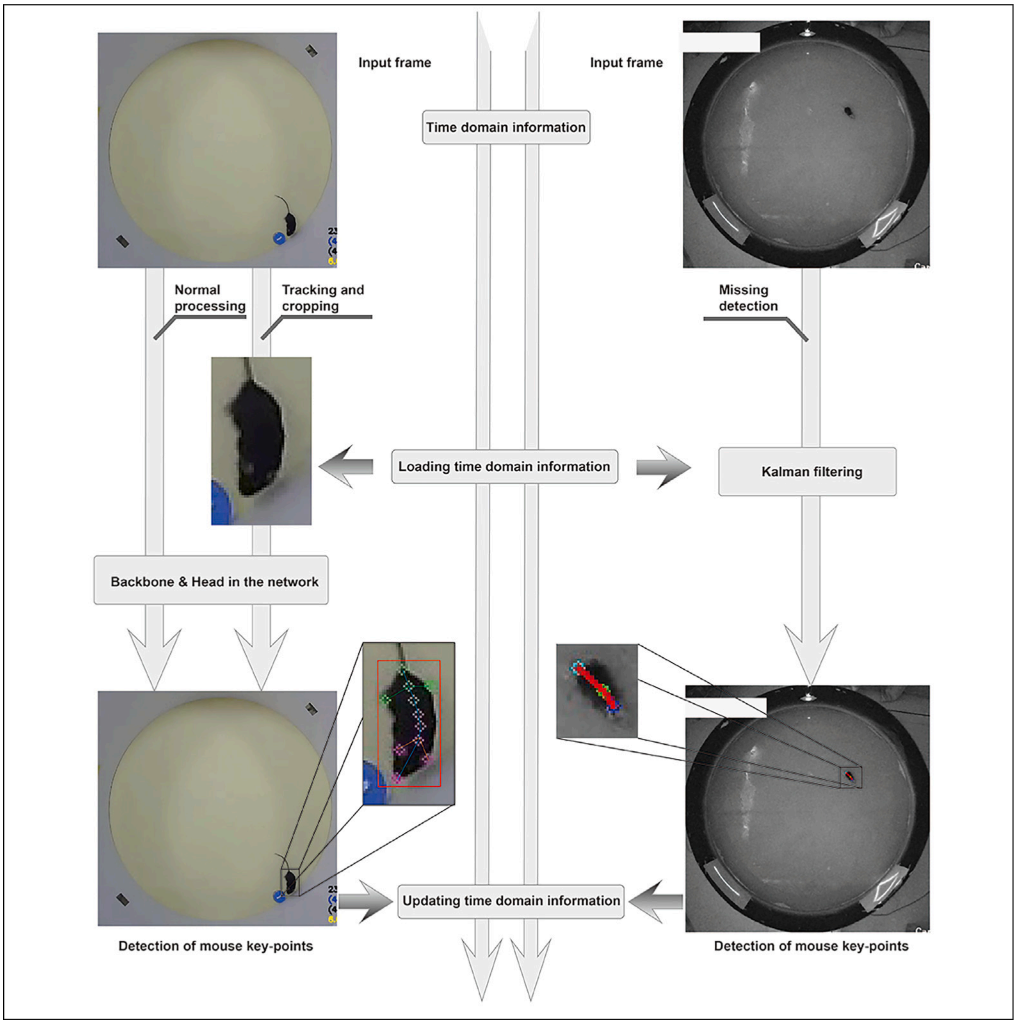
iScience

CellPress
OPEN ACCESS

Article

STPoseNet: A real-time spatiotemporal network model for robust mouse pose estimation


Input frame
Input frame
Time domain information
Normal processing
Tracking and cropping
Missing detection
Loading time domain information
Kalman filtering
Backbone & Head in the network
Updating time domain information
Detection of mouse key-points
Detection of mouse key-points

Songyan Lv,
Jincheng Wang,
Xiaowei Chen,
Xiang Liao

xiang.liao@cqu.edu.cn

Highlights

A network model STPoseNet is developed for real-time detection of mouse key points

A tracking-cropping module is designed for improving pose estimation performance

The integration of the Kalman filter facilitates the reliability of detection

Transfer learning is used to achieve accurate detection with a small training set

Lv et al., iScience *27*, 109772
May 17, 2024 © 2024 The Author(s). Published by Elsevier Inc.
https://doi.org/10.1016/j.isci.2024.109772

iScience

CellPress
OPEN ACCESS

Article

STPoseNet: A real-time spatiotemporal network model for robust mouse pose estimation

Songyan Lv,[1] Jincheng Wang,[1] Xiaowei Chen,[1] and Xiang Liao[2,3,*]

SUMMARY

Animal behavior analysis plays a crucial role in contemporary neuroscience research. However, the performance of the frame-by-frame approach may degrade in scenarios with occlusions or motion blur. In this study, we propose a spatiotemporal network model based on YOLOv8 to enhance the accuracy of key-point detection in mouse behavioral experimental videos. This model integrates a time-domain tracking strategy comprising two components: the first part utilizes key-point detection results from the previous frame to detect potential target locations in the subsequent frame; the second part employs Kalman filtering to analyze key-point changes prior to detection, allowing for the estimation of missing key-points. In the comparison of pose estimation results between our approach, YOLOv8, DeepLabCut and SLEAP on videos of three mouse behavioral experiments, our approach demonstrated significantly superior performance. This suggests that our method offers a new and effective means of accurately tracking and estimating pose in mice through spatiotemporal processing.

INTRODUCTION

The study of animal behavior has a long history dating back to ancient times, with early pioneers such as Aristotle (384–322 BC) and Erahistratus (304–250 BC) conducting experiments on animals in captivity.[1] Animal behavior analysis is a scientific and objective approach to studying the behavior of animals in different conditions. It encompasses various aspects of animal behavior, such as communication, emotional expression, social interactions, learning, and reproductive behavior. One particular area of focus in animal behavior analysis is the study of mouse behavior, which is conducted in multiple fields including neuroscience, behavioral science, and medicine.[2–4] In the context of studying mouse behavior, traditional methods involve researchers watching video replays and manually noting the timing and location of specific events to identify behavioral postures. However, these manual analyses are time-consuming and prone to inconsistencies due to variations in labeling standards and reliability among different annotators. As a result, the reproducibility of research findings is often limited.[5] Recently, the field of animal behavior research has witnessed significant advancements in information technology, leading to a substantial increase in the volume of animal behavior data. Furthermore, experiments now involve more complex and diverse situations. However, the manual labeling of behavior data becomes challenging in low-light or dim scenes, making it difficult to ensure labeling standards and accuracy.[6]

The estimation of behavioral poses plays a crucial role in a wide range of recent scientific studies. For example, Ottenheimer et al. investigated fluid consumption to gain insights into neural coding for reward prediction errors.[7] Okubo et al. explored the effects of wind on fruit fly behavior.[8] Schorscher-Petcu et al. studied the role of tactile events and nociceptors in tactile perception of freely moving mice.[9] Warren et al. examined the interaction between tactile processing in the palpebral system and its impact on movement direction in rodents.[10] Keshavarzi et al. found the relationship between eye movement and neural activity in freely behaving rodents.[11] As a result, there is an increasing focus on developing automated, reliable, and accurate methods for analyzing experimental videos of mice, particularly in the identification of mouse behavioral postures and the implementation of real-time interactive closed-loop experimental designs.[12,13]

Pose estimation methods emerged in computer vision research in the late 1970s, enabling the analysis of image frames from videos to estimate poses.[14,15] This estimation was achieved by representing the images in a dimensionally reduced space, using techniques such as principal component analysis to generate features. The estimation process involved unsupervised clustering of pose dynamics. By comparing spectra or fitting autoregressive models to low-dimensional projections, these methods can identify and classify dozens to hundreds of unique stereotypical behaviors in animals, including flies and mice.[16,17] These methods have proven valuable in uncovering underlying patterns and structures in animal behavioral data. Consequently, they have contributed significantly to studies involving social interaction, gene mutation, and neural perturbation.[18–22] However, these methods have limitations related to video properties and are challenging to adapt to different experimental settings, thus restricting their application in certain scenarios.

In animal behavioral experiments, the accuracy of posture recognition can be compromised when the recorded video quality is poor, such as in dim conditions during water maze experiments. This can result in significant position deviations of key points used for estimation. In

[1] Guangxi Key Laboratory of Special Biomedicine & Advanced Institute for Brain and Intelligence, School of Medicine, Guangxi University, Nanning 530004, China
[2] Center for Neurointelligence, School of Medicine, Chongqing University, Chongqing 400030, China
[3] Lead contact
*Correspondence: xiang.liao@cqu.edu.cn

https://doi.org/10.1016/j.isci.2024.109772

some cases, the recognition target may even be lost entirely, which can have a notable impact on the reliability of behavioral experiments. Consequently, manual identification by experimentalists is still necessary and software tools cannot fully replace this human involvement. Additionally, striking the right balance between recognition accuracy and speed poses a challenge. Real-time pose estimation is crucial for enabling closed-loop experimental setups in neuroscience research. Establishing closed-loop protocols that manipulate ongoing neuronal activity is highly valuable for studying *in vivo* electrophysiological experiments. It allows researchers to detect specific electrophysiological events in conjunction with stimulation and track specific behaviors through real-time monitoring of neuronal operation.[23,24] However, current behavioral analysis software that operates in real time often requires the attachment of sensors or tracking devices to animals, which presents additional complexities.

