The Journal of Open Source Software

SeqIKPy: a Python package for inverse kinematics in insects

Pembe Gizem Özdil [1,2], **Sibo Wang-Chen** [1], **Chuanfang Ning**[2], **Auke Ijspeert** [2], **and Pavan Ramdya** [1]

1 Neuroengineering Laboratory, Brain Mind Institute, EPFL, Lausanne, Switzerland **2** Biorobotics Laboratory, Institute of Bioengineering, EPFL, Lausanne, Switzerland

DOI: 10.21105/joss.08557

Software

- Review
- Repository
- Archive

Editor: Olexandr Konovalov

Reviewers:

- @aceglia
- @davidpagnon

Submitted: 22 January 2025
Published: 21 January 2026

Summary

SeqIKPy is a Python package for inverse kinematics (IK) calculations in animal bodies with complex joint configurations. The name stands for Sequential Inverse Kinematics in Python, as our method computes joint angles sequentially by performing IK for each joint along a kinematic chain.

Our framework contains:

- Pose alignment: map tracked keypoint locations in 3D onto an animal body template
- Inverse kinematics: calculate joint angles sequentially from 3D poses
- Visualization: plot and animate the results in 3D

SeqIKPy is aimed at researchers studying detailed joint motion in animals with complex, multiple degrees of freedom body appendages. We provide examples for the fruit fly, *Drosophila melanogaster*. However, each module can easily be extended to be used with other model organisms; the only requirements are the 3D kinematics of the target animal and its corresponding kinematic chain. Our package requires minimal Python knowledge, and extensive tutorials are available to novice users at https://nely-epfl.github.io/sequential-inverse-kinematics.

Statement of need

Over the past decade, deep-learning–based computer vision algorithms have transformed behavioral analysis in laboratory animals (Pereira et al., 2020). More recently, deep-learning–based 3D pose estimation tools (Günel et al., 2019; Karashchuk et al., 2021) and detailed biomechanical models (Lobato-Rios et al., 2022; Vaxenburg et al., 2024; Wang-Chen et al., 2024) have created a growing need for tools that translate 3D kinematics into joint-angle representations. These joint angles can then be replayed in physics-based simulations to estimate unmeasured physical quantities such as joint torques (Lobato-Rios et al., 2022).

Inverse kinematics (IK) is widely used in robotics, biomechanics, and character animation (Aristidou et al., 2018). In robotics, IK typically computes joint angles to achieve a desired end-effector position while respecting joint constraints of a robot. In contrast, biomechanical IK aims to track multiple body markers simultaneously, a process often referred to as multi-body kinematics optimization and well-established in human biomechanics (Begon et al., 2018; Delp et al., 2007; Pagnon et al., 2022; Werling et al., 2023).

In insect research, however, multi-body kinematics optimization remains relatively underdeveloped. Existing approaches lie at two extremes. Simple geometric methods compute joint angles from dot products between adjacent body segments (Karashchuk et al., 2021;

Lobato-Rios et al., 2022), often leading to cumulative errors due to the lack of iterative corrections. More advanced gradient-based optimization methods estimate joint angles in a biomechanical model of the fly (Vaxenburg et al., 2024), using simulation-based Jacobians in physics engines like MuJoCo (Todorov et al., 2012). Although these methods provide higher accuracy, they are often entangled with pose estimation and physics engines, leading to complex dependencies and heavy overhead. Here, we introduce `SeqIKPy` as a lightweight, standalone, and modular solution focused on inverse kinematics.

`SeqIKPy` consists of two main stages: marker registration (aligning measured keypoints to a 3D body template) and inverse kinematics. The latter stage builds on the open-source IKPy library (Manceron, 2016) and applies it sequentially along the kinematic chain. The sequential nature of our method constrains the solution locally and mitigates ambiguities arising from joint redundancy. Using 3D visualizations, we show that SeqIKPy reliably reconstructs fly kinematics across a range of behaviors. Previous work has demonstrated that the joint angles computed with SeqIKPy accurately reproduce both walking (Wang-Chen et al., 2024) and grooming (Özdil et al., 2024) behaviors in physics-based simulations. Although our examples mostly focus on the fly, we demonstrate that its modular design allows adaptation to other animals with articulated limbs, including a mouse forelimb.

