Animal Pose Labeling Using General-Purpose Point Trackers

Zhuoyang Pan[†,‡] Boxiao Pan[†] Guandao Yang[†] Adam W. Harley[†] Leonidas Guibas[†]

[†]Stanford University [‡]ShanghaiTech University

Abstract

Automatically estimating animal poses from videos is important for studying animal behaviors. Existing methods do not perform reliably since they are trained on datasets that are not comprehensive enough to capture all necessary animal behaviors. However, it is very challenging to collect such datasets due to the large variations in animal morphology. In this paper, we propose an animal pose labeling pipeline that follows a different strategy, i.e. *test time optimization. Given a video, we fine-tune a lightweight appearance embedding inside a pre-trained general-purpose point tracker on a sparse set of annotated frames. These annotations can be obtained from human labelers or off-the-shelf pose detectors. The fine-tuned model is then applied to the rest of the frames for automatic labeling. Our method achieves state-of-the-art performance at a reasonable annotation cost. We believe our pipeline offers a valuable tool for the automatic quantification of animal behavior. Visit our project webpage at* https://zhuoyang-pan.github.io/animal-labeling.

1. Introduction

Accurate quantification of animal behavior is essential to understanding their brain by connecting it to the studied subjects' neural activities [9]. Such quantification requires precise detection and tracking of animal poses, preferably in a markerless fashion to avoid intrusiveness. This necessitates the development of automatic pose detectors and trackers that are applicable to in-the-wild videos. Despite significant progress in human pose estimation [2, 3, 14], accurately estimating animal poses remains challenging due to the large variations in body morphology. These variations make it hard to collect comprehensive datasets and thus train generalizable models.

Previous works on animal pose estimation, therefore, focus primarily on specific animal species [4, 8–11]. These works build automated tools that generally incorporate a pose detector [9–11] and sometimes also an object tracker [4, 8]. Users first need to manually label a set of frames (normally several hundreds). Then the models are trained on this set and can be subsequently deployed on new instances from the same animal species. Such pipelines need to be carried out for each animal species of interest, and are not applicable to cases where multiple animal species are present. Other works train a single foundation model on datasets containing multiple animal species to achieve cross-species generalization [14, 16]. They, however, manifest unsatisfying generalization performance.

We argue that instead of following this traditional paradigm and attempting to achieve generalized performance across animal species and scenarios, it is more desirable and practical to adopt a *test time optimization* strategy. Specifically, we propose to train a model on each test instance. The model is trained on a sparse set of annotated frames from the input video, which can be obtained from manual annotation or a pre-trained pose detector [14]. The trained model is then applied to the rest of the frames. We build our model based on the crucial insight that we can share the knowledge of temporal tracking across all instances, while the unique knowledge we need in each example lies in appearance. Hence, we initialize our model from a state-of-the-art general-purpose point tracker [7] and only fine-tune a lightweight *appearance embedding* in it which encodes the appearance-specific information. With this strategy, our method achieves a 4-pixel accuracy of 81.6% when supervised with only 6 annotated frames for a video of 60 frames, and the optimization converges in about 3mins[1]. These values can be traded off depending on a preference for low annotation cost or high estimation quality.

Since our method does not rely on any particular assumption about animal species or morphology, it can be directly applied to any video of interest. We test our method on two datasets from different domains, namely DAVIS-Animals for quadruped animals and DeepFly3D [5] for tethered drosophilas. Our method produces high-quality per-frame annotations while achieving state-of-the-art performance on both datasets.

In summary, we propose a framework for dense animal pose annotation that follows a *test time optimization* paradigm. We initialize the model with a general-purpose point tracker, and fine-tune a lightweight appearance embedding for each test instance. Our method can annotate cross-species animal videos at a high quality and can be scaled up at a reasonable cost. We evaluate our method

[1]Correct when error is within 4 pixels. Averaged values over 15 videos.

