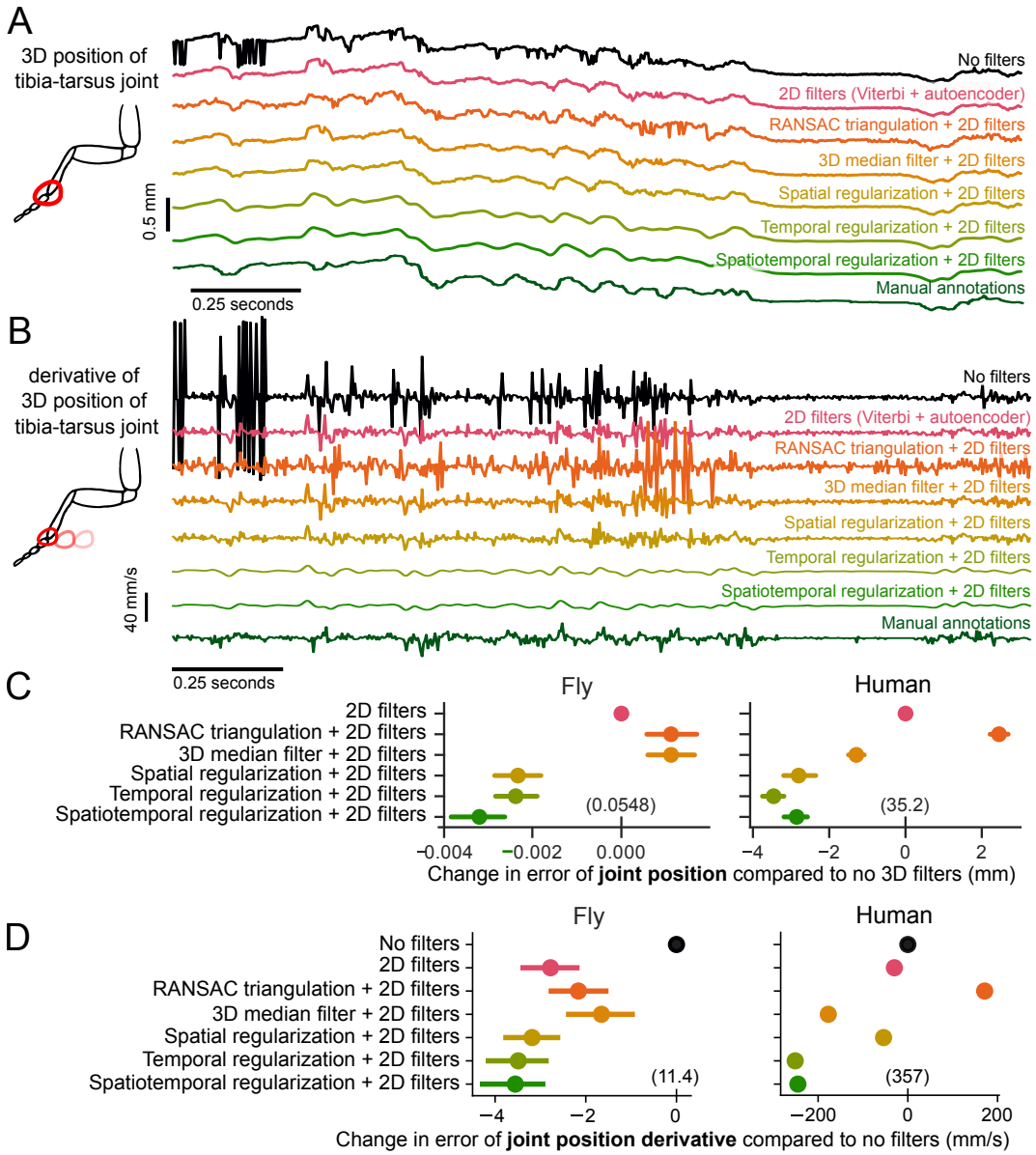

Figure 2.5: Spatiotemporal filters further improve 3D pose estimation. See Figure S2.3 for example angle and segment length traces with different filters. See Figure S2.4 for detailed evaluation of temporal regularization on a synthetic dataset. (Full caption on following page)

Figure 2.5: (A) An example trace of the tracked 3D position of the fly tibia-tarsus joint, before and after filtering. To plot a single illustrative position value, the 3D x-y-z coordinate is projected onto the longitudinal axis of the fly. Also included are comparisons with standard 3D filtering algorithms RANSAC and a 3D median filter, along with manual annotations. Filtering leads to reduction of sudden jumps and keypoint jitters, even compared to 2D filters alone. (B) An example trace of the derivative of the 3D position of the fly tibia-tarsus joint, before and after filtering. To plot a single illustrative derivative value, the 3D x-y-z joint coordinates is projected onto the longitudinal axis of the fly. Spatiotemporal regularization produces smooth derivative estimates, which are closer to the manual annotations compared to other filtering approaches. (C) Comparison of error in joint position before and after filtering. The mean difference in error for the same tracked points is plotted, along with the 95% confidence interval. The absolute error values are indicated in parentheses above the 0 tick mark for each dataset. The 2D filters are the Viterbi filter followed by the autoencoder for the fly dataset and Viterbi filter alone for the human dataset. Spatiotemporal regularization improves the estimation of joint position significantly above 2D filters in both datasets ($p < 0.001$, paired t-test). The 3D median filter improves pose estimation on the human dataset ($p < 0.001$, paired t-test) but not on the fly dataset. RANSAC triangulation does not improve pose estimation for either dataset. For the fly dataset, 1 pixel corresponds to 0.0075 mm. For the human dataset, 1 pixel corresponds to 4.8 mm. (D) Comparison of error in joint position derivative before and after filtering. The mean difference in error for the same tracked points is plotted, along with the 95% confidence interval. The absolute error values are indicated in parentheses above the 0 tick mark for each dataset. The 2D filters are the Viterbi filter followed by the autoencoder for the fly dataset and Viterbi filter alone for the human dataset. For the human dataset, due to the large number of labeled points, the confidence intervals are smaller than the size of the points. Adding filters significantly improves the estimate of the derivative.

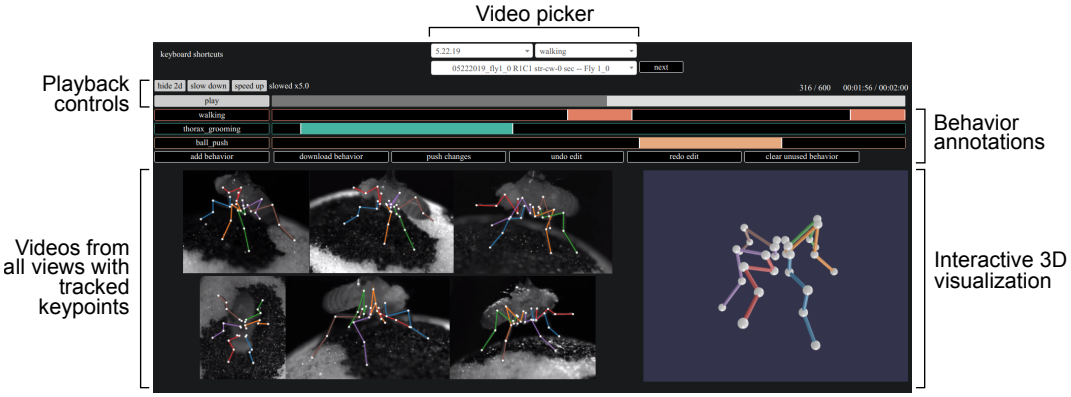

Figure 2.6: A web tool for visualizing 3D kinematics tracked with Anipose, taking advantage of the Anipose file structure shown in Figure S2.5. The videos from all views are displayed synchronously, with overlaid projections of 3D keypoints from Anipose. To the right of the videos, a dynamic 3D visualization allows the user to interact with the 3D keypoints by rotating or zooming in. Above the videos, the user can alter the playback speed or jump to different time points in the video. The user can also annotate the behavior of the animal for further analysis. Menus at the top allow the user to select specific recording dates, experimental trials, or filter trials by a specific behavior.

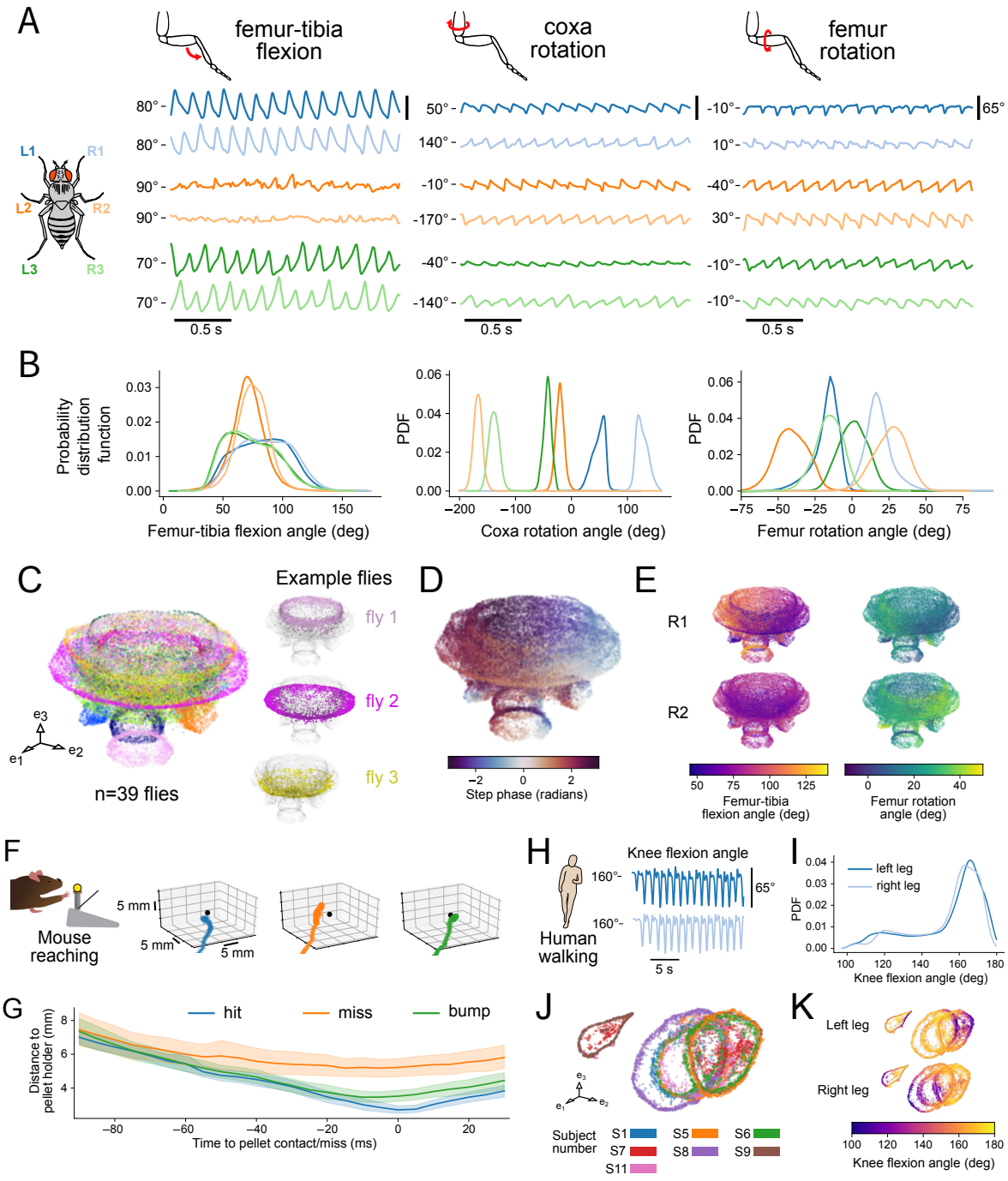

Figure 2.7: 3D tracking of fly walking reveals difference in rotation and flexion angles across legs. (Full caption on following pages)

Figure 2.7: 3D tracking enables quantification of fly, mouse, and human joint position and angles to reveal structure in behavior. (A-E) 3D tracking of fly walking reveals difference in rotation and flexion angles across legs. See Figure S2.6 for analyses across all angles. (A) Representative traces of coxa rotation, femur rotation, and femur-tibia flexion angles from tethered-walking flies. The median angle value is indicated for each angle as a reference point. (B) Probability distribution functions of coxa rotation, femur rotation, and femur-tibia flexion angles from 39 flies (1480 total seconds of walking). Only walking bouts are included. The distribution of femur-tibia flexion angles is broader for the front and rear legs, whereas the distribution of femur rotation angles is broader for the middle legs. (C) UMAP embedding of coxa rotation, femur rotation, femur-tibia flexion angles across all legs, and their derivatives. Axis units are arbitrary. Although each fly has a characteristic gait, there is a continuum across most flies, with some flies offset from the rest. (D) UMAP embedding as in C, colored by the phase of the step cycle, revealing the match between the circular structure of the embedding and the step phase. (E) UMAP embedding as in C, colored by front-right leg femur-tibia flexion and femur rotation, and middle right leg femur-tibia flexion and femur rotation. Across multiple flies, the dynamics of the middle legs are dominated by femur rotation, whereas the dynamics of the front legs are dominated by femur-tibia flexion. (caption continues on following page)

Figure 2.7: (Continued from previous page) (F) Example 3D trajectories of a mouse reaching for a food pellet. The pellet is indicated as a black dot. (G) Mean distance to pellet holder as a function of time across all 4 mice (88 hits, 69 bumps, 28 misses). Shaded areas are 95% confidence intervals. When reaches are aligned to the grasp attempt (0 ms), the hand is farther from the pellet holder on miss trials compared to hit or bump trials. Averaging across all mice reveals a clear difference between reach types. (H) Representative trace of knee flexion from a walking human, tracked with Anipose. Data is from the Human 3.6M dataset. The median angle value is indicated at left as a reference point. (I) Probability distribution function of knee flexion angle from 7 humans. Only sessions that include walking are included. (J) UMAP embedding of knee flexion, hip rotation, and hip flexion angles across all legs, and their derivatives. Axis units are arbitrary. Although each human subject has a characteristic gait, most of the walking patterns map onto a common cylinder manifold. (K) UMAP embedding as in E but colored by knee flexion for each leg. Coloring by knee flexion angle reveals the common phase alignment of the circles across subjects. See also Figures S2.6 and S2.7.

Algorithm 1 Iterative bundle adjustment

Input:

Initial camera parameters $\boldsymbol{\theta}$

Keypoint detections $\boldsymbol{u}$ from multiple cameras

Starting and ending thresholds μ_{start} and μ_{end}

for $i \leftarrow 1$ to N_{iter} **do**
  $\boldsymbol{u}_{\text{eval}} \leftarrow \text{sample}(\boldsymbol{u})$
  $\mathbf{errors}_{\text{eval}} \leftarrow \text{reprojection\_errors}(\boldsymbol{u}^{\text{eval}}, \boldsymbol{\theta})$
  $\mathbf{low} \leftarrow \text{percentile}(\mathbf{errors}_{\text{eval}}, 15\%)$
  $\mathbf{high} \leftarrow \text{percentile}(\mathbf{errors}_{\text{eval}}, 75\%)$
  $\mu_i \leftarrow \left(\frac{\mu_{\text{end}}}{\mu_{\text{start}}}\right)^{i/N_{\text{iter}}}$
  $\mu_i \leftarrow \max(\mathbf{low}, \min(\mu_i, \mathbf{high}))$
  $\mu_{\text{picked}} \leftarrow$ points from $\boldsymbol{u}_{\text{eval}}$ for which reprojection error is below μ_i
  $\boldsymbol{\theta} \leftarrow \text{bundle\_adjust}(\boldsymbol{\theta}, \boldsymbol{u}_{\text{picked}})$
end for
return $\boldsymbol{\theta}$

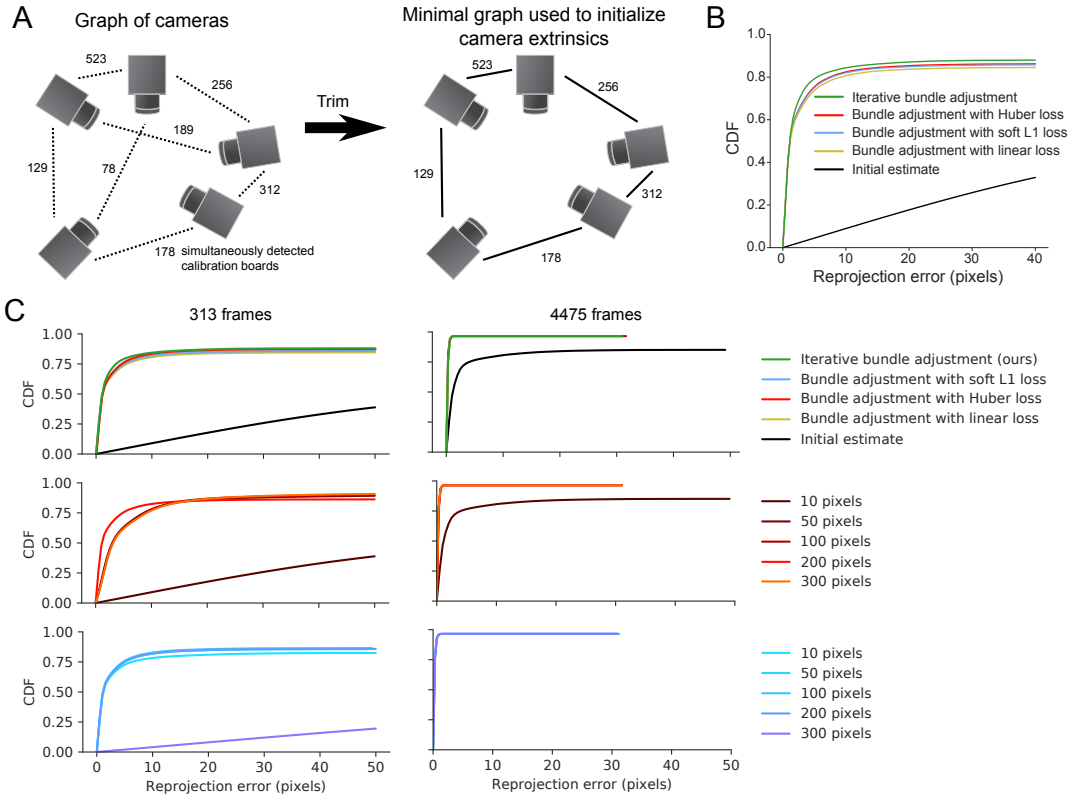

Figure S2.1: Related to Figure 2.2B. (A) Illustration of the camera parameter initialization procedure. We build a graph with each camera as a node and edge weights computed by the number of frames the calibration board is simultaneously detected by pairs of cameras. To initialize the camera calibration, we trim this graph to be a minimal, fully connected tree using a greedy approach. (B) On calibration videos from the fly dataset, bundle adjustment improves the initial calibration estimate, as measured by a reduction in reprojection error. (C) Reprojection error as a function of outlier threshold for bundle adjustment with Huber and soft L1 losses.

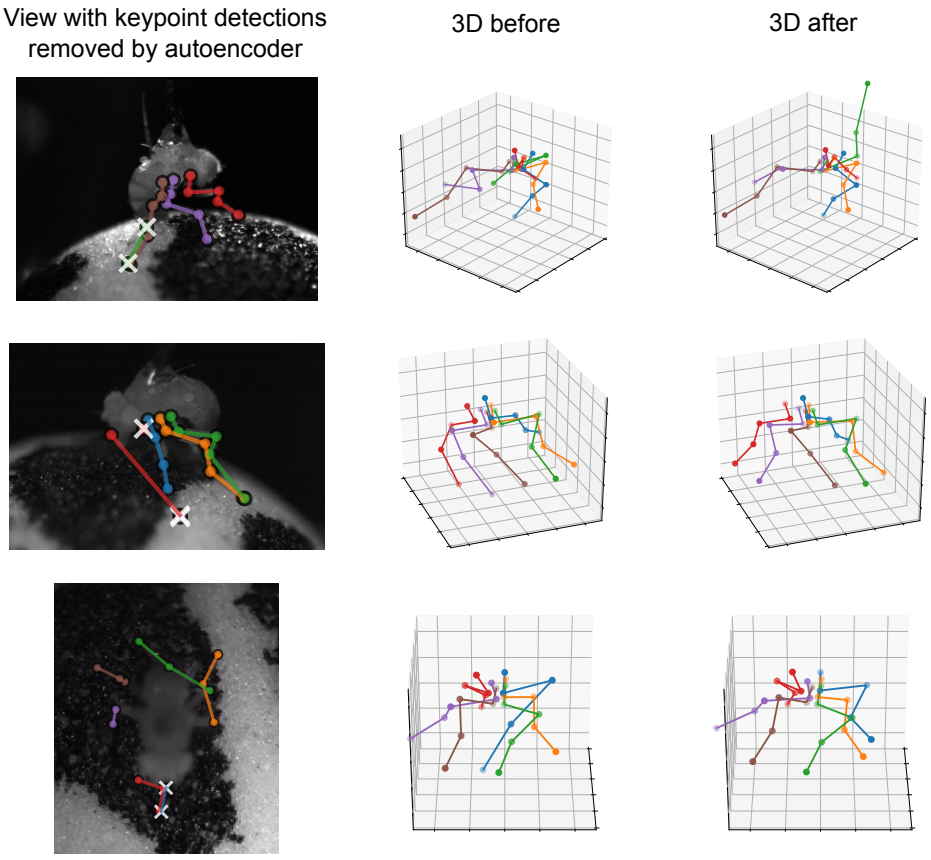

Figure S2.2: An autoencoder corrects 3D tracking by removing bad keypoint detections. Related to Figure 2.4. On the left is one view where the autoencoder lowered the confidences for particularly bad detections, thus removing them from the 3D triangulation. On the right are the 3D positions of the keypoints before and after the removal.

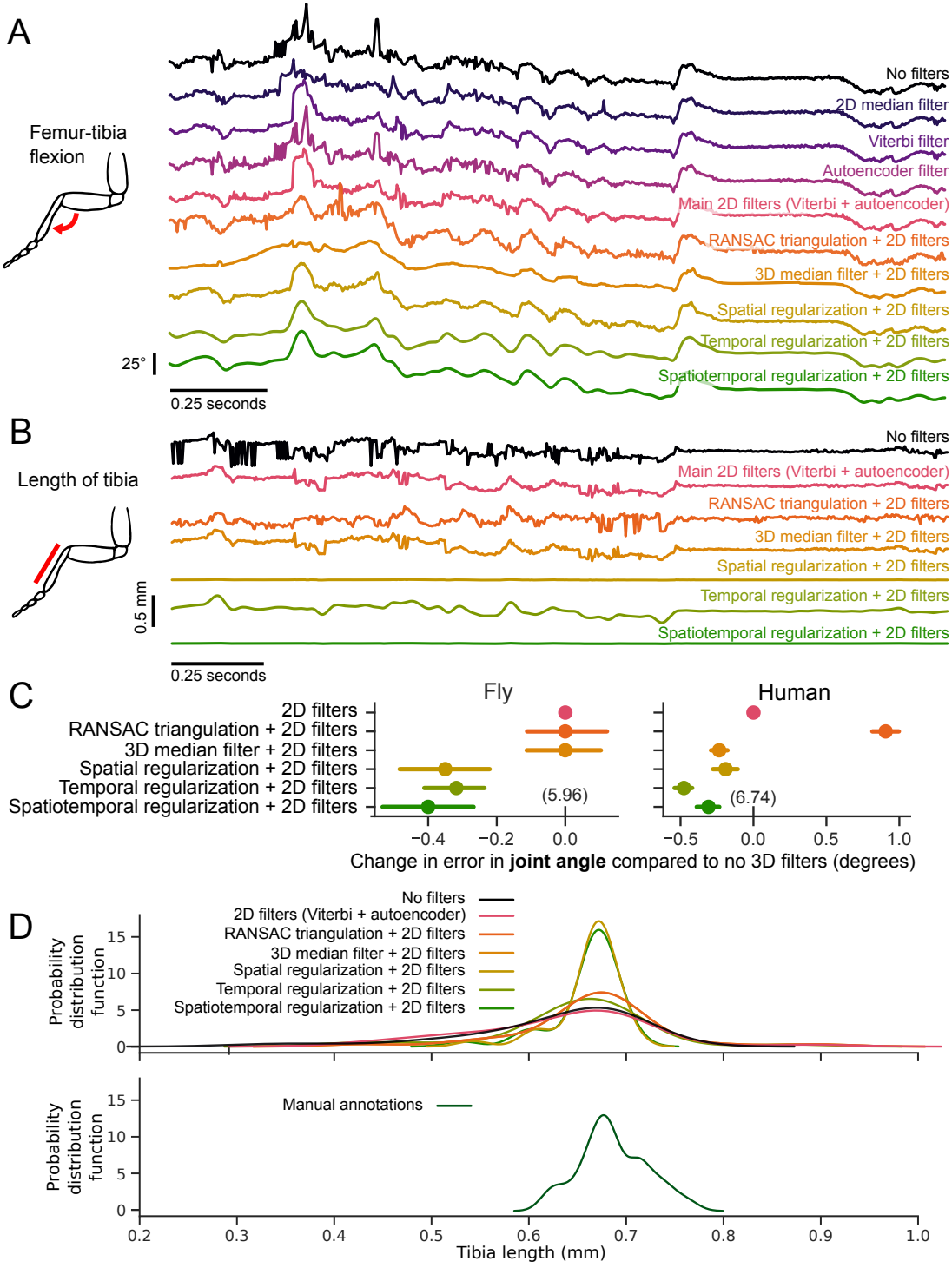

Figure S2.3: Related to Figure 2.5. Full caption on following page.

Figure S2.3: (A) Example traces of the tracked hind-leg femur tibia flexion angle, before and after filtering. (B) Estimation of tibia length over time, before and after filtering. Adding spatial regularization leads to a more stable estimate of the tibia length across frames. (C) Comparison of angle estimates before and after filtering. The mean difference and confidence intervals are plotted as in C. Spatial and temporal regularization improve angle estimation above 2D filters on both datasets ($p < 0.001$, paired t-test). The 3D median filter improves angle estimation on the human dataset ($p < 0.001$, paired t-test) but not on the fly dataset ($p > 0.8$, paired t-test). RANSAC triangulation does not improve angle estimation for either dataset. (D) Comparison of methods for estimating tibia length. Spatial regularization most closely matches the distribution of tibia lengths based on manual annotations. The plots show the distribution of tibia lengths for one fly, extending the example shown in Figure 5B, for different filtering strategies (top) and manual annotations (bottom).

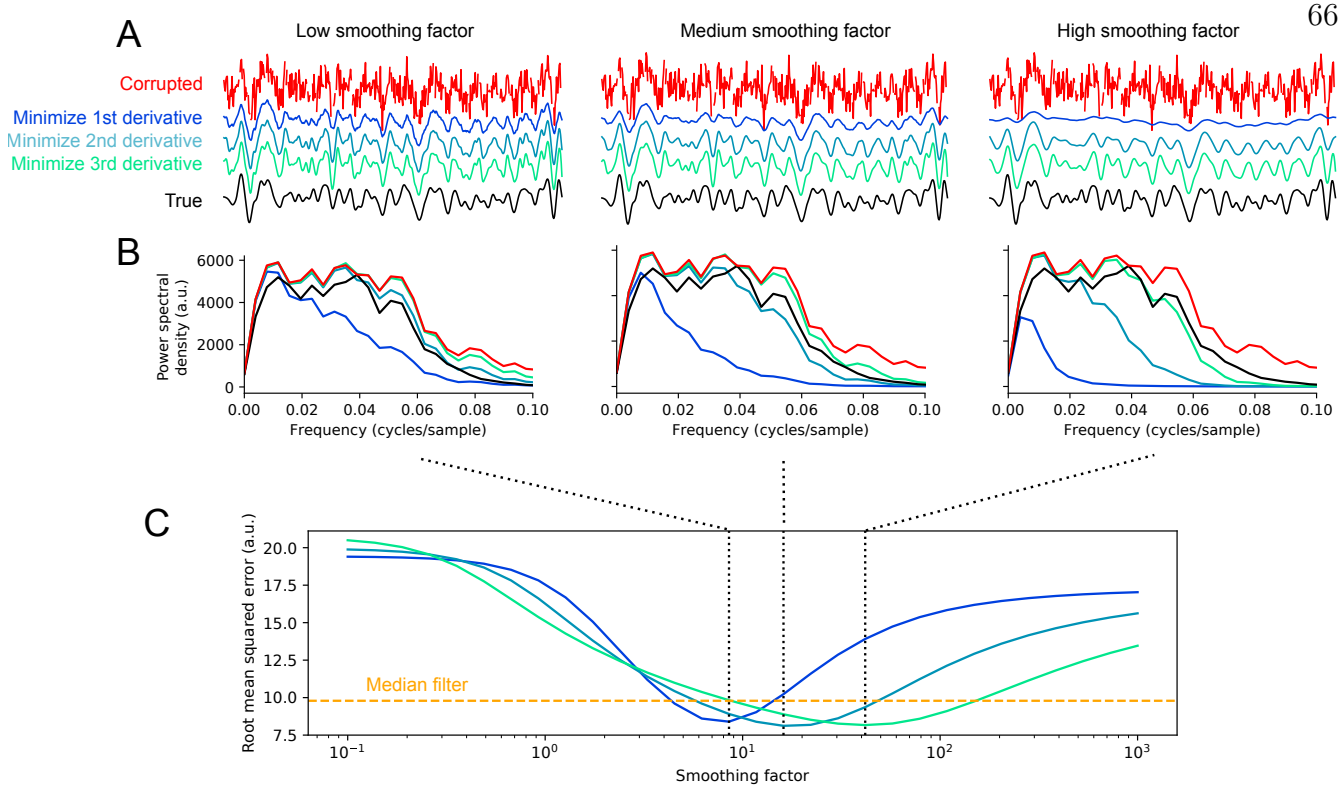

Figure S2.4: Minimizing higher order derivatives preserves high frequency dynamics and leads to lower reconstruction error. Related to Figure 2.5. (A) An example simulated trajectory along with its reconstructions using temporal regularization with different derivatives minimized. Each column shows reconstructions with different smoothing factors. (B) We synthesized 30 different trajectories with the procedure in A and compared the average power spectral density between the true, corrupted, and reconstructed trajectories with different derivatives minimized. At any smoothing factor, minimizing higher derivatives preserves more power at high frequencies. (C) The average root-mean squared error (RMSE) of reconstruction for the 30 simulated trajectories. The minimum error for a median filter (over all possible filter widths) is shown as a dashed line, for reference. Dotted lines indicate the smoothing factors shown in A and B. Note that minimizing higher derivatives is more robust to smoothing factor choice, as a wider range of factors give lower RMSE than a median filter. The best RMSE over all possible smoothing factors is lower when minimizing the 3rd derivative than 2nd or 1st.