With advancements in computer vision, deep learning, and GPU computing, new techniques have revolutionized the analysis of human behavioral posture.[25–30] The success in this field is largely attributed to the effective training and evaluation of images using convolutional neural networks.[31,32] However, when it comes to pose estimation for laboratory animals, there are subtle differences that must be taken into consideration. While algorithms developed human images can handle diverse situations in terms of human shape, environment, and quality, they typically require large training sets of labeled images.[33,34] Behavioral experiments in laboratory offer more control, but the recording conditions can be highly specific to each experimental paradigm. This makes it challenging to label data for each animal type, which is not readily available. Consequently, transfer learning becomes vital for training models for these specific scenarios. Recently, some open-source tools for animal pose estimation have been proposed, such as DeepPoseKit, DeepLabCut, and SLEAP.[35–40] Automated analysis techniques have also been explored for fruit flies, broiler chickens, primates, and zebrafish.[35,41–43] These advancements not only contribute to the field of animal behavior analysis but also facilitate standardized and scalable approaches for pose estimation in various laboratory animal species.

DeepPoseKit uses a multi-scale deep learning model (named Stacked DenseNet) and a GPU-based detection algorithm for fast peaks (estimating key-point locations with sub-pixel accuracy). They explored a number of small-scale image recognition methods, with the images having clearly defined edges. However, in the actual behavioral experiments, in order to meet some experimental conditions (such as dark field, water maze, etc.), the video quality is difficult to reach the clarity of its example dataset. At the same time, the most important point is to ensure the recognition speed. The DeepPoseKit model has a few parameters, which leads to a smaller image size that it can process (fly: 192 × 192, locust: 160 × 160, zebra: 160 × 160) and is significantly different from regular animal behavior videos. As it is not reasonable to ask experimenters to restrict the behavioral experiment itself for the sake of later video analysis, therefore it needs to meet the animal detection accuracy under normal video size and various video quality while ensuring the video processing speed.

DeepLabCut employs transfer learning on a model trained with a large image dataset (ResNet50, 25 million parameters) to achieve behavioral video recognition. While its training and prediction speeds may be slower compared to DeepPoseKit, DeepLabCut offers greater adaptability to various experimental scenarios and boasts higher detection accuracy. SLEAP, the advanced successor of the single-animal pose-estimation method LEAP, is a multi-animal pose tracking system with a sophisticated graphical user interface (GUI) akin to DeepLabCut. It facilitates effortless dataset production and training within the program, boasting exceptional recognition speed and accuracy on par with DeepLabCut.[40] SBeA is a computational framework designed to surmount data limitations by leveraging a minimal number of labeled frames for multi-animal 3D estimation. Its findings demonstrate that outstanding performance can be attained across diverse species by utilizing existing custom datasets.[44]

Moreover, YOLOv8 is one of the most well-established image recognition models in recent years. It encompasses various functions such as target recognition, image segmentation, and key-point detection. Particularly, the key-point detection component of YOLOv8-pose is capable of recognizing 17 key points on the human body. It excels in real-time recognition and offers high precision even in environments with complex backgrounds. Leveraging the strength of YOLOv8, we developed a spatiotemporal approach for mouse behavior pose estimation. By building our approach upon YOLOv8-pose, we aim to harness its robustness and accuracy in recognizing targets in complex backgrounds. This allows us to achieve precise and real-time pose estimation for mouse behavior, empowering researchers in their behavioral analysis and further advancements in the field.

The main challenge in pose estimation is to strike a balance between generality of use, recognition accuracy, processing speed, and annotation cost. To address this challenge, we propose a spatiotemporal PoseNet (STPoseNet) model that focuses on achieving high accuracy in mouse pose estimation, while maintaining fast processing speed and minimizing labeling efforts (with fewer than 100 images required for a single experimental scenario). Our approach incorporates two modules that leverage time domain information: a tracking-cropping module (TCM) and a Kalman filter-based module. We first evaluated the effectiveness of our method in two experimental scenarios: an open field, where mice chase artificial prey against a clear background, and a water maze, where mice swim in dim conditions. Then we utilized publicly accessible open field exploration experiment videos for further evaluation of performance. In our study, YOLOv8-pose, DeepLabCut and SLEAP were used as benchmarks to compare the performance of our proposed approach for mouse pose estimation. The main contributions of this study can be summarized as follows.

(1) In laboratory mouse behavior experiments, we have developed a pose estimation model based on YOLOv8 that incorporates time domain information from videos. This design ensures both accurate detection and high processing speed.
(2) To improve the accuracy of pose estimation in complex conditions, we developed a TCM based on time domain information for key-point detection and developed a matching data enhancement approach for it.
(3) Additionally, we have integrated a Kalman filter-based module to predict key points in consecutive video frames, minimizing the occurrence of missing results during the experiment.

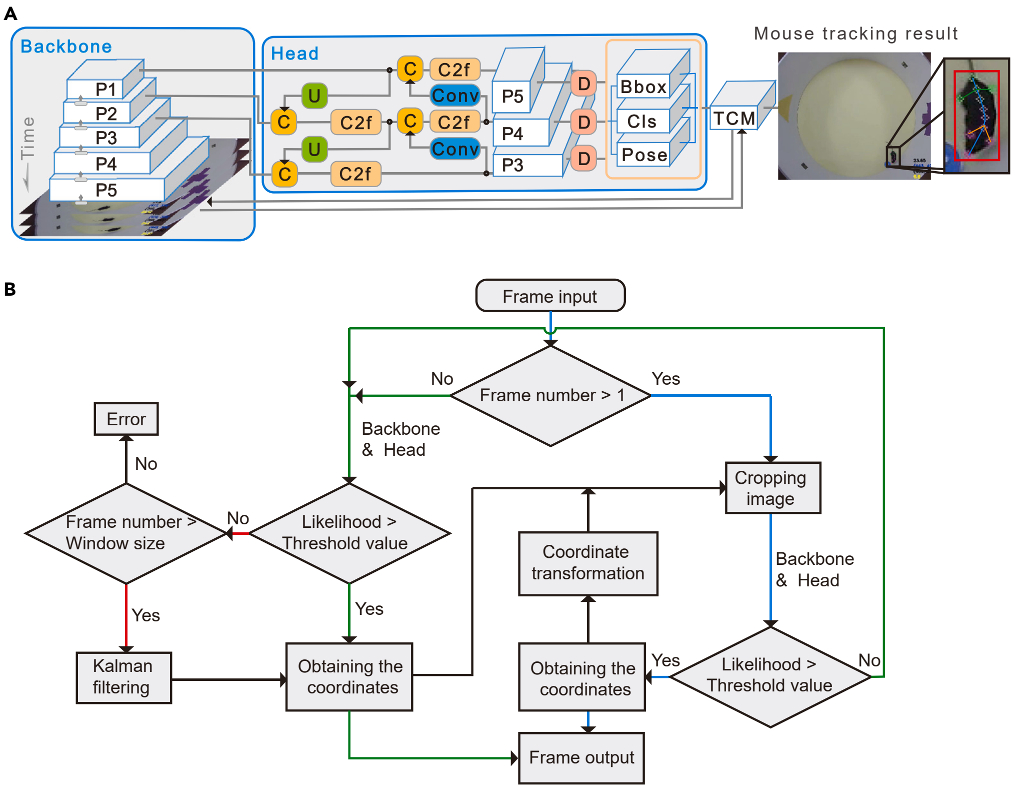


Figure 1. Schematic illustration of STPoseNet model and the processing flowchart for analyzing experimental mouse behavior videos

(A) The STPoseNet model is constructed based on the YOLOv8 network architecture. To enable spatiotemporal pose estimation, a tracking-cropping module (TCM) is introduced into the model. The TCM is independent from Backbone and Head parts. The TCM is responsible for tracking and cropping the mouse poses across consecutive frames. The data input process involves feeding images into the backbone layers (P1-P5) for feature extraction through convolution, before transferring the data to the Head. The original input images are resized to a predetermined size before being fed into the TCM. Notations used in the model: C: Concat, U: Upsample, D: Detect. The Bbox, Cls, and Pose in the Head part represent the bounding box information, the content category, and the key-points information of the content, respectively.

(B) The flowchart demonstrates the sequential processing of mouse behavior videos. The green line represents the information flow during the normal mode, where the original images are processed. The blue line illustrates the information flow when the TCM is activated, and the tracking-cropping module is utilized. This allows for improved accuracy in estimating the mouse poses over time. In certain special conditions, denoted by the red line, the TCM leverages the Kalman filter to make predictions, ensuring robust pose estimation even in challenging situations. See also Figure S1.

(4) The STPoseNet model weights we provide are pre-trained under three different experimental conditions, reducing the cost of applying them to new experimental scenarios through transfer learning.

RESULTS

Pose estimation process of STPoseNet model

STPoseNet is a neural network that we developed to incorporate both spatial and temporal information for accurate mouse pose estimation. Figure 1 illustrates the architecture and experimental video processing flow of STPoseNet. Figure 2 demonstrates the mouse tracking process and pose estimation with consecutive video frames. Figure 3 highlights the evaluation metrics utilized for assessing the accuracy of mouse pose estimation. The STPoseNet network architecture, TCM, data enhancement, error reduction estimation, and performance evaluation methods are depicted in the STAR Methods section.