`SeqIKPy` can be used for animals and robots with arbitrarily configured kinematic chains consisting of rigid bodies connected with rotational joints. However, we have focused on the fruit fly *Drosophila melanogaster* in our demonstrations. Insects are some of the oldest model organisms in the study of motor control (Delcomyn, 2004), and *Drosophila melanogaster* is particularly prominent in neuroscience research due to its compact yet versatile nervous system. Recent advances have made the fly the most complex organism with a fully mapped central nervous system (e.g., Bates et al., 2025), leading to rapid growth in the field and an increased demand for open-source data-processing tools. With `SeqIKPy`, we aim to address a critical gap in the behavioral analysis pipeline.

Overview

`SeqIKPy` assumes that the 3D pose estimation has the following orientation (Figure 1, left):

- x-axis: anteroposterior axis
- y-axis: mediolateral axis
- z-axis: dorsoventral axis

After setting this orientation, users can use the `AlignPose` class to map body keypoints to a template body model (Figure 1, middle). This step is also known as "calibration" in pose estimation literature. Despite being optional for inverse kinematics computation, this step has two benefits:

- It aligns measured kinematics to a standardized body template, facilitating replay of behaviors in body models (Figure 1, right).
- It reduces noise and variation in kinematics by standardizing body lengths.

We provide a default body template based on a CT scan of the fly (Lobato-Rios et al., 2022). Users can also define custom templates by importing SDF files using the `from_sdf` functionality, which extracts joint locations directly from the model definition, or by specifying templates manually. Utility functions are included to convert data into the required formats.

Next, the `KinematicChainSeq` class defines a pre-configured kinematic chain for fly legs. Users need a dictionary containing segment lengths and joint bounds (Figure 1, middle). Segment lengths can be derived from 3D kinematics, while joint bounds are optional. Using the defined kinematic chain, the `LegInvKin` class calculates joint angles sequentially for each leg. This process supports parallelization to perform IK on multiple legs simultaneously. Additionally, our package contains features for animation and visualization of data in 3D. For

more technical details on the implementation, please refer to the methodology section at https://nely-epfl.github.io/sequential-inverse-kinematics.

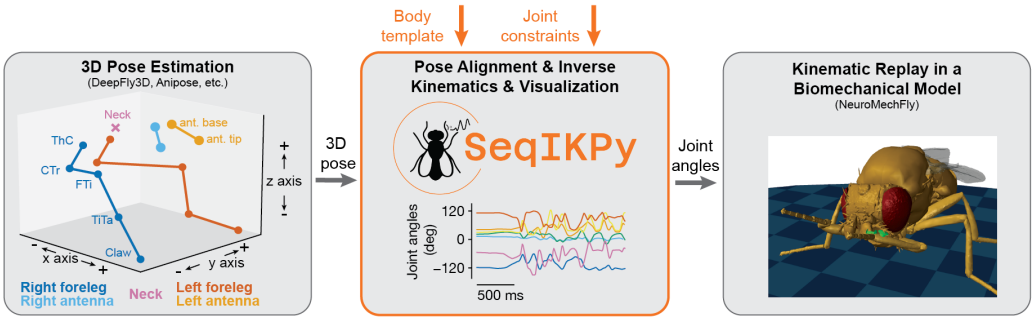

Figure 1: Overview of the SeqIKPy pipeline. **(Left)** Pose estimation tools (e.g., DeepFly3D and Anipose) estimates 3D positions of body keypoints. **(Middle)** SeqIKPy aligns these points to a body template, performs sequential inverse kinematics using joint constraints to calculate joint angles. **(Right)** The computed joint angles can be used to replay measured motions in biomechanical models, such as NeuroMechFly.