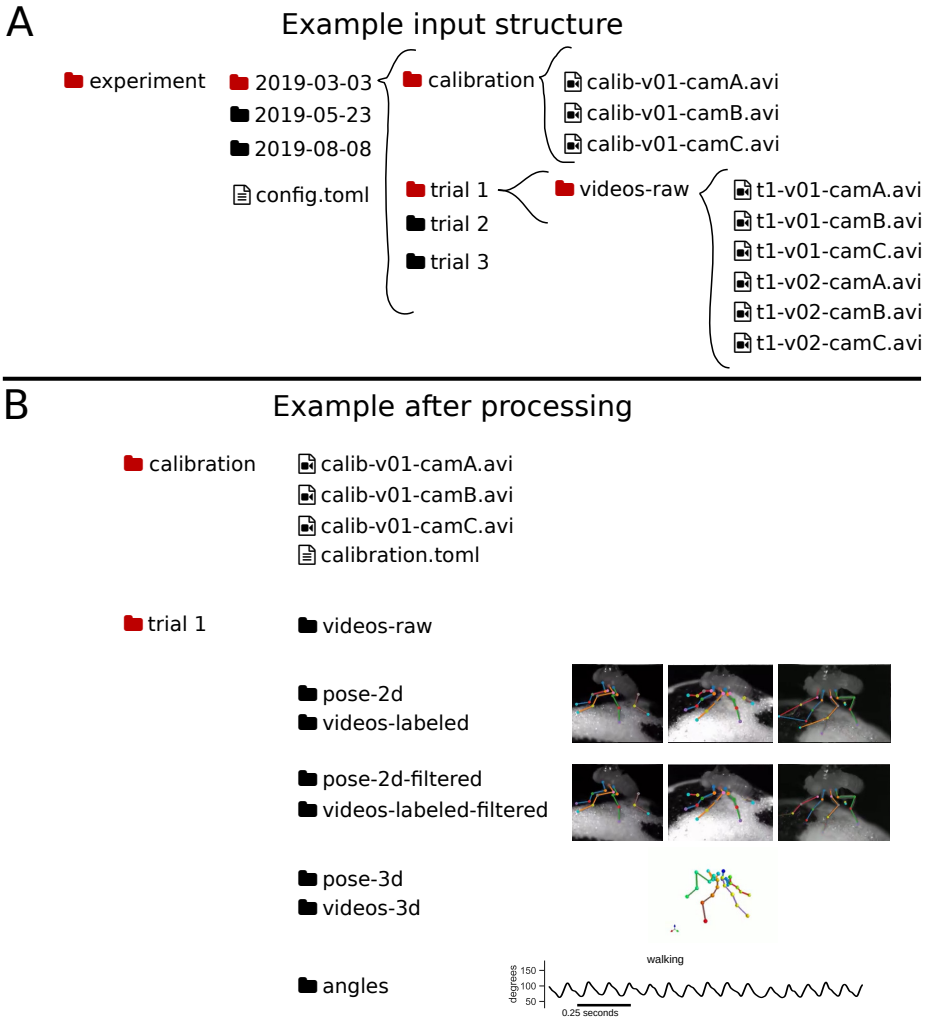

Figure S2.5: An example of the Anipose file structure. This structure enables visualization of arbitrary datasets, as shown in Figure 2.6. (A) The input file structure consists of folders nested to arbitrary depths (e.g. "experiment/2019-03-03/trial 1") with a folder for raw videos at each leaf of the directory tree. The calibration folder may be placed anywhere and will apply recursively to all folders adjacent to it. (B) When the user runs Anipose, it will create a folder for each step of processing. New folders created include "pose-2d" and "videos-labeled" which contain the unfiltered keypoint detections and visualizations of those, "pose-2d-filtered" and "videos-labeled-filtered" which contain the filtered keypoint detections and visualizations, "pose-3d" and "videos-3d" which contain the triangulated 3D keypoint detections and visualizations of these, and finally "angles" which contains angles computed based on the 3D keypoint detections.

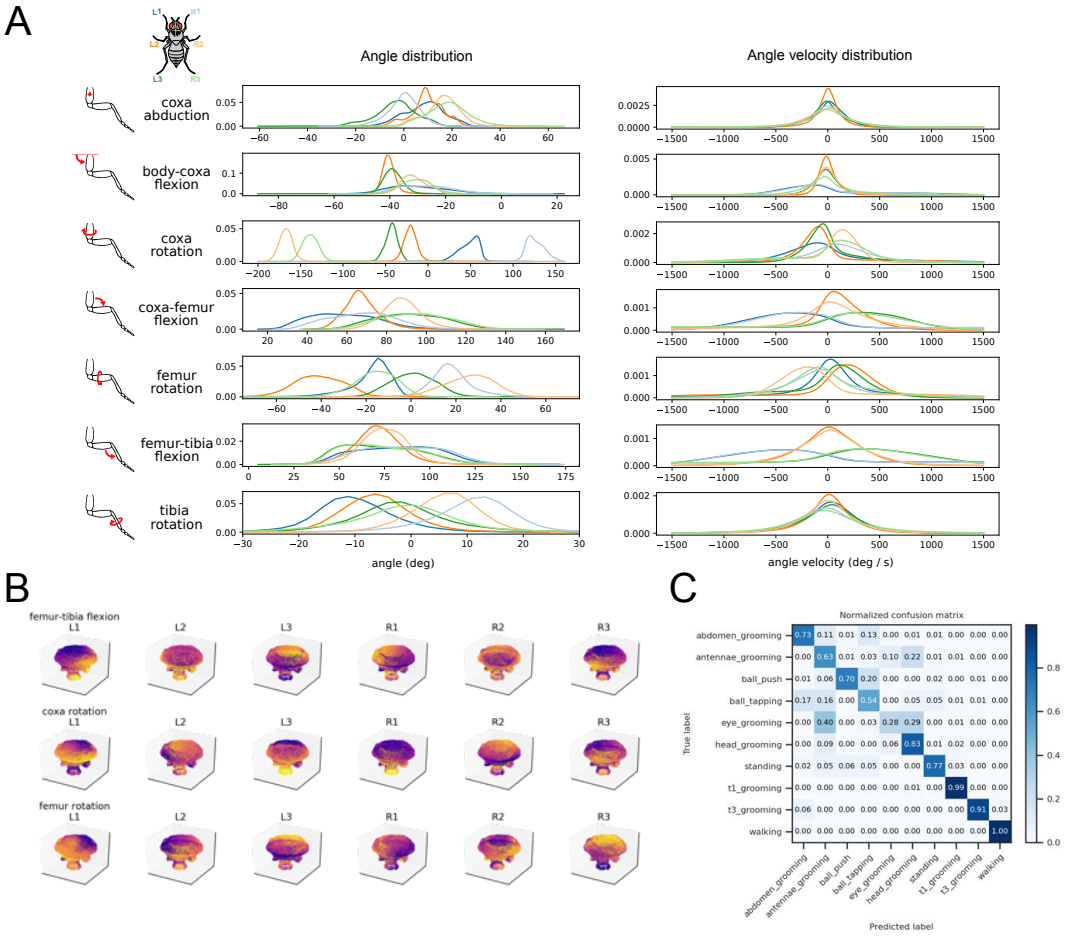

Figure S2.6: Related to Figure 2.7. (A) Probability density functions of all joint angles and derivatives for 39 wild type flies during walking, extending the subset of angles presented in Figure 7B. (B) UMAP embeddding of fly walking, as in Figure 7C, colored by each of the joint angles. The colormap is normalized to the angle within each plot. (C) Confusion matrix for the behavior classifier used to isolate walking bouts for Figure 7.

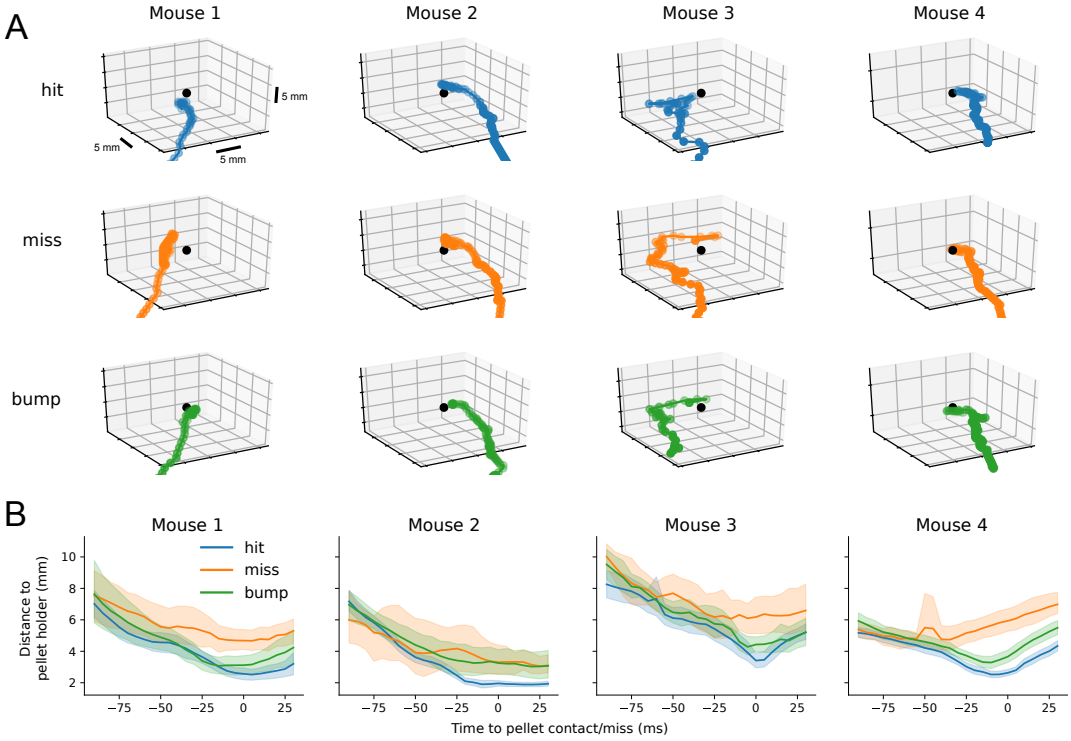

Figure S2.7: Related to Figure 2.7. 3D tracking with Anipose reveals common structure of mouse reaches. (A) 3D trajectories of example reaches of each type. The pellet holder is indicated as a black dot. (B) Mean distance to pellet holder as a function of time, for each mouse. Shaded areas are 95% confidence intervals. When reaches are aligned to grasp attempt (0 ms), the hand is farther from the pellet on miss trials compared to hit or bump trials.

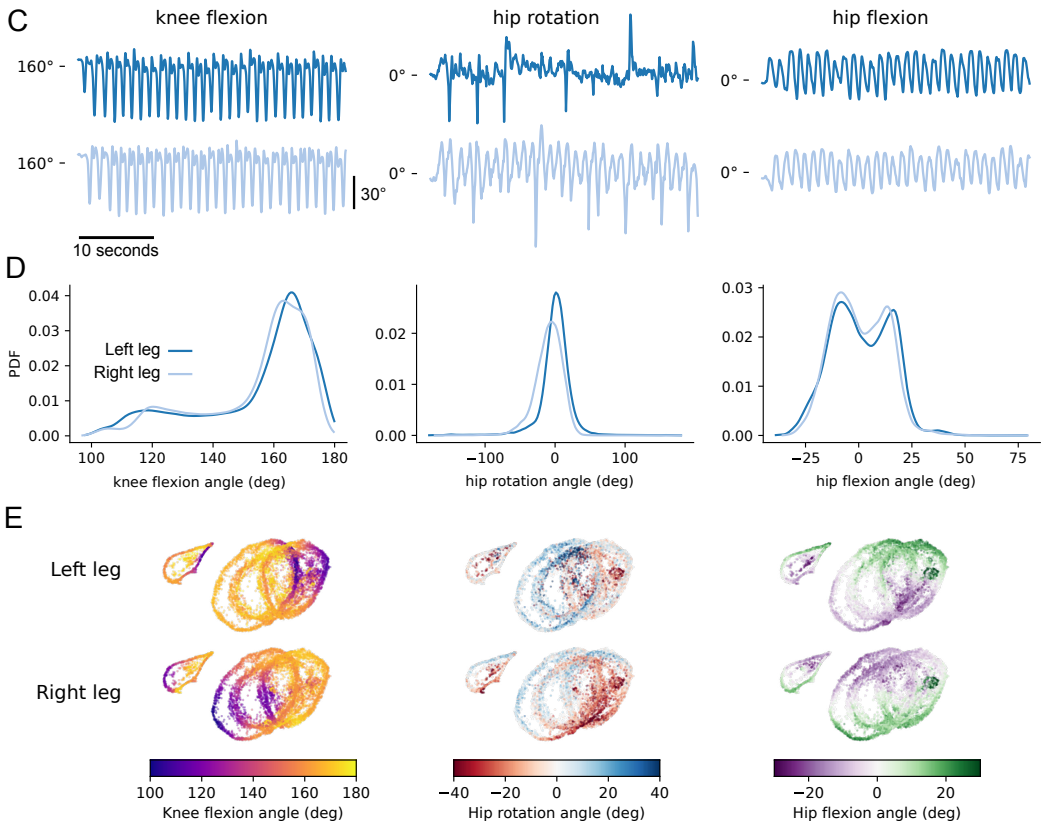

Figure S2.7: (Continued from previous page). 3D tracking of human walking enables quantification of leg angles and comparison across individuals. (C) Representative traces of knee flexion, hip rotation, and hip flexion from a walking human, tracked with Anipose. Data is from the Human 3.6M dataset. The median angle value is indicated at left as a reference point. (D) Probability distribution functions of knee flexion, hip rotation, and hip flexion angles from 7 humans. Only sessions that include walking are included. Note the asymmetry in the distributions of knee flexion and hip flexion, revealing the known non-sinusoidal pattern of knee and hip flexion during walking. (E) UMAP embedding of knee flexion, hip rotation, and hip flexion angles across all legs, and their derivatives. The UMAP embedding is colored by knee flexion and hip rotation for each leg. Coloring by knee flexion angle reveals the common phase alignment of the circles across subjects. From this phase alignment, we see that the trajectory of hip rotation for each subject is markedly different.

Chapter 3

BKIND-3D: SELF-SUPERVISED 3D KEYPOINT DISCOVERY FROM MULTI-VIEW VIDEOS

This chapter was published in *CVPR* in 2023 [262].

Authors: Jennifer J. Sun*, Lili Karashchuk*, Amil Dravid*, Serim Ryou, Sonia Fereidooni, John C. Tuthill, Aggelos Katsaggelos, Bingni W. Brunton, Georgia Gkioxari, Ann Kennedy, Yisong Yue, Pietro Perona.

* Equal contribution

3.1 Abstract

Quantifying motion in 3D is important for studying the behavior of humans and other animals, but manual pose annotations are expensive and time-consuming to obtain. Self-supervised keypoint discovery is a promising strategy for estimating 3D poses without annotations. However, current keypoint discovery approaches commonly process single 2D views and do not operate in the 3D space. We propose a new method to perform self-supervised keypoint discovery in 3D from multi-view videos of behaving agents, without any keypoint or bounding box supervision in 2D or 3D. Our method, BKinD-3D, uses an encoder-decoder architecture with a 3D volumetric heatmap, trained to reconstruct spatiotemporal differences across multiple views, in addition to joint length constraints on a learned 3D skeleton of the subject. In this way, we discover keypoints without requiring manual supervision in videos of humans and rats, demonstrating the potential of 3D keypoint discovery for studying behavior.

3.2 Introduction

All animals behave in 3D, and analyzing 3D posture and movement is crucial for a variety of applications, including the study of biomechanics, motor control, and behavior [179]. However, annotations for supervised training of 3D pose estimators are expensive and time-consuming to obtain, especially for studying diverse animal species and varying experimental contexts. Self-supervised keypoint discovery has demonstrated tremendous potential in discovering 2D keypoints from video [138, 139, 264], without the need for manual annotations. These models have not been well-explored in 3D, which is more challenging compared to 2D due to depth ambiguities, a larger search space, and the need to incorporate geometric constraints. Our goal is to enable 3D keypoint discovery of humans and animals from synchronized multi-view videos, without 2D or 3D supervision.

Self-Supervised 3D Keypoint Discovery. Previous works for self-supervised 3D keypoints typically start from a pre-trained 2D pose estimator [280, 151], and thus do not perform *keypoint discovery* (Figure 3.1). These models are suitable for studying human poses because 2D human pose estimators are widely available and the pose and body structure of humans is well-defined. However, for many scientific applications [213, 179, 264], it is important to track diverse organisms in different experimental contexts. These situations require time-consuming 2D or 3D annotations for training pose estimation models. The goal of our work is to enable 3D keypoint discovery from multi-view videos directly, without any 2D or 3D supervision, in order to accelerate the analysis of 3D poses from diverse animals in novel settings. To the best of our knowledge, self-supervised 3D keypoint discovery have not been well-explored for real-world multi-view videos.

Behavioral Videos. We study 3D keypoint discovery in the setting of behavioral videos with stationary cameras and backgrounds. We chose this for several reasons. First, this setting is common in many real-world behavior analysis datasets [249, 88, 43, 181, 213, 140, 263], where there has been an emerging trend to expand the study of behavior from 2D to 3D [179]. Thus, 3D keypoint discovery would directly benefit many scientific studies in this space

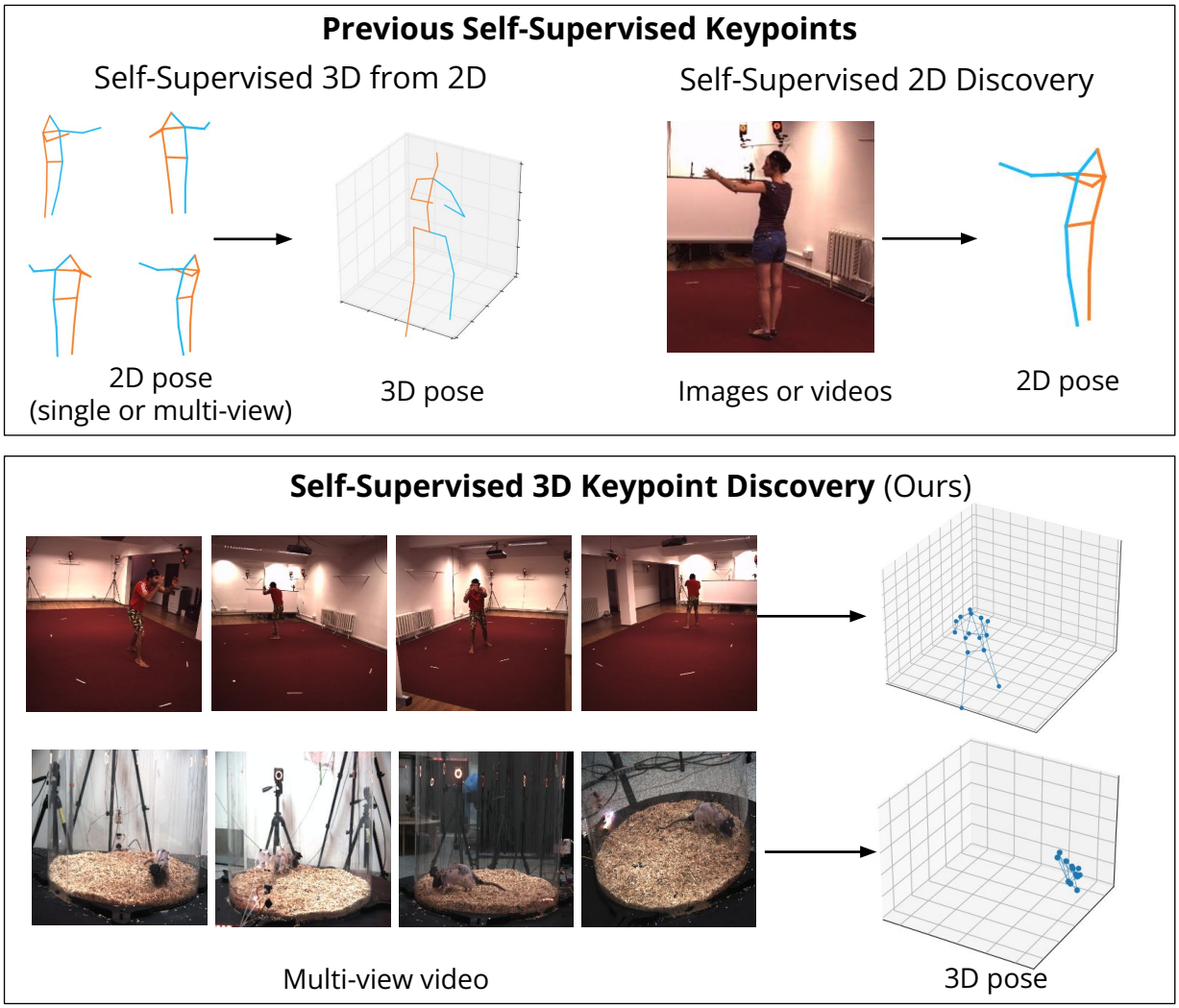

Figure 3.1: **Self-supervised 3D keypoint discovery**. Previous work studying self-supervised keypoints either requires 2D supervision for 3D pose estimation or focuses on 2D keypoint discovery. Currently, self-supervised 3D keypoint discovery is not well-explored. We propose methods for discovering 3D keypoints directly from multi-view videos of different organisms, such as human and rats, without 2D or 3D supervision. The 3D keypoint discovery examples demonstrate the results from our method.

using approaches such as biomechanics, motor control, and behavior [179]. Second, studying behavioral videos in 3D enables us to leverage recent work in 2D keypoint discovery for behavioral videos [264]. Finally, this setting enables us to tackle the 3D keypoint discovery challenge in a modular way. For example, in behavior analysis experiments, many tools are already available for camera calibration [146], and we can assume that camera parameters are known.

Our Approach. The key to our approach, which we call **B**ehavioral **K**eypo**in**t **D**iscovery in **3D** (BKinD-3D), is to encode self-supervised learning signals from videos across multiple views into a single 3D geometric bottleneck. We leverage the spatiotemporal difference reconstruction loss from [264] and use multi-view reconstruction to train an encoder-decoder architecture. Our method does not use any bounding boxes or keypoint annotations as supervision. Critically, we impose links between our discovered keypoints to discover connectivity across points. In other words, keypoints on the same parts of the body are connected, so that we are able to enforce joint length constraints in 3D. To show that our model is applicable across multiple settings, we demonstrate our approach on multi-view videos from different organisms. To summarize:

- We introduce self-supervised 3D keypoint discovery, which discovers 3D pose from real-world multi-view behavioral videos of different organisms, without any 2D or 3D supervision.

- We propose a novel method (BKinD-3D) for end-to-end 3D discovery from video using multi-view spatiotemporal difference reconstruction and 3D joint length constraints.

- We demonstrate quantitatively that our work significantly closes the gap between supervised 3D methods and 3D keypoint discovery across different organisms (humans and rats).

Method	3D sup.	2D sup.	camera params	data type
Isakov et al. [135] DANNCE [79]	✓	✓	intrinsics extrinsics	real
Rhodin et al. [227]	✓	optional	intrinsics	real
Anipose [146] DeepFly3D [113]	×	✓	intrinsics extrinsics	real
EpipolarPose [151] CanonPose [284]	×	✓	optional	real
MetaPose [280]	×	✓	×	real
Keypoint3D [50]	×	×	intrinsics extrinsics	simulation
Ours (3D discovery)	×	×	intrinsics extrinsics	real

Table 3.1: **Comparison of our work with representative related work for 3D pose using multi-view training**. Previous works require either 3D or 2D supervision, or simulated environments to train jointly with reinforcement learning. Our method addresses a gap in discovering 3D keypoints from real videos without 2D or 3D supervision.

3.3 Related Work

3D Pose Estimation. There has been a large body of work studying 3D human pose estimation from images or videos, as reviewed in [237, 285], with recent works also focusing on 3D animal poses [79, 179, 108, 146, 113]. Most of these methods are fully supervised from visual data [135, 265, 53], with some models perform lifting starting from 2D poses [183, 51, 208, 225]. We focus our discussion on multi-view 3D pose estimation methods, but all of these models require either 3D or 2D supervision during training. This 2D supervision is typically in the form of pre-trained 2D detectors [151], or ground truth 2D poses [280]. In comparison, our method uses multi-view videos to discover 3D keypoints without 2D or 3D supervision.

Methods more closely related to our work are those that also leverage multi-view structure to estimate 3D pose (Table 3.1). [135] proposed a supervised method that uses learnable triangulation to aggregate 2D information across views to 3D. Here we study similar approaches for representing 3D information, but using self-supervision instead of supervised 3D annotations. Other methods in this space propose training methods such as enforcing consistency of predicted poses across views [227], regression to 3D pose estimated from epipolar geometry of multi-view 2D [151], constraining 3D poses to project to realistic 2D pose [52], or estimates camera parameters using detected and ground truth 2D poses [280]. While we also leverage multi-view information, our goal is different from the work above, in that our approach aims to discover 3D poses without 2D or 3D supervision, given camera parameters.

Self-supervised Keypoint Discovery. 2D keypoint discovery has been studied from images [138, 307, 119] and videos [139, 264]. Our approach focuses on behavioral videos, similar to [264], but we aim to use multi-view information to discover 3D keypoints, instead of 2D. Many approaches use an encoder-decoder setup to disentangle appearance and geometry information [307, 138, 162, 264]. Our setup also consists of encoders and decoders, but our encoder maps information across views to aggregate 2D information into a 3D geometry bottleneck. The discovery model most similar to our approach is Keypoint3D [50], which

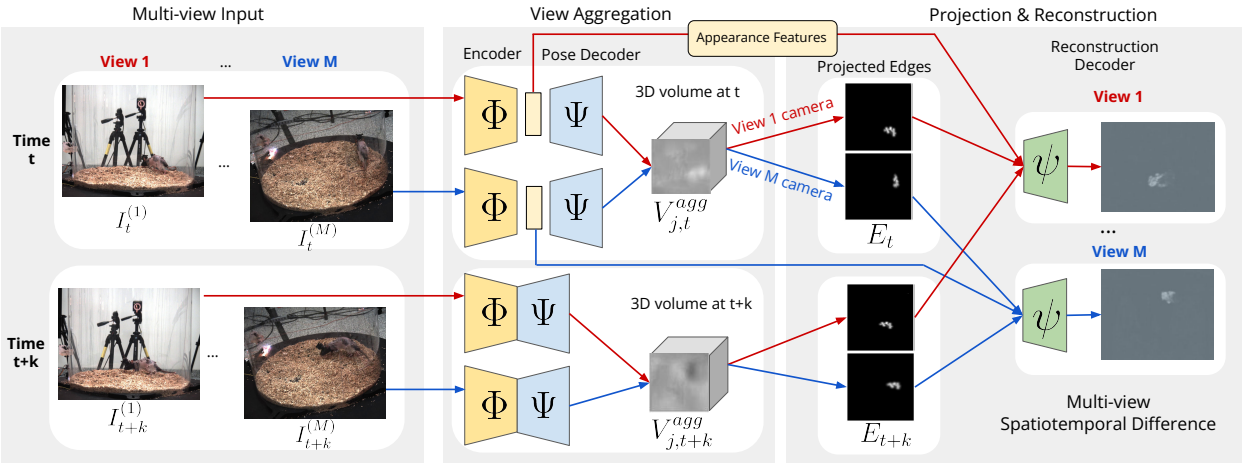

Figure 3.2: **BKinD-3D: 3D keypoint discovery using 3D volume bottleneck**. We start from input multi-view videos with known camera parameters, then unproject feature maps from geometric encoders into 3D volumes for timestamps t and $t+k$. We next aggregate 3D points from volumes into a single edge map at each timestamp, and use edges as input to the decoder alongside appearance features at time t. The model is trained using multi-view spatiotemporal difference reconstruction. Best viewed in color.

discovers 3D keypoints for control from virtual agents, using a combination of image reconstruction and reinforcement learning. However, this setup is designed for simulated data and does not translate well to real videos, since updating the keypoints through a reinforcement learning policy requires videos generated through the simulated environment. Keypoint discovery models typically represent discovered parts as 2D Gaussian heatmaps [138, 264] or 2D edges [119]. While we also use an edge-based representation, our edges are in 3D, which enables our training objective to enforce joint length consistency.

Behavioral Video Analysis. Pose estimation is a common intermediate step in automated behavior quantification; behavioral videos are commonly captured with stationary camera and background, with moving agents. To date, supervised 2D pose estimators are most often used for analyzing behavior videos [143, 124, 87, 184, 83, 249]. However, 2D pose estimation is inadequate for many applications: it cannot reliably capture the angle of joints for kinematics, fails to generalize across views, is sensitive to occlusion, and cannot incorporate body plan constraints as skeleton length or range of motion of joints. Thus, there has recently been an accelerating trend to study behavior in 3D [146, 179, 79, 108]. These models typically require more expensive 3D training annotations compared to 2D poses. While 2D self-supervision has been studied for behavioral videos [264], 3D keypoint discovery in real-world behavioral videos have not been well-explored.

3.4 Method

Our goal is to discover 3D keypoints from multi-view behavioral videos without 2D or 3D supervision (Figure 3.2). Our approach is inspired by BKinD [264], which uses spatiotemporal difference reconstruction to discover 2D keypoints in behavioral videos. In these videos, the camera and background is stationary, and spatiotemporal difference provides a strong signal for encoding agent movement.

We develop several approaches for 3D keypoint discovery, but focus on our volumetric model (Figure 3.2) in this section, as this model generally performed the best in our evaluations. More details on other approaches are in Section 3.5.1 and supplemental materials.

In our volumetric model (BKinD-3D, Figure 3.2) we use multi-view spatiotemporal reconstruction to train an encoder-decoder architecture with 2D information aggregated to a 3D volumetric heatmap. Projections from the 3D heatmap in the form of agent skeletons are then used to reconstruct movement, represented by spatiotemporal difference, in each view.

3.4.1 3D Keypoint Discovery

Given behavioral videos captured from M synchronized camera views, with known camera projection matrix $P^{(i)}$ for each camera $i \in \{1...M\}$, we aim to discover a set of J 3D keypoints $U_t \in \mathbb{R}^{J\times 3}$ on a single behaving agent, at each timestamp t. We assume access to camera projection matrices so that our model discovers 3D keypoints in the global coordinate frame.

During training, our model uses two timestamps in the video t and $t+k$ to compute the spatiotemporal difference in each view as the reconstruction target. In other words, for each camera view i, our training starts with a frame $I_t^{(i)}$ and a future frame $I_{t+k}^{(i)}$. During inference, only a single timestamp is required: once the model is trained, the model only needs $I_t^{(i)}$ for each camera view i.

In our model setup, the appearance encoder Φ, geometry decoder Ψ, and reconstruction decoder ψ are shared across views and timestamps (in previous work [264], these networks are shared across timestamps, but only a single view is addressed). The appearance encoder Φ is used to generate appearance features, which are decoded into 2D heatmaps by the geometry decoder Ψ. These 2D heatmaps are then aggregated across views to form a 3D volumetric bottleneck (Section 3.4.1), which is processed by a volume-to-volume network ρ. We compute the 3D keypoints using spatial softmax on the 3D volume. Then, we project these keypoints to 2D, compute edges between points, and output these edges into the reconstruction decoder ψ (Section 3.4.1) for training. The reconstruction decoder ψ is only used during training, and not required for inference.

Feature Encoding

To start, we first compute appearance features from frame pairs $I_t^{(i)}$ and $I_{t+k}^{(i)}$ using the appearance encoder Φ: $\Phi(I_t^{(i)})$ and $\Phi(I_{t+k}^{(i)})$. These appearance features are then fed into the geometry decoder Ψ to generate 2D heatmaps $\Psi(\Phi(I_t^{(i)})) = H_t^{(i)}$ and $H_{t+k}^{(i)}$. Each 2D heatmap has C channels, where $H_{t,c}^{(i)}$ represents channel c of $H_t^{(i)}$.

View Aggregation using Volumetric Model

To aggregate information across views, we unproject our 2D heatmaps to a 3D volumetric bottleneck. We perform view aggregation separately across timestamps t and $t + k$.

We aggregate 2D heatmaps into a 3D volume similar to [135], which used previously for supervised 3D human pose estimation. One important difference is that in the supervised setting, an $L \times L \times L$ sized volume is drawn around the human pelvis, with L being around twice the size of a person. As we perform keypoint discovery, we do not have information on the location or size of the agent. Instead, we initialize our volume with L representing the maximum size of the space/room for the behaving agent.