The data preprocessing, network model training, and testing were conducted on a Windows 10 system featuring an Intel i7-11800H processor, 32 GB of RAM, and an NVIDIA GeForce RTX 3080 Laptop GPU. In the training process, STPoseNet and YOLOv8 were initialized with preloaded weights from YOLOv8, whereas DeepLabCut utilized the initial training weights of ResNet50. To ensure robust performance, the network models were trained using three datasets from two distinct experimental scenarios. The first dataset included 1,647 images (labeled with 3 key points) from the open field scenario, the second dataset comprised 300 images (labeled with 11 key points) from the same open field scenario, and the last dataset consisted of 1,392 images (labeled with 3 key points) from the water maze scenario. To assess mouse pose estimation across varied experimental scenes, we employed 221 continuous video frames from the open field and 500 from the water maze for

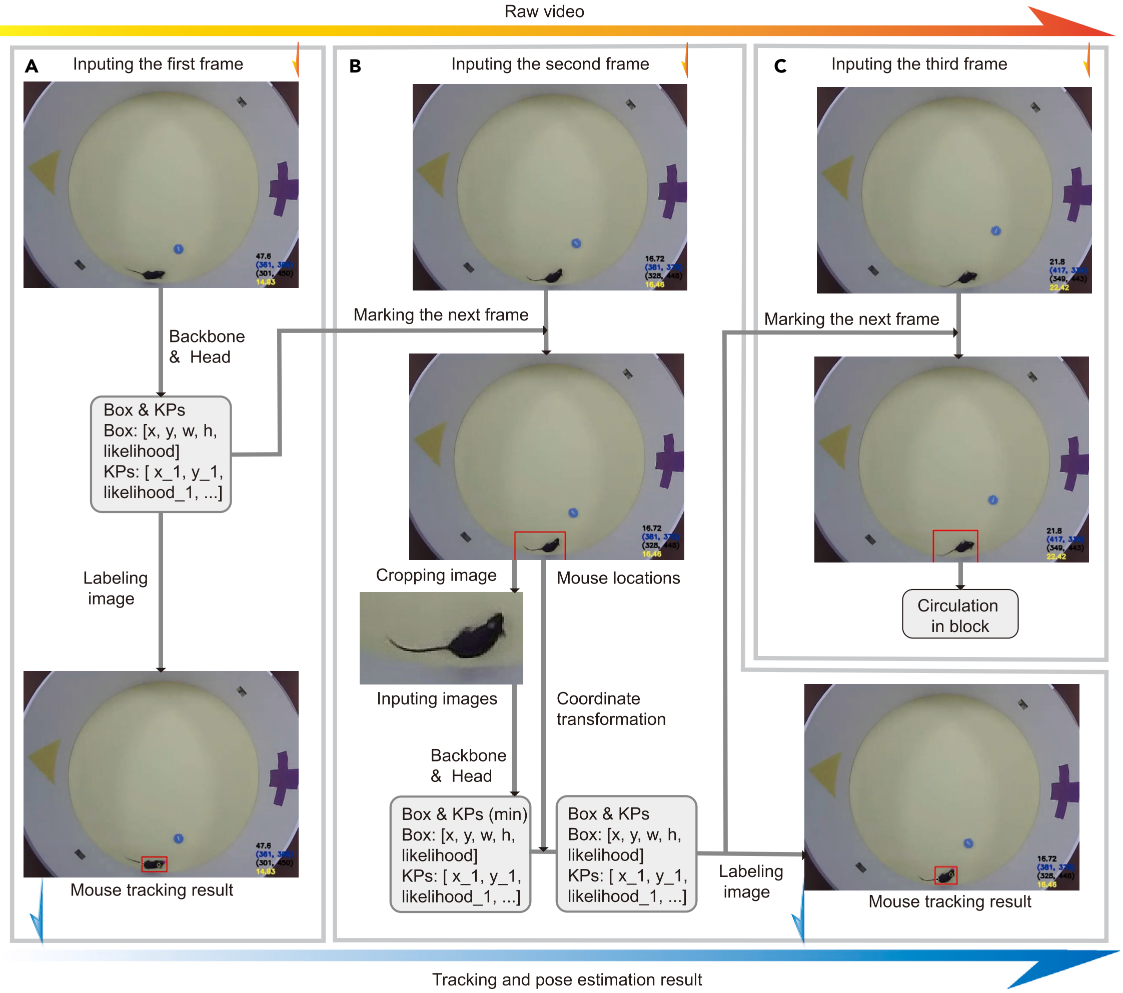


Figure 2. Process of mouse tracking and pose estimation using consecutive video frames

(A) The mouse tracking result of the first video frame serves as the basis for the following calculation.

(B) The area with the detected mouse is cropped for new frame processing.

(C) Loop is run over the process in (B) until detection failure or the frame is at the end of the video.

evaluating model performance. Our objective was to assess the efficacy and adaptability of the models across diverse experimental conditions through training on these heterogeneous datasets.

Following the completion of model training, the STPoseNet algorithm progresses to the testing phase by estimating the mouse pose using the original input frames. Once the key points are obtained, they are passed to the TCM for marking the corresponding locations in the subsequent frame. The spatiotemporal processing of the video continues until either the video reaches the end or the confidence level of the cropped image detection becomes too low. In cases where the mouse detection fails even with the original image (i.e., under extreme circumstances), the Kalman filter is applied to predict the key points obtained in the previous frame. This robust approach ensures that pose estimation can still be achieved, even in challenging scenarios. By incorporating tracking, cropping, and the use of the Kalman filter, the STPoseNet methodology provides a comprehensive solution for accurately estimating mouse pose throughout a video sequence. This dynamic approach enhances the reliability and robustness of the pose estimation results.

In assessing the robustness of STPoseNet, we conducted pose estimation evaluations across various video resolutions (320 × 240, 480 × 360, 640 × 480, 800 × 600, and 960 × 720) and frame rates (10 Hz, 15 Hz, 30 Hz, 50 Hz, and 60 Hz). The consistent and reliable estimations of the 3 key points (nose, neck, and tail) under these diverse conditions serve as a testament to the efficacy of our approach (Figure S1).

Comparison of STPoseNet with other methods for distance error

To assess the pose estimation performance for mouse behavior data, we compared the proposed method with the original YOLOv8, DeepLabCut and SLEAP methods in the two experimental scenarios, open field and water maze. For the two experimental scenarios, the image quality of the open field experiment is good, and the mouse features are obvious, which can be used to compare the accuracy

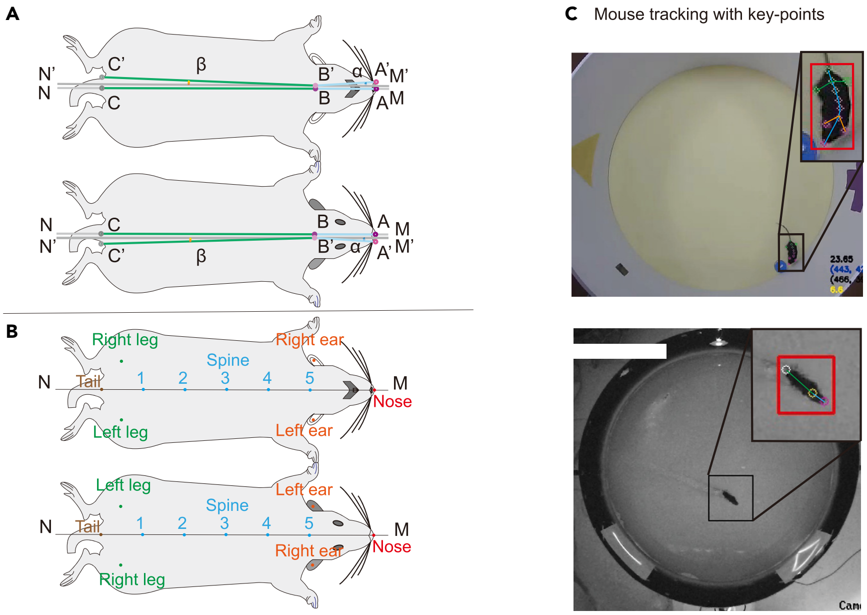


Figure 3. Illustration of the evaluation metrics used for mouse pose estimation

(A) The schematic diagrams depict the mouse abdomen (left) and the mouse back (right). In both cases, M and N represent the centrosymmetric line of the mouse (ground truth), which is based on the thoracic spine orientation of mouse, while M′ and N′ represent the centrosymmetric line estimated using the key-point detection method. A–C represent the key points of the mouse's nose, neck, and tail (ground truth), which are based on the front end of the head, cervical spine, and tail vertebrae of mouse, while A′–C′ represent the estimated key-points using the detection method. The angle α indicates the deviation of the head, and the angle β indicates the deviation of the body.

(B) The schematic diagrams depict the locations of 11 mouse key points, including nose, left and right ears, five positions evenly distributed along the mouse spine, left and right hind legs, and tail.

(C) The tracking and key-point estimation results in the open field (top) and water maze (bottom) experiments, respectively. The red rectangle represents the detection bounding box. The examples demonstrate the detection of multiple key-points in two experiments.

of the pose estimation methods in simple scenario. In the water maze experiment, the image is relatively dim, and the mouse features are partially occluded, which can be used to compare the reliability of the pose estimation methods in complex scenario. The four models were trained on the same dataset, and the training loss values for each model converged after a set number of epochs (Figure S2).

The representative results depicted in Figures 4A and 4B illustrate that the key-point distance error of the STPoseNet model is smaller than that of YOLOv8, DeepLabCut, and SLEAP. Overall, STPoseNet demonstrated superior stability and accuracy in detecting multiple key points (distance errors: 20.826 and 7.149), while the other three methods showed inconsistencies in recognizing the direction of the mouse body, leading to significant distance errors. This indicates that STPoseNet is more reliable than DeepLabCut, YOLOv8, and SLEAP in conditions such as mouse leaning, turning, and being underwater.