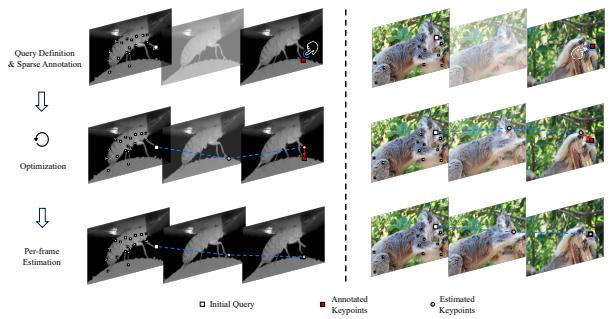

Figure 1. **System overview.** Our pipeline consists of three stages. First, users define query keypoints and provide sparse annotations. Our model is then optimized w.r.t. these annotations. Finally, the optimized model is applied to the remaining frames for dense pose labeling. We show two examples from the DeepFly3D (left) and DAVIS-Animals (right) datasets.

on datasets covering different animal species and scenarios, and show that it produces state-of-the-art results. Check the project webpage for video results.

2. Method

An overview of our pipeline is shown in Fig. 1. Our pipeline takes an input video sequence $\{\mathcal{I}_t\}_{t=1}^T$, query points $\{(\mathbf{x}_0^q, \mathbf{y}_0^q)\}$ on the first frame that define the keypoints to track, and a sparse set of annotated future keypoint positions $\{(\mathbf{x}_t^a, \mathbf{y}_t^a)\}_{t\in T_a}$, where T_a represents the time steps at which the keypoints are annotated. The goal is to predict the keypoint positions in the remaining frames by leveraging general-purpose point trackers. We next provide essential background on a state-of-the-art point tracker CoTracker3 [6], upon which we build our pipeline. After that, we describe our test time optimization strategy in detail.

2.1. Preliminaries: CoTracker3

CoTracker3 [7] takes a video sequence $\{\mathcal{I}_t\}_{t=1}^T$ and query points $\{(\mathbf{x}_t^a, \mathbf{y}_t^a)\}$ as input to generate estimated point positions in each frame.

Feature encoding. CoTracker3 first computes dense feature maps with a convolutional neural network for each video frame, *i.e.* $\Phi_t = \Phi(\mathcal{I}_t), t = 1, ..., T$. The network downsamples the input video by a spatial factor $k = 4$ and computes feature maps at $S = 4$ different scales, *i.e.* $\Phi_t^s \in \mathbb{R}^{d\times \frac{H}{k2^{s-1}}\times \frac{W}{k2^{s-1}}}, \quad s = 1, \ldots, S$.

Tracking features. Points in CoTracker3 are described by extracting a square neighborhood with size $(2\Delta + 1)^2$ of features at different scales, *i.e.* $\phi_t^s = \left[\Phi_t^s\left(\frac{\mathbf{x}_t}{ks} + \delta, \frac{\mathbf{y}_t}{ks} + \delta\right) : \delta \in \mathbb{Z}, \|\delta\|_\infty \leq \Delta\right], s = 1, \ldots, S$. $\Phi_t^s(\mathbf{x}_t, \mathbf{y}_t)$ denotes bineanly interpolating Φ_t^s around $(\mathbf{x}_t, \mathbf{y}_t)$, which can be either the query points or the current track estimates.

Iterative updates. Inference is carried out as an iterative update process. The track estimates are first initialized from the positions of the query points. Each iteration begins by measuring the similarity between tracking features for current estimates ϕ_t^s and query points ϕ_0^s, represented as correlation features $\text{Corr}_t = \text{MLP}\left(\langle\phi_0^s, \phi_t^s\rangle\right), s = 1, \ldots, S$, where $\langle\phi_0^s, \phi_t^s\rangle = \text{stack}((\phi_0^s)^\intercal \phi_t^s)$. These correlation features along with other information are subsequently passed into a transformer to predict the position deltas. We refer to the original paper for all details.