This process aggregates 2D heatmaps $H_{t,c}^{(i)}$ for cameras $i \in \{1...M\}$ and channels $c \in \{1...C\}$ to 3D keypoints U_t, for timestamp t. Our volume is first discretized into voxels $V_{coords} \in \mathbb{R}^{B \times B \times B \times 3}$, where B represents the number of distinct coordinates in each dimension. Each voxel corresponds to a global 3D coordinate. These 3D coordinates are projected to a 2D plane using the projection matrices in each camera view i: $V_{proj}^{(i)} = P^{(i)} V_{coords}$. A volume $V_c^{(i)}$ is then created and filled for each camera view i and each channel c using bilinear sampling [137] from the corresponding 2D heatmap: $V_c^{(i)} = H_{t,c}^{(i)}\{V_{proj}^{(i)}\}$, where $\{\cdot\}$ denotes bilinear sampling.

We then aggregate these $V_c^{(i)}$ across views for each channel c using a softmax approach [135]:

$$V_c^{agg} = \sum_i \frac{\exp(V_c^{(i)})}{\sum_j \exp(V_c^{(j)})} \odot V_c^{(i)}.$$

V^{agg} is then mapped to 3D heatmaps corresponding to each joint using a volumetric con-

volutional network [195] ρ: $V^{agg*} = \rho(V^{agg})$. We compute the 3D spatial softmax over the volume, for each channel j of V_j^{agg*}, $j \in \{1...J\}$, to obtain the 3D keypoint locations U_t for timestamp t, as in [135]. In many supervised works, the keypoint locations U_t are optimized to match to ground truth 3D poses; however, we aim to discover 3D keypoints, and train our network by using U_t to decode spatiotemporal difference across views.

Projection and Reconstruction

In this step, we project the discovered 3D keypoints to a 2D representation in each view using camera parameters. For training, 2D representations in timestamps t and $t+k$ are used as input to the reconstruction decoder ψ. We train the 3D keypoints U_t at each timestamp t using multi-view spatiotemporal difference reconstruction. The target spatiotemporal difference is computed using the 2D image pair $I_t^{(i)}$ and $I_{t+k}^{(i)}$ at each view i.

First, we project the 3D keypoints using camera projection matrices into 2D keypoints $u_t^{(i)} = P^{(i)}U_t$. We create an edge representation for each view for each timestamp, which enables us to discover connections between points and enforce 3D joint length constraints. For each keypoint pair $u_{t,m}^{(i)}$ and $u_{t,n}^{(i)}$, we draw a differentiable edge map as a Gaussian along the line connecting them, similar to [119]:

$$E_{t,(m,n)}^{(i)}(\mathbf{p}) = \exp(d_{m,n}^{(i)}(\mathbf{p})^2/\sigma^2),$$

where σ controls the line thickness and $d_{m,n}(\mathbf{p})^{(i)}$ is the distance between pixel $\mathbf{p}$ and the line connecting $u_{t,m}^{(i)}$ and $u_{t,n}^{(i)}$. We then aggregate the edge heatmaps at each timestamp using a set of learned weights $w_{m,n}$ for each edge, where $w_{m,n}$ is shared across all timestamps and all views. An edge is active and connects two points if $w_{m,n} > 0$, otherwise the points are not connected. Finally, we aggregate all edge heatmaps using the max across all edge pairs [119]:

$$E_t^{(i)}(\mathbf{p}) = \max_{m,n} w_{m,n} E_{t,(m,n)^{(i)}}(\mathbf{p}).$$

In our framework, for each view i, the decoder ψ uses the edge maps $E_t^{(i)}$ and $E_{t+k}^{(i)}$ as well as the appearance feature $\Phi(I_t^{(i)})$ for reconstructing the spatiotemporal difference across

each view. The ground truth spatiotemporal difference is computed from the original images $S(I_t^{(i)}, I_{t+k}^{(i)})$. The reconstruction from the model is $\hat{S} = \psi(E_t^{(i)}, E_{t+k}^{(i)}, \Phi(I_t^{(i)}))$, through the 3D volumetric bottleneck in order to discover informative 3D keypoints for reconstructing agent movement.

3.4.2 Learning Formulation

The entire training pipeline (Figure 3.2) is differentiable, and we train the model end-to-end. We note that our model is only given multi-view video and corresponding camera parameters, without keypoint or bounding box supervision.

Multi-View Reconstruction Loss

Our multi-view spatiotemporal difference reconstruction is based on the single-view spatiotemporal difference studied for 2D keypoint discovery [264]. We compute the Structural Similarity Index Measure (SSIM) [286] as a reconstruction target in each view. SSIM has been used to measure perceived differences between images based on luminance, contrast, and structure features. Here, we use SSIM as a reconstruction target and we compute a similarity map using local SSIM on corresponding patches between $I_t^{(i)}$ and $I_{t+k}^{(i)}$. This similarity map is negated to obtain the dissimilarity map used as the target: $S(I_t^{(i)}, I_{t+k}^{(i)})$.

We use perceptual loss [141] in each view between the target S and the reconstruction $\hat{S}$. This loss computes the L2 distance between features of the target and reconstruction computed from the VGG network ϕ [251]:

$$\mathcal{L}_{recon}^{(i)} = \left\| \phi(S(I_t^{(i)}, I_{t+T}^{(i)})) - \phi(\hat{S}(I_t^{(i)}, I_{t+T}^{(i)})) \right\|_2 . \tag{3.1}$$

The error is computed by comparing features from intermediate convolutional blocks of the network. Our final perceptual loss is summed over each view $\mathcal{L}_{recon} = \sum_i \mathcal{L}_{recon}^{(i)}$.

Learned Length Constraint

Since many animals have a rigid skeletal structure, we encourage that the length of active edges ($w_{m,n} > 0$ for point pairs m and n) are consistent across samples. We do not assume that these lengths and connections are known, such as previous work [280]; rather, they are learned during training. We do this by maintaining a running average of the length of all active edges $l_{avg(m,n)}$, and minimizing the difference between the average length and each sample $l_{m,n}$:

$$\mathcal{L}_{\text{length}} = \sum_m \sum_n \mathbb{1}_{w_{m,n}>0} \left\| l_{avg(m,n)} - l_{m,n} \right\|_2 . \tag{3.2}$$

During training, we update $l_{avg(m,n)}$ using an exponential running average and $w_{m,n}$ indicating edge weights for every pair is learned. Both of these parameters are shared across all viewpoints and timestamps. Notably, the length constraint is only applied to active edges, since there are many point pairs without rigid connections (e.g. elbow to feet), while we want to enforce this constraint only for rigid connections (e.g. elbow to wrist).

Separation Loss

To encourage unique keypoints to be discovered, we apply separation loss to our 3D keypoints, which has been previously studied in 2D [307, 264]. On a set of 3D keypoints U_{it}, where i is the index of a keypoint and t is the time, the separation loss is:

$$\mathcal{L}_s = \sum_{i \neq j} \exp\left(\frac{-(U_{it} - U_{jt})^2}{2\sigma_s^2}\right), \tag{3.3}$$

where σ_s is a hyperparameter that controls the strength of separation.

Training Objective

Our full training objective is the sum of the multi-view spatiotemporal reconstruction loss $\mathcal{L}_{recon}$, learned length constraints $\mathcal{L}_{length}$, and separation loss $\mathcal{L}_s$:

$$\mathcal{L} = \mathcal{L}_{recon} + \mathbb{1}_{epoch>e}(\omega_r \mathcal{L}_{length} + \omega_s \mathcal{L}_s). \tag{3.4}$$

Our model is trained using curriculum learning [26]. We only apply $\mathcal{L}_{length}$ and $\mathcal{L}_s$ when the keypoints are more consistent, after e epochs of training using reconstruction loss.

3.5 Experiments

We demonstrate BKinD-3D using real-world behavioral videos, using a human dataset and a recently released large-scale rat dataset (Section 3.5.1). We evaluate our discovered keypoints using a standard linear regression protocol based on previous works for 2D keypoint discovery [138, 264] (also described in Section 3.5.1). Here, we present results on pose regression (Section 5.3) with ablation studies (Section 3.5.3), with additional results in supplementary materials.

3.5.1 Experimental Setup

Datasets

We demonstrate our method by evaluating it on two representative datasets: Human 3.6M and Rat7M. The datasets have different environments and focus on subjects of different sizes, with humans being about 1700mm tall and rats about 250mm long.

Human 3.6M. We evaluate our method on Human3.6M to compare to recent works in self-supervised 3D from 2D [280]. Human 3.6M [131] is a large-scale motion capture dataset with videos from 4 viewpoints. We follow the standard evaluation protocol [135, 151] to use subjects 1, 5, 6, 7, and 8 for training and 9 and 11 for testing. Our test set matches the set specified in [280] using every 16th frame (8516 test frame sets). Notably, unlike baselines such as [135], our method does not require any pre-processing with 2D bounding box annotations but rather is directly applied to the full image frame.

Rat7M. We also evaluate our method on Rat7M [79], a 3D pose dataset of rats moving in a behavioral arena. This dataset most closely matches the expected use case for our method, which is a dataset of non-human animal behavior in a static environment. Rat7M consists videos from 6 viewpoints captured at 1328×1048 resolution and 120Hz, along with ground

truth annotations obtained from marker-based tracking. We train on subjects 1, 2, 3, 4, and test on subject 5, as in [79]. We train and evaluate on every 240th frame of each video (3083 train, 1934 test frame sets).

Model Comparisons

We compare our method with three main categories of baselines: supervised 3D pose estimation methods (ex: [135]), 3D pose estimation methods from 2D supervision (ex: [280]), and a 3D keypoint discovery method developed for control in simulation [50]. A more detailed comparison of methods in this space is in Table 3.1. For baselines with model variations, we use evaluation results from the version that is the closest to our model (multi-view inference, and camera parameters during inference). We note that all previous methods require additional 3D or 2D supervision, or jointly training a reinforcement learning policy in simulation [50], which we do not require for 3D keypoint discovery in real videos. Another notable difference is that previous methods typically pre-process video frames using detected or ground truth 2D bounding boxes [135], while our method does not require this pre-processing step.

Since 3D keypoint discovery has not been thoroughly explored, we additionally study methods in this area using multi-view 2D discovery and triangulation (Triang.+Reproj.), and multi-view 2D discovery with a depth map estimates (Depth Map), in addition to our volumetric approach (Section 3.4, BKinD-3D). For multi-view 2D discovery and triangulation, we use BKinD [264] to discover 2D keypoints in each view, and perform triangulation using camera parameters to obtain 3D keypoints. We then project the 3D keypoints for multi-view reconstruction. We add an additional loss on the reprojection error to learn keypoints consistent across multiple views. For the depth map approach, in each camera view, we estimate 2D heatmaps corresponding to each keypoint alongside a view-specific depthmap estimate. The final 3D keypoints are then computed from a confidence-weighted average of each view's estimated 3D keypoint coordinates (from the per-view 2D heatmaps and depth estimates). More details on each method are in the supplementary materials.

Training and Evaluation Procedure

We train our volumetric approach using the full objective (Eq 3.4). We scale images to 256×256 for training, with a frame gap of 0.4s for Human3.6M and 0.66s for Rat7M. We use a maximum volume size of 7500mm for Human3.6M and 1000mm for Rat7M. The results are computed for all 3D keypoint discovery methods with 15 keypoints unless otherwise specified. We train using videos from the train split with camera parameters provided by each dataset.

We evaluate our 3D keypoint discovery through keypoint regression based on similar methods from 2D, using a linear regressor without a bias term [264, 138, 307]. For this regression step, we extract our discovered 3D keypoints from a frozen network, and learn a linear regressor to map our discovered keypoints to the provided 3D keypoints in each of the training sets. We then perform evaluation on regressed keypoints on the test set.

For metrics, we compute Mean Per Joint Position Error (MPJPE) in line with previous works in 3D pose estimation [135, 133], which is the L2 distance between the regressed and ground truth 3D poses, accounting for the mean shift between the regressed and ground truth points. To compare to methods that require addition alignment before MPJPE computation (e.g. [280] which does not use camera parameters during inference), we also compute Procrustes aligned MPJPE (PMPJPE) [280, 151, 133]. PMPJPE applies the optimal rigid alignment to the predicted and ground truth 3D poses before metric computation.

3.5.2 Results

We evaluate our discovered keypoints quantitatively using keypoint regression on Human3.6M (Table 3.2) and Rat7M (Table 3.3). Over both datasets with diverse organisms, our approach generally outperforms all other fully self-supervised 3D keypoint discovery approaches. Additionally, among all the approaches we developed for 3D keypoint discovery, BKinD-3D using the volumetric bottleneck performs the best overall. Results demonstrate that BKinD-3D is directly applicable to discover 3D keypoints on novel model organisms, potentially very different in appearance or size, without 2D or 3D supervision.

Method	Supervision	PMPJPE ↓	MPJPE ↓
	Supervised 3D		
Anipose [146]	2D only	-	33
Rhodin et al. [227]	3D/2D	52	67
Isakov et al. [135]	3D/2D	-	21
	Supervised 2D + self-supervised 3D		
CanonPose [284]	2D	53	74
EpipolarPose [151]	2D	67	77
Iqbal et al. [133]	2D	55	69
MetaPose [280]	2D	32	-
	3D Discovery + Regression		
Keypoint3D [50]	×	168	368
Ours:			
Triang+reproj	×	134	241
Depth Map	×	122	161
BKinD-3D	×	105	125

Table 3.2: **Comparing performance with related work on Human3.6M**. We note that previous approaches typically require additional 2D or 3D supervision, whereas our model discovers 3D keypoints directly from multi-view video. The 3D keypoint discovery models are evaluated using a linear regression protocol (Section 3.5.1).

Notably, on Humam3.6M, Keypoint3D [50], developed for control of simulated videos, does not work well in our setting with real videos, and qualitative results demonstrate that this method was not able to discover keypoints that tracked the agent (supplementary materials).

Qualitative results. We find that the discovered points and skeletons are reasonable and look similar to the ground truth annotations for Human3.6M (Figure 3.3) and Rat7M (Figure 3.4). Furthermore, we find that a volumetric model with 30 keypoints learns a more detailed human skeleton representation than a model with 15 keypoints. For example, the model with 30 keypoints is able to track both legs, while the 15 keypoint model only tracks 1 leg; however, both models miss the knees. Importantly, our model discovers the skeleton in global coordinates, and is able to track the agent as they move around the space. More examples are in supplementary materials.

While there exists a gap in terms of quantitative metrics between supervised methods and self-supervised 3D keypoint discovery, supervised methods require users to invest time and resources for annotations. In comparison, our method can be deployed out-of-the-box on new datasets and experiments with multi-view cameras. Our approach has closed the gap substantially to supervised methods compared to previous work, without requiring time-consuming 2D or 3D annotations. Qualitative results demonstrate that our approach is able to discover structure across diverse model organisms, providing a method for accelerating the study of organism movements in 3D.

Downstream Analysis. To further evaluate our keypoint discovery method, we use BKinD-3D keypoints as input to a 1D convolutional neural network (previously used in [263]) to predict action labels on Human3.6M. Notably, we found that our keypoints performs similarly to ground truth 3D points for action recognition, where Top 5 accuracy is 64.8% (GT), 61.0% (15 kpts), and 64.9% (30 kpts) (supplementary material).

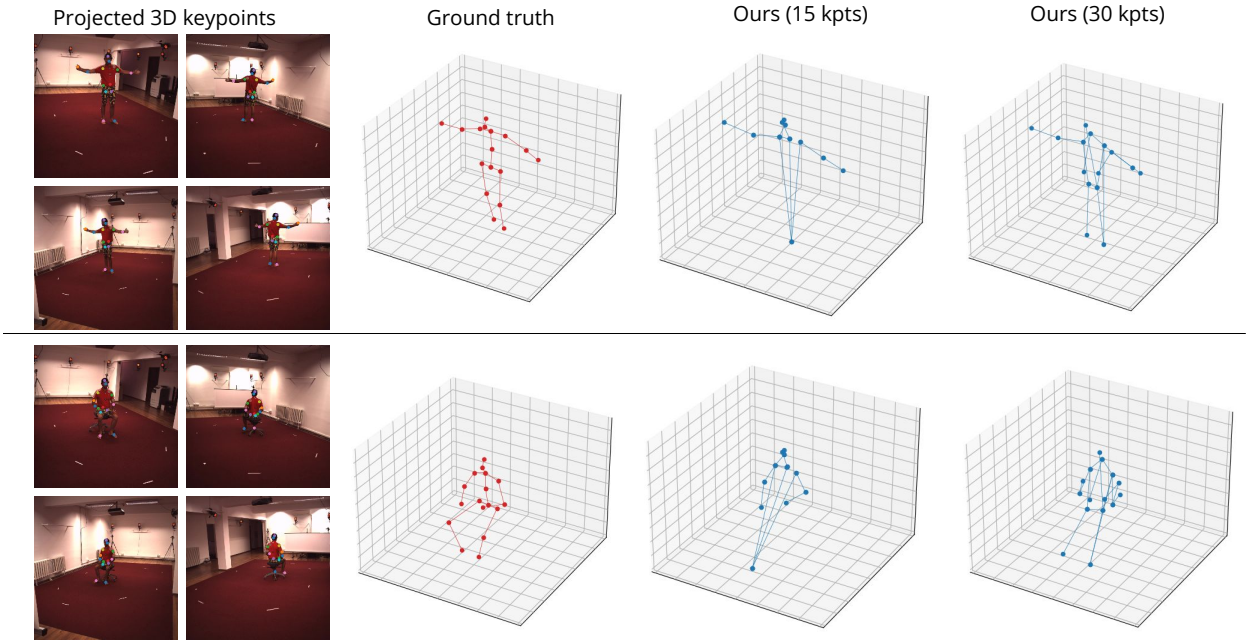

Figure 3.3: **Qualitative results for 3D keypoint discovery on Human3.6M**. Representative samples of 3D keypoints discovered from BKinD-3D without regression or alignment for 15 and 30 total discovered keypoints. We visualize all keypoints that are connected using the learned edge weights, and the projected 3D keypoints in the leftmost column are from the keypoint model with 30 discovered keypoints.

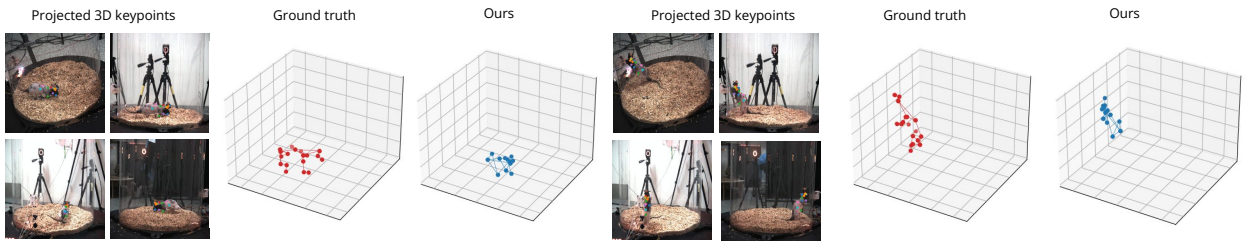

Figure 3.4: **Qualitative results for 3D keypoint discovery on Rat7M**. Representative samples of 3D keypoints discovered from BKinD-3D without regression or alignment. We visualize all connected keypoints using the learned edge weights and visualize the first 4 cameras (out of 6 cameras) in Rat7M for projected 3D keypoints.

Method	Supervision	PMPJPE ↓	MPJPE ↓
	Supervised 3D		
DANNCE [79]	3D	11	-
	3D Discovery + Regression		
Ours:			
Triang+reproj	×	21	108
Depth Map	×	27	56
BKinD-3D	×	24	76

Table 3.3: **Comparison with 3D keypoint discovery methods on Rat7M**. Results from the top three 3D keypoint discovery methods on Rat7M. The 3D keypoint discovery models are evaluated using a linear regression protocol (Section 3.5.1).

Method	PMPJPE ↓	MPJPE ↓
BKinD-3D (8 kpts)	120	149
BKinD-3D (15 kpts)	105	125
BKinD-3D (30 kpts)	109	130
BKinD-3D (point)	110	137
BKinD-3D (edge, without length)	108	129
BKinD-3D (edge, full objective)	105	125

Table 3.4: **Ablation results on Human3.6M**. We perform an ablation study of our volumetric bottleneck method comparing different numbers of keypoints as well as variations to the edge bottleneck with length constraints.

3.5.3 Ablation

We perform an ablation study of our model (Table 3.4), focused on BKinD-3D as it is the best performing approach on Human3.6M. Results show that 15 keypoints performed the best quantitatively, but 30 keypoints is comparable and qualitatively provides a more informed skeleton (Figure 3.3). We perform additional regression experiments using a 2-layer MLP regressor (supplementary material), and we found that the keypoints discovered by the 30 keypoints model (94 PMPJPE) perform better relative to 15 keypoints (98 PMPJPE). This suggests that the linear model may have been underfitting our 30 keypoints model.

We additionally find that adding edge information has a quantitative improvement on performance and provides more qualitative information on connectivity between joints (Figures 3.3, 3.4). In our 3D setting, we found that the point bottleneck (studied in previous works in 2D [264, 138]) did not work as well as the edge bottleneck (studied in previous works in 2D [119]). By studying edge bottlenecks in 3D and expanding beyond 2D, our approach is able to enforce joint length constraints through the discovered edge connectivity.

3.6 Discussion

We present a method for 3D keypoint discovery directly from multi-view video, without any requirement for 2D or 3D supervision. Our method discovers 3D keypoint locations as well as joint connectivity in behaving organisms using a volumetric heatmap with multi-view spatiotemporal difference reconstruction. Results show that our work has closed the gap significantly to supervised methods for studying 3D pose, and is applicable to different organisms.

Our approach focuses on behavioral videos with stationary cameras and background, with known camera parameters. The applicability of 3D keypoint discovery can be further improved with future work to jointly estimate camera parameters, camera movement, and pose from visual data. Additionally, the lack of publicly available multi-view datasets of animals could limit model development and evaluation. Open-sourcing more datasets in

this area would encourage the development of pose estimation models with broader impacts beyond humans. Despite these challenges, 3D keypoint discovery has the potential to enable studying behavior of diverse organisms, without the need for expensive and time-consuming annotations. Our goal is to encourage more efforts in 3D keypoint discovery, to study the capabilities of vision models and to facilitate the study of behavior in new organisms and across diverse experimental setups.

3.7 Acknowledgments

This work is generously supported by the Amazon AI4Science Fellowship (to JJS), NIH NINDS (R01NS102333 to JCT), the Air Force Office of Scientific Research (AFOSR FA9550-19-1-0386 to BWB), and NSF (1918865 to YY).

Chapter 4

COMPUTATIONAL MODELS OF PROPRIOCEPTION

This chapter was published in *Current Opinion in Physiology* in 2021 [66].

Authors: Chris J. Dallmann*, Lili Karashchuk*, Bingni W. Brunton, and John C. Tuthill

* Equal contribution

4.1 Abstract

Dexterous motor control requires feedback from proprioceptors, internal mechanosensory neurons that sense the body's position and movement. An outstanding question in neuroscience is how diverse proprioceptive feedback signals contribute to flexible motor control. Genetic tools now enable targeted recording and perturbation of proprioceptive neurons in behaving animals; however, these experiments can be challenging to interpret, due to the tight coupling of proprioception and motor control. Here, we argue that understanding the role of proprioceptive feedback in controlling behavior will be aided by the development of multiscale models of sensorimotor loops. We review current phenomenological and structural models for proprioceptor encoding and discuss how they may be integrated with existing models of posture, movement, and body state estimation.

4.2 Experimental perturbations probe the role of proprioceptive feedback in motor control

All animals possess specialized sensory neurons that monitor the mechanical consequences of their actions. These sensors, known as proprioceptors, are essential for coordinating body movement and maintaining body posture [224, 275]. While it is possible for some isolated motor circuits to generate structured output in the absence of proprioceptive feedback,

behaviors driven by purely feedforward motor signals are often clumsy and ineffectual [107]. Understanding how proprioceptive feedback interacts with motor circuits to control the body remains a fundamental problem in neuroscience.

An effective method to investigate the function of sensory circuits is to perturb neural activity and measure the effect on an animal's behavior. For example, activating or silencing neurons in the mammalian visual cortex [180] or insect optic glomeruli [298] has identified the circuitry and patterns of activity that underlie visually-guided behaviors. However, due to the distributed nature of proprioceptive sensors and their tight coupling with motor control circuits, perturbations to the proprioceptive system can be difficult to execute and tricky to interpret.

4.2.1 Mechanical perturbation experiments

Early efforts to understand the behavioral contributions of proprioceptive feedback relied on lesions and mechanical perturbations. A classic example of a proprioceptive perturbation experiment is the use of vibration to artificially excite primary muscle spindle afferents. In humans, muscle vibration creates the illusory perception that a muscle is being stretched [106]. If a person is walking, vibrating their hamstring produces an increase in forward walking speed, while vibrating their quadriceps has little effect on walking kinematics [136] (Figure 4.1, top left). During backward walking, however, these effects are reversed, suggesting that proprioceptive feedback acts in a context-dependent manner.

Longer-term mechanical perturbations have also been used to investigate the dynamics of sensorimotor adaptation. Bässler et al. [46] inverted the sign of feedback signals from the femoral chordotonal organ in the stick insect leg by surgically crossing the receptor tendon (Figure 4.1, bottom left). This manipulation causes a walking stick insect to either "salute" or "drag" her leg, and a standing stick insect to rhythmically wave her tibia. Over the course of a month, Bässler and colleagues observed that the walking salute and dragging remained unchanged, while the waving movements gradually decreased. This observation suggests that the postural feedback control system recovered, while the control system for leg movement

did not.

These examples illustrate several hazards to consider when executing and interpreting proprioceptive perturbation experiments. First, the effects of proprioceptive feedback on motor output are often context-dependent, such as during forward and backward walking. Second, proprioceptive signals are typically used for controlling both posture and movement, often at the same time. An additional challenge of mechanical perturbations is that they often lack specificity, leaving it unclear which specific proprioceptor neurons underlie the observed behavior.

4.2.2 Neural perturbation experiments

With the emergence of genetic tools to label and manipulate specific cell-types, it has recently become possible to genetically activate or silence specific proprioceptive neurons to assess their role in behavior. However, such manipulation experiments come with their own challenges [295]. For example, Agrawal et al. [6] used optogenetics to study a population of second-order proprioceptive neurons in *Drosophila* that encode extension of the fly's tibia. Activation of these neurons caused walking flies to slow down, suggesting that they may participate in controlling walking speed (Figure 4.1, top right). However, fine-grained kinematic analysis of leg joints revealed that the same perturbation produced reflexive flexion of the tibia whenever the leg was unloaded, not just during walking. Thus, the decrease in walking speed was likely driven by the fly's reaction to a perceived extension of her tibia, rather than a disruption to walking speed control.

Another recent study used intersectional genetic methods to test the role of local proprioceptive feedback in rodent locomotion. When Takeoka et al. [270] expressed Diphtheria toxin (DTX) in muscle spindles that innervate the front or rear legs of adult mice, they observed only subtle changes in spontaneous walking gait. The inter-limb coordination deficits produced by these ablations became apparent only when mice were forced to run at high speeds on a motorized treadmill (Figure 4.1, bottom right).

These examples illustrate how it can be misleading to infer a direct, causal relationship

between neural activity and behavior when the perturbation impacts multiple feedback loops at different levels of the nervous system. For example, activating or silencing proprioceptors may alter postural reflexes that interfere with an ongoing behavior, such as walking. In other cases, changes to motor output produced by genetic perturbations may only become visible in specific behavioral contexts. Finally, removing proprioceptive feedback can lead to general deficits to motor coordination that alter an animal's ability or desire to perform certain behavioral tasks.

4.3 How models can help

Our thesis is that the design and interpretation of such perturbation experiments would benefit from the integration of models for proprioceptive sensing with theoretical frameworks for motor control. Because of the tight feedback between proprioceptive and motor circuits at multiple levels, the emergent behavior of a sensorimotor system depends on how its components are integrated. Therefore, computational simulations are a means to generate specific, quantitative predictions about the roles of key parameters in the system, including sensory delay, gain modulation, and encoding nonlinearities. By formalizing how the components contribute to overall behavior, an integrated model can illustrate how different manipulations may produce similar results and aid in the design of experiments to disambiguate among multiple hypotheses.

Our overall goal is to guide the design and interpretation of experiments to understand the role of proprioceptive neurons in flexible control of behavior. We first review current approaches to model sensory coding in proprioceptive sensory neurons. Next, we discuss the role of proprioceptive feedback in existing models of posture, locomotion, skilled movements, and body state estimation. Although proprioception plays an important role in motor learning, here we focus on innate behaviors driven by movement of the limbs, such as walking. We conclude by discussing three obstacles to achieving a systems-level understanding of proprioception: (1) the diversity of proprioceptive sensors, (2) a lack of experimental separability between proprioceptive and motor circuits, and (3) context-dependent modulation

of proprioceptive feedback for guiding behavior.

4.4 Computational models of sensory coding in proprioceptors

Proprioceptors are mechanosensory neurons located within muscles, tendons, and joints that convert mechanical forces in the body into patterns of neural activity. Proprioceptors can be classified into different functional subtypes that encode limb displacement, load, or their time derivatives [275]. For example, the proprioceptors innervating a muscle spindle encode either muscle length or a combination of muscle length and its rate of change (velocity) [168]. Similarly, proprioceptors in the insect femoral chordotonal organ encode distinct kinematic parameters, including position, velocity, and acceleration [173, 90]. Recent studies have begun to map these functional subtypes onto neuronal cell-types defined using genetic markers [173, 206]. These tools now enable targeted perturbation experiments to investigate the role of proprioceptor subtypes in specific motor tasks.

Due to a combination of neural and mechanical properties [21, 236], proprioceptors are tuned to biologically relevant stimuli. These same properties can make proprioceptor responses vary as a function of stimulus history and behavioral context. Such nonlinearities make proprioceptors different from most engineered sensors, which are designed to directly measure a physical quantity, such as a joint angle or torque. Because sensory neurons constrain the information available for other downstream computations in the nervous system, the construction of proprioceptor models is an essential first step in quantitative analysis of sensorimotor loops.

Current models of proprioceptors fall into two broad categories: phenomenological models and structural models. Phenomenological models reproduce the computational properties of a proprioceptor in abstraction, for instance by deriving a mathematical function, often a transfer function, from experimentally determined relations between mechanics and neural activity [159, 172, 218, 220, 269, 125]. This function is often derived from recorded neural activity in response to simple ramp-and-hold or sinusoidal stimuli using linear systems theory or non-linear curve fitting.

Phenomenological models are compact and computationally efficient, which allows them to be integrated with models of the motor system. For example, the simple muscle spindle models by Prochazka and Gorassini [218, 220] have been incorporated into musculoskeletal models of arms and legs [148, 217, 176, 235]. Similarly, the Golgi tendon organs models by Houk and Simon [125] and Rosenthal et al. [232] have been used in models of the stretch reflex [159, 231]. A drawback of phenomenological models derived from ramp-and-hold or sinusoidal stimuli is that they might perform well for only a narrow subset of stimuli [218, 171]. This overfitting is particularly problematic when proprioceptor activity is context-dependent, such as in muscle spindles. Measuring the responses of proprioceptors to more complex stimuli (e.g., white noise or naturalistic limb trajectories) and incorporating nonlinearities are both likely to produce more generalizable models of spiking responses to sensory stimuli [95, 94].

Unlike abstract phenomenological models, structural models derive firing patterns of proprioceptors by approximating their anatomical structure [32, 31, 117, 160, 192, 193, 240]. For example, structural models of muscle spindles simulate intrafusal muscle fibers and their interaction with the extrafusal muscle and tendon. In a recent muscle spindle model by Blum et al. [32, 31], the forces of intrafusal muscle fibers are calculated based on measurements of cross-bridge kinetics. Importantly, this knowledge of mechanics allows the model to predict varied and complex spindle firing patterns, including history-dependent features. Structural models are more computationally demanding and make more assumptions about parameters than phenomenological models, but they tend to generalize better across conditions because firing patterns are emergent rather than built-in. Thus, structural models can be particularly effective for proprioceptors with nonlinear tuning, such as muscle spindles.