We conducted a comprehensive comparison of STPoseNet, YOLOv8, DeepLabCut, and SLEAP regarding error rates in two different experiment scenarios. In the open field experiment, our findings indicate that STPoseNet achieved an impressive error rate of less than 8% of body length in 100% (nose), 100% (neck), and 95.023% (tail) of the testing dataset. In contrast, DeepLabCut only achieved 78.28% (nose), 45.25% (neck), and 78.73% (tail). SLEAP achieved 90.5% (nose), 61.99% (neck), and 99.5% (tail) under the same conditions. Moving on to the water maze experiment, we observed that STPoseNet attained an error rate of less than 8% of body length in 90.2% (nose), 89.4% (neck), and 85.2% (tail) of the testing dataset. Comparatively, DeepLabCut yielded 68.0% (nose), 57.6% (neck), and 66.4% (tail), SLEAP achieved 90.6% (nose), 84.4% (neck), and 91.6% (tail) accuracy under the same conditions. To further highlight the enhanced efficacy of pose estimation using STPoseNet, we conducted a statistical comparison with the original YOLOv8, DeepLabCut, and SLEAP, as shown in Figure 5. The comparative analysis of the error of a single key point reveals that STPose outperformed YOLOv8, DeepLabCut, and SLEAP significantly in terms of distance errors for nose and neck (Spine 1) estimations in the open field experiment ($p < 0.05$, two-sided Z test, $n = 221$ images), as well as neck and tail estimations in the water maze experiments ($p < 0.05$, two-sided Z test, $n = 500$ images). In the open field experiment, distance error comparisons for the other 8 key points are detailed in Figure S3. Furthermore, when comparing four models using root-mean-square error (RMSE) for distance errors in the two experimental scenarios, the results consistently showed that STPoseNet exhibited the smallest RMSE among all tested methods, as detailed in Table S1. These findings suggest that STPoseNet offers superior and precise pose estimation compared to the other three methods, making it suitable for various applications across different scenarios.

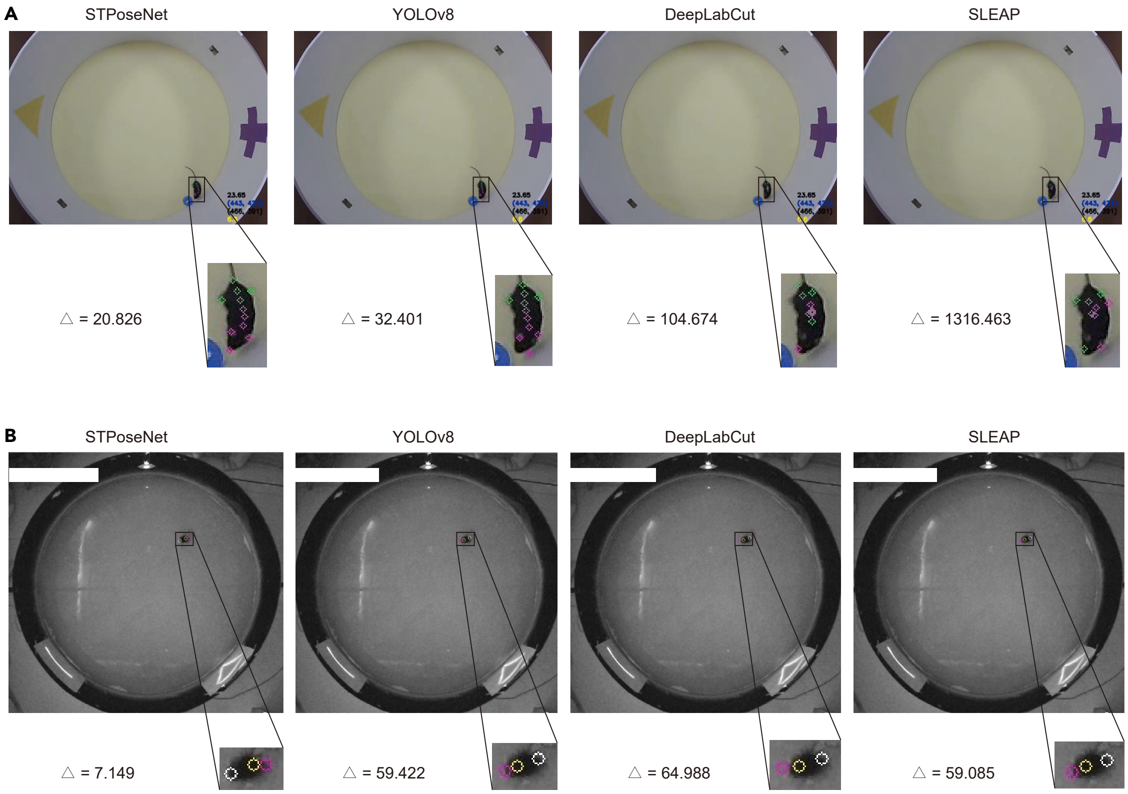


Figure 4. Pose estimation examples of four methods in two experiments

(A and B) The examples demonstrate the key-point detection with STPoseNet, YOLOv8, DeepLabCut, and SLEAP in the open field (A) and water maze (B) experiments, respectively. To quantify the comparison, the sum of the minimum distance errors of the three key points (Δ represents the sum of the straight-line distances from all identified points to marked points) is utilized. See also Figure S2.

Comparison of STPoseNet with other methods for angle deviation

We conducted a comparative analysis of STPoseNet, DeepLabCut, and SLEAP based on error angles in two different experiment scenarios. In the open field experiment, we achieved an error angle of less than 10° in 84.16% (head orientation difference, i.e., the angle between the nose and neck) and 90.05% (body orientation difference, i.e., the angle between the neck and tail) of the testing dataset using STPoseNet. Under the same conditions, DeepLabCut achieved 35.75% (head orientation difference) and 85.52% (body orientation difference), SLEAP achieved 48.42% (head orientation difference) and 64.71% (body orientation difference). Moving on to the water maze experiment, we achieved an error angle of less than 10° in 68.8% (head orientation difference) and 86.8% (body orientation difference) of the testing dataset using STPoseNet. Comparatively, DeepLabCut yielded 67.6% (head orientation difference) and 85.4% (body orientation difference) and SLEAP yielded 65.8% (head orientation difference) and 88.6% (body orientation difference) under the same conditions. We also performed a statistical comparison of the angle differences for head orientation and body orientation estimation among the four methods, as shown in Figure 6. The results indicate that STPoseNet consistently exhibited significantly smaller error angles compared to YOLOv8, DeepLabCut and SLEAP ($p < 0.05$, two-sided Z test, $n = 221$ images for the open field, $n = 500$ images for the water maze). Furthermore, we conducted a comparison of four models using RMSE for error angles in the two experimental scenarios, and the results revealed that, on average, STPoseNet demonstrated the lowest RMSE among all tested methods, as presented in Table S2. Thus, STPoseNet exhibited reduced error angles in pose estimation compared to the other three methods.

Comparison of STPoseNet with other methods for processing speed

In Figure 7A, we present the mouse tracking results obtained using STPoseNet for the open field and water maze experiments, demonstrating how it accurately tracks the movement of the mouse. The mouse position and pose estimation data provided by STPoseNet can be combined with neural recordings to analyze the behavioral and neural mechanisms. We demonstrate mouse pose estimation with 3 key-point detection in Video S1, 11 key-point detection in an open field in Video S2, and 3 key-point detection in a water maze in Video S3. Moreover, we assessed the processing speed of the four methods, as depicted in Figure 7B. We found that STPoseNet achieved a processing speed of 30 and 25 frames per second when analyzing the mouse behavior videos from the two experiments. This processing speed is sufficient for real-time pose estimation. While the processing speed of STPoseNet is slower than the original YOLOv8 and SLEAP due to the inclusion of TCM