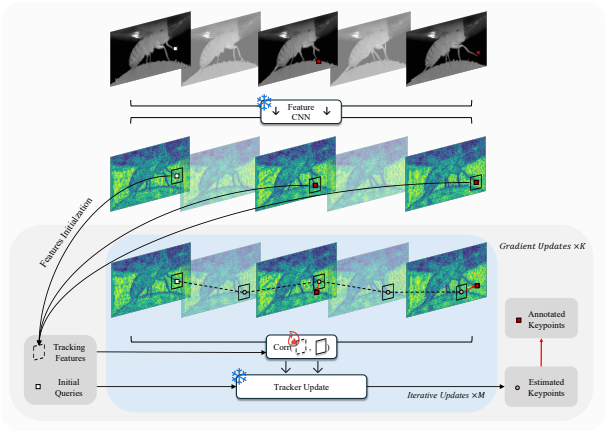

Figure 2. **Test time optimization.**

Discussions. We find in our experiments that CoTracker3 does not perform reliably when applied out of the box to animal videos, possibly due to the scarcity of high-quality annotations across multiple animal species. It often produces stationary or highly jittery point estimates because it confuses the keypoints to track with points on background or other objects, which makes it unsuitable to be used in animal behavioral studies. In the following, we propose a test time optimization strategy that finetunes CoTracker3 on each test instance. This strategy greatly improves the pose detection performance at a manageable cost.

2.2. Test Time Optimization

Our key idea is to optimize the tracking features for each query point $\hat{\phi}_0$ on a single video while keeping other components fixed. These optimized features essentially serve as an "appearance embedding" that encodes video-specific appearance information, hence greatly helping improve tracking performance. We describe this process in detail below, which is also illustrated in Fig. 2.

Feature initialization. Instead of initializing the query features only from the square neighborhood of the first frame's feature map ϕ_0, we choose to initialize with the square neighborhood of features from all the annotated frames $t \in T_a$:

$$\hat{\phi}_0 := \frac{1}{|T_a| + 1}\left(\phi_0^1 + \sum_{t\in T_a}\phi_t^1\right) \tag{1}$$

Iterative updates. The updated correlation features are calculated by

$$\widehat{\text{Corr}}_t = \left(\text{MLP}\left(\langle\hat{\phi}_0, \phi_t^1\rangle\right), \ldots, \text{MLP}\left(\langle\hat{\phi}_0, \phi_t^S\rangle\right)\right) \quad (2)$$

We then repeat the iterative update process as done in the original CoTracker3, using the updated correlation features.

Loss function. Our loss function consists of a primary tracking loss followed by a regularization term. For the tracking loss, we follow CoTracker3 [7] to supervise both the visible and occluded tracks using the Huber loss with a threshold of 6 and exponentially increasing weights:

$$\mathcal{L}_{\text{track}}(\mathcal{P}, \mathcal{P}^\star) = \sum_{m=1}^{M} \gamma^{M-m} L_H(\mathcal{P}, \mathcal{P}^\star) \quad (3)$$

where $\mathcal{P}$ and $\mathcal{P}^\star$ denote the annotated and estimated keypoints, respectively. M is the number of update iterations, $\gamma = 0.8$ is a discount factor, and L_H is the Huber Loss.

To ensure the tracking features remain close to the original sampled features, we apply an additional regularization term that penalizes significant deviations:

$$\mathcal{L}_{\text{reg}} = \frac{1}{N} \sum_{i=1}^{N} \|\phi_0^{1i} - \hat{\phi}_0^i\|_1 \quad (4)$$

where N is the number of keypoints. Our final loss is thus:

$$\mathcal{L} = \mathcal{L}_{\text{track}} + \lambda \mathcal{L}_{\text{reg}} \quad (5)$$

We use $\lambda = 0.01$ in all experiments.

Optimization details. For each video, we optimize for 1,000 gradient update steps, starting with a learning rate of 1×10^{-3} that linearly decays to 1×10^{-5} using the Adam optimizer. For a 100-frame 480p video, the optimization converges in about 3 minutes on an NVIDIA RTX A6000.