4.5 Behavioral functions of proprioceptive feedback

Proprioceptive information is involved in nearly all aspects of motor control, from rapid recovery after a stumble to searching for a light switch with your toes in a dark room while your hands are bound tightly behind your back. When trying to investigate the role of proprioceptive feedback in a specific motor task, it is important to consider the impact of

perturbing proprioceptive sensors at different levels of the sensorimotor hierarchy. Below, we delineate the multiple levels of motor function for which proprioceptive feedback plays an essential role. Although we discuss posture and movement separately, we note that they are ultimately integrated behaviors that are not necessarily controlled independently [7].

4.5.1 Reflexive control of body posture

Proprioceptors from the limbs project to the spinal cord (in vertebrates) and the ventral nerve cord (in invertebrates), where they provide excitatory synaptic input to populations of projection neurons, local interneurons, and motor neurons (Figure 4.2, left). The architecture of these circuits has been extensively studied [40, 44], although only recently have genetic tools begun to reveal the connectivity of identified cell types [216, 6].

Behaviors mediated by direct (monosynaptic) or nearly direct feedback from proprioceptors are typically characterized as "reflexive" because they occur with a shorter latency than voluntary movements [226]. In their simplest form, proprioceptive reflexes rapidly (<50 ms) stabilize body posture against external perturbations. For example, when a physician taps the tendon near your knee cap, muscle spindles elicit a compensatory reaction by activating the quadriceps and inhibiting the antagonistic hamstring. Similarly, when a grasshopper's tibia is flexed, proprioceptive feedback from the femoral chordotonal organ activates the extensor muscle and inhibits the flexor muscle.

Such stabilization reflexes can be modeled as negative feedback controllers, which produce corrective motor output to minimize the error between the sensed posture and a reference posture. These controllers may act locally to stabilize individual joints, or they may act across joints to produce a coordinated response that stabilizes a task-level variable, such as the position of the hand while grasping an object [289] or the position of the body's center of mass while standing [9].

A key parameter to achieve stability in negative feedback controllers is the ratio of motor output to sensory input, known as feedback gain. Tuning feedback gains can be accomplished in modeling studies by identifying the range of parameters in which the model

is stable or by fitting model predictions to experimental data. In rare cases, gains have also been measured experimentally. In one such experiment, Weiland et al. [288] measured the gain of the feedback loop that controls the posture of the stick insect femur-tibia joint. This experiment was only possible because the anatomy of the stretch receptor (femoral chordotonal organ) allowed its mechanical stimulation without affecting the surrounding muscles. Today, optogenetic stimulation may enable equivalent analyses of reflex loops that cannot be manipulated mechanically.

In addition to their established function in basic postural control, elementary reflex loops also participate in more complex behaviors that involve multiple joints or limbs [226]. Understanding the context-dependent role and tuning of reflexes remains an important direction for future research, and computational models will likely be useful in exploring the underlying circuits and algorithms [221, 104].

4.5.2 Feedback control of locomotion

Walking and other forms of locomotion are characterized by rhythmic movements of the body and limbs. The rhythmic pattern of these movements can, in some cases, be generated without feedback by intrinsically rhythmic circuits in the spinal cord and ventral nerve cord, referred to as central pattern generators (CPGs) [30, 110]. However, proprioceptive feedback has long been known to regulate phase transitions, stabilize ongoing movements, and adjust locomotion to changes in the environment [209, 219].

Several recent models of locomotion control combine feedforward motor commands from CPGs with proprioceptive feedback. Such models have been useful in testing the contribution of each control pathway in producing coordinated movements. For example, in a CPG model of cat walking, Markin et al. [176] found that the integration of feedback from muscle spindles and Golgi tendon organs is critical for providing body weight support and coordinating the legs' transitions between stance and swing phases. Proctor and Holmes [222] modeled cockroach running and found that the integration of feedback from leg joints (roughly corresponding to load feedback from campaniform sensilla) with a CPG model helps recover

the heading direction after a lateral perturbation. Similarly, feedback from body stretch receptors effectively tunes rhythmic CPG patterns for swimming and crawling [115, 266]. In fact, locomotion can be successfully modeled without CPGs [241, 72, 253, 74], emphasizing the functional importance of integrating proprioceptive feedback in low-level motor circuits. In high-level motor circuits, proprioceptive information may be used for planning movement on a longer timescale. For instance, during walking, a high-level controller may use proprioceptive feedback to plan desired foot placements, which then serve as target inputs to a low-level controller [253, 241].

The use of proprioceptive feedback to drive or fine-tune motor control of locomotion is ultimately constrained by neural conduction delays and muscle kinetics. During fast running, for example, proprioceptive feedback might not be fast enough to modulate muscle activity on a step-by-step basis [311]. In such situations, stable locomotion can arise from interactions between feedforward control and passive body mechanics [154, 123]. Using direct experimental perturbations of proprioceptive neurons has the potential to disambiguate the relative contributions of feedforward and feedback signals in locomotion control, but interpreting these experiments may require careful modeling of inputs and outputs at multiple scales.

4.5.3 Feedback control of skilled limb movements

The control of non-rhythmic movements, such as putting on a sock or dodging a projectile, also critically depends on proprioceptive feedback. A particularly useful framework for modeling such goal-directed movements is optimal feedback control [273, 92]. Optimal control posits that the motor system attempts to minimize a set of cost functions that describe the performance criteria of a given task—for example, movement effort or accuracy. The role of proprioception in this framework is to help generate and update an accurate estimate of the state of the body, which in turn determines how the controller initiates and refines movements. Unlike the simple negative feedback controllers discussed above, optimal feedback controllers correct for perturbations only to the degree that they interfere with task success.

For example, when throwing a ball to a target, an optimal feedback controller minimizes the variability of hand trajectories only around the time of release while allowing variability at other times [273].

Although optimal feedback control can predict many common features of goal-directed movement, it remains unclear how this strategy may be implemented in the nervous system. It is typically assumed that feedback control of skilled movements is mediated by higher-order regions, such as cortex in vertebrates, but recent experiments suggest that even fast spinal reflexes can produce sophisticated control reminiscent of optimal feedback [289, 290]. In humans and other primates, there is a strong tradition of combining mechanical perturbations with computational models to understand the sensory inputs and algorithms that underlie feedback control of reaching [296, 150]. Recent studies of primate motor cortex have revealed that neural activity during skilled movements exhibits rotational dynamics that lie on a low-dimensional manifold [58]. The role of proprioceptive signals in constructing these representations is currently unclear, although a recent modeling study showed that recurrent neural network models trained to control a limb exhibited similar rotational dynamics [144]. These activity patterns have also been modeled using an implementation of a feedback controller in a spiking neural network [73]. Going forward, these modeling frameworks may also help understand the results of genetic perturbations to neural circuits that underlie skilled motor behavior. As optimal feedback control models generally have few parameters, they may prove particularly useful for providing experimental predictions of behavior when combined with models of proprioceptors.

4.5.4 Body state estimation

Natural proprioceptive signals are noisy and subject to conduction delays, creating problems for continuous feedback control. To quantify and manage the uncertainty introduced by noise and delays, the central nervous system may implement a form of state estimation (Figure 4.2, right), where sensory feedback is combined with a prediction of the body's state from an internal model—a neural representation of the system that is being controlled [189]. This

comparison of sensory feedback against an internal prediction can compensate for sensory delays and make control more robust to sensory noise. Robustness to noise makes state estimation a useful strategy for controlling motor tasks, including posture [155], reaching [191], and locomotion [170]. For example, in a dynamical simulation of posture control during standing, both feedback and state-estimation controllers were found to be stable against external perturbations without noise, but in the presence of proprioceptive noise, only state-estimation-based control was stable [155].

It is typically assumed that state estimation occurs in higher-level circuits, such as the cerebellum and parietal cortex in vertebrates [189], or the mushroom bodies in insects [287]. But an intriguing idea is that locomotor CPGs perform computations equivalent to state estimation and can thus be understood as observers of feedback control rather than just generators of limb motion [233]. This idea suggests that a form of state estimation may occur at multiple timescales and levels of the sensorimotor hierarchy (Figure 4.2, right). Indeed, signals from different proprioceptor subtypes can converge as early as in second-order neurons [6, 44, 39, 38]. However, little is known about how circuits at different levels represent the body and to what degree they combine signals from the diverse proprioceptor subtypes. Going forward, one useful role of computational models could be to predict the proprioceptive information available during natural movement, for example by combining models of proprioceptors with realistic models of the muscles and limbs [235].

State estimates enabled by proprioceptive feedback are also used beyond the motor system. Recent work in the *Drosophila* central complex has revealed circuits that encode the body in both self-centered and world-centered coordinates [167, 166, 129]. Neural encoding of several key quantities that guide spatial navigation during walking, including heading in allocentric coordinates and velocity in egocentric coordinates, persist in darkness [247], which suggests that they rely on proprioceptive signals from the legs. However, it remains unknown precisely which proprioceptive signals are used to compute these signals. Computational models that predict the proprioceptive information available during natural movement by simulating both proprioceptors and the mechanics of the limbs could help explore which signals are suited to

extract key quantities like heading and velocity.

4.6 Approaches for systems-level modeling of proprioception

We propose that developing multiscale computational models that span proprioceptive sensing and motor control will make it easier to understand how neural circuits flexibly control the body. Below, we outline several possible approaches, including developing tractable models that grapple with proprioceptor diversity, integrating across levels of the sensorimotor hierarchy, and accounting for the roles of sensory delay, noise, and contextual modulation of proprioceptive signals.

4.6.1 Diversity of proprioceptive sensors and their models

A major challenge in modeling proprioceptors is the diversity of sensory neuron subtypes. Unlike sensory systems that are concentrated into specialized organs (e.g. the eye, nose, and ear), proprioceptors are distributed throughout the body. Furthermore, the receptive field of each proprioceptor is idiosyncratic to the tissue in which it is embedded and how that tissue moves through and interacts with the environment. Therefore, a model that describes the input-output function of a muscle spindle in a leg muscle may have little in common with a muscle spindle in a jaw muscle. Existing models may be tuned to match the properties of different proprioceptors (e.g. [292]), but in most cases, additional physiological data is needed. In addition, there remain many proprioceptors for which models do not currently exist. Historically, there has been a strong focus on modeling the activity of mammalian muscle spindles and Golgi tendon organs [159, 172, 218, 220, 32, 31, 117, 160, 192, 193, 240]. In contrast, computational models of invertebrate proprioceptors are only beginning to emerge (e.g. [4, 269]), despite the growing literature characterizing encoding properties of chordotonal organs, campaniform sensilla, slit sensilla, and other proprioceptors [276, 96, 75]. Models of insect proprioceptors would be particularly useful to complement the available genetic tools to experimentally manipulate specific proprioceptive neurons during behavior [276].

Even where precise proprioceptor models are available, many recent neuromechanical

simulations rely on simplified phenomenological proprioceptor models [235, 102, 176, 304, 148, 292]. Whether such simplifications are appropriate may depend on the level of system being modeled, and further experimental and computational analyses are needed to identify constraints on useful approximations. One attractive possibility is that a transformation of coordinates in the input and output variables may help simplify proprioceptor models, as in the case of using force instead of fiber length to model muscle spindle responses [31] or transforming signals between eye, head, or body reference frames to model proprioceptive encoding in the cerebellar nuclei [12]. More generally, a powerful modeling framework would capture diverse proprioceptors with a tractable number of tunable parameters, such as location on body, hysteresis, and maximum firing rate. Of course, simulation parameters and code should be openly shared to facilitate reproducibility and reuse.

4.6.2 Integration of models for proprioception and motor control

A major challenge in understanding the behavioral functions of proprioceptive feedback is the lack of experimental separability between proprioceptive and motor circuits across the sensorimotor hierarchy. An integration of models across levels can provide unique insights into the control of posture, movement, and state estimation that only emerge when considering these systems as a whole.

To study how specific proprioceptive signals contribute to posture control, Prochazka et al. [221] integrated models of muscle spindles and Golgi tendon organs into a feedback-controlled stimulation of arm and leg muscles. The authors found that positive force feedback from Golgi tendon organs, which is generally considered unstable, can actually stabilize a muscle's response to increased load if one also incorporates sensory delays and natural filtering properties of mammalian muscle. The conclusion that positive force feedback is a stable means to control load-bearing motor tasks would not have been clear in studying the controller or proprioceptors alone.

Several recent neuromechanical models of movement control have also simulated the dynamic activity of proprioceptors [102, 176, 234, 304, 148]. For example, Goldsmith et al.

[102] integrated models of proprioceptors into a simulation and robotic model of a walking fruit fly. Using feedback signals inspired by the tuning properties of insect femoral chordotonal neurons and campaniform sensilla enabled the robot to adapt to continuous changes in load. These models could eventually be used to study the effects of conduction delays, noise, and modulation of specific proprioceptive signals on locomotion and skilled limb movements, and to generate testable predictions for perturbation experiments. We hope that emerging methods for neuromechanical modeling will make these models easier to construct and use [61, 268, 24].

Systems-level models can also provide insights into how proprioceptive circuits represent the body and suggest potential roles for feedback in motor control. For example, Hamlet et al. [115] used a multiscale model of lamprey swimming to investigate how different proprioceptive signals, encoding the direction and magnitude of body curvature, may exert different effects on locomotor speed and energy consumption. In another example, Sandbrink et al. [235] recently showed that some proprioceptive representations naturally arise in a deep neural network trained to recognize characters from arm motions given a biologically realistic model of a human arm. Finally, Ache and Dürr [4] modeled the proprioceptive hairs of an insect antenna to predict how downstream neurons encode antennal movements. These studies illustrate how systems-level models provide a useful framework to investigate the role of proprioceptive feedback in guiding natural movements.

4.6.3 Context-dependent modulation

An important characteristic of proprioception is that its effects on motor control are not fixed, but can be tuned by the nervous system under different behavioral contexts. Modeling experimental results with such context-dependent changes requires information about interactions across multiple levels, further motivating systems-level models.

A clear example of such a top-down modulation is the case of reflex reversal, in which a reflex changes sign depending on the behavioral state of the animal. For instance, stimulating neurons in the femoral chordotonal organ in the stick insect leads to either extension or

flexion of the tibia depending on whether the leg is in stance or swing [22]. Although there have been detailed studies of the circuits that mediate reflex reversal [59], this phenomenon is not commonly built into models for sensorimotor control. One notable exception is the recent model by Goldsmith et al. [104], which investigated different perturbations in a neuromechanical model and found that reflex reversal may be due to inhibition of the flexion-tuned position- and velocity-sensitive neurons. In another example, Bacqué-Cazenave et al. [18] built a neuromechanical model of the crayfish leg circuitry which was able to reproduce state-dependent reflex reversal observed during locomotion [57].

Reflex modulation may also play an important role in steering, possibly via descending input which modulates sensory feedback depending on the steering direction. For example, Schilling and Cruse [241] proposed a fully decentralized model of walking control that produces curved walking by adjusting the setpoint of local control loops in each leg. Ultimately, we expect that most behaviors will involve not only a unique set of descending motor commands, but also bespoke tuning of proprioceptive feedback pathways.

4.6.4 Conclusion

Due to advances in genetic targeting of neuronal cell-types, our ability to perform targeted perturbations in proprioceptive and motor circuits is rapidly expanding. As the space of possible experiments grows, there is a pressing need for computational frameworks to help guide experimental design and interpretation. Models developed for proprioceptors and motor control at different scales can be integrated to yield new insights about how proprioception interacts with motor control to support dexterous and flexible movements. These biological principles may also inspire the design of novel robotic systems and contribute to our understanding and treatment of movement disorders.

4.7 Acknowledgments

We are grateful for insightful comments on the manuscript from Joshua L. Proctor, Nicholas Szczecinski, and members of the Tuthill and Brunton laboratories. CJD was supported by

a Research Fellowship from the German Research Foundation (DFG DA 2322/1-1). LK was supported by a National Science Foundation Graduate Research Fellowship. BWB was supported by the Washington Research Foundation and the Air Force Office of Scientific Research grants FA9550-19-1-0386 and FA9550-18-1-0114. JCT is a New York Stem Cell Foundation – Robertson Investigator and was additionally supported by the Searle Scholar Program, the Pew Biomedical Scholar Program, the McKnight Foundation, and National Institute of Health grants R01NS102333 and U19NS104655.

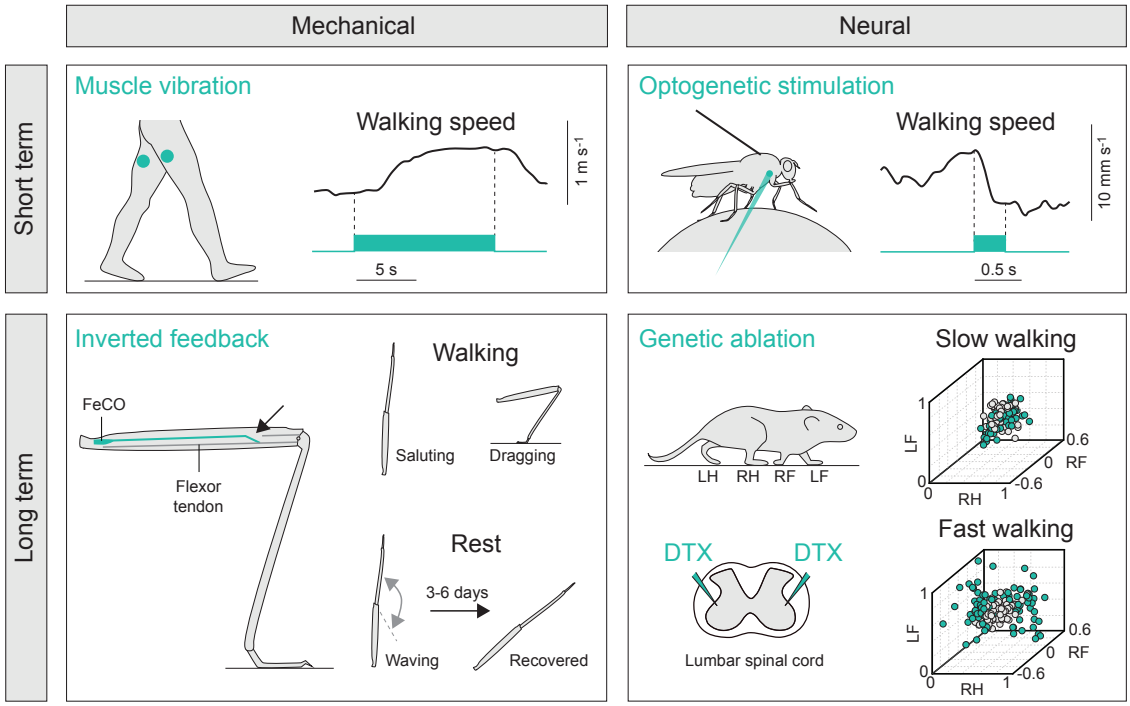

Figure 4.1: Examples of experimental perturbations used to probe the role of proprioceptive feedback in motor control. Top left: Vibration of hamstring muscles in humans stimulated muscle spindles, leading to an increase in walking speed [136]. Bottom left: Inverting sensory feedback from the femoral chordotonal organ of a stick insect front leg by crossing the receptor tendon (arrow) led to saluting or dragging of the leg during walking and waving during rest (front leg in the air, other legs on the ground). The normal rest posture of the front leg (extended in air) recovered after a few days, but saluting and dragging during walking remained unchanged [46]. Top right: Optogenetic stimulation of second-order proprioceptive neurons in tethered *Drosophila* walking on a treadmill caused a decrease in walking speed. Trace shows mean of multiple animals [6]. Bottom right: Genetic ablation of hindlimb muscle spindles in the mouse lumbar spinal cord caused inter-limb coordination deficits only during fast walking. 3D plots show timing (phase) of right hindlimb (RH), right forelimb (RF), and left forelimb (LF) relative to left hindlimb (LH) for gait cycles with (gray) and without (green) muscle spindles [270].

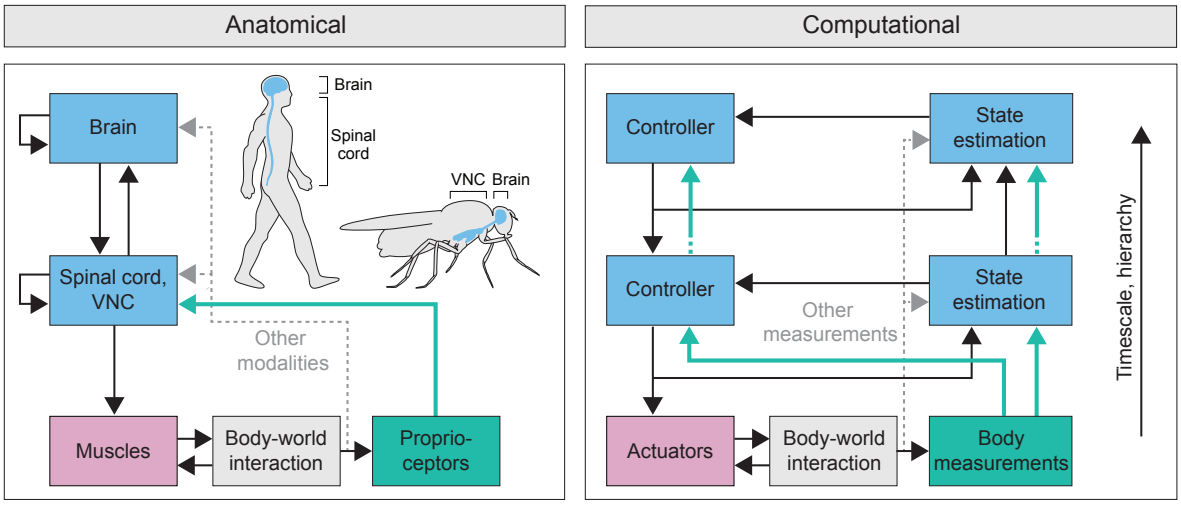

Figure 4.2: Anatomical and computational view of proprioceptive feedback in motor control. Left: Proprioceptors from the limbs project to the spinal cord (in vertebrates) and the ventral nerve cord (VNC, in invertebrates), where they transmit sensory signals to distributed circuits for posture and movement control. Right: Proprioceptive signals affect motor control on different timescales through multiple nested feedback loops. Proprioceptive signals are integrated in low- and high-level controllers for posture, movement, and planning. They may also be used to estimate the state of the body. Note that each box represents a computation, not necessarily an anatomically confined circuit. Computations higher up in the hierarchy are more abstract and operate at longer timescales.

Chapter 5

DYNAMICS OF FRUIT FLY LOCOMOTION ARE AT THE BOUNDARY OF PHYSIOLOGICAL DELAYS

This chapter is a work in progress.

Authors: Lili Karashchuk*, Jing Shuang (Lisa) Li*, Steven Brunton, John C. Tuthill, and Bingni W. Brunton

* Equal contribution

5.1 Abstract

We present a computational model of multi-legged locomotion in *Drosophila* flies. The model contains three functional layers, or modules: a data-driven neural network that generates realistic kinematics for each leg, an optimal controller that enacts the desired kinematics on some physical environment, and an inter-leg coordinator. The optimal controller is designed to account for sensory and motor delays, which critically constrain the animal's ability to respond to unpredictable perturbations. Our model generates realistic walking at a range of forward and turning speeds, achieving simulated joint angles within 6 degrees of recorded trajectories of real flies. Additionally, this layered architecture facilitates generalization — the model maintains walking behavior in the presence of impulse and persistent stochastic perturbations, despite never being trained for such perturbations. Using this model, we gain insight on fundamental physiological limits of locomotor control, characterizing the allowable range of sensory and motor delays that preserve robust fly walking.

5.2 Introduction

Animals rely on robust locomotor control to survive. One of the most common modalities of terrestrial locomotion is legged locomotion, which is employed by diverse species ranging from insects to humans. Legged locomotion requires coordination between many elements: central pattern generators and inter-leg coupling, control of leg poses via muscles, and integration of proprioceptive feedback [123]. The study of legged locomotion aims to produce analyses and models that unite physiology, mechanics, and behaviors; the production of such a model remains an open question in the field.

Insects, particularly flies, are popular organisms to study with regards to legged locomotion [70, 105, 203, 204]. They display robust legged locomotion, and their status as model organisms confer many experimental advantages. By studying and modeling insects, we hope to extract design principles behind legged locomotion and sensorimotor control that apply not only to insects but more broadly to other organisms (including human locomotion) and even legged robots.

Three broad approaches have been taken to model legged locomotion. The first approach models legs as coupled oscillators, where each leg is a single oscillator [64, 223]. The network of oscillators is tuned to recreate oscillatory gait from data, but the models do not include mechanical details of leg joints. The second approach focuses on physical details of the legs and its joints. [242] uses a decentralized reactive controller to recreate hexapod gaits. Further, [103] and [100] introduce a robotic platform, while [161] introduces a virtual simulation in a physical engine. These models are also driven by coupled oscillators with tuned parameters. The resultant walking behaviors, although dynamically valid, deviate substantially from those of real animals' joint kinematics. The third approach is to use a normative learning and optimization approach to generate walking behaviors *de novo* [99, 120]. These models produce walking with varying degrees of realism, but require clever selection of objectives and constraints, and are computationally expensive. Additionally, they do not provide much in the way of neurophysiological insight, as they are fundamentally black box models.

We summarize these key related works in Table 5.1. Generally, existing models are unable to simultaneously incorporate mechanical details and realistic behavior while providing neurophysiological insights—instead, they have focused on one or two of these goals.

Table 5.1: Survey of related works. Symbols: ✓: true, ✗: false, ✓✗: partially true, N/A: not applicable

Related work	Includes mechanics	Reproduces realistic joint kinematics	Offers neural insights
Proctor and Holmes (2018) [223]	✗	N/A	✓
Couzin-Fuchs et al. (2015) [64]	✗	N/A	✓
Goldsmith et al. (2019) [103]	✓	✗	N/A
Goldsmith et al. (2020) [100]	✓	✗	✓
Lobato-Rios et al. (2022) [161]	✓	✗	N/A
Schilling et al. (2013) [243]	✓	N/A	✓
Geijtenbeek et al. (2013) [99]	✓	✓	N/A
Heess et al. (2017) [120]	✓	✓✗	N/A
Karashchuk & Li et al. (this work)	✓✗	✓	✓

In this paper, we present a model with a multi-layer architecture that generates realistic joint kinematics, is robust to untrained perturbations, and offers insights about the constraints on control imposed by neurophysiological sensory and motor delays. This model combines and extends features of many previous studies and models, including coupled oscillators, data-driven neural networks, optimal controllers, and simple link-and-joint leg dynamics. As schematized in Fig. 5.1, our architecture is directly inspired by the hierarchical anatomical organization of the fly neuro-muscular-skeletal system. The leg dynamics are controlled by the optimal control layer, which receives proprioceptive feedback and maintain realistic trajectories while rejecting external perturbations and accommodating sensory and motor

delay. Realistic joint trajectories are generated by a neural network, which is trained on joint angle kinematic data collected from walking flies via Anipose [146]. The top-most layer consists of coupled oscillators, which coordinate walking phases between legs and modulates forward and turning speeds.

The central feature of this model is its modularity; together, the layers of the model achieve more than the sum of its parts. A neural network fit to joint trajectories is physiologically uninformative and does not generalize to physical perturbations, while an optimal controller cannot produce realistic kinematics. However, here we show that combining the two produces realistic kinematics *and* is robust to external perturbations. In other words, the incorporation of the optimal controller allows the data-driven model to generalize to new scenarios as well as serves with an interface with physical dynamics. Additionally, we use a novel controller formulation to incorporate sensory and motor delays, which are key physiological factors in biological locomotion that are typically difficult to incorporate into standard models without sacrificing performance and realism. We use this model to explore the effect of a range of sensory and motor delays on walking, especially the ability to maintain walking gait under large perturbations. Our approach to model joint kinematics of walking for two- to many-legged animals is a generalizable framework that can be widely applied to connect the neurophysiology and mechanics of locomotion, with broad impact for sensorimotor neuroscience, behavior, and robotics.

5.3 Results

We propose an end-to-end model of *Drosophila* walking, from single leg joint dynamics to inter-leg coordination. The efficacy of the model is demonstrated by comparing model-generated simulations with real data.

5.3.1 Layered architecture facilitates realistic walking with dynamics and physiological delays

The model employs three functional layers, which interface with a dynamics model, shown in Fig. 5.1. The three layers are the optimal controller, the trajectory generator, and the phase

coordinator. Each individual leg is governed by its own dynamics, optimal controller, and trajectory generator, while inter-leg coordination is accomplished by the phase coordinator.

The use of this layered architecture allows us to unify different aspects of walking; individual leg dynamics, individual leg kinematics, and inter-leg coupling. Each layer provides an abstraction for the layer above it, such that the different aspects of walking can be modularly integrated.

Dynamics and control

Leg dynamics for each leg are derived from link-and-joint models. The dynamics equations relate the state (angle and angular velocity of each joint) of the leg model with the motor output (muscle-generated torque on each joint). We linearize the dynamics equations, and use a linear quadratic regulator as the controller; this controller senses the state of the leg via proprioceptive input, then determines the optimal motor output for walking. This controller is also formulated to include sensoryand motor delay. Detailed formulations are described in Methods and Materials.

At regular intervals, the controller receives a time series of desired state trajectories from the trajectory generator layer. The controller then produces the necessary torques to track (i.e. recreate) this trajectory. External perturbations, when present, enter through the dynamics and affect the states; the controller then senses the state and reacts to them accordingly.

We remark that the walking and perturbation-rejecting capabilities of the overall model are not contingent upon any specific dynamics or controller formulation; any controller that adequately tracks the trajectory generator will suffice.