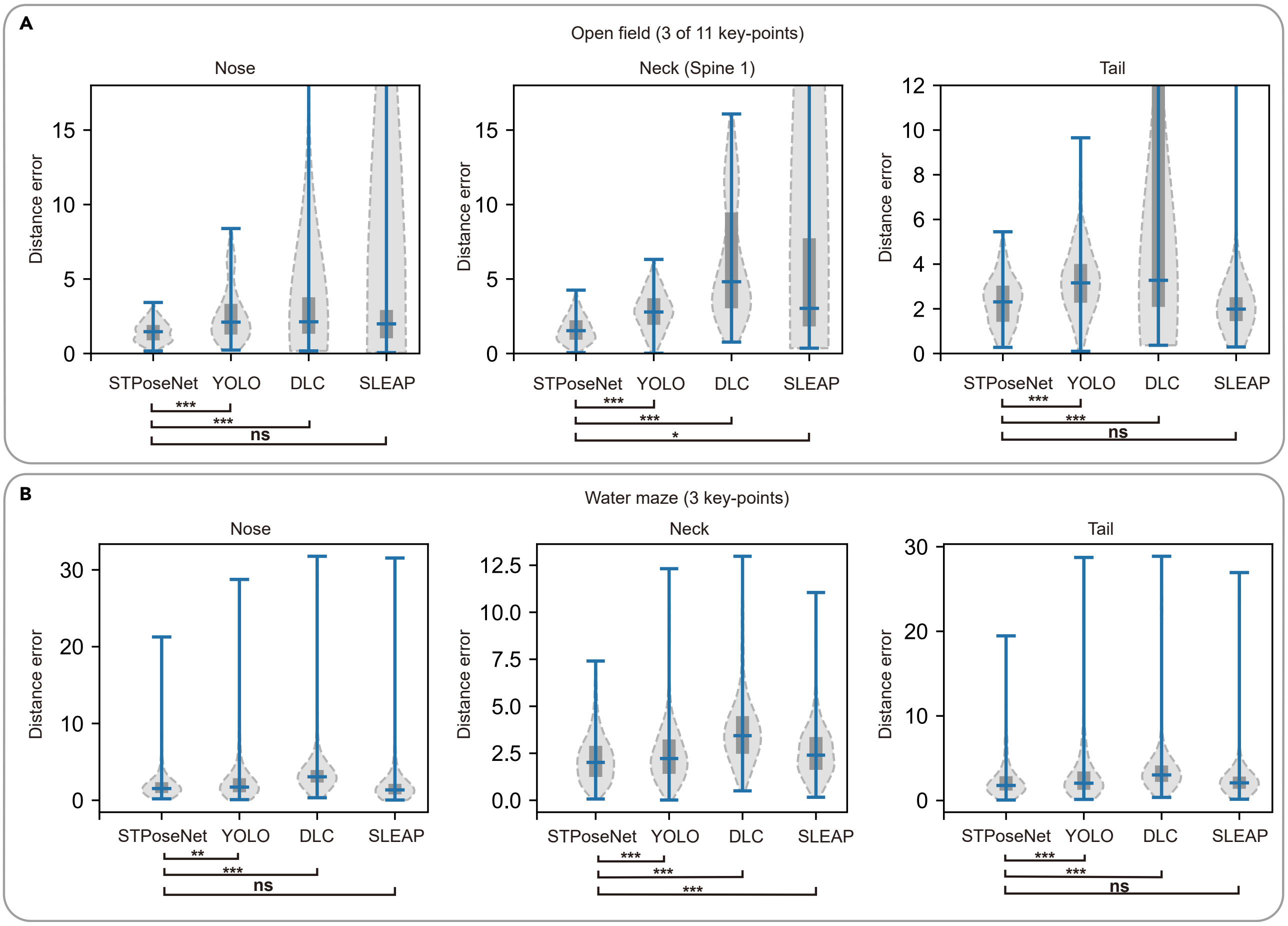


Figure 5. Distance error comparison of four methods

(A and B) Distance errors for the nose (left), neck (middle, Spine 1 in 11 key-points) and tail (right) estimation in the open field (A) and water maze (B) experiments, respectively. Comparison of STPoseNet with other methods, one-way ANOVA, open field experiment: F = 5.83, $p = 6.0 \times 10^{-4}$ (nose); F = 6.62, $p = 2.0 \times 10^{-4}$ (neck); F = 89.9, $p = 1.0 \times 10^{-50}$ (tail); water maze experiment: F = 26.98, $p = 4.2 \times 10^{-17}$ (nose); F = 76.74, $p = 5.6 \times 10^{-47}$ (neck); F = 15.47, $p = 6.0 \times 10^{-10}$ (tail). Violin plots are used for showing statistical comparison, two-sided Z test, $*p < 0.05$, $**p < 0.01$, $***p < 0.001$; ns, not significant. YOLOv8 is abbreviated as YOLO, and DeepLabCut is abbreviated as DLC. See also Figures S3 and S4; Table S1.

and Kalman filter modules, it is notably faster than DeepLabCut. This highlights the advantage of STPoseNet, as it provides accurate pose estimation while maintaining a reasonable processing speed. Researchers can benefit from utilizing STPoseNet for real-time analysis of mouse behavior and its integration with neural recordings for a comprehensive understanding of behavioral-related neural mechanisms.

It is worth noting that the input images were resized to 640 × 640, and the resolution of the image itself has minimal impact on processing speed. The primary reason for the faster processing speed in the water maze compared to the open field is may be due to the larger size of the mouse in the open field, resulting in faster relative movement than in the water maze. This variance affects the continuity of detection and tracking in TCM. In case of poor recognition quality in the segmented image, a secondary round of original image recognition was carried out. This resulted in certain frames in the video being processed twice, thereby contributing significantly to the reduction in processing speed.

Comparison of STPoseNet with other methods using public dataset

To enhance the verification of the generality and stability of our model, we utilized publicly accessible mouse behavioral videos.[45] Within this dataset, we adopted the 6 key points referenced in the literature, comprising the nose, left forehand, right forehand, left hindhand, right hindhand, and tail. We utilized two videos, one for the training dataset (300 images) and the other for the testing dataset (100 frames), to conduct a comparative analysis of pose estimation using four methods. The training loss values exhibited convergence after a set number of iterations (Figure S2). In Figure 8A, it is evident that STPoseNet exhibited a smaller distance error compared to the other three methods. For instance, SLEAP failed to detect the right forehand and thus exhibited a large error level. Through statistical analysis (Figure 8B), the results indicate that STPoseNet had a significantly lower distance error than the other three methods, which is consistent with the findings presented in the preceding section.

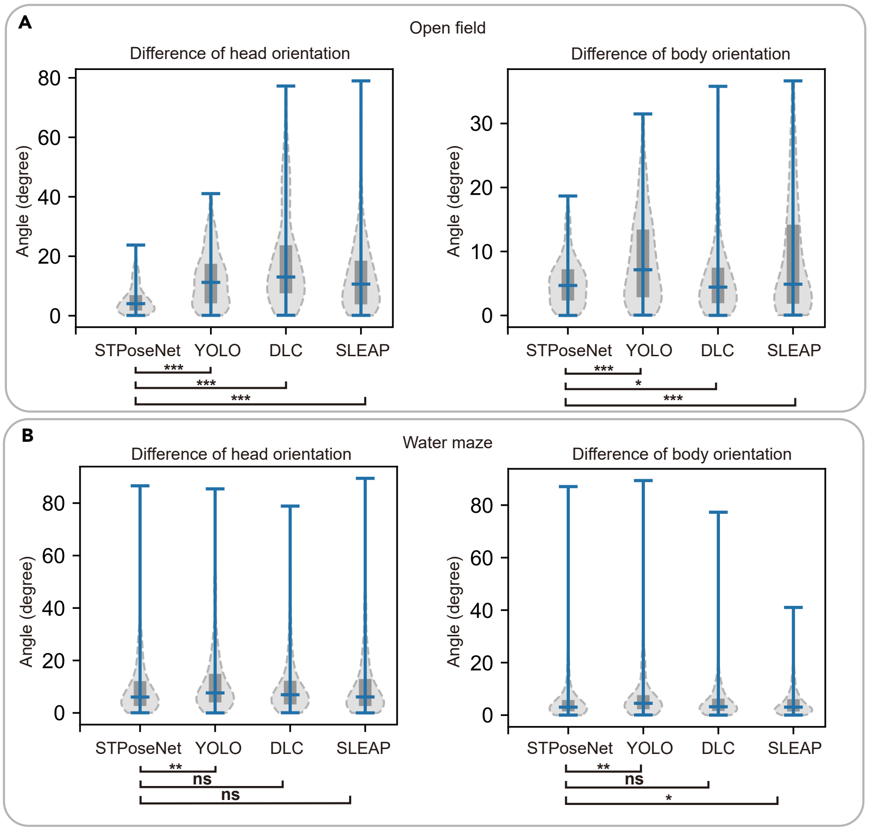


Figure 6. Error angle comparison of four methods

(A and B) The angle of head orientation difference (left), as well as the angle of body orientation difference (bottom) in the open field (A) and water maze (B) experiments, respectively. Comparison of STPoseNet with other methods, one-way ANOVA, open field experiment: F = 53.56, $p = 8.5 \times 10^{-32}$ (head); F = 18.30, $p = 1.6 \times 10^{-11}$ (body); water maze experiment: F = 2.62, $p = 4.9 \times 10^{-2}$ (head); F = 9.98, $p = 1.6 \times 10^{-6}$ (body). Violin plots are used for showing statistical comparison, two-sided Z test, $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$; ns, not significant. YOLOv8 is abbreviated as YOLO, and DeepLabCut is abbreviated as DLC). See also Table S2.

By comparing the performance metrics including distance error, angle, and processing speed in the mouse behavioral experiments, it is evident that STPoseNet surpasses YOLOv8, DeepLabCut and SLEAP in terms of accuracy. This superiority is evident not only in our own experiments (water maze and open field) but also in the open field exploration experiment conducted by another laboratory. Thus, STPoseNet consistently delivers reliable and robust pose estimation results. Furthermore, STPoseNet exhibits a promising processing speed for video data, allowing for real-time recognition. This means that researchers can rely on STPoseNet for instant analysis and tracking of pose in behavioral experiments.

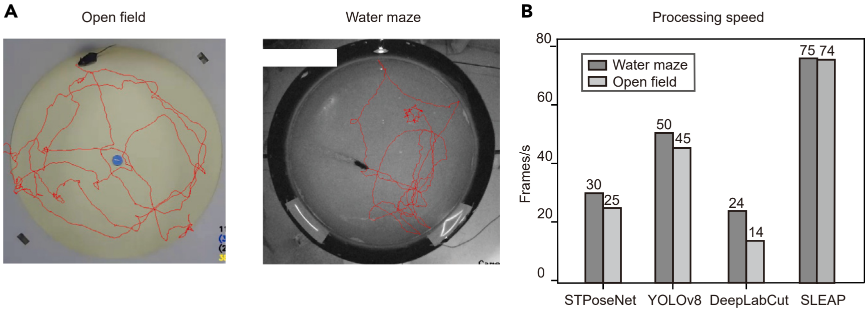


Figure 7. Mouse tracking results and video processing speed

(A) The example mouse pose tracking (the key points of nose tip) results from open field (left) and water maze (right) experimental scenarios.

(B) Comparison of processing speed (the average time of the recognition process) for the four methods tested in the two experimental conditions (video image resolution is 1280 × 720 for water maze and 720 × 480 for open field). See also Videos S1, S2, and S3.

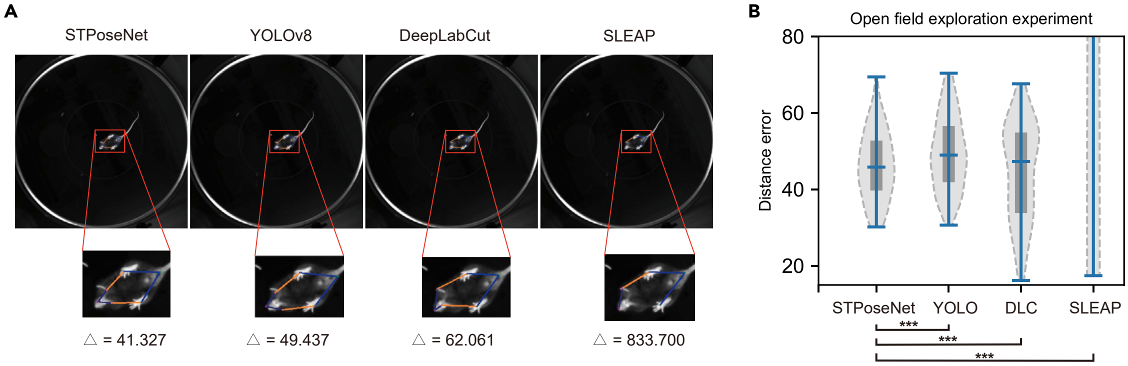


Figure 8. Testing the pose estimation methods with publicly available dataset

(A) The examples demonstrate the pose estimation with STPoseNet, YOLOv8, DeepLabCut, and SLEAP in the open field exploration experiment. To quantify the comparison, the sum of the minimum distance errors of the key points (Δ represents the sum of the straight-line distances from all identified points to marked points) is utilized.