3. Experiments

Datasets. Animal videos with dense high-quality pose annotations are scarce. APT36k [15] consists of 36k annotated frames from 2.4k videos of 30 animal species, but the keypoint annotations are very jittery. BADJA [1] samples 7 animal videos from the DAVIS [12] dataset along with 2 other videos and provides high-quality dense annotations. However, the keypoint definition of BADJA differs from that of ViTPose [14], which is an important baseline to compare. Hence, we sample 15 videos from DAVIS and manually annotate them, following the keypoint definition of ViTPose. We refer to this dataset as *DAVIS-Animals*.

To demonstrate that our pipeline can be directly applied to videos of different animal species, we further evaluate our method on *DeepFly3D* [5], a dataset specifically for tethered drosophilas. We sample 15 videos from the dataset, each for 100 frames with 19 annotated keypoints.

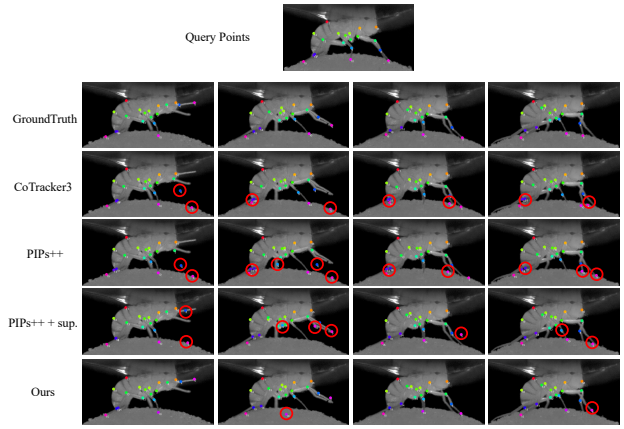

Figure 3. **Qualitative comparison on DeepFly3D.** Query points are shown in the frame on top, while estimated points in other frames. Keypoints with the same color / number denote correspondence. Red circles highlight estimation mistakes.

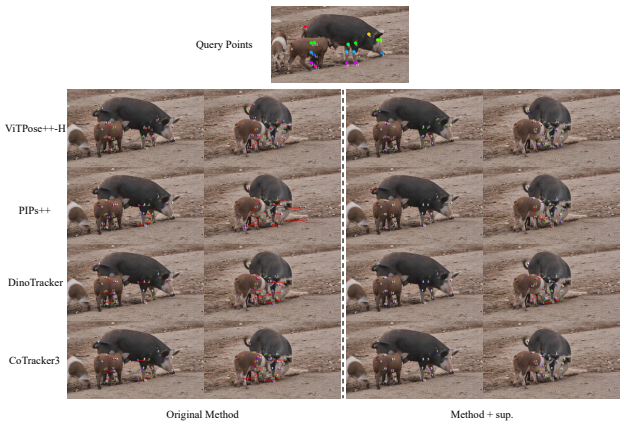

Figure 4. **Qualitative comparison on DAVIS-Animals.** Query points are color-coded in the frame on top. In other frames, estimated points are shown in blue, while ground-truth points are in green. Red lines indicate tracking errors relative to the corresponding ground truth positions. Note that CoTracker3 + sup. (bottom right) is our adapted version.

Metrics. We follow SuperAnimal [16] to report 1) δ_{avg}, which measures the average position accuracy of all points across 5 thresholds: 1, 2, 4, 8, and 16 pixels. Following [17], images are resized to 256 × 256 pixels before calculation; 2) a jittering metric J, which is the average of the unsigned speed across all examples and keypoints. For a given keypoint k and example e, $J_{k,e}$ is computed as:

$$J_{k,e} = \frac{1}{N_{k,e}} \sum_{i=1}^{N_{k,e}} |v_{k,e,i}| \quad (6)$$

where $N_{k,e}$ is the total number of speed measurements

for keypoint k in example e. 3) a masked jittering metric J_{masked}, which enhances the jittering metric by focusing on correctly localized keypoints while penalizing incorrect ones. The metric is computed as:

$$J_{\text{masked},k,e} = \frac{1}{N_{k,e}} \sum_{i=1}^{N_{k,e}} v_{k,e,i} \cdot \begin{cases} 1, & d_{k,e,i} < 4 \\ 10, & d_{k,e,i} \geq 4 \end{cases} \quad (7)$$

where $d_{k,e,i}$ is the distance to the ground truth.