Trajectory generator

The trajectory generator is a per-leg neural network, trained on biological data. It takes the current leg phase, joint angles, and joint angular velocities as input, and outputs the leg phase, joint angles, and angular velocities for the next time step. It can be used to generate

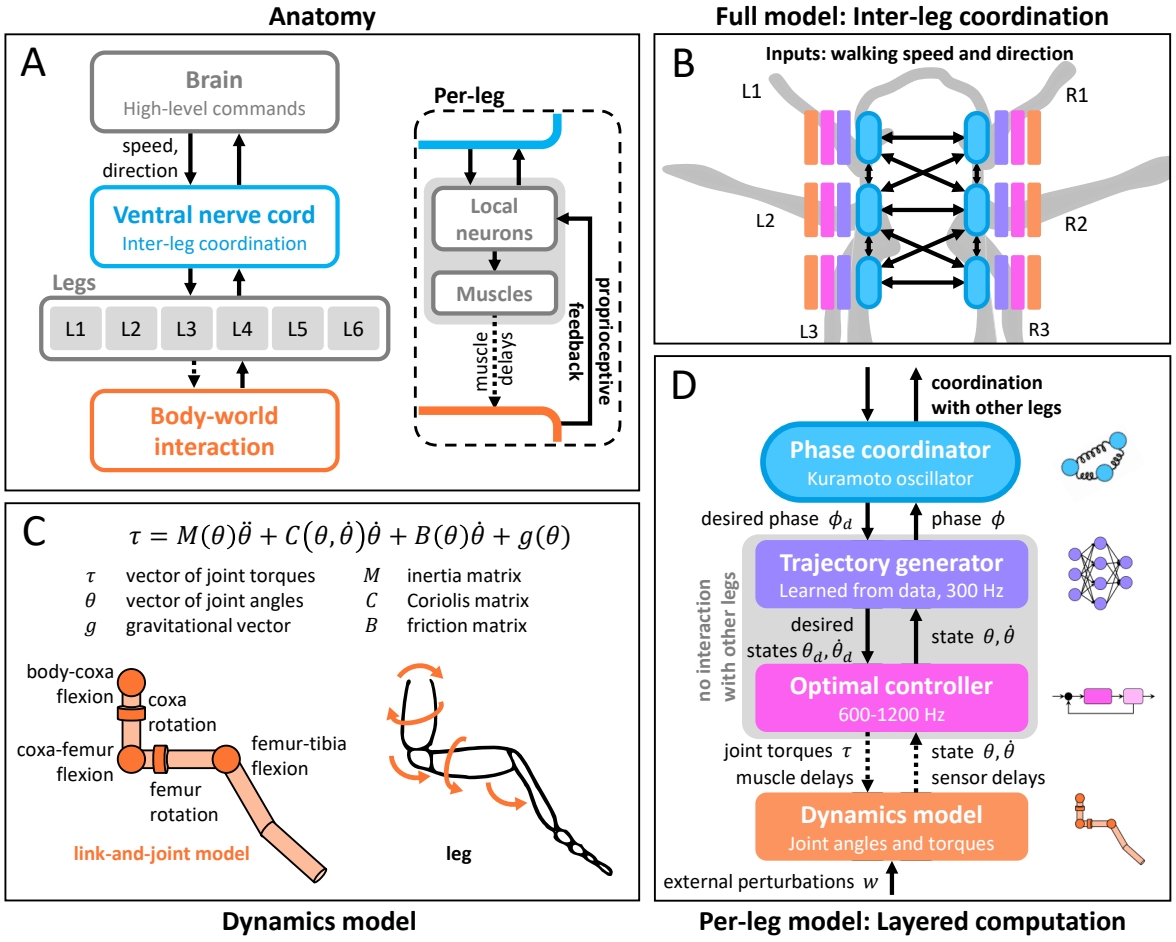

Figure 5.1: Summary of layered locomotion model and relation to anatomy. **(A)** Anatomy involved in walking. The brain sends high-level commands (e.g. walking speed and direction) to the ventral nerve cord (VNC), which coordinates across legs. Each leg's neurons and muscles take inputs from the VNC and acts on the environment; this body-world interaction and its effect for leg joints, joint angular velocities, torques, etc. are reported back to local circuits on the leg via proprioceptive feedback. **(B)** Full model: per-leg models are coupled through their phase coordinators (blue ovals). **(C)** Per-leg dynamics model, derived from link-and-joint models and Euler-Lagrange equations. **(D)** Layered per-leg model. Body-world interactions are modeled through a dynamics model. We assume proprioceptive feedback provides information on joint angles and angular velocities. Each leg contains an optimal controller which interfaces with dynamics, and a trajectory generator which generates realistic gaits. The trajectory generator interfaces with the phase coordinator, a Kuramoto oscillator which induces inter-leg coupling. The trajectory generator and optimal controller mimic local circuits and do not interact with other legs.

a time series of phase, joint angles, and angular velocities for each leg. Details on training and network properties are described in Methods and Materials.

The trajectory generator periodically receives proprioceptive information from the controller on the true state (joint angles and angular velocities) of the leg. It also receives information on the desired phase from the phase coordinator. Using this information, it generates the desired angle and angular velocity trajectories for some future time interval, and sends this back to the controller; it also generates phase values, which are sent to the phase coordinator.

In the absence of external perturbations, the trajectory generator generates realistic walking; it is the key to the overall model's biologically accurate kinematics. When external perturbations are present, their effects are largely mitigated by the controller layer; the trajectory generator itself generally receives low magnitudes of perturbation, and maintains relatively accurate kinematics. We remark that all data used to train the trajectory generator come from experimental conditions with no external perturbations.

Phase coordinator

The phase coordinator ensures realistic inter-leg coordination. We used a Kuramoto model [3, 261] to describe this coordination process. The model takes a set of phases across all legs and adjusts them so that they are synchronized, accounting for the target offsets between pairs of legs. The model couples the legs where the trajectory generators on their own would lead to uncoupled dynamics. See Methods and Materials for equations and implementation details.

5.3.2 Realistic model-generated walking

Various metrics and visualizations of simulated vs. real walking are shown in Fig. 5.2; these metrics are explained in the following subsections. All simulated trajectories were produced with sensory delay of 10ms and motor delay of 30ms, which is consistent with experimental values [277, 15].

Example time series and videos

We include example time series of simulated walking vs. real walking, as plots of angles and angular velocities versus time for specific joints on a specific leg. This is shown in Fig. 5.2, panels A and C. Simulated values resemble real values; real values are comparatively noisier, but the general pseudo-triangular shape, as well as amplitude and frequency of angular oscillations, are similar. We also include videos of simulated walking, side-by-side with real walking. The purpose of these visualizations is to demonstrate *qualitative* rather than quantitative resemblance between simulated and real walking.

Angles and angular velocities versus phase

We compare simulated walking and real walking data by comparing the angles and angular velocities of the two. The naive approach is to compare between time series — this can be misleading, since even in data, the time series for two flies walking in the same direction at the same speed may differ significantly due to phase misalignment. The appropriate comparison is to compare angles and angular velocities as functions of phase. Here, we make the distinction between *generated phase*, the per-leg phases produced by the phase coordinator of the model, and *computed phase*, which can be computed for each joint from time series data. For real walking data, we do not have access to the generated phase; thus, comparisons must be made between computed phases.

In panels B and D of Fig. 5.2, we plot the angle and angular velocity vs. phase for representative joints (e.g. femur-tibia flexion) for real vs. simulated walking over 4 bouts[1], at varying forward-walking speeds (8, 10, 12, and 14 mm/s), with no turning or side-stepping. The real and simulated data are highly similar.

The aggregate differences between simulated and real data over a range of forward-walking, backward-walking, turning, and side-stepping speeds over 500 bouts are shown in panel E. The dotted line indicates the uncertainty (5.9 degrees) associated with data collection [146]

[1]In the data, for a given forward-walking speed, typically only 4-10 distinct bouts are available

— we see that the average difference between the model and data is comparable to data collection uncertainty. We note that the errors for angular velocity are several orders of magnitude larger; this is partially an artefact of the high sampling rate (300 Hz) used in data acquisition.

Phase coupling within and across legs

We are also interested in the difference between computed phases of joints within a leg and across legs, shown in panels A and B of Fig. 5.3, respectively. As with Figure 2E, the data for these figures comes from over 500 bouts of walking with various values of forward and backward walking, turning, and side-stepping.

For phase coupling within a leg, shown in panel A, we choose a representative joint for the leg (denoted "target" on the image), and observe the coupling between this joint and other joints on the leg. Peaks in coupling indicate synchronization; a single peak at zero on the horizontal axis indicates that the two joint phases are coupled to match; a single peak elsewhere indicates that the two joint phases are coupled with some offset. A lack of peaks indicates that the two joints are very weakly or not coupled. We see that in the real data, joints exhibit a mixture of strong (e.g. L3 coxa-femur flexion to femur-tibia flexion) and weak (e.g. R2 femur-tibia flexion to femur rotation) flexion coupling. All synchronizations (i.e. peaks) in data are captured by the model; however, the model generally exhibits stronger synchronization (i.e. higher peaks) than data. In some cases (e.g. L3 femur rotation), the model exhibits synchronizations that are not present in the data. In the model, the trajectory generator is responsible for capturing the per-leg inter-joint coupling; these results suggest that the neural network learns stronger coupling values than are present in nature. We also note that synchronizations in the data are, interestingly, not fully symmetric across legs — for instance, compare L2 and R2 coxa-femur flexion.

For phase coupling across legs, shown in panel B, we represent each leg with the phase of its representative joint, and compare the phases across legs. Once again, peaks indicate strong coupling, while a lack of peaks indicates a lack of correlation. Coupling properties of the

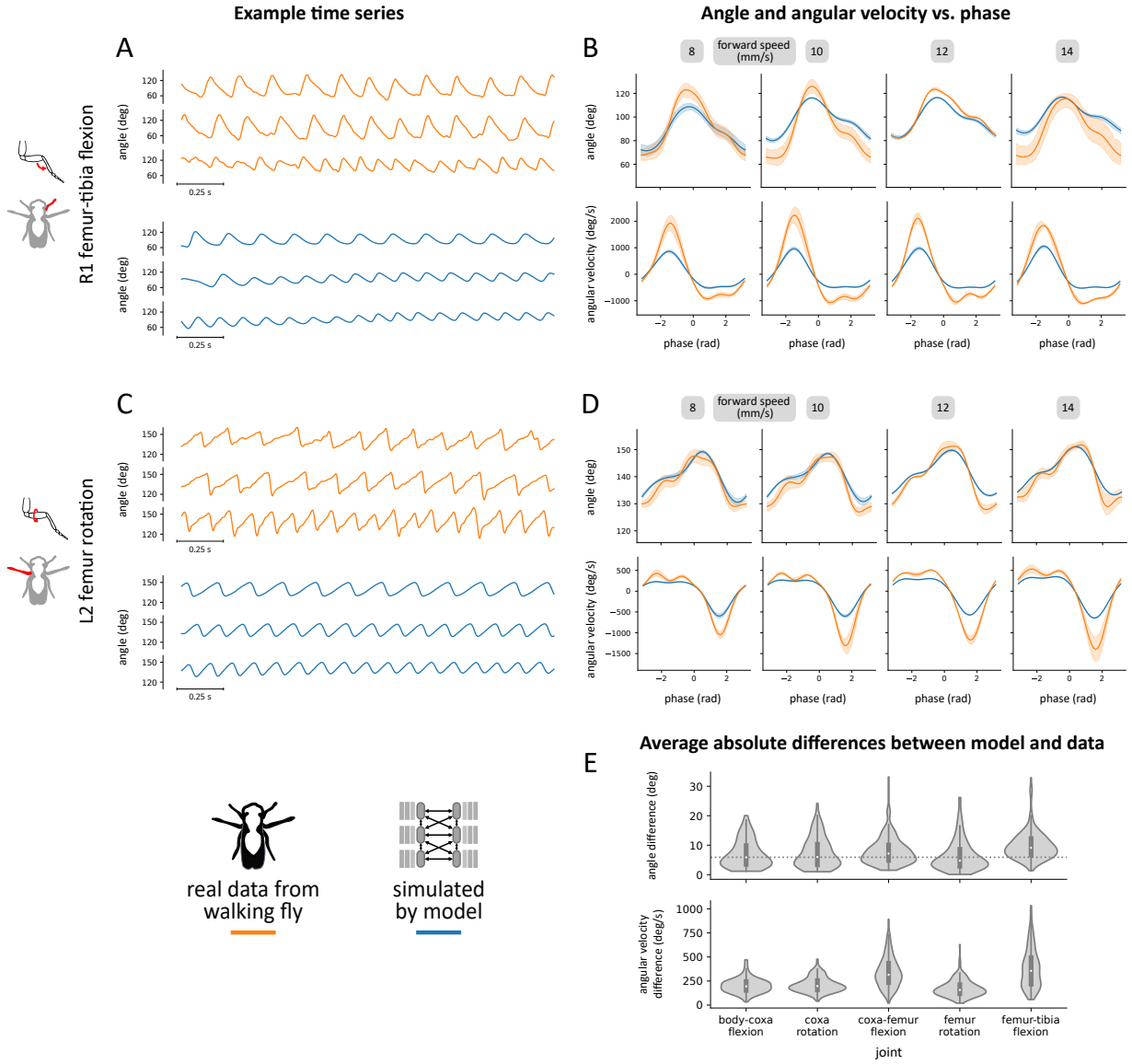

Figure 5.2: Comparison of walking behavior generated by the model (blue) vs. walking behavior recorded from real flies (orange). **(A,C)** Example time-series of femur-tibia flexion on leg R1 and femur rotation on leg L2 for three different walking speeds: 8, 10, and 12 mm/s. Real data exhibits slightly more variability than model simulations. **(B, D)** Angle and angular velocity vs. computed per-leg phase of femur-tibia flexion on leg R1 and femur rotation on leg L2 for four different walking speeds. **(E)** Average differences between model simulations and data. The dotted line indicates average in-sample differences between different bouts of real walking — we see that the average difference between the model and data is comparable to the in-sample differences in the data itself.

model are qualitatively similar to coupling properties of the data, with moderate variations in phase difference (i.e. peak location) and synchronization strength (i.e. peak height). In the model, the phase coordinator is responsible for representing inter-leg coupling; these results suggest that the underlying Kuramoto oscillator model is indeed able to capture naturalistic coupling patterns.

5.3.3 Realistic responses to external perturbations

We simulate the model with external dynamic perturbations, and assess the realism of the resulting walking. Two types of perturbations are considered: impulse perturbations and persistent stochastic perturbations. For impulse perturbations, an instantaneous velocity was added to all joints at a single point in time (corresponding to some instantaneous force and acceleration). The perturbation values are random and normally distributed — we use *perturbation strength* (rad/s) to represent the mean of the distribution. Impulse perturbations approximate real-life situations in which the animal encounters a brief, unexpected force (e.g. slipping on the walking surface). For persistent stochastic perturbations, we applied impulse perturbations with smaller perturbation strengths applied stochastically following a Poisson distribution (with rate 10Hz) to all joints. These perturbations approximate real-life collisions with slippery surfaces or wind-related forces. Ranges of perturbation strengths included in simulation correspond to biologically plausible values — calculations are included in the Appendix.

We show time series data in Fig. 5.5, and observe that joint kinematics appear different, but still generally oscillatory during perturbations. After perturbations cease, joint kinematics recover to pre-perturbation patterns. Thus, the model is able to maintain walking-like behavior in the presence of perturbations, and recovers after perturbations end.

Since no experimental data with external perturbations are available, our previous quantitative metrics (which rely on per-bout comparison with experimental data) are not applicable here — it is necessary to introduce a new metric. For a given bout of walking, we compute the *likelihood* of it occurring in nature (i.e. data). The process to compute this is shown in

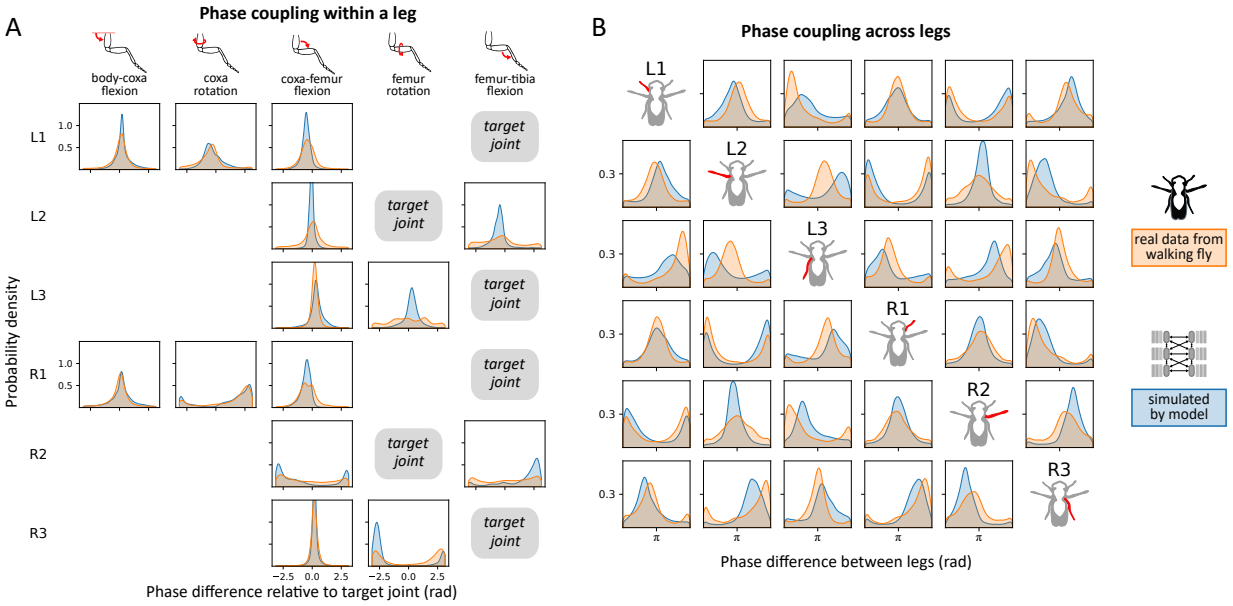

Figure 5.3: Phase coupling within and between legs. **(A)** Phase coupling within each leg. For each leg, we compare phases between a representative joint for the leg (denoted "target" on the image) and other joints on the leg. Gaps are present as we did not include all 5 joints for all legs in the model. Peaks in coupling indicate synchronization. All synchronizations (i.e. peaks) in data are captured by the model; however, the model generally exhibits stronger synchronization (i.e. higher peaks). **(B)** Phase coupling across legs. We compare phases of representative joints across legs. Model coupling qualitatively resembles data coupling, with moderate variations in phase difference (i.e. peak location) and synchronization strength (i.e. peak height).

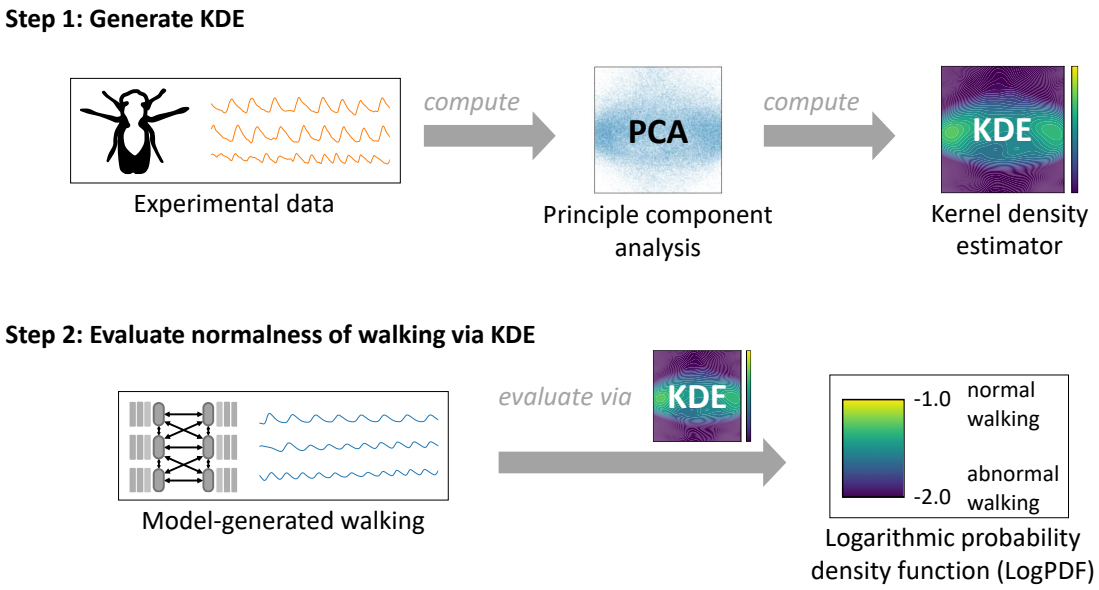

Figure 5.4: Process diagram of quantifying normality of model-generated walking. First, we compute a principle component analysis on the full set of experimental data from the flies, then use this to compute a Gaussian kernel density estimator (KDE). To quantify the normality of a given bout of walking, we apply to KDE to the bout to evaluate its log probability density function (LogPDF), which is a scalar metric corresponding to the normality of walking. High values of LogPDF indicate normal walking, while low values indicate abnormal walking.

Fig. 5.4. First, a principle component analysis is computed on the set of experimental data. A Gaussian kernel density estimator (KDE) is fitted to this data. This KDE takes a bout as input, and gives the log probability density function (LogPDF) as an output. This is a scalar value corresponding to likelihood.

5.3.4 Effect of sensory and motor delay on walking

The inclusion of delays in the model allows us to explore varying sensoryand motor delays and how this affects walking. For all previous results, we used nominal values of 10ms of

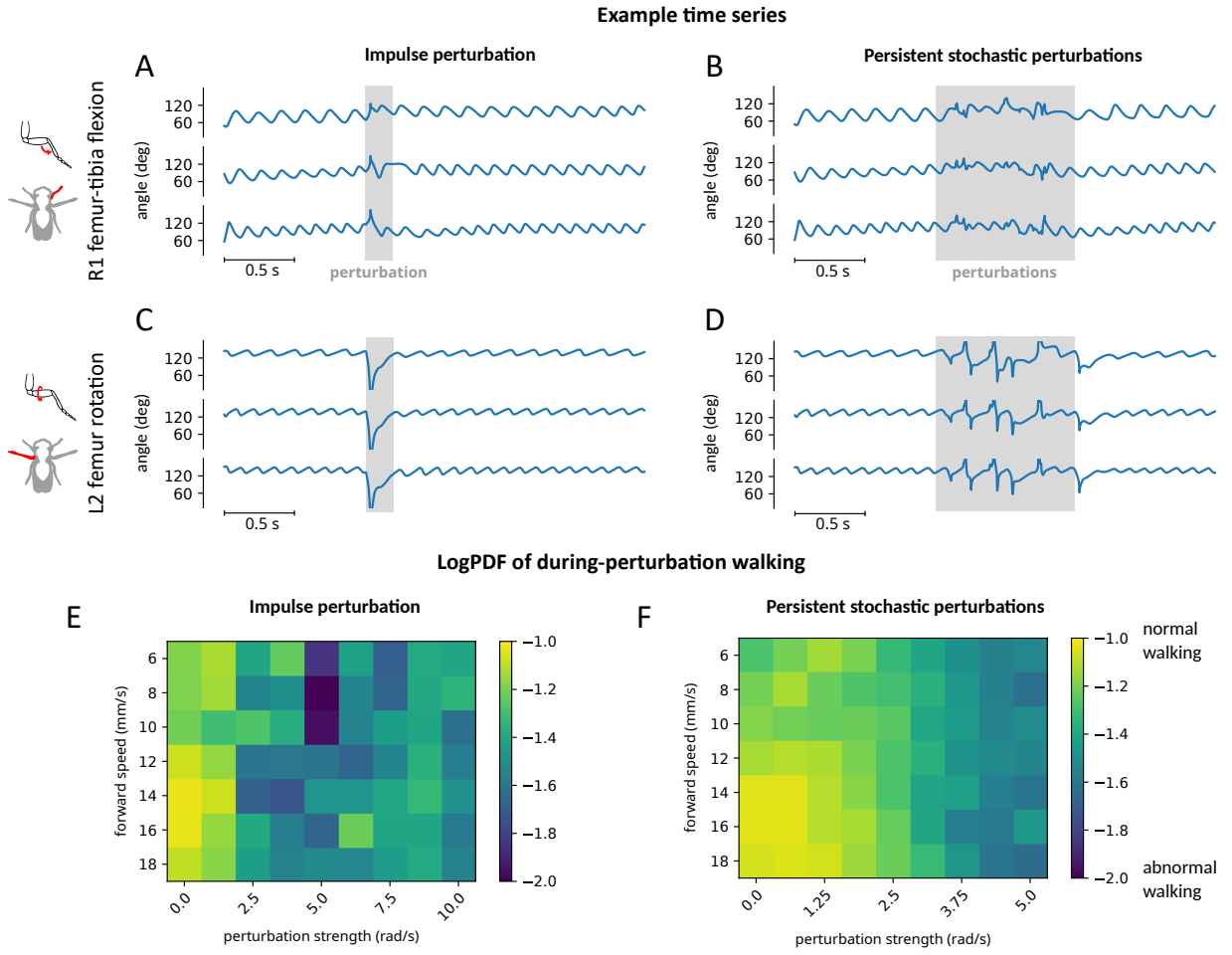

Figure 5.5: Model-generated walking under impulse and persistent stochastic perturbations. **(A, B, C, D)** Example time-series of femur-tibia flexion on leg R1 and femur rotation on leg L2 for three different walking speeds (8, 10, and 12 mm/s) before, during, and after perturbation. Angle trajectories are visibly different during perturbation (shaded grey area), but recover after perturbation. Perturbation effects appear similar across speeds. Perturbations strengths shown are 4.75. **(E, F)** LogPDF values of during-perturbation walking for impulse and persistent stochastic perturbations. For each square of the heatmap, four simulations with different initial condition were simulated and evaluated. Abnormality of walking appears to increase slightly with increased perturbation strength and decreased walking speed.

sensory delay and 30ms of motor delay. Now, we explore how variations in delay values affect walking behavior.

Without external perturbations, the model is able to produce realistic walking with arbitrary delays — the model can use the trajectory generator to predict into the future and effectively compensate for delay effects. However, the assumption of perfectly predictable, perturbation-free walking is unrealistic; thus, we consider the joint effect of persistent stochastic perturbations and delays, and visualize the results in Fig. 5.6 (similar results are presented for impulse perturbations in the Appendix). For a given perturbation strength, the effect is much more pronounced for higher sensory and motor delays — but even for large perturbations and delays, the model mostly recovers after perturbations end. We characterize the abnormality of during-perturbation walking for various motor delays (with fixed sensory delay of 10ms) and various sensory delays (with fixed motor delay of 30ms) in panels E and F. Across a range of perturbation strengths and forward speeds of walking, the model maintains normal walking up to about 25-30ms of motor delay, and 4-6ms of sensory delay.

5.4 Discussion

5.4.1 Biological insights

Layered model produces robust walking and facilitates local control

One key finding of this model is that robust walking in the presence of external perturbations can organically result from a reflex-like controller layer in combination with a trajectory generation layer that is pre-programmed for perturbation-free conditions (i.e. trained on data from perturbation-free walking). The model maintains robust walking in the presence of perturbations without requiring re-training. The key enabler of robustness here is the reflex-like controller layer, which rejects perturbations while maintaining a walking gait. In nature, many external perturbations are encountered during legged locomotion; this layered model suggests how robust locomotion may be maintained using a reflex layer, requiring minimal adaptation from non-reflex neurons.

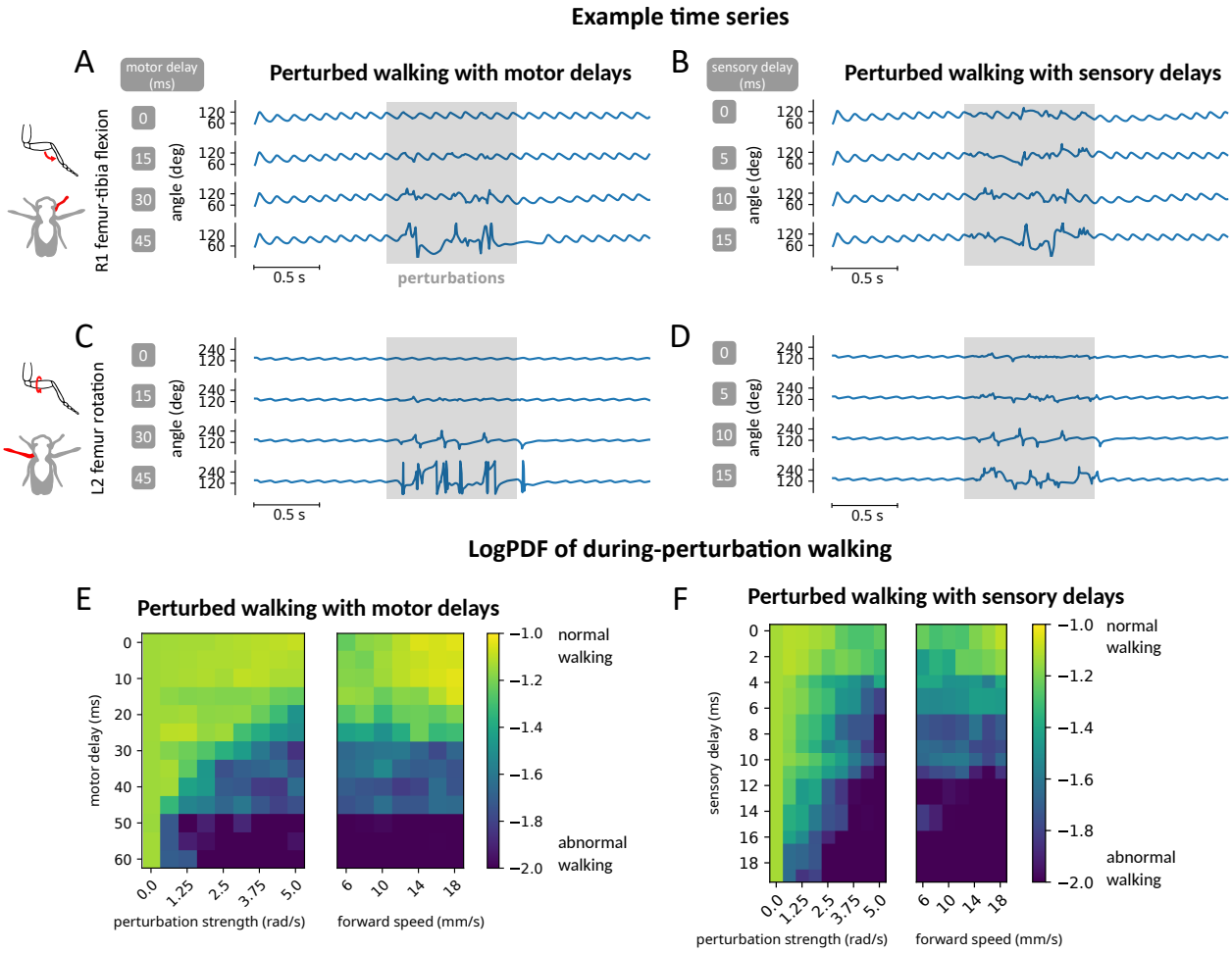

Figure 5.6: Model-generated walking under persistent stochastic perturbations and various motor and sensory delay values. Values for motor and sensory delay are 30ms and 10ms unless otherwise stated. **(A, B, C, D)** Example time-series of femur-tibia flexion on leg R1 and femur rotation on leg L2 for a walking speed of 12 mm/s, under various values of motor (0, 15, 30, 45ms) and sensory delay (0, 5, 10, 15ms). Perturbation effects are more noticeable with increased delay values, and appear to be more sensitive to sensory delay. All time-series shown correspond to the maximum perturbation strength (5.0). **(E, F)** LogPDF values of during-perturbation walking for various values of delay, perturbation strength, and forward speeds. For each square of the heatmap, four simulations with different initial condition were simulated and evaluated. Abnormality of walking appears to increase slightly with increased perturbation strength and decreased walking speed, though it increases mostly strongly with increased delays. The model maintains normal walking (LogPDF $>$ -1.6) up to about 25-30ms of motor delay, and 4-6ms of sensory delay across a range of conditions.

The separation of functions between layers in this model also reinforce that locomotion can be produced by mostly local (i.e. per-leg) signals. In this model, the only signals that need to be communicated between legs are signals about per-leg phases; no information about individual joints needs to be included.

Fundamental constraints on sensory and motor delay

This model includes sensory and motor delays. Sensory and motor delays are ubiquitous in animal locomotion due to the biophysics of muscles and neurons. Their values are governed by two opposing design principles. Firstly, from an energy perspective, it is more expensive to manufacture and maintain muscles and neurons with low delay [258] — thus, higher delays are preferable. In contrast, from a performance perspective, delays result in performance degradation and slower reactions, which may impede survival [197]. As seen in our simulations, increased delays lead to worse walking — thus, lower delays are preferable. Our work reconciles these two design principles by showing that evolution selects the maximum value of delay that will still preserve performance (i.e. walking). Across a range of perturbation strengths and forward speeds of walking, our model maintains normal walking for up to about 25-30ms of motor delay, and 4-6ms of sensory delay. These values are remarkably close to known values in *Drosophila* physiology: 20-30ms of motor delay [15] and 6-9ms of sensory delay [277].