(B) Comparison of STPoseNet with other methods, one-way ANOVA: F = 230.28, $p = 1.9 \times 10^{-86}$, n = 100 frames. Violin plots are used for statistical comparison, two-sided Z test, ***$p < 0.001$.

Performance improvement with Kalman filter

The Kalman filter module is integrated into STPoseNet to make predictions when the mouse is not detected, considering image conditions such as occlusion and motion blur. The activation of the Kalman filter is initiated when the likelihood value falls below a certain threshold. In the context of utilizing the Kalman filter module, the key-point detection results in the water maze indicate that the Kalman filter effectively decreased the distance error (as shown in Figure 9A) once the likelihood value exceeded a certain threshold (illustrated in Figure 9B). Incorporating the Kalman filter in mouse pose estimation leads to enhanced and dependable performance.

Transfer learning with different new training dataset sizes

To explore the training data requirements for transfer learning in different animal behavior experiments, we conducted experiments to compare the dataset sizes needed to apply STPoseNet to various tasks. We trained a pre-trained weight using data from three different experiments: eight-arm maze (n = 894 images), water maze (n = 1,253 images), and conditioned fear (n = 400 images). After migration training

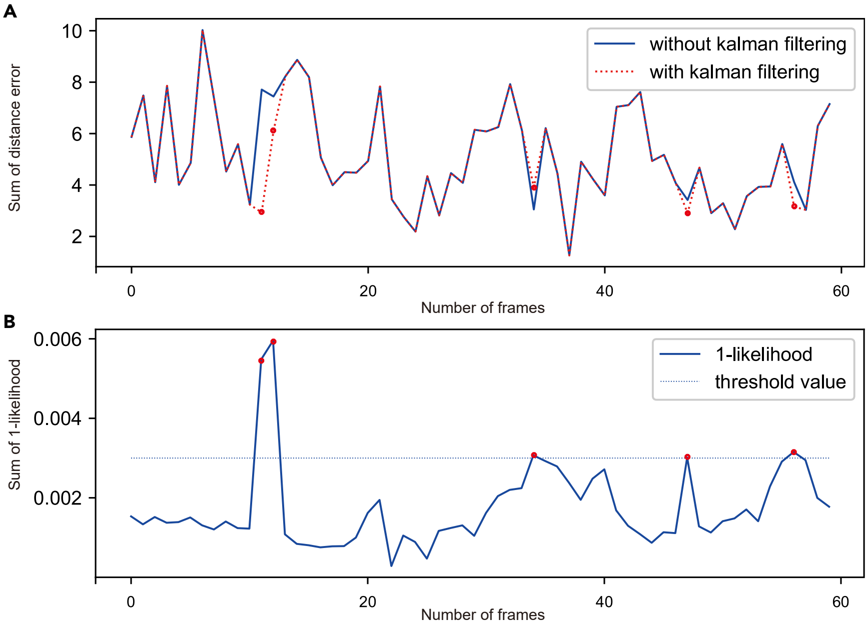


Figure 9. Pose estimation improvement with Kalman filtering

(A) Comparison of distance errors of the model with Kalman filtering (red dashed line) and without Kalman filtering (blue solid line).

(B) The confidence changes associated with using Kalman filter. The five red dots represent the frames that used the Kalman filter module.

with this dataset, we validated the model using the dataset from the open field experiment, which consisted of different numbers of images (25, 50, 100, 200, 400, and 1,483, respectively). The results demonstrate that when the number of images is small (e.g., 25), the model struggles to learn, and the likelihood values fail to reach a high and stable level. However, as the number of images increases to 50, there is a significant improvement in the confidence of key-point identification, and the likelihood values for mAP50 (Box), mAP50-95 (Box), mAP50 (Pose), and mAP50-95 (Pose) converge to a high level (as shown in Figure 10A–10D). These performance patterns continue to improve with 100, 200, and 400 images.

Furthermore, we compared the shortest distance error of each key point for the open field video (consisting of 221 images). In Figures 10E–10G, we observed that even with a reduced number of images (around 50), the average accuracy still achieved an error of less than 3 pixels, with the maximum error not exceeding 7 pixels (considering the mouse body length is approximately 60 pixels, the average error is 5%, and the maximum error is 11%). This indicates that our STPoseNet model can be successfully transferred to new behavioral experiments with a small number of labeled datasets using the pre-trained model while achieving promising performance. This advantage allows our model to be easily applied to various experimental scenarios, as it significantly reduces the burden of gathering large amounts of labeled data. Researchers can leverage the model's transfer learning capability and achieve reliable results with minimal data requirements.

DISCUSSION

In this study, we propose an end-to-end network model that tackles the challenges of automatic, accurate, and robust key-point detection in mouse behavior experiments. Our network model is developed upon YOLOv8 and incorporates the TCM and Kalman filter modules (Figure 1). This approach effectively tracks mouse images, especially in extreme cases, and supplements key-point locations based on time domain information, which not only avoids error amplification but also enhances the reliability of key-point detection. By leveraging video time domain information and implementing tracking and cropping approach, we have achieved significantly improved accuracy in mouse pose estimation.

Comparative results illustrate that our model surpassed the performance of the original YOLOv8, DeepLabCut, and SLEAP (Figures 4, 5, and 6). Due to the frame-by-frame approach of DeepLabCut and SLEAP in pose estimation, their effectiveness may deteriorate in scenarios with occlusions, motion blur, or missed features in certain frames.[37] On the contrary, STPoseNet integrates spatial and temporal information through the TCM and Kalman filter, thereby improving the consistency of pose estimation throughout video frames and reinstating key-point detection with predictive capabilities (Figure 9). The thorough examination of performance metrics firmly establishes STPoseNet as an advanced and reliable solution for pose estimation. Its superior accuracy and real-time processing capabilities render it an indispensable tool for researchers in the field of neuroscience. The results also demonstrate the convenience of training and testing with STPoseNet, the end-to-end training process minimizes the possibility of errors. Even with a small training set, our model can accurately estimate mouse posture key points. Overall, our approach stands out from conventional methods by leveraging time domain information, enabling accurate and fast key-point identification across different experimental conditions.

Limitations of the study

Our work is subject to certain limitations:

(1) In this study, we did not implement the separation of the original image recognition model and the cropped image recognition model. Consequently, the same weight is used to predict both the small target with a large background and the large target with a small background. This approach could benefit from further improvement to enhance recognition accuracy.

(2) The network model we pretrained for this study was trained using only three experimental datasets. To enhance the practical performance of our pre-trained model, we recognize the need to engage with a wider range of experimental scenarios and incorporate more diverse training data. For instance, mice implanted with optical fibers or electrodes are frequently used in neuroscience experiments to facilitate closed-loop real-time studies.[46] Nevertheless, accurate pose estimation in such scenarios poses a significant challenge due to the obscured head and difficult detection. Therefore, expanding our research to tackle these challenges by integrating this type of dataset will greatly enhance key-point detection in neuroscience experiments.

(3) We conducted key-point detection tasks involving 3, 6, and 11 key points. The detection of 11 key points significantly enhances the amount of information available compared to just 3 key points. This expanded set of key points enables a more in-depth analysis of intricate mouse behaviors such as leaning and turning, thereby propelling the software beyond the capabilities of existing commercially available models. As the key-point quantity being constrained by video quality and mouse size, integrating video restoration techniques with STPoseNet will enhance the efficacy of multi-key-point detection in challenging situations with low-quality data and complex environments.

Conclusions

Our study presents a fast, robust, and well-adapted model for the analysis of laboratory mouse behavioral experiments. To further enhance the performance and applicability of our model, future research should address the limitations. By addressing these areas of improvement, we can continue to advance the field of animal behavior analysis and provide valuable tools for researchers in various fields.