Baselines. We compare against state-of-the-art general-purpose point trackers (*PIPS++* [17], *DINO-Tracker* [13], and *CoTracker3* [7]), a cross-species pose estimator developed specifically for quadrupeds (*SuperAnimal* [16]), as well as a general-purpose pose estimator applicable for both humans and animals(*ViTPose* [14]). For a fair comparison, we also fine-tune them using the same strategy as our method (*+sup.*). We next describe each in more detail.

PIPs++ (+sup.) Similar to CoTracker3 [7], *PIPs++* [17] computes the cross correlation between the features at current timestep and that at previous timestep. For *PIPS++ +sup.*, we instead compute the correlations between the current features and an optimizable features.

DINO-Tracker (+sup.) For each video, DINO-Tracker [13] fine-tunes a Delta-DINO encoder and a CNN-refiner through optical flows and DINO-feature correspondence. At inference time, they calculate the similarity between the refined DINO features sampled at the query point. Since this method is self-supervised, for a fair comparison, we add our tracking loss in addition to their original loss designs and finetune their original networks.

ViTPose++-H (+sup.) ViTPose++-H [14] is a pose estimation model trained on multiple humans and animals dataset. It is the SOTA model for animal pose estimation. We fine-tune this model on the annotated frames and test the performance of the fine-tuned model on the testing frames.

SuperAnimal (+sup.) We use the SuperAnimal-Quadruped model provided by Superanimal [16], which is trained on multiple quadruped datasets. We enable their video-adaption option, which is another test time optimization technique they propose to improve estimation accuracy and mitigate jittering. They first obtain the pseudolabels every 10 frames by estimating the poses, and then use these estimated poses to finetune the estimator. We change their pseudolabels to the ground truth annotations and use these ground truth labels to finetune the estimator.

3.1. Quantitative Comparison

For this experiment, we train our method and baselines with additional supervision on frames 0, 10, ..., and test on frames 5, 15, ... The results are provided in Tab. 1. Our method achieves the highest δ_{avg} score on both datasets, surpassing baselines that are applied out of the box or fine-tuned with test time optimization (*+sup.*). For the jittering metric, our method performs the best compared to baselines fine-tuned with test time optimization, but is worse than PIPs++ and CoTracker3. This is because PIPs++ and CoTracker3 produce estimates that confuse with the background more often, which tend to move less and result in a deceptively low jittering score (*e.g.* keypoints 4 and 5 in Fig. 3). To better evaluate the tracking quality, we use the masked jittering metric J_{masked}, which focuses on correctly localized keypoints while penalizing incorrect ones. As shown in Tab. 1, our method achieves the lowest J_{masked} score across all methods.

Method	DeepFly3D			DAVIS-Animals		
	$\delta_{avg} \uparrow$	$J \downarrow$	$J_{masked} \downarrow$	$\delta_{avg} \uparrow$	$J \downarrow$	$J_{masked} \downarrow$
SuperAnimal [16]	—	—	—	41.07	15.19	10.90
ViTPose++-H [14]	—	—	—	52.56	9.52	9.22
PIPs++ [17]	45.91	**0.55**	5.32	46.19	**4.41**	7.67
DINO-Tracker [13]	48.51	1.95	5.40	49.09	8.41	8.61
CoTracker3 [7]	50.26	0.60	4.92	49.59	6.26	8.03
SuperAnimal +sup.	—	—	—	57.54	10.36	9.19
ViTPose++-H +sup.	—	—	—	62.17	8.21	7.94
PIPs++ +sup.	58.89	2.14	3.72	51.17	8.20	8.13
DINO-Tracker+sup.	67.29	2.91	3.39	63.33	9.11	8.67
Ours	**70.80**	**1.46**	**2.54**	**67.53**	**7.04**	**7.15**

Table 1. Quantitative comparison with baselines.