Motor delay necessitates compensatory prediction

This model uses a novel controller formulation from to include sensory and motor delays using Linear Quadratic Gaussian techniques. During model development, we discovered that for the task of tracking a reference trajectory, prediction is necessary for good model performance. In all simulations, we allow the controller to use a prediction horizon that matches the motor delay. For instance, if the motor delay is 10 timesteps, then the controller has access to the planned trajectory (from the trajectory generator) up to 10 timesteps into the future. A motor delay of 10 timesteps means that the current planned motor action will not take effect until 10 timesteps in the future, so intuitively it makes sense that we should

know the planned trajectory at this point in the future. We also experimented with altering the prediction horizon to be less than the motor delay, with catastrophic consequences — the model mostly produced only noise. Overall, the model suggests that future predictions are crucial in compensating for motor delays, an idea also proposed in [215] and [197].

5.4.2 Broader impacts

General framework for models of animal locomotion

The general framework of a three-layer model for locomotion is applicable to any multi-limbed organism for any modality (e.g. flying, swimming). The key ingredients required for this model are: (1) a functional inter-limb coordinator, (2) sufficient data to train a trajectory-generator layer, and (3) a controller that adequately tracks the trajectory for some dynamical model of the organism. The dynamical model may be linearized (as is done in this paper) and controlled with a standard controller, or nonlinear techniques such as feedback linearization may be employed. Overall, this framework allows scientists and engineers to build upon existing models of inter-limb coupling and take advantage of various emerging datasets on animal locomotion to create more fully integrated models of locomotion for various organisms.

Studying physiological delays

Using a novel controller formulation, this model allows us to simulate and study physiological sensory and motor delays — to the best of our knowledge, it is the first work to investigate and estimate allowable delay bounds. Signaling delays are ubiquitous in biological systems, and incorporating realistic physiology will inevitably lead to the incorporation of various signaling delays. This work provides a model that incorporates delay without sacrificing behavioral realism, and lays the groundwork for future studies on the effect of delay on various aspects of locomotion such as inter-leg coordination, biomechanical design, etc.

Generative model for bio-mimetic gaits

This model has potential applications for bio-mimetic robotic locomotion. In the interests of parsimony, the current model includes a fairly basic dynamical model consisting of linearized link and joints; however, we could also replace this dynamical model with that of a hexapod robot, and recompute a controller accordingly. Due to the modular nature of the model, other modules (trajectory generator, inter-leg coordinator) could remain unchanged — and the resulting code would, in theory, generate bio-mimetic gaits for some robot. Overall, the model can be thought of as performing a layered implementation of imitation learning, which is a popular technique in robotics.

5.4.3 Limitations

The focus of this model is to produce realistic behavior with reasonable physiological considerations, i.e. link-and-joint dynamics, sensory and motor delays. In order to do so, we have made a number of physiological simplifications. Though quantitative insights are made in the study, numerical values may be sensitive to these simplifications. In particular, ground contact interactions, muscle models, and proprioceptor models are not explicitly included in the dynamics, though they are implicitly taken into account by the trajectories learned by the neural network. We defer the full inclusion of these elements to future work. We anticipate that the inclusion of proprioceptor models will make the system slightly more difficult to control and therefore lower the allowable values of delay; while the inclusion of muscle models and other biomechanical details will make the system slightly easier to control and therefore increase the allowable values of delay. The reasoning for the latter is that biomechanics (for instance, compliance in the tarsus) are typically somewhat optimized for locomotion.

Another key simplification made by this model is the omission of dynamical coupling. Currently, the legs are only coupled neurally, through the phase coordinator — however, in real life, the legs are also dynamically coupled through the body and its weight distribution upon the legs. This is also a feature we plan to incorporate in future work, as we move toward

a more biomechanically realistic model. One avenue of planned investigation is integration with a physics-based model [161]. We anticipate that the inclusion of dynamical coupling may require some additional coordination between the legs. Additionally, the current controller uses a position-based method, while the inclusion of dynamical coupling will likely necessitate a switch to impedance-based methods.

5.5 Methods and Materials

5.5.1 Dynamics and controller formulation

All techniques used for dynamics formulation are standard tools from undergraduate-level controls theory. We begin with a link-and-joint model of the fly leg, as depicted in Fig. 5.1. For simplicity, we only model joints that are crucial to natural leg movements during walking and turning. For instance, varying femur rotation is important to the movements of the middle legs, but the front legs exhibit near-constant femur rotation; thus, a femur rotation joint is included for the middle and hind legs only. The joints included for each leg is shown in Table 5.2.

Table 5.2: Joints included for leg models

Joint	Front legs	Middle legs	Hind legs
Body-coxa flexion	✓		
Coxa rotation	✓		
Coxa-femur flexion	✓	✓	✓
Femur rotation		✓	✓
Femur-tibia flexion	✓	✓	✓

We write the Denavit-Hartenberg (DH) table of the leg model, and use this to systematically

derive the Euler-Langrange matrix equations of motion:

$$\tau = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + B(\theta)\dot{\theta} + g(\theta) \tag{5.1}$$

where τ is the vector of joint torques; θ, $\dot{\theta}$, and $\ddot{\theta}$ are vectors of joint angles, angular velocity, and angular acceleration; M, C, B, are the inertia, Coriolis, and friction matrices, and g is the gravity vector.

We then define state $q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}$ and input τ, and rearrange (5.1) into the form $\dot{q} = F(q, \tau)$, i.e.

$$\begin{bmatrix} \dot{q_1} \\ \dot{q_2} \end{bmatrix} = \begin{bmatrix} q_2 \\ -M(q_1)^{-1}\left(C(q_1, q_2)q_2 + B(q_1)q_2 + g(q_1)\right) \end{bmatrix} + \begin{bmatrix} 0 \\ M(q_1)^{-1} \end{bmatrix} \tau \tag{5.2}$$

We then choose equilibrium values $\bar{q}$ and $\bar{\tau}$, such that $F(\bar{q}, \bar{\tau}) = 0$. $\bar{q_1}$ is chosen based on the average joint angles for each leg, from the data. Then, $\bar{q_2} = 0$ and $\bar{\tau} = g(\bar{q_1})$ gives the desired equilibrium. We linearized about this equilibrium point, which leads to the following equations:

$$\dot{x} = A_c x + B_c u \tag{5.3}$$

where A and B are the Jacobians with respect to q and τ, respectively, i.e. $A = \frac{\partial F}{\partial q}(\bar{q}, \bar{\tau})$, $B = \frac{\partial F}{\partial \tau}(\bar{q}, \bar{\tau})$; and $x := q - \bar{q}$ and $u := \tau - \bar{\tau}$. In our code, we use the SymPyBotics toolbox [254] to obtain symbolic equations for the quantities in (5.1), then numerically compute Jacobian values.

Next, we discretize the system using some sampling interval T, which is typically chosen to be an integer multiple of sampling interval from the data ($T = 1/300$). In our simulations, we use $T = 1/600$. The discretized dynamics are written as:

$$x(t + T) = Ax(t) + Bu(t) \tag{5.4}$$

where $A = I + A_c T$ and $B = B_c T$.

Finally, we perform a coordinate shift to error dynamics. This allows us to apply standard control techniques for trajectory tracking. Define tracking error $y = q - q_d$. This error obeys

the following dynamics:

$$y(t+T) = Ay(t) + Bu(t) + w(t) + w_{traj}(t) \tag{5.5a}$$

$$w_{traj}(t) = A(q_d(t) - \bar{q}) + \bar{q} - q_d(t+1) \tag{5.5b}$$

where w is external perturbation, and w_{traj} represents the effect of constantly changing trajectories. Error y is of size n_y (8 for front legs, and 6 for the other legs), and input u is of size $n_u := n_y/2$.

To include motor delay (or *actuation* delay, as is standard in controls literature) and sensory delay, we make use of augmented state formulations as introduced in [257]. Let the motor delay be d_{motor} steps, and let the sensory delay be d_{sense} steps. Our augmented state vector $z(t)$ is written as follows:

$$z(t) = \begin{bmatrix} y(t) \\ f(t) \\ a(t) \\ s(t) \\ g(t) \end{bmatrix} \tag{5.6}$$

where y is the error from (5.5); f is size $d_{motor} * n_y$, containing predicted future errors up to d_{motor} time steps in the future; a is size $d_{motor} * n_u$, containing motor signals from up to d_{motor} steps ago; s is size $d_{sense} * n_y$, containing sensing signals from up to d_{sense} steps ago; and g is size $d_{motor} * n_y$, containing information about future trajectory effects w_{traj} up to d_{motor} time steps in the future.

We write the overall system in the form of

$$z(t+1) = Fz(t) + Gu(t) + w_{aug}(t) \tag{5.7a}$$

$$r(t) = Hz(t) \tag{5.7b}$$

where F, G, and H are formulated using A and B according to techniques in [257], and $w_{aug}(t)$ contains the perturbations from (5.5), appropriately rearranged and zero-padded.

Note that u represents the motor signal, which must be delayed for some amount of timesteps (via a) before affecting the system; similarly, the sensory signal is delayed for several timesteps (via s) before reaching the controller via r.

To achieve effective trajectory tracking, we seek a control law under which y remains small. This can be achieved using the Linear Quadratic Gaussian controller, a standard technique. The controller is governed by the following equations:

$$\hat{z}(t+1) = F\hat{z}(t) + Gu(t) + L(r(t) - H\hat{z}(t)) \tag{5.8a}$$

$$u(t) = K\hat{z}(t) \tag{5.8b}$$

where $\hat{z}$ is the estimate of the state, estimated via a steady-state Kalman filter. L and K are the optimal observer and controller matrices, respectively, synthesized via discrete algebraic Ricatti equations.

5.5.2 D. melanogaster walking data

We obtained *D. melanogaster* walking data following the procedure outlined in [146]. Briefly, a fly was tethered to a tungsten wire and positioned on a frictionless spherical treadmill ball suspended on compressed air. Six cameras captured the movement of all of the fly's legs. Using Anipose, we tracked 30 keypoints on the fly, which are the following 5 points on each of the 6 legs: body-coxa, coxa-femur, femur-tibia, and tibia-tarsus joints, as well as the tip of the tarsus.

We used the behavior classifier described in [146] to subset the data to bouts when the fly is walking. We further restricted our analysis to bouts at least 0.5 seconds (150 video frames) in length, and where the femur-tibia flexion angle of the left front leg had a range of at least 30 degrees.

5.5.3 Trajectory generator

Model

We formulate the trajectory generator as the function

$$(\ddot{\theta}, \dot{p}) = F(\theta, \dot{\theta}, v, p)$$

where θ is a vector of joint angles, v is a vector of desired speeds, and p is the phase of the leg. Note that we model trajectory of each leg by a different function F. To compute a trajectory given an input v and an initial p, θ, and $\dot{\theta}$, we integrate the function F numerically using the midpoint method [163]. Following [122, 305], we represent the function F as a multilayer perceptron network with 2 hidden layers, with 512 units each. We used ELU [60] as our nonlinearity. In total, the multilayer perceptron had 274,437 parameters for T1 legs and 272,388 parameters for T2 and T3 legs, with the slight different in parameters due to the different number of joint angles (θ) modeled for a given leg.

Data

To train the multilayer perceptron network used to represent F, we used the walking data as described in the section above for our ground truth data for θ, $\dot{t}heta$, $\ddot{\theta}$, and v. We computed the phase p using a Hilbert transform over the femur-tibia flexion angle for T1 legs, femur rotation angle for T2 legs, and coxa-femur angle for T3 legs. In order to get at the oscillation, we filter each angle using a first-order Butterworth bandpass filter with 3 Hz and 60 Hz as critical frequencies, using the scipy library [283].

Training procedure

For the training, we followed a two step procedure. In the first step, we minimized the error of F for predicting $(\ddot{\theta}, \dot{p})$ over one time step, given the corresponding $(\theta, \dot{\theta}, v, p)$ from the training data. We minimized the mean squared error of the prediction, normalized by the variance for each dimension. We trained our network for 300 iterations over the full training

data using a batch size of 2500 training samples, using gradient descent with the Adam algorithm [149]. To ensure a robust function at this step, we apply dropout to a random 5% of the hidden units [256]. We standardized the input and output training data to the multilayer perceptron so that has a mean of 0 and standard deviation of 1.

In the second step, we minimze the error of F for predicting a trajectory θ when numerically integrated. Specifically, we integrate F over 60 steps given initial conditions $(\theta, \dot{\theta}, v, p)$ to produce an estimated trajectory of $\hat{\theta}(t)$. We minimize the loss:

$$\sum_t ||\cos(\hat{\theta}(t)) - \cos(\theta(t))||_2^2 + ||\sin(\hat{\theta}(t)) - \sin(\theta(t))||_2^2$$

using gradient descent with the Adam alogrithm [149]. During training, we clip gradients to a norm of 10 to stabilize training.

We implemented the training using Tensorflow [1] running on a computer with NVIDIA GeForce RTX 2070 GPU and AMD Ryzen Threadripper 1920X 12-Core Processor.

Chapter 6

FUTURE DIRECTIONS

In this thesis, we have developed new measuring and modeling approaches for understanding the sensorimotor control of walking flies. Our approaches are quite general. Since its publication, Anipose has already been used by more than a dozen different labs across the world in order to quantify animal 3D kinematics, covering flies, beetles, mice, macaques, spiders, horses, and even humans! We hope that the model we lay out in Chapter 5 will similarly be useful to understand animal movement both in flies going forward and across other species.

Here, we lay out some possible future steps for both animal pose estimation and walking models, based on our conversations with many people.

6.1 3D animal pose estimation

6.1.1 Improved documentation and software for recording videos from multiple cameras

When getting started with 3D pose estimation, many labs struggle with setting up the system for recording from multiple cameras in a synchronized fashion. They often ask me which cameras we used and how we synchronized them. In fact, when I started setting up another rig for 3D pose estimation in walking flies, I ended up spending another month on tuning software for recording. This is often too big of a hurdle for many labs to even get started. A few labs have made some bespoke solutions for specific cameras which could all be integrated into a common framework that works across multiple different camera types. Future work in this space should focus on packaging up better documentation and software for acquiring and synchronizing videos from multiple cameras.

6.1.2 Easy and robust camera calibration

Once the cameras are set up, the next biggest hurdle is in getting the cameras calibrated to allow triangulation. Sometimes this is possible to do in days, but I have seen labs struggle with this for months. The current process of calibration used in Anipose is described in Section 2.3.1. Briefly, a user takes a calibration board with a specific checkerboard pattern and waves it around in different orientations in front of the cameras. Anipose detects the corners within the checkerboard using OpenCV and runs a robust optimization to find a good camera calibration.

There are many improvements that could be made to the calibration protocol and algorithm to make this process easier and more robust. First, the checkerboard corners could be detected more robustly with a neural network [127], helping calibrate setups in dark environments or for imaging setups with narrow depths of field leading to blurring. Second, different object types could be supported for calibration, such as wands or custom 3D objects, in order to help calibrate camera configurations where multiple cameras cannot simultaneously image a plane (as in 2 cameras opposite each other, for instance). Third, perhaps it is possible to get rid of the calibration board entirely and calibrate the cameras based on commonly detected points in the environment or on the animal. Fourth, the whole calibration process should have an interactive graphical interface (such as in [228]) so that the experimenter can assess in real time whether their calibration worked or whether they should record another calibration video.

6.1.3 Improved behavior visualization and annotation

The behavior visualization interface in Anipose is described in Section 2.3.6 and Figure 2.6. It has been in use for the past few years within the Tuthill lab for understanding fly behavior. As of this writing, an instance of the visualization for the Tuthill lab lives at `https://flyviz.biz` . To my knowledge, this is currently the only way to visualize and annotate behavioral data obtained from multiple camera views on a web browser.

Although it has been tremendously useful internally, the behavior visualization and annotation tool has not been adopted outside of our lab. With some work, I anticipate that this vizualizer tool could become useful to more researchers doing 3D tracking. One place for future development is to describe the visualizer in detail in the current Anipose documentation, to help researchers become aware of it in the first place. The visualization could also be independent from Anipose and to support data in more formats, so that researchers could use the tool to review and annotate their behavior regardless of how or if they are doing 3D tracking. Finally, the visualization could integrate the parallel data that scientists often need to see in a common interface along their behavioral videos, such as neural recordings or metadata about an experiment.

6.1.4 Reduce annotation

In Chapter 3, we presented BKinD-3D, a method to discover 3D keypoints from multiview videos without annotations. The method is promising, but as can be seen in Tables 3.2 and 3.3, it still falls behind supervised approaches by a factor of 2 (for rats) to 5 (for humans). In addition, these unsupervised methods are currently not integrated into an easy to use framework made for biologists. As a result, a scientist interested in quantifying 3D kinematics of an animal would likely default to a supervised approach.

Future work could focus on resolving these issues. BKinD-3D and other unsupervised approaches could be adapted for use in a semi-supervised setting to reach supervised accuracy with less annotations. Both semi-supervised and unsupervised approaches could be integrated into a framework like Anipose to make it widely available to scientists studying animal behavior.

6.1.5 Improved annotation of 3D pose

I have tried a few different ways to annotate keypoints throughout my PhD. I started with Fiji [244], then tried the DeepLabCut annotation tool [185], a custom Matlab solution, before finally settling on VGG VIA [81] to do our annotations. The reason for sticking with VIA

has been that (1) it is web-based so easy for annotators to get started with and (2) it is very responsive which makes annotation less frustrating. However, it's a generic annotation tool, so it doesn't have good support for deleting and correcting keypoints corresponding to specific bodyparts. Furthermore, none of the tools listed above handle the unique constraints of 3D pose labeling, namely having multiple views where you can reproject one keypoint to other views.

DeepFly3D [113] has support for web-based annotation of 3D pose, but its code architecture is too specific for their setup, making it hard to apply to new projects without editing the code. The JARVIS toolbox [48] looks really great for 3D animal annotation, but is not web based so users must have the framework set up locally along with their data to annotate. Having everything locally can actually be an advantage sometimes, but I think there is also room for a general web-based 3D pose annotation tool. I'm excited to see where 3D pose annotation will be in the future.

Additionally, the advent of self-supervised keypoint discovery (as in Chapter 3) provides an exciting starting point of annotation of 3D pose. Besides reducing the need for annotation (as described in previous section), the keypoint discovery approaches could guide users to decide where to place keypoints to model movement. For instance, in Figure 3.3, BKinD-3D places multiple keypoints which could be used to model hip rotation, whereas the ground truth keypoints (manually chosen) only had a few keypoints which are insufficient to get hip rotation.

6.1.6 3D multi-animal tracking

Although there are a few frameworks to estimate multi-animal 2D pose, notably SLEAP [214] and DeepLabCut [184], there currently aren't any frameworks for performing multi-animal 3D pose estimation.

This is due to several reasons. First, collecting multi-view behavioral data is still somewhat niche currently and thus the intersection of multi-view and multi-animal is even more niche. Nevertheless, a few labs have started to collect this kind of data [178, 200, 300], and it

will likely become more prevalent in the future. Second, the algorithms for building multi-view correspondence are qualitatively different from algorithms for multi-animal 2D pose estimation, and few people have experience with both currently. Thus, there has been a thriving literature on multi-person 3D pose estimation within computer vision [54, 128] and it has yet to be translated for the general animal behavior community.

6.2 Feedback modeling of movement

6.2.1 Model the proprioceptors

In our layered model, we modeled the proprioceptors as perfect sensors of the joint angles and derivatives except for some sensory delay. However, proprioceptors are generally more complicated. In fly legs, there are three broad classes of proprioceptors: chordotonal organs (sensing joint angles and derivatives), hair plates (sensing joint limits), and campaniform sensilla (sensing leg load) [275]. Furthermore, mechanosensation from bristles likely contributes to body state estimation as well [276].

There is some initial data to build an initial model of some of proprioceptors, but many proprioceptors in the fly leg have yet to be characterized. Akira Mamiya has characterized the femoral chordotonal neurons in the T1 leg enough to build an initial model of their responses [173], but the chordotonal organ in the coxa of T1 or in femur of T2 and T3 have yet to be characterized in fruit flies. Similarly, the functional responses of hair plates to kinematics and campaniform sensilla to load forces need to be quantified in more detail to include them in a feedback walking model. Until this data becomes available, some initial guesses of the encoding properties could be made based on literature in other insects, biomechanical models based on anatomy [174], and general function attributed to each sensor. Perhaps it may even be possible to place constraints on what kind of proprioceptive encoding would be required for generating robust walking.

6.2.2 More realistic biomechanics

Our model of biomechanics within Chapter 5 was a rather simplified model of a fly leg. Specifically, we did not model the contribution of muscles, tendons, ground, and viscous forces. Here we briefly review what models and data are already available for future work to integrate into a feedback model.

An ideal muscle model would encompass the muscle attachment points, the dynamics of muscle activation, and the relationship between muscle activity and force exerted the joint at the attachment point. The muscle attachment points may be estimated from x-ray holographic nano-tomography images [153] [13] or perhaps inferred through some optimality condition on walking [120]. The dynamics of muscle activity could be modeled following existing models in the literature such as a spring and damper Ekeberg model [84][161], a Hill-type model [121], or more modern musculoskeletal-type models [71] [126]. The ideal muscle model would also include consideration for muscle activation and deactivation delays, which place constraints on movement dynamics [41, 202]. Note however that a lot of existing models are built for verterbrate muscles and as data is collected on muscle dynamics and relationship to neuronal firing [16], more precise models may be built by fitting the data itself.

Paralleling the ideal muscle model, an ideal tendon model would include the attachment points, the tendon dynamics, and the constraints the tendon induces on the kinematics. As in muscles, the tendon attachment points may be estimated from x-ray holographic nano-tomography [174]. The tendon dynamics may be modeled following biomechanics from the attachment points [174] and from these dynamics one could derive the constraints on kinematics.

How the tendon and muscle attachment points influence the kinematics of the leg depends critically on the shape of the legs, thorax, and abdomen. There is ongoing effort to model these and integrate attachment points [161]. Future work could perhaps build off of these integrated models.

Finally, animals interact with the world through physical dynamics and these should be modeled too for a complete understanding of the sensorimotor loop that drives behavior. This could be done by using a physics simulation toolbox such as Pybullet [62, 161], Mujoco [272], or others [85]. Additionaly, flies are comparatively small and at their scale viscous forces from the air become non-negligible [267] and should be accounted for in a complete neuromechanical model of walking. Integrating ground collisions and viscous forces poses an additional challenge from a modeling perspective, as these introduce nonlinearities into the dynamics. In the layered model presented in Chapter 5, we used a linear controller in order to provide guarantees on optimality with sensory and actuation delays added. However, this controller fails when interacting with unexpected nonlinearities from the ground collisions. In addition, our data-driven model for kinematics cannot be adjusted for different physical interactions. Similar models currently get around these problems by providing diverse environments and optimizing a broader objective (like walking speed) in a reinforcement learning framework. How to integrate physics in a feedback model while preserving interpretability and match observed kinematics remains a key issue for future work.

6.2.3 Learn representations constrained by connectome

In another step towards biological realism, the trajectory generator and optimal controller may be somehow constrained by the possible pathways within the connectome or potentially replaced by a connectome-based simulation entirely. This is now becoming feasible as the connectome is being increasingly characterized for fruit flies, with recent reports characterizing the connections of premotor neurons [158], motor neurons [175], and their muscle targets [13].

Some challenges from using a connectome within a feedback model are that the ventral nerve cord is not completely mapped and that more realistic biomechanics models are still missing. The incompleteness of the connectome means that some of the critical interneurons for walking may be missing from the map, impeding the construction of feedback model. Furthermore, the mapping from motor neurons to muscles [13] is promising for building a complete feedback model of walking, but we may still need more detailed muscle and

biomechanics models to effectively close the loop.

Our layered approach could be useful for overcoming these challenges and learning from closed loop models despite having incomplete information about the biological system. For instance, the trajectory generator artificial neural network (ANN) model could be replaced by a combination of connectome simulation and ANN for the missing neurons, with the constraint on the ANN dynamics coming from the complete circuit generating kinematics. Similarly, the biomechanical model within the feedback model could be progressively expanded with more biological realism as the data and models become available. As more realism gets added to the model, we gain increasing power in probing how different aspects of the full circuit contribute to the generation of walking within a feedback paradigm.

6.2.4 Expand to other animals and behaviors

Finally, our feedback modeling approach could be expanded to other behaviors beyond walking and animals beyond flies, even ones where the physiology is not well known or accessible. The requirements for our model are (1) 3D kinematics of a specific behavior, (2) a biomechanical model (possibly simplified) of the animal's limbs, and (3) phase coupling parameters across limbs. The phase coupling parameters may be extracted from kinematics, so really only kinematics and simple biomechanics model are required. This makes the model quite general and in principle it could be applied to model the sensorimotor loop in other animals such as mice, horses, beetles, macaques, and others performing a wide variety of behaviors.

6.3 Conclusion

In this thesis, we have described in detail how to quantify the fine kinematics of animal movement and then use these kinematics to shed light on sensorimotor loops, focusing on walking in fruit flies. Along the way, we discovered a new degree of freedom in fruit fly legs and placed limits on the physiological delays for robust walking with the observed kinematics.

A lot of mysteries remain about the feedback circuits that generate the fluid movement of animals. How do they coordinate so many joints? How are the neural circuits underlying

locomotion organized? How have these circuits evolved differently or similarly across animals with different environmental requirements? How do nervous systems combine sensory inputs with internal state when generating fast kinematics? Why is there such a close match between the physiological delays and the kinematics in our simulations?

It will likely take decades if not centuries for neuroscience to fully describe the principles underlying the feedback circuitry of movement. I hope that the work described here points in a fruitful direction towards this description.

BIBLIOGRAPHY

[1] Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. 2015.

[2] Ahmed S. Abdelfattah, Takashi Kawashima, Amrita Singh, Ondrej Novak, Hui Liu, Yichun Shuai, Yi-Chieh Huang, Luke Campagnola, Stephanie C. Seeman, Jianing Yu, Jihong Zheng, Jonathan B. Grimm, Ronak Patel, Johannes Friedrich, Brett D. Mensh, Liam Paninski, John J. Macklin, Gabe J. Murphy, Kaspar Podgorski, Bei-Jung Lin, Tsai-Wen Chen, Glenn C. Turner, Zhe Liu, Minoru Koyama, Karel Svoboda, Misha B. Ahrens, Luke D. Lavis, and Eric R. Schreiter. Bright and photostable chemigenetic indicators for extended in vivo voltage imaging. *Science*, 365(6454):699–704, 2019.

[3] Juan A Acebrón, Luis L Bonilla, Conrad J Pérez Vicente, Félix Ritort, and Renato Spigler. The kuramoto model: A simple paradigm for synchronization phenomena. *Reviews of modern physics*, 77(1):137, 2005.

[4] Jan M. Ache and Volker Dürr. A computational model of a descending mechanosensory pathway involved in active tactile sensing. *PLOS Computational Biology*, 11:e1004263, 2015.

[5] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Richard Szeliski. Building Rome in a day. *Communications of the ACM*, 54(10):105–112, October 2011.

[6] Sweta Agrawal, Evyn S Dickinson, Anne Sustar, Pralaksha Gurung, David Shepherd, James W Truman, and John C Tuthill. Central processing of leg proprioception in *Drosophila*. *eLife*, 9:e60299, 2020.

[7] Scott T Albert, Alkis M Hadjiosif, Jihoon Jang, Andrew J Zimnik, Demetris S Soteropoulos, Stuart N Baker, Mark M Churchland, John W Krakauer, and Reza Shadmehr. Postural control of arm and fingers through integration of movement commands. *eLife*, 9:e52507, 2020.

[8] David E. Alexander. *Nature's Machines: An Introduction to Organismal Biomechanics*. Academic Press, 1 edition, 2017.

[9] Jessica L. Allen and Lena H. Ting. Why is neuromechanical modeling of balance and locomotion so hard? In Boris I. Prilutsky and Donald H. Edwards, editors, *Neuromechanical Modeling of Posture and Locomotion*, pages 197–223. Springer New York, New York, NY, 2016.

[10] Sikandar Amin, Mykhaylo Andriluka, Marcus Rohrbach, and Bernt Schiele. Multi-view Pictorial Structures for 3D Human Pose Estimation. In *Procedings of the British Machine Vision Conference 2013*, pages 45.1–45.11, Bristol, 2013. British Machine Vision Association.

[11] Zahra Aminzare, Vaibhav Srivastava, and Philip Holmes. Gait Transitions in a Phase Oscillator Model of an Insect Central Pattern Generator. *SIAM Journal on Applied Dynamical Systems*, 17(1):626–671, 2018.

[12] Dora E. Angelaki and Kathleen E. Cullen. Vestibular system: the many facets of a multimodal sense. *Annual Review of Neuroscience*, 31:125–150, 2008.

[13] Anthony Azevedo, Ellen Lesser, Brandon Mark, Jasper Phelps, Leila Elabbady, Sumiya Kuroda, Anne Sustar, Anthony Moussa, Avinash Kandelwal, Chris J. Dallmann, Sweta Agrawal, Su-Yee J. Lee, Brandon Pratt, Andrew Cook, Kyobi Skutt-Kakaria, Stephan Gerhard, Ran Lu, Nico Kemnitz, Kisuk Lee, Akhilesh Halageri, Manuel Castro, Dodam Ih, Jay Gager, Marwan Tammam, Sven Dorkenwald, Forrest Collman, Casey Schneider-Mizell, Derrick Brittain, Chris S. Jordan, Michael Dickinson, Alexandra Pacureanu, H. Sebastian Seung, Thomas Macrina, Wei-Chung Allen Lee, and John C. Tuthill. Tools for comprehensive reconstruction and analysis of *Drosophila* motor circuits. Preprint, Neuroscience, December 2022.

[14] Anthony W Azevedo, Evyn S Dickinson, Pralaksha Gurung, Lalanti Venkatasubramanian, Richard S Mann, and John C Tuthill. A size principle for recruitment of Drosophila leg motor neurons. *eLife*, 9:e56754, June 2020.

[15] Anthony W Azevedo, Evyn S Dickinson, Pralaksha Gurung, Lalanti Venkatasubramanian, Richard S Mann, and John C Tuthill. A size principle for recruitment of drosophila leg motor neurons. *Elife*, 9:e56754, 2020.

[16] Anthony W Azevedo, Evyn S Dickinson, Pralaksha Gurung, Lalanti Venkatasubramanian, Richard S Mann, and John C Tuthill. A size principle for recruitment of Drosophila leg motor neurons. *eLife*, 9:e56754, 2020.

[17] Eiman Azim, Juan Jiang, Bror Alstermark, and Thomas M Jessell. Skilled reaching relies on a V2a propriospinal internal copy circuit. *Nature*, 508(7496):357–363, 2014.

[18] Julien Bacqué-Cazenave, Bryce Chung, David W Cofer, Daniel Cattaert, and Donald H Edwards. The effect of sensory feedback on crayfish posture and locomotion. II. Neuromechanical simulation of closing the loop. *Journal of Neurophysiology*, 113:1772–1783, 2015.

[19] Praneet C. Bala, Benjamin R. Eisenreich, Seng Bum Michael Yoo, Benjamin Y. Hayden,

Hyun Soo Park, and Jan Zimmermann. Automated markerless pose estimation in freely moving macaques with OpenMonkeyStudio. *Nature Communications*, 11(1):4560, September 2020.

[20] Gustavo Balbinot, Clarissa Pedrini Schuch, Matthew S. Jeffers, Matthew W. McDonald, Jessica M. Livingston-Thomas, and Dale Corbett. Post-stroke kinematic analysis in rats reveals similar reaching abnormalities as humans. *Scientific Reports*, 8(1):8738, December 2018.

[21] Friedrich G. Barth. Mechanics to pre-process information for the fine tuning of mechanoreceptors. *Journal of Comparative Physiology A*, 205:661–686, 2019.