Limitations and mitigation

Like `IKPy`, `SeqIKPy` solves inverse kinematics through per-frame optimization. While this approach generally leads to a more faithful fit between raw data and the inferred kinematics, a main disadvantage is that processing can be slow. For use cases requiring higher throughput but not requiring sequential fitting over the whole kinematic chain, we refer the user to the `ik` module (Dauphin, 2017) from the Aversive++ project. Alternatively, inverse kinematics can be implemented using approaches that do not require explicit optimization loops for every frame. These include methods based on Extended Kalman Filters (e.g., Bonnet et al., 2017; Ceglia et al., 2025; Fohanno et al., 2010), and methods based on deep neural networks (e.g., Toquica et al., 2021; Wang et al., 2021). These methods can also implicitly solve for the angles of all degrees of freedom on each kinematic chain simultaneously, instead of running a sequential optimization. At the cost of losing explicit per-frame accuracy, these methods are much faster and, in many cases, can run in real-time.

For use cases where sequential inverse kinematics is required, we have implemented parallel processing in `SeqIKPy` in order to improve the throughput. Parallelism is implemented over different kinematic chains and over time, with the size of each atomic task determined adaptively to balance the trade-off between load balancing and overhead. On a typical machine, the overall processing rate scales near-linearly up to the number of physical CPU cores (~90% efficiency) and, to a lesser extent, up to the number of logical threads (~80% efficiency). On an Intel Xeon Platinum 8360Y processor, this translates to ~2ms per frame with 36 processes.

Acknowledgements

We acknowledge the contributors and maintainers of the open-source software tools that `SeqIKPy` builds upon, including Python, NumPy, SciPy, Matplotlib, and IKPy (Manceron, 2016), among others. PGÖ acknowledges support from a Swiss Government Excellence Scholarship for Doctoral Studies and a Google PhD Fellowship. SWC acknowledges support from a Boehringer Ingelheim Fonds PhD fellowship. PR acknowledges support from an SNSF Project Grant (175667) and an SNSF Eccellenza Grant (181239).

References

Aristidou, A., Lasenby, J., Chrysanthou, Y., & Shamir, A. (2018). Inverse kinematics techniques in computer graphics: A survey. *Computer Graphics Forum*, *37*, 35–58. https://doi.org/10.1111/cgf.13310

Bates, A. S., Phelps, J. S., Kim, M., Yang, H. H., Matsliah, A., Ajabi, Z., Perlman, E., Delgado, K. M., Osman, M. A. M., Salmon, C. K., Gager, J., Silverman, B., Renauld, S., Collie, M. F., Fan, J., Pacheco, D. A., Zhao, Y., Patel, J., Zhang, W., … Lee, W.-C. A. (2025). *Distributed control circuits across a brain-and-cord connectome*. https://doi.org/10.1101/2025.07.31.667571

Begon, M., Andersen, M. S., & Dumas, R. (2018). Multibody kinematics optimization for the estimation of upper and lower limb human joint kinematics: A systematized methodological review. *Journal of Biomechanical Engineering*, *140*(3), 030801. https://doi.org/10.1115/1.4038741

Bonnet, V., Dumas, R., Cappozzo, A., Joukov, V., Daune, G., Kulić, D., Fraisse, P., Andary, S., & Venture, G. (2017). A constrained extended Kalman filter for the optimal estimate of kinematics and kinetics of a sagittal symmetric exercise. *Journal of Biomechanics*, *62*, 140–147. https://doi.org/10.1016/j.jbiomech.2016.12.027

Ceglia, A., Facon, K., Begon, M., & Seoud, L. (2025). Real-time, accurate, and open source upper-limb musculoskeletal analysis using a single RGBD camera — an exploratory hand-cycling study. *Computers in Biology and Medicine*, *184*, 109434. https://doi.org/10.1016/j.compbiomed.2024.109434

Dauphin, L. (2017). *Aversive++* (Version v17.02). GitHub. https://github.com/AversivePlusPlus/AversivePlusPlus

Delcomyn, F. (2004). Insect walking and robotics. *Annual Review of Entomology*, *49*(1), 51–70. https://doi.org/10.1146/annurev.ento.49.061802.123257

Delp, S. L., Anderson, F. C., Arnold, A. S., Loan, P., Habib, A., John, C. T., Guendelman, E., & Thelen, D. G. (2007). OpenSim: Open-source software to create and analyze dynamic simulations of movement. *IEEE Transactions on Biomedical Engineering*, *54*(11), 1940–1950. https://doi.org/10.1109/TBME.2007.901024