STAR★METHODS

Detailed methods are provided in the online version of this paper and include the following:

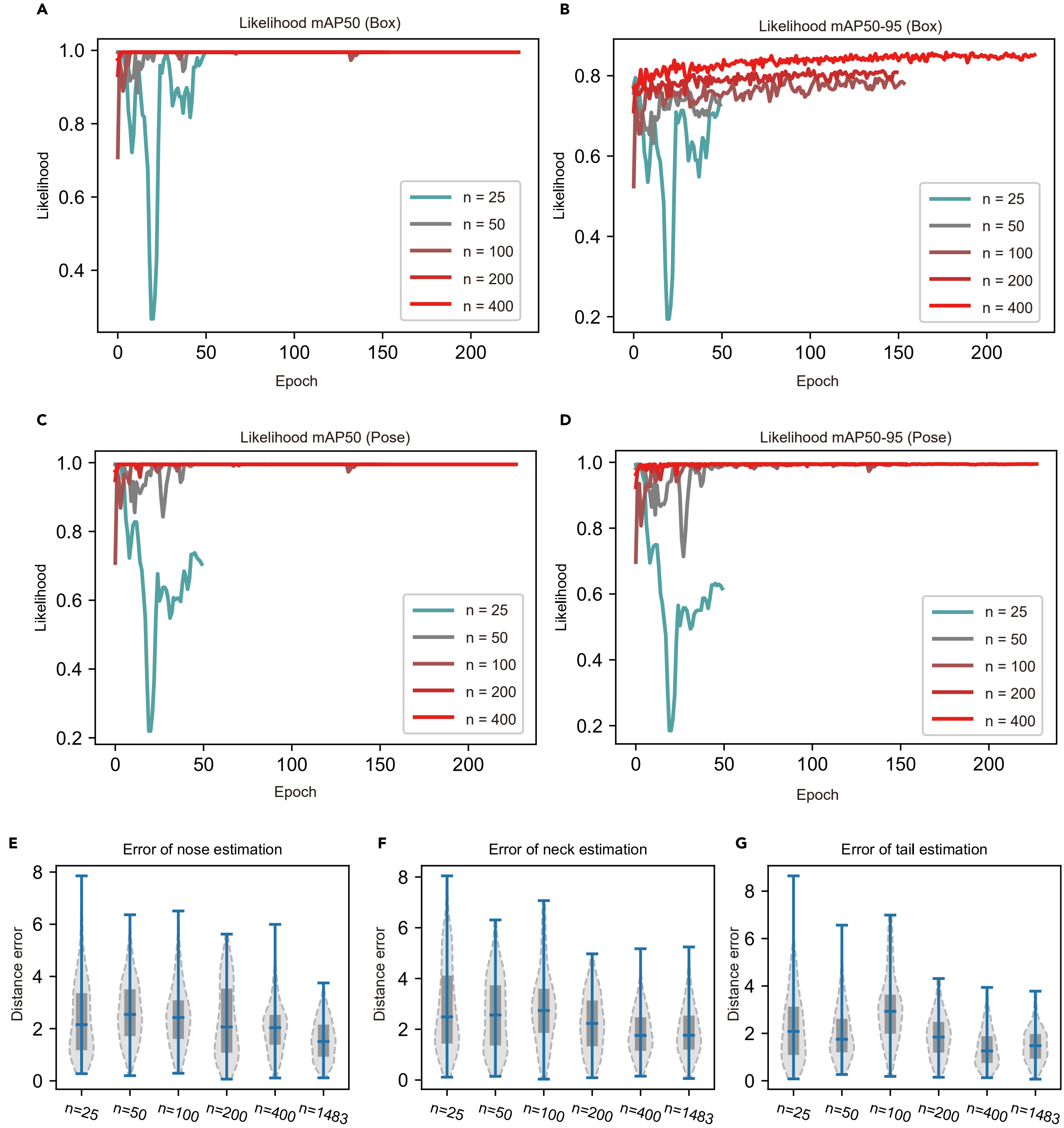


Figure 10. Comparison of different training dataset sizes

(A–D) The datasets are from open field experiment (3 key points). The likelihood changes of mAP50 (Box) (A), mAP50-95 (Box) (B), mAP50 (Pose) (C), and mAP50-95 (Pose) (D) along with epochs during model training. Five training data sizes, from *n* = 25 to *n* = 400 images, are compared for the performance.
(E–G) The distance errors of nose (E), neck (F) and tail (G) estimation are compared for the six training data sizes, from *n* = 25 to *n* = 1,483 images.

- KEY RESOURCES TABLE
- RESOURCE AVAILABILITY
  - Lead contact
  - Materials availability
  - Data and code availability

- EXPERIMENTAL MODEL AND STUDY PARTICIPANT DETAILS
  - ○ Dataset of open field experiment
  - ○ Dataset of water maze experiment
  - ○ Dataset annotation
- METHOD DETAILS
  - ○ Neural network architecture
  - ○ Overall process of the tracking-cropping module
  - ○ Tracking and cropping process for mouse detection
  - ○ Data enhancement for TCM
  - ○ Estimation of error ratio
  - ○ Prediction module for missing pose data
  - ○ Performance evaluation
- QUANTIFICATION AND STATISTICAL ANALYSIS

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.isci.2024.109772.

ACKNOWLEDGMENTS

The authors are grateful to Dr. Wenjun Jin for his comments and suggestions on the manuscript. This work was supported by grants from the National Natural Science Foundation of China (no. 32171096, 31925018, and 32127801), Guangxi Science and Technology Base & Talents Fund (GUIKE AD22035948).

AUTHOR CONTRIBUTIONS

X.C. and X.L. contributed to the design of the study. S.L. and J.W. performed the experiments and acquired the data. S.L., X.C., and X.L. designed the method. S.L. and J.W. processed the datasets. X.L. wrote the manuscript with help from all the other authors.

DECLARATION OF INTERESTS

The authors declare no competing interests.

Received: December 30, 2023
Revised: March 15, 2024
Accepted: April 15, 2024
Published: April 18, 2024

REFERENCES

1. Hajar, R. (2011). Animal testing and medicine. Heart Views *12*, 42. https://doi.org/10.4103/1995-705X.81548.
2. Datta, S.R., Anderson, D.J., Branson, K., Perona, P., and Leifer, A. (2019). Computational Neuroethology: A Call to Action. Neuron *104*, 11–24. https://doi.org/10.1016/j.neuron.2019.09.038.
3. Mathis, A., Schneider, S., Lauer, J., and Mathis, M.W. (2020). A Primer on Motion Capture with Deep Learning: Principles, Pitfalls, and Perspectives. Neuron *108*, 44–65. https://doi.org/10.1016/j.neuron.2020.09.017.
4. Mathis, M.W., and Mathis, A. (2020). Deep learning tools for the measurement of animal behavior in neuroscience. Curr. Opin. Neurobiol. *60*, 1–11. https://doi.org/10.1016/j.conb.2019.10.008.
5. Kafkafi, N., Agassi, J., Chesler, E.J., Crabbe, J.C., Crusio, W.E., Eilam, D., Gerlai, R., Golani, I., Gomez-Marin, A., Heller, R., et al. (2018). Reproducibility and replicability of rodent phenotyping in preclinical studies. Neurosci. Biobehav. Rev. *87*, 218–232. https://doi.org/10.1016/j.neubiorev.2018.01.003.
6. Anderson, D.J., and Perona, P. (2014). Toward a science of computational ethology. Neuron *84*, 18–31. https://doi.org/10.1016/j.neuron.2014.09.005.
7. Ottenheimer, D.J., Bari, B.A., Sutlief, E., Fraser, K.M., Kim, T.H., Richard, J.M., Cohen, J.Y., and Janak, P.H. (2020). A quantitative reward prediction error signal in the ventral pallidum. Nat. Neurosci. *23*, 1267–1276. https://doi.org/10.1038/s41593-020-0688-5.
8. Okubo, T.S., Patella, P., D'Alessandro, I., and Wilson, R.I. (2020). A Neural Network for Wind-Guided Compass Navigation. Neuron *107*, 924–940.e18. https://doi.org/10.1016/j.neuron.2020.06.022.
9. Schorscher-Petcu, A., Takacs, F., and Browne, L.E. (2021). Scanned optogenetic control of mammalian somatosensory input to map input-specific behavioral outputs. Elife *10*, e62026. https://doi.org/10.7554/eLife.62026.
10. Warren, R.A., Zhang, Q., Hoffman, J.R., Li, E.Y., Hong, Y.K., Bruno, R.M., and Sawtell, N.B. (2021). A rapid whisker-based decision underlying skilled locomotion in mice. Elife *10*, e63596. https://doi.org/10.7554/eLife.63596.
11. Keshavarzi, S., Bracey, E.F., Faville, R.A., Campagner, D., Tyson, A.L., Lenzi, S.C., Branco, T., and Margrie, T.W. (2022). Multisensory coding of angular head velocity in the retrosplenial cortex. Neuron *110*, 532–543.e9. https://doi.org/10.1016/j.neuron.2021.10.031.
12. Isik, S., and Unal, G. (2023). Open-source software for automated rodent behavioral analysis. Front. Neurosci. *17*, 1149027. https://doi.org/10.3389/fnins.2023.1149027.
13. Luxem, K., Sun, J.J., Bradley, S.P., Krishnan, K., Yttri, E., Zimmermann, J., Pereira, T.D., and Laubach, M. (2023). Open-source tools for behavioral video analysis: Setup, methods, and best practices. Elife *12*, e79305. https://doi.org/10.7554/eLife.79305.
14. Nevatia, K., and Binford, T.O. (1973). Structured descriptions of complex objects. In Proceedings of the 3rd International Joint Conference on Artificial Intelligence, pp. 641–647.
15. Marr, D., and Nishihara, H.K. (1978). Representation and recognition of the spatial organization of three-dimensional shapes. Proc. R. Soc. Lond. B Biol. Sci. *200*, 269–294. https://doi.org/10.1098/rspb.1978.0020.