[22] Ulrich Bässler. Afferent control of walking movements in the stick insect *Cuniculina impigra*. *Journal of Comparative Physiology A*, 158:351–362, 1986.

[23] Matthew I. Becker and Abigail L. Person. Cerebellar Control of Reach Kinematics for Endpoint Precision. *Neuron*, 103(2):335–348.e5, July 2019.

[24] Trevor Bekolay, James Bergstra, Eric Hunsberger, Travis DeWolf, Terrence C. Stewart, Daniel Rasmussen, Xuan Choo, Aaron Russell Voelker, and Chris Eliasmith. Nengo: A Python tool for building large-scale functional brain models. *Frontiers in Neuroinformatics*, 7:48, 2014.

[25] John A. Bender, Elaine M. Simpson, and Roy E. Ritzmann. Computer-Assisted 3D Kinematic Analysis of All Leg Joints in Walking Insects. *PLoS ONE*, 5(10):e13617, October 2010.

[26] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *ICML*, 2009.

[27] Volker Berendes, Sasha N. Zill, Ansgar Büschges, and Till Bockemühl. Speed-dependent interplay between local pattern-generating activity and sensory signals during walking in Drosophila. *Journal of Experimental Biology*, page jeb.146720, January 2016.

[28] Gordon J. Berman, Daniel M. Choi, William Bialek, and Joshua W. Shaevitz. Mapping the stereotyped behaviour of freely moving fruit flies. *Journal of The Royal Society Interface*, 11(99):20140672, 2014.

[29] Jacob G. Bernstein, Paul A. Garrity, and Edward S. Boyden. Optogenetics and thermogenetics: technologies for controlling the activity of targeted cells within intact neural circuits. *Current Opinion in Neurobiology*, 22(1):61–71, February 2012.

[30] Salil S. Bidaye, Till Bockemühl, and Ansgar Büschges. Six-legged walking in insects: How CPGs, peripheral feedback, and descending signals generate coordinated and adaptive motor rhythms. *Journal of Neurophysiology*, 119(2):459–475, October 2017.

[31] Kyle P Blum, Kenneth S Campbell, Brian C Horslen, Paul Nardelli, Stephen N Housley, Timothy C Cope, and Lena H Ting. Diverse and complex muscle spindle afferent firing properties emerge from multiscale muscle mechanics. *eLife*, 9:e55177, 2020.

[32] Kyle P. Blum, Boris Lamotte D'Incamps, Daniel Zytnicki, and Lena H. Ting. Force encoding in muscle spindles during stretch of passive muscle. *PLOS Computational Biology*, 13:e1005767, 2017.

[33] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[34] Mary Ann Branch, Thomas F. Coleman, and Yuying Li. A subspace, interior, and conjugate gradient method for large-scale bound-constrained minimization problems. *SIAM Journal on Scientific Computing*, 21(1):1–23, 1999.

[35] Kristin Branson. Animal part tracker. `https://github.com/kristinbranson/APT`.

[36] Kristin Branson, Alice A. Robie, John Bender, Pietro Perona, and Michael H. Dickinson. High-throughput ethomics in large groups of drosophila. *Nature Methods*, 6(6):451–457, 2009.

[37] F. V. Van Breugel, J. N. Kutz, and B. W. Brunton. Numerical Differentiation of Noisy Data: A Unifying Multi-Objective Optimization Framework. *IEEE Access*, 8:196865–196877, 2020.

[38] A. G. Brown. *Nerve cells and nervous systems: an introduction to neuroscience*. Springer, London, second edition, 2001.

[39] A. G. Brown and D. N. Franz. Responses of spinocervical tract neurones to natural stimulation of identified cutaneous receptors. *Experimental Brain Research*, 7:231–249, 1969.

[40] Alan Geoffrey Brown. *Organization in the spinal cord: the anatomy and physiology of identified neurones*. Springer Science & Business Media, 2012.

[41] Ian E. Brown and Gerald E. Loeb. Measured and modeled properties of mammalian skeletal muscle: IV. Dynamics of activation and deactivation. *Journal of Muscle Research & Cell Motility*, 21:33–47, 2000.

[42] E. Buchner. Elementary movement detectors in an insect visual system. *Biological Cybernetics*, 24(2):85–101, 1976.

[43] Xavier P Burgos-Artizzu, Piotr Dollár, Dayu Lin, David J Anderson, and Pietro Perona. Social behavior recognition in continuous video. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1322–1329. IEEE, 2012.

[44] Malcolm Burrows. *The neurobiology of an insect brain*. Oxford University Press, 1996.

[45] Richard H. Byrd, Robert B. Schnabel, and Gerald A. Shultz. Approximate solution of the trust region problem by minimization over two-dimensional subspaces. *Mathematical programming*, 40(1):247–263, 1988.

[46] Ulrich Bässler, Harald Wolf, and Wolfgang Stein. Functional recovery following manipulation of muscles and sense organs in the stick insect leg. *Journal of Comparative Physiology A*, 193:1151–1168, 2007.

[47] Z. Cao, G. Hidalgo Martinez, T. Simon, S.-E. Wei, and Y. A. Sheikh. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2019.

[48] Jarvis-3D Motion Capture. JARVIS-AcquisitionTool, June 2023.

[49] Cristian Sminchisescu Catalin Ionescu, Fuxin Li. Latent structured models for human pose estimation. In *International Conference on Computer Vision*, 2011.

[50] Boyuan Chen, Pieter Abbeel, and Deepak Pathak. Unsupervised learning of visual 3d keypoints for control. In *International Conference on Machine Learning*, pages 1539–1549. PMLR, 2021.

[51] Ching-Hang Chen and Deva Ramanan. 3d human pose estimation= 2d pose estimation+ matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7035–7043, 2017.

[52] Ching-Hang Chen, Ambrish Tyagi, Amit Agrawal, Dylan Drover, Stefan Stojanov, and James M Rehg. Unsupervised 3d pose estimation with geometric self-supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5714–5724, 2019.

[53] Long Chen, Haizhou Ai, Rui Chen, Zijie Zhuang, and Shuang Liu. Cross-view tracking for multi-human 3d pose estimation at over 100 fps. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3279–3288, 2020.

[54] Long Chen, Haizhou Ai, Rui Chen, Zijie Zhuang, and Shuang Liu. Cross-View Tracking for Multi-Human 3D Pose Estimation at Over 100 FPS. In *2020 IEEE/CVF Conference*

on Computer Vision and Pattern Recognition (CVPR), pages 3276–3285, Seattle, WA, USA, June 2020. IEEE.

[55] Yu Cheng, Bo Yang, Bo Wang, Yan Wending, and Robby Tan. Occlusion-Aware Networks for 3D Human Pose Estimation in Video. *ICCV*, pages 723–732, October 2019.

[56] Hiroshi Chiba, Satoru Ebihara, Naoki Tomita, Hidetada Sasaki, and James P Butler. Differential gait kinematics between fallers and non-fallers in community-dwelling elderly people. *Geriatrics and Gerontology International*, 5(2):127–134, June 2005.

[57] Bryce Chung, Julien Bacqué-Cazenave, David W Cofer, Daniel Cattaert, and Donald H Edwards. The effect of sensory feedback on crayfish posture and locomotion. I. Experimental analysis of closing the loop. *Journal of Neurophysiology*, 113:1763–1771, 2015.

[58] Mark M. Churchland, John P. Cunningham, Matthew T. Kaufman, Justin D. Foster, Paul Nuyujukian, Stephen I. Ryu, and Krishna V. Shenoy. Neural population dynamics during reaching. *Nature*, 487:51–56, 2012.

[59] François Clarac, Daniel Cattaert, and Didier Le Ray. Central control components of a 'simple' stretch reflex. *Trends in Neurosciences*, 23:199–208, 2000.

[60] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[61] David Cofer, Gennady Cymbalyuk, James Reid, Ying Zhu, William J. Heitler, and Donald H. Edwards. AnimatLab: A 3D graphics environment for neuromechanical simulations. *Journal of Neuroscience Methods*, 187:280–288, 2010.

[62] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. 2016.

[63] Einat Couzin-Fuchs, Tim Kiemel, Omer Gal, Amir Ayali, and Philip Holmes. Intersegmental coupling and recovery from perturbations in freely running cockroaches. 218:285–297.

[64] Einat Couzin-Fuchs, Tim Kiemel, Omer Gal, Amir Ayali, and Philip Holmes. Intersegmental coupling and recovery from perturbations in freely running cockroaches. *Journal of Experimental Biology*, 218(2):285–297, 2015.

[65] Chris J. Dallmann, Volker Dürr, and Josef Schmitz. Motor control of an insect leg during level and incline walking. *The Journal of Experimental Biology*, 222(7):jeb188748, April 2019.

[66] Chris J. Dallmann, Pierre Karashchuk, Bingni W. Brunton, and John C. Tuthill. A leg to stand on: Computational models of proprioception. *Current Opinion in Physiology*, March 2021.

[67] Hod Dana, Yi Sun, Boaz Mohar, Brad K. Hulse, Aaron M. Kerlin, Jeremy P. Hasseman, Getahun Tsegaye, Arthur Tsang, Allan Wong, Ronak Patel, John J. Macklin, Yang Chen, Arthur Konnerth, Vivek Jayaraman, Loren L. Looger, Eric R. Schreiter, Karel Svoboda, and Douglas S. Kim. High-performance calcium sensors for imaging activity in neuronal populations and microcompartments. *Nature Methods*, 16(7):649–657, July 2019.

[68] David G. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[69] Brian D DeAngelis, Jacob A Zavatone-Veth, and Damon A Clark. The manifold structure of limb coordination in walking Drosophila. *eLife*, 8:e46409, 2019.

[70] Brian D. Deangelis, Jacob A. Zavatone-Veth, and Damon A. Clark. The manifold structure of limb coordination in walking drosophila. *eLife*, 8:1–33, 2019.

[71] Scott L. Delp, Frank C. Anderson, Allison S. Arnold, Peter Loan, Ayman Habib, Chand T. John, Eran Guendelman, and Darryl G. Thelen. OpenSim: open-source

software to create and analyze dynamic simulations of movement. *IEEE transactions on biomedical engineering*, 54(11):1940–1950, 2007.

[72] Jack E. Denham, Thomas Ranner, and Netta Cohen. Signatures of proprioceptive control in *Caenorhabditis elegans* locomotion. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373:20180208, 2018.

[73] Travis DeWolf, Terrence C. Stewart, Jean-Jacques Slotine, and Chris Eliasmith. A spiking neural model of adaptive arm control. *Proceedings of the Royal Society B: Biological Sciences*, 283:20162134, 2016.

[74] Andrea Di Russo, Dimitar Stanev, Stéphane Armand, and Auke Ijspeert. Sensory modulation of gait characteristics in human locomotion: a neuromusculoskeletal modeling study. *bioRxiv*, 2020.

[75] Bradley H Dickerson, Jessica L Fox, and Simon Sponberg. Functional diversity from generic encoding in insect campaniform sensilla. *Current Opinion in Physiology*, 19:194–203, 2021.

[76] Junting Dong, Wen Jiang, Qixing Huang, Hujun Bao, and Xiaowei Zhou. Fast and Robust Multi-Person 3D Pose Estimation from Multiple Views. *CVPR*, January 2019.

[77] Duffy Joseph B. GAL4 system in drosophila: A fly geneticist's swiss army knife. *genesis*, 34(1-2):1–15, September 2002.

[78] Timothy W. Dunn, Jesse D. Marshall, Kyle S. Severson, Diego E. Aldarondo, David G. C. Hildebrand, Selmaan N. Chettih, William L. Wang, Amanda J. Gellis, David E. Carlson, Dmitriy Aronov, Winrich A. Freiwald, Fan Wang, and Bence P. Ölveczky. Geometric deep learning enables 3D kinematic profiling across species and environments. *Nature Methods*, 18(5):564–573, May 2021.

[79] Timothy W Dunn, Jesse D Marshall, Kyle S Severson, Diego E Aldarondo, David GC Hildebrand, Selmaan N Chettih, William L Wang, Amanda J Gellis, David E Carlson,

Dmitriy Aronov, et al. Geometric deep learning enables 3d kinematic profiling across species and environments. *Nature methods*, 18(5):564–573, 2021.

[80] A. Dutta, A. Gupta, and A. Zissermann. VGG image annotator (VIA). `http://www.robots.ox.ac.uk/~vgg/software/via/`, 2016.

[81] Abhishek Dutta and Andrew Zisserman. The VIA annotation software for images, audio and video. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM '19, New York, NY, USA, 2019. ACM.

[82] Rudi D'Hooge and Peter P. De Deyn. Applications of the morris water maze in the study of learning and memory. *Brain research reviews*, 36(1):60–90, 2001.

[83] SE Roian Egnor and Kristin Branson. Computational analysis of behavior. *Annual review of neuroscience*, 39:217–236, 2016.

[84] Örjan Ekeberg. A combined neuronal and mechanical model of fish swimming. *Biological Cybernetics*, 69(5):363–374, September 1993.

[85] Tom Erez, Yuval Tassa, and Emanuel Todorov. Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4397–4404. IEEE, 2015.

[86] Maria Soledad Esposito, Paolo Capelli, and Silvia Arber. Brainstem nucleus MdV mediates skilled forelimb motor tasks. *Nature*, 508(7496):351–356, April 2014.

[87] Eyrun Eyjolfsdottir, Kristin Branson, Yisong Yue, and Pietro Perona. Learning recurrent representations for hierarchical behavior modeling. *ICLR*, 2017.

[88] Eyrun Eyjolfsdottir, Steve Branson, Xavier P Burgos-Artizzu, Eric D Hoopfer, Jonathan Schor, David J Anderson, and Pietro Perona. Detecting social actions of fruit flies. In *European Conference on Computer Vision*, pages 772–787. Springer, 2014.

[89] Tracy D. Farr and Ian Q. Whishaw. Quantitative and Qualitative Impairments in Skilled Reaching in the Mouse (Mus musculus) After a Focal Motor Cortex Stroke. *Stroke*, 33(7):1869–1875, July 2002.

[90] L.H. Field and T. Matheson. Chordotonal organs of insects. *Advances in Insect Physiology*, 27:1–56, 1998.

[91] Brett W. Fling, Geetanjali Gera Dutta, Heather Schlueter, Michelle H. Cameron, and Fay B. Horak. Associations between Proprioceptive Neural Pathway Structural Connectivity and Balance in People with Multiple Sclerosis. *Frontiers in Human Neuroscience*, 8, October 2014.

[92] D.W. Franklin and D.M. Wolpert. Computational mechanisms of sensorimotor control. *Neuron*, 72:425–442, 2011.

[93] Leonid Frantsevich and Weiying Wang. Gimbals in the insect leg. *Arthropod Structure & Development*, 38(1):16–30, 2009.

[94] A. S. French and K. Pfeiffer. Nonlinearization: naturalistic stimulation and nonlinear dynamic behavior in a spider mechanoreceptor. *Biological Cybernetics*, page 403–413, 2018.

[95] Andrew S. French, Esa-Ville Immonen, and Roman V. Frolov. Static and dynamic adaptation of insect photoreceptor responses to naturalistic stimuli. *Frontiers in Physiology*, 7:477, 2016.

[96] Andrew S French, Päivi H Torkkeli, and Ernst-August Seyfarth. From stress and strain to spikes: mechanotransduction in spider slit sensilla. *Journal of Comparative Physiology A*, 188:739–752, 2002.

[97] Claudiane A. Fukuchi, Reginaldo K. Fukuchi, and Marcos Duarte. A public dataset of overground and treadmill walking kinematics and kinetics in healthy individuals. *PeerJ*, 6:e4640, April 2018.

[98] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, June 2014.

[99] Thomas Geijtenbeek, Michiel Van De Panne, and A Frank Van Der Stappen. Flexible muscle-based locomotion for bipedal creatures. *ACM Transactions on Graphics (TOG)*, 32(6):1–11, 2013.

[100] C. A. Goldsmith, N. S. Szczecinski, and R. D. Quinn. Neurodynamic modeling of the fruit fly drosophila melanogaster. *Bioinspiration and Biomimetics*, 15(6), 2020.

[101] C A Goldsmith, N S Szczecinski, and R D Quinn. Neurodynamic modeling of the fruit fly *Drosophila melanogaster*. *Bioinspiration & Biomimetics*, 15(6):065003, 2020.

[102] C A Goldsmith, N S Szczecinski, and R D Quinn. Neurodynamic modeling of the fruit fly *Drosophila melanogaster*. *Bioinspiration & Biomimetics*, 15:065003, 2020.

[103] Clarissa Goldsmith, Nicholas Szczecinski, and Roger Quinn. Drosophibot: A fruit fly inspired bio-robot. In *Biomimetic and Biohybrid Systems*, pages 146–157. Springer International Publishing, 2019.

[104] Clarissa Goldsmith, Nicholas S. Szczecinski, and Roger D. Quinn. Response of a neuromechanical insect joint model to inhibition of fCO sensory afferents. In Vasiliki Vouloutsi, Anna Mura, Falk Tauber, Thomas Speck, Tony J. Prescott, and Paul F. M. J. Verschure, editors, *Biomimetic and Biohybrid Systems*, Lecture Notes in Computer Science, pages 141–152, Cham, 2020. Springer International Publishing.

[105] Ana I. Goncalves, Jacob A. Zavatone-Veth, Megan R. Carey, and Damon A. Clark. Parallel locomotor control strategies in mice and flies. *Current Opinion in Neurobiology*, 73:102516, 2022.

[106] Guy M. Goodwin, D. Ian McCloskey, and Peter B. C. Matthews. Proprioceptive illusions induced by muscle vibration: Contribution by muscle spindles to perception? *Science*, 175:1382–1384, 1972.

[107] JAMES Gordon, Maria Felice Ghilardi, and Claude Ghez. Impairments of reaching movements in patients without proprioception. I. Spatial errors. *Journal of Neurophysiology*, 73:347–360, 1995.

[108] Adam Gosztolai, Semih Günel, Victor Lobato-Ríos, Marco Pietro Abrate, Daniel Morales, Helge Rhodin, Pascal Fua, and Pavan Ramdya. Liftpose3d, a deep learning-based approach for transforming two-dimensional to three-dimensional poses in laboratory animals. *Nature methods*, 18(8):975–981, 2021.

[109] Jacob M Graving, Daniel Chae, Hemal Naik, Liang Li, Benjamin Koger, Blair R Costelloe, and Iain D Couzin. Deepposekit, a software toolkit for fast and robust animal pose estimation using deep learning. *eLife*, 8:e47994, oct 2019.

[110] Pierre A Guertin. The mammalian central pattern generator for locomotion. *Brain Research Reviews*, 62:45–56, 2009.

[111] Jian-Zhong Guo, Austin R Graves, Wendy W Guo, Jihong Zheng, Allen Lee, Juan Rodríguez-González, Nuo Li, John J Macklin, James W Phillips, Brett D Mensh, Kristin Branson, and Adam W Hantman. Cortex commands the performance of skilled movement. *eLife*, 4:e10774, December 2015.

[112] K. G. Götz. Visual control of locomotion in the walking fruitfly Drosophila. *Journal of Comparative Physiology*, 85(3):235–266, 1973.

[113] Semih Günel, Helge Rhodin, Daniel Morales, João Campagnolo, Pavan Ramdya, and Pascal Fua. Deepfly3d, a deep learning-based approach for 3d limb and appendage tracking in tethered, adult *Drosophila*. *eLife*, 8:e48571, oct 2019.

[114] Jamin Halberstadt, Joshua Conrad Jackson, David Bilkey, Jonathan Jong, Harvey Whitehouse, Craig McNaughton, and Stefanie Zollmann. Incipient Social Groups: An Analysis via In-Vivo Behavioral Tracking. *PLOS ONE*, 11(3):e0149880, 2016.

[115] Christina L. Hamlet, Kathleen A. Hoffman, Eric D. Tytell, and Lisa J. Fauci. The role of curvature feedback in the energetics and dynamics of lamprey swimming: a closed-loop model. *PLOS Computational Biology*, 14:e1006324, 2018.

[116] Richard I. Hartley and Peter Sturm. Triangulation. *Computer vision and image understanding*, 68(2):146–157, 1997.

[117] Z Hasan. A model of spindle afferent response to muscle stretch. *Journal of Neurophysiology*, 49:989–1006, 1983.

[118] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *CVPR*, 2016. arXiv: 1512.03385.

[119] Xingzhe He, Bastian Wandt, and Helge Rhodin. Autolink: Self-supervised learning of human skeletons and object outlines by linking keypoints. *arXiv preprint arXiv:2205.10636*, 2022.

[120] Nicolas Heess, Dhruva TB, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, SM Eslami, et al. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*, 2017.

[121] Archibald Vivian Hill. The heat of shortening and the dynamic constants of muscle. *Proceedings of the Royal Society of London. Series B - Biological Sciences*, 126(843):136–195, 1938.

[122] Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. *ACM Transactions on Graphics*, 36(4):42:1–42:13, July 2017.

[123] Philip Holmes, Robert J. Full, Dan Koditschek, and John Guckenheimer. The dynamics of legged locomotion: Models, analyses, and challenges. *SIAM Review*, 48(2):207–304, 2006.

[124] Weizhe Hong, Ann Kennedy, Xavier P Burgos-Artizzu, Moriel Zelikowsky, Santiago G Navonne, Pietro Perona, and David J Anderson. Automated measurement of mouse social behaviors using depth sensing, video tracking, and machine learning. *Proceedings of the National Academy of Sciences*, 112(38):E5351–E5360, 2015.

[125] J Houk and W Simon. Responses of Golgi tendon organs to forces applied to muscle tendon. *Journal of Neurophysiology*, 30:1466–1481, 1967.

[126] Bingshan Hu, Haoran Tao, Hongrun Lu, Xiangxiang Zhao, Jiantao Yang, and Hongliu Yu. An Improved EMG-Driven Neuromusculoskeletal Model for Elbow Joint Muscle Torque Estimation. *Applied Bionics and Biomechanics*, 2021:1985741, October 2021.

[127] Danying Hu, Daniel DeTone, Vikram Chauhan, Igor Spivak, and Tomasz Malisiewicz. Deep ChArUco: Dark ChArUco Marker Pose Estimation. *arXiv:1812.03247 [cs]*, December 2018.

[128] Congzhentao Huang, Shuai Jiang, Yang Li, Ziyue Zhang, Jason Traish, Chen Deng, Sam Ferguson, and Richard Yi Da Xu. End-to-end Dynamic Matching Network for Multi-view Multi-person 3D Pose Estimation. In *Computer Vision – ECCV 2020*, volume 12373, pages 477–493. Springer International Publishing, Cham, 2020.

[129] Brad K Hulse, Hannah Haberkern, Romain Franconville, Daniel B Turner-Evans, Shinya Takemura, Tanya Wolff, Marcella Noorman, Marisa Dreher, Chuntao Dan, Ruchi Parekh, et al. A connectome of the *Drosophila* central complex reveals network motifs suitable for flexible navigation and context-dependent action selection. *bioRxiv*, 2020.

[130] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt

Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. 2016.

[131] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013.

[132] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[133] Umar Iqbal, Pavlo Molchanov, and Jan Kautz. Weakly-supervised 3d human pose learning via multi-view images in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5243–5252, 2020.

[134] Karim Iskakov, Egor Burkov, Victor Lempitsky, and Yury Malkov. Learnable triangulation of human pose. In *International Conference on Computer Vision (ICCV)*, 2019.

[135] Karim Iskakov, Egor Burkov, Victor Lempitsky, and Yury Malkov. Learnable triangulation of human pose. *arXiv preprint arXiv:1905.05754*, 2019.

[136] Y. P. Ivanenko, R. Grasso, and F. Lacquaniti. Influence of leg muscle vibration on human walking. *Journal of Neurophysiology*, 84:1737–1747, 2000.

[137] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015.

[138] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Unsupervised learning of object landmarks through conditional image generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[139] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Self-supervised learning of interpretable keypoints from unlabelled videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[140] Hueihan Jhuang, Estibaliz Garrote, Xinlin Yu, Vinita Khilnani, Tomaso Poggio, Andrew D Steele, and Thomas Serre. Automated home-cage behavioural phenotyping of mice. *Nature communications*, 1(1):1–10, 2010.

[141] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016.

[142] James J. Jun, Nicholas A. Steinmetz, Joshua H. Siegle, Daniel J. Denman, Marius Bauza, Brian Barbarits, Albert K. Lee, Costas A. Anastassiou, Alexandru Andrei, Cagatay Aydın, Mladen Barbic, Timothy J. Blanche, Vincent Bonin, João Couto, Barundeb Dutta, Sergey L. Gratiy, Diego A. Gutnisky, Michael Häusser, Bill Karsh, Peter Ledochowitsch, Carolina Mora Lopez, Catalin Mitelut, Silke Musa, Michael Okun, Marius Pachitariu, Jan Putzeys, P. Dylan Rich, Cyrille Rossant, Wei-lung Sun, Karel Svoboda, Matteo Carandini, Kenneth D. Harris, Christof Koch, John O'Keefe, and Timothy D. Harris. Fully integrated silicon probes for high-density recording of neural activity. *Nature*, 551(7679):232–236, November 2017.

[143] Mayank Kabra, Alice A Robie, Marta Rivera-Alba, Steven Branson, and Kristin Branson. Jaaba: interactive machine learning for automatic annotation of animal behavior. *Nature methods*, 10(1):64, 2013.

[144] Hari Teja Kalidindi, Kevin P. Cross, Timothy P. Lillicrap, Mohsen Omrani, Egidio Falotico, Philip N. Sabes, and Stephen H. Scott. Rotational dynamics in motor cortex are consistent with a feedback controller. *bioRxiv*, 2020.

[145] Gary A Kane, Gonçalo Lopes, Jonny L Saunders, Alexander Mathis, and Mackenzie W

Mathis. Real-time, low-latency closed-loop feedback using markerless posture tracking. *eLife*, 9:e61909, December 2020. Publisher: eLife Sciences Publications, Ltd.

[146] Pierre Karashchuk, Katie L Rupp, Evyn S Dickinson, Sarah Walling-Bell, Elischa Sanders, Eiman Azim, Bingni W Brunton, and John C Tuthill. Anipose: a toolkit for robust markerless 3d pose estimation. *Cell reports*, 36(13):109730, 2021.

[147] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg. A Survey of Research on Cloud Robotics and Automation. *IEEE Transactions on Automation Science and Engineering*, 12(2):398–409, April 2015.

[148] P. Kibleur, S. R. Tata, N. Greiner, S. Conti, B. Barra, K. Zhuang, M. Kaeser, A. Ijspeert, and M. Capogrosso. Spatiotemporal maps of proprioceptive inputs to the cervical spinal cord during three-dimensional reaching and grasping. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 28:1668–1677, 2020.

[149] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, January 2017.

[150] Dinant A. Kistemaker, Jeremy D. Wong, and Paul L. Gribble. The cost of moving optimally: kinematic path selection. *Journal of Neurophysiology*, 112:1815–1824, 2014.

[151] Muhammed Kocabas, Salih Karagoz, and Emre Akbas. Self-supervised learning of 3d human pose using multi-view geometry. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1077–1086, 2019.

[152] Stephanie C. Koch, Marta Garcia Del Barrio, Antoine Dalet, Graziana Gatto, Thomas Gunther, Jingming Zhang, Barbara Seidler, Dieter Saur, Roland Schuele, and Martyn Goulding. RORβ spinal interneurons gate sensory transmission during locomotion to secure a fluid walking gait. *Neuron*, 96(6):1419–1431.e5, December 2017.

[153] Aaron T. Kuan, Jasper S. Phelps, Logan A. Thomas, Tri M. Nguyen, Julie Han, Chiao-Lin Chen, Anthony W. Azevedo, John C. Tuthill, Jan Funke, Peter Cloetens, Alexandra

Pacureanu, and Wei-Chung Allen Lee. Dense neuronal reconstruction through X-ray holographic nano-tomography. *Nature Neuroscience*, 23(12):1637–1643, 2020.

[154] R. Kukillaya, J. Proctor, and P. Holmes. Neuromechanical models for insect locomotion: stability, maneuverability, and proprioceptive feedback. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 19:026107, 2009.

[155] Arthur D Kuo. An optimal state estimation model of sensory integration in human postural balance. *Journal of Neural Engineering*, 2:S235–S249, 2005.

[156] Arthur D. Kuo and J. Maxwell Donelan. Dynamic Principles of Gait and Their Clinical Implications. *Physical Therapy*, 90(2):157–174, February 2010.

[157] Sen-Lin Lai and Tzumin Lee. Genetic mosaic with dual binary transcriptional systems in Drosophila. *Nature Neuroscience*, 9(5):703, May 2006.

[158] Ellen Lesser, Anthony W. Azevedo, Jasper S. Phelps, Leila Elabbady, Andrew Cook, Brandon Mark, Sumiya Kuroda, Anne Sustar, Anthony Moussa, Chris J. Dallmann, Sweta Agrawal, Su-Yee J. Lee, Brandon Pratt, Kyobi Skutt-Kakaria, Stephan Gerhard, Ran Lu, Nico Kemnitz, Kisuk Lee, Akhilesh Halageri, Manuel Castro, Dodam Ih, Jay Gager, Marwan Tammam, Sven Dorkenwald, Forrest Collman, Casey Schneider-Mizell, Derrick Brittain, Chris S. Jordan, H. Sebastian Seung, Thomas Macrina, Michael Dickinson, Wei-Chung Allen Lee, and John C. Tuthill. Synaptic architecture of leg and wing motor control networks in *Drosophila*. Preprint, Neuroscience, May 2023.

[159] Chou-Ching K Lin and Patrick E Crago. Neural and mechanical contributions to the stretch reflex: a model synthesis. *Annals of Biomedical Engineering*, 30:54–67, 2002.

[160] Chou-Ching K Lin and Patrick E Crago. Structural model of the muscle spindle. *Annals of Biomedical Engineering*, 30:68–83, 2002.

[161] Victor Lobato-Rios, Shravan Tata Ramalingasetty, Pembe Gizem Ozdil, Jonathan Arreguit, Auke Jan Ijspeert, and Pavan Ramdya. Neuromechfly, a neuromechanical model of adult drosophila melanogaster. *Nature Methods*, 19(5):620–627, 2022.

[162] Dominik Lorenz, Leonard Bereska, Timo Milbich, and Björn Ommer. Unsupervised part-based disentangling of object shape and appearance. In *CVPR*, 2019.

[163] Mark Lotkin. A Note on the Midpoint Method of Integration. *Journal of the ACM*, 3(3):208–211, July 1956.

[164] Gus K. Lott, Merri J. Rosen, and Ronald R. Hoy. An inexpensive sub-millisecond system for walking measurements of small animals based on optical computer mouse technology. *Journal of Neuroscience Methods*, 161(1):55–61, March 2007.

[165] Aloysius Y. T. Low, Ayesha R. Thanawalla, Alaric K. K. Yip, Jinsook Kim, Kelly L. L. Wong, Martesa Tantra, George J. Augustine, and Albert I. Chen. Precision of Discrete and Rhythmic Forelimb Movements Requires a Distinct Neuronal Subpopulation in the Interposed Anterior Nucleus. *Cell Reports*, 22(9):2322–2333, February 2018.

[166] Jenny Lu, Elena A. Westeinde, Lydia Hamburg, Paul M. Dawson, Cheng Lyu, Gaby Maimon, Shaul Druckmann, and Rachel I. Wilson. Transforming representations of movement from body- to world-centric space. *bioRxiv*, 2020.

[167] Cheng Lyu, L.F. Abbott, and Gaby Maimon. A neuronal circuit for vector computation builds an allocentric traveling-direction signal in the *Drosophila* fan-shaped body. *bioRxiv*, 2020.

[168] Vaughan G. Macefield and Thomas P. Knellwolf. Functional properties of human muscle spindles. *Journal of Neurophysiology*, 120:452–467, 2018.