3.2. Qualitative Comparison

We compare qualitatively with selected baselines in Figs. 3 and 4. In Fig. 3, both CoTracker3 [7] and PIPs++ [17] have obvious errors on keypoints on legs. For PIPs++, tracking is noticeably more accurate after test time optimization but errors still occur frequently. Comparatively, our method is able to track the keypoints most reliably and produce the least errors. In Fig. 4 we observe similar trends. For all methods, fine-tuning with test time optimization noticeably reduces tracking errors. Among all methods, our method is the most accurate, also producing the least jitter.

4. Conclusion

We propose an animal pose labeling pipeline that features high-quality pose labeling while at a reasonable cost for manual annotation. Our key idea is to adopt a test time optimization strategy to fine-tune a pretrained model with sparse annotations on the example we wish to annotate. By leveraging a general-purpose point tracker, we inherit valuable knowledge of temporal tracking and only fine-tune a lightweight appearance embedding that encodes example-specific appearance information. Our method achieves state-of-the-art pose estimation performance across different animal species, offering a valuable tool for accurate animal behavior quantification.

Acknowledgements. This work is supported in part by a Vannevar Bush Faculty Fellowship and by an ARL grant W911NF-21-2-0104.

References

[1] Benjamin Biggs, Thomas Roddick, Andrew Fitzgibbon, and Roberto Cipolla. Creatures great and SMAL: Recovering the shape and motion of animals from video. In *ACCV*, 2018. 3

[2] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 1

[3] Shubham Goel, Georgios Pavlakos, Jathushan Rajasegaran, Angjoo Kanazawa, and Jitendra Malik. Humans in 4d: Reconstructing and tracking humans with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14783–14794, 2023. 1

[4] 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, 2019. 1

[5] 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, 2019. 1, 3

[6] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023. 2

[7] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. *arXiv preprint arXiv:2410.11831*, 2024. 1, 2, 3, 4

[8] Jessy Lauer, Mu Zhou, Shaokai Ye, William Menegas, Steffen Schneider, Tanmay Nath, Mohammed Mostafizur Rahman, Valentina Di Santo, Daniel Soberanes, Guoping Feng, et al. Multi-animal pose estimation, identification and tracking with deeplabcut. *Nature Methods*, 19(4):496–504, 2022. 1

[9] 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):1281–1289, 2018. 1

[10] 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.

[11] 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):117–125, 2019. 1

[12] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017. 3

[13] Narek Tumanyan, Assaf Singer, Shai Bagon, and Tali Dekel. Dino-tracker: Taming dino for self-supervised point tracking in a single video. In *European Conference on Computer Vision*, pages 367–385. Springer, 2025. 4

[14] Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. Vitpose: Simple vision transformer baselines for human pose estimation. *Advances in Neural Information Processing Systems*, 35:38571–38584, 2022. 1, 3, 4

[15] Yuxiang Yang, Junjie Yang, Yufei Xu, Jing Zhang, Long Lan, and Dacheng Tao. Apt-36k: A large-scale benchmark for animal pose estimation and tracking. *Advances in Neural Information Processing Systems*, 35:17301–17313, 2022. 3

[16] Shaokai Ye, Anastasiia Filippova, Jessy Lauer, Steffen Schneider, Maxime Vidal, Tian Qiu, Alexander Mathis, and Mackenzie Weygandt Mathis. Superanimal pretrained pose estimation models for behavioral analysis. *Nature Communications*, 15(1):5165, 2024. 1, 3, 4

[17] Yang Zheng, Adam W Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19855–19865, 2023. 3, 4