Fohanno, V., Colloud, F., Begon, M., & Lacouture, P. (2010). Estimation of the 3D kinematics in kayak using an extended Kalman filter algorithm: A pilot study. *Computer Methods in Biomechanics and Biomedical Engineering*, *13*(sup1), 55–56. https://doi.org/10.1080/10255842.2010.491958

Günel, S., Rhodin, H., Morales, D., Campagnolo, J., Ramdya, P., & Fua, P. (2019). DeepFly3D, a deep learning-based approach for 3D limb and appendage tracking in tethered, adult Drosophila. *eLife*, *8*, e48571. https://doi.org/10.7554/eLife.48571

Karashchuk, P., Rupp, K. L., Dickinson, E. S., Walling-Bell, S., Sanders, E., Azim, E., Brunton, B. W., & Tuthill, J. C. (2021). Anipose: A toolkit for robust markerless 3D pose estimation. *Cell Reports*, *36*(13), 109730. https://doi.org/10.1016/j.celrep.2021.109730

Lobato-Rios, V., Ramalingasetty, S. T., Özdil, P. G., Arreguit, J., Ijspeert, A. J., & Ramdya, P. (2022). NeuroMechFly, a neuromechanical model of adult Drosophila melanogaster. *Nature Methods*, *19*(5), 620–627. https://doi.org/10.1038/s41592-022-01466-7

Manceron, P. (2016). *IKPy* (Version v3.4.2). GitHub. https://doi.org/10.5281/zenodo.6551105

Özdil, P. G., Arreguit, J., Scherrer, C., Ijspeert, A., & Ramdya, P. (2024). Centralized brain networks underlie body part coordination during grooming. *bioRxiv*. https://doi.org/10.1101/2024.12.17.628844

Pagnon, D., Domalain, M., & Reveret, L. (2022). Pose2Sim: An open-source Python package for multiview markerless kinematics. *Journal of Open Source Software*, *7*(77), 4362. https://doi.org/10.21105/joss.04362

Pereira, T. D., Shaevitz, J. W., & Murthy, M. (2020). Quantifying behavior to understand the brain. *Nature Neuroscience*, *23*(12), 1537–1549. https://doi.org/10.1038/s41593-020-00734-z

Todorov, E., Erez, T., & Tassa, Y. (2012). Mujoco: A physics engine for model-based control. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 5026–5033. https://doi.org/10.1109/IROS.2012.6386109

Toquica, J. S., Oliveira, P. S., Souza, W. S. R., Motta, J. M. S. T., & Borges, D. L. (2021). An analytical and a deep learning model for solving the inverse kinematic problem of an industrial parallel robot. *Computers &Amp; Industrial Engineering*, *151*, 106682. https://doi.org/10.1016/j.cie.2020.106682

Vaxenburg, R., Siwanowicz, I., Merel, J., Robie, A. A., Morrow, C., Novati, G., Stefanidi, Z., Card, G. M., Reiser, M. B., Botvinick, M. M., Branson, K. M., Tassa, Y., & Turaga, S. C. (2024). *Whole-body simulation of realistic fruit fly locomotion with deep reinforcement learning*. https://doi.org/10.1101/2024.03.11.584515

Wang, X., Liu, X., Chen, L., & Hu, H. (2021). Deep-learning damped least squares method for inverse kinematics of redundant robots. *Measurement*, *171*, 108821. https://doi.org/10.1016/j.measurement.2020.108821

Wang-Chen, S., Stimpfling, V. A., Lam, T. K. C., Özdil, P. G., Genoud, L., Hurtak, F., & Ramdya, P. (2024). NeuroMechFly v2: Simulating embodied sensorimotor control in adult drosophila. *Nature Methods*, 1–10. https://doi.org/10.1038/s41592-024-02497-y

Werling, K., Bianco, N. A., Raitor, M., Stingel, J., Hicks, J. L., Collins, S. H., Delp, S. L., & Liu, C. K. (2023). AddBiomechanics: Automating model scaling, inverse kinematics, and inverse dynamics from human motion data through sequential optimization. *Plos One*, *18*(11), e0295152. https://doi.org/10.1371/journal.pone.0295152