[169] Ana S Machado, Dana M Darmohray, João Fayad, Hugo G Marques, and Megan R Carey. A quantitative framework for whole-body coordination reveals specific deficits in freely walking ataxic mice. *eLife*, 4:e07892, October 2015.

[170] Rodrigo S. Maeda, Shawn M. O'Connor, J. Maxwell Donelan, and Daniel S. Marigold. Foot placement relies on state estimation during visually guided walking. *Journal of Neurophysiology*, 117:480–491, 2017.

[171] P. Malik, Nuha Jabakhanji, and K. E. Jones. An assessment of six muscle spindle models for predicting sensory information during human wrist movements. *Frontiers in Computational Neuroscience*, 9:154, 2016.

[172] Mitchell G Maltenfort and R E Burke. Spindle model responsive to mixed fusimotor inputs and testable predictions of beta feedback effects. *Journal of Neurophysiology*, 89:2797–809, 2003.

[173] Akira Mamiya, Pralaksha Gurung, and John C. Tuthill. Neural Coding of Leg Proprioception in Drosophila. *Neuron*, 100(3):636–650.e6, 2018.

[174] Akira Mamiya, Anne Sustar, Igor Siwanowicz, Yanyan Qi, Tzu-Chiao Lu, Pralaksha Gurung, Chenghao Chen, Jasper S. Phelps, Aaron T. Kuan, Alexandra Pacureanu, Wei-Chung Allen Lee, Hongjie Li, Natasha Mhatre, and John C. Tuthill. Origins of proprioceptor feature selectivity and topographic maps in the Drosophila leg, May 2023.

[175] Jasper T. Maniates-Selvin, David Grant Colburn Hildebrand, Brett J. Graham, Aaron T. Kuan, Logan A. Thomas, Tri Nguyen, Julia Buhmann, Anthony W. Azevedo, Brendan L. Shanny, Jan Funke, John C. Tuthill, and Wei-Chung Allen Lee. Reconstruction of motor control circuits in adult Drosophila using automated transmission electron microscopy. *bioRxiv*, page 2020.01.10.902478, 2020.

[176] Sergey N. Markin, Alexander N. Klishko, Natalia A. Shevtsova, Michel A. Lemay, Boris I. Prilutsky, and Ilya A. Rybak. A neuromechanical model of spinal control of locomotion. In *Neuromechanical Modeling of Posture and Locomotion*, pages 21–65. Springer New York, New York, NY, 2016.

[177] Jesse D. Marshall, Diego E. Aldarondo, Timothy W. Dunn, William L. Wang, Gordon J. Berman, and Bence P. Ölveczky. Continuous whole-body 3d kinematic recordings across the rodent behavioral repertoire. *Neuron*, 109(3):420–437.e8, 2021.

[178] Jesse D. Marshall, Ugne Klibaite, Amanda Gellis, Diego E. Aldarondo, Bence P. Ölveczky, and Timothy W. Dunn. The PAIR-R24M Dataset for Multi-animal 3D Pose Estimation, November 2021.

[179] Jesse D Marshall, Tianqing Li, Joshua H Wu, and Timothy W Dunn. Leaving flatland: Advances in 3d behavioral measurement. *Current Opinion in Neurobiology*, 73:102522, 2022.

[180] James H. Marshel, Yoon Seok Kim, Timothy A. Machado, Sean Quirin, Brandon Benson, Jonathan Kadmon, Cephra Raja, Adelaida Chibukhchyan, Charu Ramakrishnan, Masatoshi Inoue, Janelle C. Shane, Douglas J. McKnight, Susumu Yoshizawa, Hideaki E. Kato, Surya Ganguli, and Karl Deisseroth. Cortical layer–specific critical dynamics triggering perception. *Science*, 365, 2019.

[181] Julian Marstaller, Frederic Tausch, and Simon Stock. Deepbees-building and scaling convolutional neuronal nets for fast and large-scale visual monitoring of bee hives. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[182] Julieta Martinez, Rayat Hossain, Javier Romero, and James J. Little. A simple yet effective baseline for 3d human pose estimation. In *ICCV*, 2017.

[183] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2640–2649, 2017.

[184] Alexander Mathis, Pranav Mamidanna, Kevin M. Cury, Taiga Abe, Venkatesh N.

Murthy, Mackenzie W. Mathis, and Matthias Bethge. Deeplabcut: markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*, 2018.

[185] Alexander Mathis, Pranav Mamidanna, Kevin M. Cury, Taiga Abe, Venkatesh N. Murthy, Mackenzie Weygandt Mathis, and Matthias Bethge. DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*, 21(9):1281, September 2018.

[186] Alexander Mathis, Steffen Schneider, Jessy Lauer, and Mackenzie Weygandt Mathis. A Primer on Motion Capture with Deep Learning: Principles, Pitfalls, and Perspectives. *Neuron*, 108(1):44–65, October 2020.

[187] Mackenzie Weygandt Mathis and Alexander Mathis. Deep learning tools for the measurement of animal behavior in neuroscience. *Current Opinion in Neurobiology*, 60:1–11, February 2020.

[188] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29):861, 2018.

[189] Daniel McNamee and Daniel M. Wolpert. Internal models in biological control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2:339–364, 2019.

[190] César S Mendes, Imre Bartos, Turgay Akay, Szabolcs Márka, and Richard S Mann. Quantification of gait parameters in freely walking wild type and sensory deprived Drosophila melanogaster. *eLife*, 2, January 2013.

[191] R. Chris Miall, Lars O. D Christensen, Owen Cain, and James Stanley. Disruption of state estimation in the human lateral cerebellum. *PLOS Biology*, 5:e316, 2007.

[192] Milana P Mileusnic, Ian E Brown, Ning Lan, and Gerald E Loeb. Mathematical models of proprioceptors. I. Control and transduction in the muscle spindle. *Journal of Neurophysiology*, 96:1772–88, 2006.

[193] Milana P. Mileusnic and Gerald E. Loeb. Mathematical models of proprioceptors. II. Structure and function of the Golgi tendon organ. *Journal of Neurophysiology*, 96:1789–1802, 2006.

[194] Stephen Miller, Jur van den Berg, Mario Fritz, Trevor Darrell, Ken Goldberg, and Pieter Abbeel. A geometric approach to robotic laundry folding. *The International Journal of Robotics Research*, 31(2):249–267, February 2012.

[195] Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee. V2v-posenet: Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation from a single depth map. In *Proceedings of the IEEE conference on computer vision and pattern Recognition*, pages 5079–5088, 2018.

[196] Richard J. D. Moore, Gavin J. Taylor, Angelique C. Paulk, Thomas Pearson, Bruno van Swinderen, and Mandyam V. Srinivasan. FicTrac: A visual method for tracking spherical motion and generating fictive animal paths. *Journal of Neuroscience Methods*, 225:106–119, March 2014.

[197] Heather L More and J Maxwell Donelan. Scaling of sensorimotor delays in terrestrial mammals. *Proceedings of the Royal Society B*, 285(1885):20180613, 2018.

[198] Lars Mündermann, Stefano Corazza, and Thomas P. Andriacchi. The evolution of methods for the capture of human movement leading to markerless motion capture for biomechanical applications. *Journal of NeuroEngineering and Rehabilitation*, 3(1):6, March 2006.

[199] Daniel Murphy. *Markerless 3D Pose Estimation from RGB Data.* Bachelor's thesis, Brown University, 2019.

[200] Hemal Naik, Alex Hoi Hang Chan, Junran Yang, Mathilde Delacoux, Iain D. Couzin, Fumihiro Kano, and Máté Nagy. 3D-POP – An automated annotation approach to

facilitate markerless 2D-3D tracking of freely moving birds with marker-based motion capture, March 2023.

[201] Tanmay Nath, Alexander Mathis, An Chi Chen, Amir Patel, Matthias Bethge, and Mackenzie Weygandt Mathis. Using DeepLabCut for 3d markerless pose estimation across species and behaviors. *Nature Protocols*, 14(7):2152–2176, 2019.

[202] R. R. Neptune and S. A. Kautz. Muscle activation and deactivation dynamics: The governing properties in fast cyclical human movement performance? *Exercise and sport sciences reviews*, 29(2):76–81, 2001.

[203] Izaak D. Neveln, Amoolya Tirumalai, and Simon Sponberg. Information-based centralization of locomotion in animals and robots. *Nature Communications*, 10(1):1–11, 2019.

[204] Jun Nishii. Legged insects select the optimal locomotor pattern based on the energetic cost. *Biological Cybernetics*, 83(5):435–442, 2000.

[205] Juan Carlos Núñez, Raúl Cabido, José F. Vélez, Antonio S. Montemayor, and Juan José Pantrigo. Multiview 3D human pose estimation using improved least-squares and LSTM networks. *Neurocomputing*, 323:335–343, January 2019.

[206] Katherine M Oliver, Danny M Florez-Paz, Tudor Constantin Badea, George Z Mentis, Vilas Menon, and Joriene C de Nooij. Molecular correlates of muscle spindle and Golgi tendon organ afferents. *Nature communications*, 12:1451, 2021.

[207] David S. Olton. Mazes, maps, and memory. *American psychologist*, 34(7):583, 1979.

[208] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7753–7762, 2019.

[209] Keir G Pearson. Proprioceptive regulation of locomotion. *Current Opinion in Neurobiology*, 5:786–791, 1995.

[210] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[211] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. DeepMimic: Example-Guided Deep Reinforcement Learning of Physics-Based Character Skills. *ACM Transactions on Graphics*, 37(4):1–14, July 2018.

[212] Talmo D. Pereira, Diego E. Aldarondo, Lindsay Willmore, Mikhail Kislin, Samuel S.-H. Wang, Mala Murthy, and Joshua W. Shaevitz. Fast animal pose estimation using deep neural networks. *Nature Methods*, 16(1):117, 2019.

[213] Talmo D Pereira, Joshua W Shaevitz, and Mala Murthy. Quantifying behavior to understand the brain. *Nature neuroscience*, 23(12):1537–1549, 2020.

[214] Talmo D. Pereira, Nathaniel Tabris, Junyu Li, Shruthi Ravindranath, Eleni S. Papadoyannis, Z. Yan Wang, David M. Turner, Grace McKenzie-Smith, Sarah D. Kocher, Annegret L. Falkner, Joshua W. Shaevitz, and Mala Murthy. SLEAP: Multi-animal pose tracking. *bioRxiv*, page 2020.08.31.276246, September 2020.

[215] Laurent U Perrinet, Rick A Adams, and Karl J Friston. Active inference, eye movements and oculomotor delays. *Biological cybernetics*, 108:777–801, 2014.

[216] Iliodora V Pop, Felipe Espinosa-Becerra, Megan Goyal, Bishakha Mona, Mark Landy, Osita Ogujiofor, Kevin M Dean, Channabasavaiah M Gurumurthy, and Helen C Lai. Evidence for genetically distinct direct and indirect spinocerebellar pathways mediating proprioception. *bioRxiv*, 2020.

[217] Boris I. Prilutsky, Alexander N. Klishko, Douglas J. Weber, and Michel A. Lemay. Computing motion dependent afferent activity during cat locomotion using a forward dynamics musculoskeletal model. In *Neuromechanical Modeling of Posture and Locomotion*, pages 273–307. Springer New York, New York, NY, 2016.

[218] A Prochazka and M Gorassini. Models of ensemble firing of muscle spindle afferents recorded during normal locomotion in cats. *The Journal of Physiology*, 507:277–91, 1998.

[219] Arthur Prochazka. Proprioceptive feedback and movement regulation. In *Handbook of Physiology. Exercise: Regulation and Integration of Multiple Systems*, pages 89–127. American Physiological Society, 1996.

[220] Arthur Prochazka. Quantifying proprioception. *Progress in Brain Research*, 123:133–142, 1999.

[221] Arthur Prochazka, D. Gillard, and D. J. Bennett. Implications of positive feedback in the control of movement. *Journal of Neurophysiology*, 77:3237–3251, 1997.

[222] J. L. Proctor and P. Holmes. The effects of feedback on stability and maneuverability of a phase-reduced model for cockroach locomotion. *Biological Cybernetics*, 112:387–401, 2018.

[223] J. L. Proctor and P. Holmes. The effects of feedback on stability and maneuverability of a phase-reduced model for cockroach locomotion. *Biological Cybernetics*, 112(4):387–401, 2018.

[224] Uwe Proske and Simon C. Gandevia. The proprioceptive senses: their roles in signaling body shape, body position and movement, and muscle force. *Physiological Reviews*, 92:1651–1697, 2012.

[225] Mir Rayat Imtiaz Hossain and James J Little. Exploiting temporal information for 3d human pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 68–84, 2018.

[226] Sasha Reschechtko and J. Andrew Pruszynski. Stretch reflexes. *Current Biology*, 30:R1025–R1030, 2020.

[227] Helge Rhodin, Jörg Spörri, Isinsu Katircioglu, Victor Constantin, Frédéric Meyer, Erich Müller, Mathieu Salzmann, and Pascal Fua. Learning monocular 3d human pose estimation from multi-view images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8437–8446, 2018.

[228] Andrew Richardson, Johannes Strom, and Edwin Olson. AprilCal: Assisted and repeatable camera calibration. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1814–1821, Tokyo, November 2013. IEEE.

[229] Nicole J. Rinehart, Mark A. Bellgrove, Bruce J. Tonge, Avril V. Brereton, Debra Howells-Rankin, and John L. Bradshaw. An Examination of Movement Kinematics in Young People with High-functioning Autism and Asperger's Disorder: Further Evidence for a Motor Planning Deficit. *Journal of Autism and Developmental Disorders*, 36(6):757–767, August 2006.

[230] Mary Roberts, David Mongeon, and Francois Prince. Biomechanical parameters for gait analysis: A systematic review of healthy human gait. *Physical Therapy and Rehabilitation*, 4(1):6, 2017.

[231] W J Roberts, N P Rosenthal, and C A Terzuolo. A control model of stretch reflex. *Journal of Neurophysiology*, 34:620–634, 1971.

[232] N P Rosenthal, T A McKean, W J Roberts, and C A Terzuolo. Frequency analysis of stretch reflex and its main subsystems in triceps surae muscles of the cat. *Journal of Neurophysiology*, 33:713–749, 1970.

[233] Hansol X. Ryu and Arthur D. Kuo. An optimality principle for locomotor central pattern generators. *bioRxiv*, 2020.

[234] Oliver Röhrle, Utku S. Yavuz, Thomas Klotz, Francesco Negro, and Thomas Heidlauf. Multiscale modeling of the neuromuscular system: coupling neurophysiology and skeletal muscle mechanics. *WIREs Systems Biology and Medicine*, 11:e1457, 2019.

[235] Kai J. Sandbrink, Pranav Mamidanna, Claudio Michaelis, Mackenzie Weygandt Mathis, Matthias Bethge, and Alexander Mathis. Task-driven hierarchical deep neural network models of the proprioceptive pathway. *bioRxiv*, 2020.

[236] Sanjay P. Sane and Matthew J. McHenry. The biomechanics of sensory organs. *Integrative and Comparative Biology*, 49:i8–i23, 2009.

[237] Nikolaos Sarafianos, Bogdan Boteanu, Bogdan Ionescu, and Ioannis A Kakadiaris. 3d human pose estimation: A review of the literature and analysis of covariates. *Computer Vision and Image Understanding*, 152:1–20, 2016.

[238] István Sárándi, Timm Linder, Kai O. Arras, and Bastian Leibe. Synthetic Occlusion Augmentation with Volumetric Heatmaps for the 2018 ECCV PoseTrack Challenge on 3D Human Pose Estimation. November 2018.

[239] Jonny L. Saunders and Michael Wehr. Autopilot: Automating behavioral experiments with lots of raspberry pis. *bioRxiv*, 2019.

[240] A Schaafsma, E Otten, and J D Van Willigen. A muscle spindle model for primary afferent firing based on a simulation of intrafusal mechanical events. *Journal of Neurophysiology*, 65:1297–312, 1991.

[241] Malte Schilling and Holk Cruse. Decentralized control of insect walking: a simple neural network explains a wide range of behavioral and neurophysiological results. *PLOS Computational Biology*, 16:e1007804, 2020.

[242] Malte Schilling, Thierry Hoinville, Josef Schmitz, and Holk Cruse. Walknet, a bio-inspired controller for hexapod walking. *Biological Cybernetics*, 107(4):397–419, August 2013.

[243] Malte Schilling, Jan Paskarbeit, Thierry Hoinville, Arne Hüffmeier, Axel Schneider, Josef Schmitz, and Holk Cruse. A hexapod walker using a heterarchical architecture for action selection. *Frontiers in Computational Neuroscience*, 7, 2013. Publisher: Frontiers.

[244] Johannes Schindelin, Ignacio Arganda-Carreras, Erwin Frise, Verena Kaynig, Mark Longair, Tobias Pietzsch, Stephan Preibisch, Curtis Rueden, Stephan Saalfeld, Benjamin Schmid, Jean-Yves Tinevez, Daniel James White, Volker Hartenstein, Kevin Eliceiri, Pavel Tomancak, and Albert Cardona. Fiji: an open-source platform for biological-image analysis. *Nature Methods*, 9(7):676–682, July 2012.

[245] Johannes L. Schönberger. *Robust Methods for Accurate and Efficient 3D Modeling from Unstructured Imagery*. Doctoral Thesis, ETH Zurich, 2018.

[246] Mac Schwager, Carrick Detweiler, Iuliu Vasilescu, Dean M. Anderson, and Daniela Rus. Data-driven identification of group dynamics for motion prediction and control. *Journal of Field Robotics*, 25(6-7):305–324, 2008.

[247] Johannes D. Seelig and Vivek Jayaraman. Neural dynamics for landmark orientation and angular path integration. *Nature*, 521:186–191, 2015.

[248] Nidhi Seethapathi, Shaofei Wang, Rachit Saluja, Gunnar Blohm, and Konrad P. Kording. Movement science needs different pose tracking algorithms. *arXiv:1907.10226 [cs, q-bio]*, 2019. arXiv: 1907.10226.

[249] Cristina Segalin, Jalani Williams, Tomomi Karigo, May Hui, Moriel Zelikowsky, Jennifer J. Sun, Pietro Perona, David J. Anderson, and Ann Kennedy. The mouse action

recognition system (mars): a software pipeline for automated analysis of social behaviors in mice. *bioRxiv https://doi.org/10.1101/2020.07.26.222299*, 2020.

[250] Ajay Seth, Jennifer L. Hicks, Thomas K. Uchida, Ayman Habib, Christopher L. Dembia, James J. Dunne, Carmichael F. Ong, Matthew S. DeMers, Apoorva Rajagopal, Matthew Millard, Samuel R. Hamner, Edith M. Arnold, Jennifer R. Yong, Shrinidhi K. Lakshmikanth, Michael A. Sherman, Joy P. Ku, and Scott L. Delp. OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement. *PLOS Computational Biology*, 14(7):e1006223, July 2018.

[251] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[252] C. Solaro, G. Brichetto, M. Casadio, L. Roccatagliata, P. Ruggiu, G.L. Mancardi, P.G. Morasso, P. Tanganelli, and V. Sanguineti. Subtle upper limb impairment in asymptomatic multiple sclerosis subjects. *Multiple Sclerosis Journal*, 13(3):428–432, April 2007.

[253] S. Song and H. Geyer. A neural circuitry that emphasizes spinal feedback generates diverse behaviours of human locomotion. *Journal of Physiology*, 593:3493–3511, 2015.

[254] Cristvo Duarte Sousa. Sympybotics v1.0, 2013.

[255] Richard B. Souza. An Evidence-Based Videotaped Running Biomechanics Analysis. *Physical Medicine and Rehabilitation Clinics of North America*, 27(1):217–236, February 2016.

[256] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[257] Josefin Stenberg, Jing Shuang Li, Anish A. Sarma, and John C. Doyle. Internal feedback in biological control: Diversity, delays, and standard theory. In *Proceedings of the IEEE American Control Conference*, pages 462–467, 2022.

[258] Peter Sterling and Simon B. Laughlin. *Principles of neural design*. MIT Press, 2015.

[259] Henning Stolze, Stephan Klebe, Christoph Baecker, Christiane Zechlin, Lars Friege, Sabine Pohle, and Günther Deuschl. Prevalence of gait disorders in hospitalized neurological patients. *Movement Disorders*, 20(1):89–94, 2005.

[260] R. Strauß and M. Heisenberg. Coordination of legs during straight walking and turning in Drosophila melanogaster. *Journal of Comparative Physiology A*, 167(3), August 1990.

[261] Steven H. Strogatz. From Kuramoto to Crawford: Exploring the onset of synchronization in populations of coupled oscillators. *Physica D: Nonlinear Phenomena*, 143(1-4):1–20, September 2000.

[262] Jennifer J. Sun, Lili Karashchuk, Amil Dravid, Serim Ryou, Sonia Fereidooni, John C. Tuthill, Aggelos Katsaggelos, Bingni W. Brunton, Georgia Gkioxari, and Ann Kennedy. BKinD-3D: Self-Supervised 3D Keypoint Discovery from Multi-View Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9001–9010, 2023.

[263] Jennifer J Sun, Tomomi Karigo, Dipam Chakraborty, Sharada P Mohanty, David J Anderson, Pietro Perona, Yisong Yue, and Ann Kennedy. The multi-agent behavior dataset: Mouse dyadic social interactions. *arXiv preprint arXiv:2104.02710*, 2021.

[264] Jennifer J Sun, Serim Ryou, Roni H Goldshmid, Brandon Weissbourd, John O Dabiri, David J Anderson, Ann Kennedy, Yisong Yue, and Pietro Perona. Self-supervised keypoint discovery in behavioral videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2171–2180, 2022.

[265] Xiao Sun, Bin Xiao, Fangyin Wei, Shuang Liang, and Yichen Wei. Integral human pose regression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 529–545, 2018.

[266] Xiyang Sun, Yingtao Liu, Chang Liu, Koichi Mayumi, Kohzo Ito, Akinao Nose, and Hiroshi Kohsaka. A neuromechanical model and kinematic analyses for *Drosophila* larval crawling based on physical measurements. *bioRxiv*, 2020.

[267] Nicholas S. Szczecinski, Till Bockemühl, Alexander S. Chockley, and Ansgar Büschges. Static stability predicts the continuum of interleg coordination patterns in Drosophila. *Journal of Experimental Biology*, 221(22):jeb189142, November 2018.

[268] Nicholas S. Szczecinski, Alexander J. Hunt, and Roger D. Quinn. A functional subnetwork approach to designing synthetic nervous systems that control legged robot locomotion. *Frontiers in Neurorobotics*, 11:37, 2017.

[269] Nick Szczecinski, Sasha Zill, Chris J Dallmann, and Roger Quinn. Modeling the dynamic sensory discharges of insect campaniform sensilla. In V. Vouloutsi, A. Mura, F. Tauber, T. Speck, T. J. Prescott, and P. F. M. J. Verschure, editors, *Biomimetic and Biohybrid Systems. Living Machines 2020. Lecture Notes in Computer Science*, volume 12413, pages 342–353. Springer, 2020.

[270] Aya Takeoka and Silvia Arber. Functional local proprioceptive feedback circuits initiate and maintain locomotor recovery after spinal cord injury. *Cell Reports*, 27:71–85.e3, 2019.

[271] William J. Tippett, Adam Krajewski, and Lauren E. Sergio. Visuomotor Integration Is Compromised in Alzheimer's Disease Patients Reaching for Remembered Targets. *European Neurology*, 58(1):1–11, 2007.

[272] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-

based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[273] Emanuel Todorov and Michael I. Jordan. Optimal feedback control as a theory of motor coordination. *Nature Neuroscience*, 5:1226–1235, 2002.

[274] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment — a modern synthesis. In Bill Triggs, Andrew Zisserman, and Richard Szeliski, editors, *Vision Algorithms: Theory and Practice*, volume 1883, pages 298–372. Springer Berlin Heidelberg, 2000.

[275] John C. Tuthill and Eiman Azim. Proprioception. *Current Biology*, 28:R194–R203, 2018.

[276] John C. Tuthill and Rachel I. Wilson. Mechanosensation and adaptive motor control in insects. *Current Biology*, 26:R1022–R1038, 2016.

[277] John C Tuthill and Rachel I Wilson. Parallel transformation of tactile signals in central circuits of drosophila. *Cell*, 164(5):1046–1059, 2016.

[278] Thomas M. Tzschentke. Review on CPP: Measuring reward with the conditioned place preference (CPP) paradigm: update of the last decade. *Addiction biology*, 12(3):227–462, 2007.

[279] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. DeMoN: Depth and Motion Network for Learning Monocular Stereo. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5622–5631, July 2017.

[280] Ben Usman, Andrea Tagliasacchi, Kate Saenko, and Avneesh Sud. Metapose: Fast 3d pose from multiple views without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6759–6770, 2022.

[281] Sydney K. Vaughan, Zachary Kemp, Theo Hatzipetros, Fernando Vieira, and Gregorio Valdez. Degeneration of proprioceptive sensory nerve endings in mice harboring amyotrophic lateral sclerosis–causing mutations. *Journal of Comparative Neurology*, 523(17):Spc1–Spc1, 2015.

[282] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[283] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C. J. Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, and Paul van Mulbregt. SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3):261–272, March 2020.

[284] Bastian Wandt, Marco Rudolph, Petrissa Zell, Helge Rhodin, and Bodo Rosenhahn. Canonpose: Self-supervised monocular 3d human pose estimation in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13294–13304, 2021.

[285] Jinbao Wang, Shujie Tan, Xiantong Zhen, Shuo Xu, Feng Zheng, Zhenyu He, and Ling Shao. Deep 3d human pose estimation: A review. *Computer Vision and Image Understanding*, 210:103225, 2021.

[286] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 13(4):600–612, 2004.

[287] Barbara Webb. Neural mechanisms for prediction: do insects have forward models? *Trends in Neurosciences*, 27:278–282, 2004.

[288] G. Weiland, U. Bässler, and M. Brunner. A biological feedback control system with electronic input: the artificially closed femur-tibia control system of stick insects. *Journal of Experimental Biology*, 120:369–385, 1986.

[289] Jeffrey Weiler, Paul L Gribble, and Andrew Pruszynski. Spinal stretch reflexes support efficient hand control. *Nature Neuroscience*, 22:529–533, 2019.

[290] Jeffrey Weiler, Paul L Gribble, and J. Andrew Pruszynski. Spinal stretch reflexes support efficient control of reaching. *bioRxiv*, 2020.

[291] K. White, E. Tahaoglu, and H. Steller. Cell killing by the Drosophila gene reaper. *Science (New York, N.Y.)*, 271(5250):805–807, February 1996.

[292] Ian Williams and Timothy G Constandinou. Computationally efficient modeling of proprioceptive signals in the upper limb for prostheses: a simulation study. *Frontiers in Neuroscience*, 8:181, 2014.

[293] Markus Windolf, Nils Götzen, and Michael Morlock. Systematic accuracy and precision analysis of video motion capturing systems exemplified on the Vicon-460 system. *Journal of Biomechanics*, 41(12):2776–2780, August 2008.

[294] Joanne E. Wittwer, Kate E. Webster, and Hylton B. Menz. A longitudinal study of measures of walking in people with Alzheimer's Disease. *Gait & Posture*, 32(1):113–117, May 2010.

[295] Steffen BE Wolff and Bence P Ölveczky. The promise and perils of causal circuit manipulations. *Current Opinion in Neurobiology*, 49:84–94, 2018.

[296] Daniel M Wolpert, Zoubin Ghahramani, and Michael I Jordan. Are arm trajectories planned in kinematic or dynamic coordinates? An adaptation study. *Experimental Brain Research*, 103:460–470, 1995.

[297] Anqi Wu, E. Kelly Buchanan, Matthew R. Whiteway, Michael Schartner, Guido Meijer, Jean-Paul Noel, Erica Rodriguez, Claire Everett, Amy Norovich, Evan Schaffer, Neeli Mishra, C. Daniel Salzman, Dora Angelaki, Andrés Bendesky, The International Brain Laboratory, John Cunningham, and Liam Paninski. Deep Graph Pose: A semi-supervised deep graphical model for improved animal pose tracking. *bioRxiv*, page 2020.08.20.259705, October 2020.

[298] Ming Wu, Aljoscha Nern, W Ryan Williamson, Mai M Morimoto, Michael B Reiser, Gwyneth M Card, and Gerald M Rubin. Visual projection neurons in the *Drosophila* lobula link feature detection to distinct behavioral programs. *Elife*, 5:e21022, 2016.

[299] Yuxin Wu et al. Tensorpack. `https://github.com/tensorpack/`, 2016.

[300] Shiting Xiao, Yufu Wang, Ammon Perkes, Bernd Pfrommer, Marc Schmidt, Kostas Daniilidis, and Marc Badger. Multi-view Tracking, Re-ID, and Social Network Analysis of a Flock of Visually Similar Birds in an Outdoor Aviary, November 2022.

[301] Wei Yang, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. End-to-End Learning of Deformable Mixture of Parts and Deep Convolutional Neural Networks for Human Pose Estimation. *CVPR*, pages 3073–3082, June 2016.

[302] Yuan Yao, Yasamin Jafarian, and Hyun Soo Park. MONET: Multiview semi-supervised keypoint detection via epipolar divergence. In *International Conference on Computer Vision (ICCV)*, 2019.

[303] Ryan A. York, Lisa M. Giocomo, and Thomas R. Clandinin. TREBLE: A generalizable framework for high-throughput behavioral analysis. *bioRxiv*, page 2020.09.30.321406, October 2020.

[304] H. Zhang, F. Mo, L. Wang, M. Behr, and P. Arnoux. A framework of a lower limb musculoskeletal model with implemented natural proprioceptive feedback and its progressive evaluation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 28:1866–1875, 2020.

[305] He Zhang, Sebastian Starke, Taku Komura, and Jun Saito. Mode-adaptive neural networks for quadruped motion control. *ACM Transactions on Graphics*, 37(4):145:1–145:11, July 2018.

[306] Libby Zhang, Tim Dunn, Jesse Marshall, Bence Olveczky, and Scott Linderman. Animal pose estimation from video data with a hierarchical von Mises-Fisher-Gaussian model. In *International Conference on Artificial Intelligence and Statistics*, pages 2800–2808. PMLR, March 2021.

[307] Yuting Zhang, Yijie Guo, Yixin Jin, Yijun Luo, Zhiyuan He, and Honglak Lee. Unsupervised discovery of object landmarks as structural representations. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2018.

[308] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[309] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. 2016.

[310] Xingyi Zhou, Qixing Huang, Xiao Sun, Xiangyang Xue, and Yichen Wei. Towards 3D Human Pose Estimation in the Wild: A Weakly-supervised Approach. *arXiv:1704.02447 [cs]*, July 2017.

[311] Sasha N. Zill and David T. Moran. The exoskeleton and insect proprioception. III. Activity of tibial campaniform sensilla during walking in the American cockroach, *Periplaneta americana*. *Journal of Experimental Biology*, 94:57–75, 1981.

[312] Christian Zimmermann, Artur Schneider, Mansour Alyahyay, Thomas Brox, and Ilka Diester. FreiPose: A Deep Learning Framework for Precise Animal Motion Capture in 3D Spaces. Preprint, Neuroscience, February 2020.

VITA

Lili Karashchuk was born in Russia to Ukrainian parents, moved to France at 6, and moved again to the US at 12. In high school, she learned of the beauty of statistics and computer science and started studying these in depth on her own time. She proceeded to obtain a B.A. in Statistics and Computer Science from UC Berkeley. Along the way, she grew enamored with the statistics of nervous systems, some of the most complex natural information processing systems. This led her to pursue a PhD in Neuroscience at the University of Washington. Outside of research, she enjoys painting watercolors, dancing, and hanging out with her friends, whom she deeply cherishes.