16. Berman, G.J., Choi, D.M., Bialek, W., and Shaevitz, J.W. (2014). Mapping the stereotyped behaviour of freely moving fruit flies. J. R. Soc. Interface *11*, 20140672. https://doi.org/10.1098/rsif.2014.0672.
17. Wiltschko, A.B., Johnson, M.J., Iurilli, G., Peterson, R.E., Katon, J.M., Pashkovski, S.L., Abraira, V.E., Adams, R.P., and Datta, S.R. (2015). Mapping Sub-Second Structure in Mouse Behavior. Neuron *88*, 1121–1135. https://doi.org/10.1016/j.neuron.2015.11.031.
18. Vogelstein, J.T., Park, Y., Ohyama, T., Kerr, R.A., Truman, J.W., Priebe, C.E., and Zlatic, M. (2014). Discovery of brainwide neural-behavioral maps via multiscale unsupervised structure learning. Science *344*, 386–392. https://doi.org/10.1126/science.1250298.
19. Berman, G.J., Bialek, W., and Shaevitz, J.W. (2016). Predictability and hierarchy in Drosophila behavior. Proc. Natl. Acad. Sci. USA *113*, 11943–11948. https://doi.org/10.1073/pnas.1607601113.
20. Wang, Q., Taliaferro, J.M., Klibaite, U., Hilgers, V., Shaevitz, J.W., and Rio, D.C. (2016). The PSI-U1 snRNP interaction regulates male mating behavior in Drosophila. Proc. Natl. Acad. Sci. USA *113*, 5269–5274. https://doi.org/10.1073/pnas.1600936113.
21. Klibaite, U., Berman, G.J., Cande, J., Stern, D.L., and Shaevitz, J.W. (2017). An unsupervised method for quantifying the behavior of paired animals. Phys. Biol. *14*, 015006. https://doi.org/10.1088/1478-3975/aa5c50.
22. Cande, J., Namiki, S., Qiu, J., Korff, W., Card, G.M., Shaevitz, J.W., Stern, D.L., and Berman, G.J. (2018). Optogenetic dissection of descending behavioral control in Drosophila. Elife *7*, e34275. https://doi.org/10.7554/eLife.34275.
23. Couto, J., Linaro, D., De Schutter, E., and Giugliano, M. (2015). On the Firing Rate Dependency of the Phase Response Curve of Rat Purkinje Neurons In Vitro. PLoS Comput. Biol. *11*, e1004112. https://doi.org/10.1371/journal.pcbi.1004112.
24. Hu, Y., Ferrario, C.R., Maitland, A.D., Ionides, R.B., Ghimire, A., Watson, B., Iwasaki, K., White, H., Xi, Y., Zhou, J., and Ye, B. (2023). LabGym: Quantification of user-defined animal behaviors using learning-based holistic assessment. Cell Rep. Methods *3*, 100415. https://doi.org/10.1016/j.crmeth.2023.100415.
25. Tompson, J., Jain, A., LeCun, Y., and Bregler, C. (2014). Joint training of a convolutional network and a graphical model for human pose estimation. Adv. Neural Inf. Process. Syst. *27*.
26. Toshev, A., and Szegedy, C. (2014). DeepPose: Human Pose Estimation via Deep Neural Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1653–1660.
27. Carreira, J., Agrawal, P., Fragkiadaki, K., and Malik, J. (2016). Human Pose Estimation with Iterative Error Feedback. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4733–4742.
28. Wei, S.E., Ramakrishna, V., Kanade, T., and Sheikh, Y. (2016). Convolutional Pose Machines. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 4724–4732.
29. Tome, D., Russell, C., and Agapito, L. (2017). Lifting from the Deep: Convolutional 3D Pose Estimation from a Single Image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5689–5698.
30. Cao, Z., Hidalgo, G., Simon, T., Wei, S.E., and Sheikh, Y. (2021). OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields. IEEE T. Pattern Anal. *43*, 172–186. https://doi.org/10.1109/TPAMI.2019.2929257.
31. Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In Medical Image Computing and Computer-Assisted Intervention, *2015*, N. Navab, J. Hornegger, W. Wells, and A. Frangi, eds., pp. 234–241. https://doi.org/10.1007/978-3-319-24574-4_28.
32. Shelhamer, E., Long, J., and Darrell, T. (2017). Fully Convolutional Networks for Semantic Segmentation. IEEE T. Pattern Anal. *39*, 640–651. https://doi.org/10.1109/TPAMI.2016.2572683.
33. Andriluka, M., Pishchulin, L., Gehler, P., and Schiele, B. (2014). 2D Human Pose Estimation: New Benchmark and State of the Art Analysis. In Proceedings of the IEEE Conference on computer Vision and Pattern Recognition, pp. 3686–3693.
34. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C.L. (2014)held in Cham, 2014. In Microsoft COCO: Common Objects in Context, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds. (Springer International Publishing), pp. 740–755. https://doi.org/10.1007/978-3-319-10602-1_48.
35. Mathis, A., Mamidanna, P., Cury, K.M., Abe, T., Murthy, V.N., Mathis, M.W., and Bethge, M. (2018). DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. Nat. Neurosci. *21*, 1281–1289. https://doi.org/10.1038/s41593-018-0209-y.
36. Graving, J.M., Chae, D., Naik, H., Li, L., Koger, B., Costelloe, B.R., and Couzin, I.D. (2019). DeepPoseKit, a software toolkit for fast and robust animal pose estimation using deep learning. Elife *8*, e47994. https://doi.org/10.7554/eLife.47994.
37. Nath, T., Mathis, A., Chen, A.C., Patel, A., Bethge, M., and Mathis, M.W. (2019). Using DeepLabCut for 3D markerless pose estimation across species and behaviors. Nat. Protoc. *14*, 2152–2176. https://doi.org/10.1038/s41596-019-0176-0.
38. Pereira, T.D., Aldarondo, D.E., Willmore, L., Kislin, M., Wang, S.S.H., Murthy, M., and Shaevitz, J.W. (2019). Fast animal pose estimation using deep neural networks. Nat. Methods *16*, 117–125. https://doi.org/10.1038/s41592-018-0234-5.
39. Lauer, J., Zhou, M., Ye, S., Menegas, W., Schneider, S., Nath, T., Rahman, M.M., Di Santo, V., Soberanes, D., Feng, G., et al. (2022). Multi-animal pose estimation, identification and tracking with DeepLabCut. Nat. Methods *19*, 496–504. https://doi.org/10.1038/s41592-022-01443-0.
40. Pereira, T.D., Tabris, N., Matsliah, A., Turner, D.M., Li, J., Ravindranath, S., Papadoyannis, E.S., Normand, E., Deutsch, D.S., Wang, Z.Y., et al. (2022). SLEAP: A deep learning system for multi-animal pose tracking. Nat. Methods *19*, 486–495. https://doi.org/10.1038/s41592-022-01426-1.
41. Fang, C., Zhang, T., Zheng, H., Huang, J., and Cuan, K. (2021). Pose estimation and behavior classification of broiler chickens based on deep neural networks. Comput. Electron. Agric. *180*, 105863. https://doi.org/10.1016/j.compag.2020.105863.
42. Marks, M., Jin, Q., Sturman, O., von Ziegler, L., Kollmorgen, S., von der Behrens, W., Mante, V., Bohacek, J., and Yanik, M.F. (2022). Deep-learning-based identification, tracking, pose estimation and behaviour classification of interacting primates and mice in complex environments. Nat. Mach. Intell. *4*, 331–340. https://doi.org/10.1038/s42256-022-00477-5.
43. Li, J., Keselman, M., and Shlizerman, E. (2023). OpenLabCluster: Active Learning Based Clustering and Classification of Animal Behaviors in Videos Based on Automatically Extracted Kinematic Body Keypoints. Preprint at bioRxiv. https://doi.org/10.1101/2022.10.10.511660.
44. Han, Y., Chen, K., Wang, Y., Liu, W., Wang, Z., Wang, X., Han, C., Liao, J., Huang, K., Cai, S., et al. (2024). Multi-animal 3D social pose estimation, identification and behaviour embedding with a few-shot learning framework. Nat. Mach. Intell. *6*, 48–61. https://doi.org/10.1038/s42256-023-00776-5.
45. Luxem, K., Mocellin, P., Fuhrmann, F., Kürsch, J., Miller, S.R., Palop, J.J., Remy, S., and Bauer, P. (2022). Identifying behavioral structure from deep variational embeddings of animal motion. Commun. Biol. *5*, 1267. https://doi.org/10.1038/s42003-022-04080-7.
46. Király, B., Balázsfi, D., Horváth, I., Solari, N., Sviatkó, K., Lengyel, K., Birtalan, E., Babos, M., Bagaméry, G., Máthé, D., et al. (2020). In vivo localization of chronically implanted electrodes and optic fibers in mice. Nat. Commun. *11*, 4686. https://doi.org/10.1038/s41467-020-18472-y.
47. Qin, H., Fu, L., Hu, B., Liao, X., Lu, J., He, W., Liang, S., Zhang, K., Li, R., Yao, J., et al. (2018). A Visual-Cue-Dependent Memory Circuit for Place Navigation. Neuron *99*, 47–55.e44. https://doi.org/10.1016/j.neuron.2018.05.021.
48. Maji, D., Nagori, S., Mathew, M., and Poddar, D. (2022). YOLO-Pose: Enhancing YOLO for Multi Person Pose Estimation Using Object Keypoint Similarity Loss. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2637–2646.
49. Kovvali, N., Banavar, M., and Spanias, A. (2014). The Kalman Filter. In An Introduction to Kalman Filtering with MATLAB Examples, N. Kovvali, M. Banavar, and A. Spanias, eds. (Springer International Publishing), pp. 23–41. https://doi.org/10.1007/978-3-031-02536-5_3.
50. Chen, Z., Zhang, R., Fang, H.S., Zhang, Y.E., Bal, A., Zhou, H., Rock, R.R., Padilla-Coreano, N., Keyes, L.R., Zhu, H., et al. (2023). AlphaTracker: a multi-animal tracking and behavioral analysis tool. Front. Behav. Neurosci. *17*, 1111908. https://doi.org/10.3389/fnbeh.2023.1111908.

STAR★METHODS

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited data		
VAME dataset	Luxem et al.[45]	https://figshare.com/articles/media/VAME_Data/19213272
Software and algorithms		
Python 3.8.11	Python Software Foundation	https://www.python.org/downloads/release/python-3811/
Opencv-Python 4.7.0	Opencv software	https://opencv.org/blog/release/opencv-4-7-0/
Pytorch 1.90	Pytorch software	https://pytorch.org/blog/pytorch-1.9-released/
Matplotlab 3.4.3	Matplotlab software	https://matplotlab.org/3.4.3/
Deeplabcut	Mathis et al.[35]	https://github.com/DeepLabCut/DeepLabCut
SLEAP	Pereira et al.[40]	https://sleap.ai/
YOLOv8	Maji et al.[48]	https://docs.ultralytics.com/
Other		
Source code	Github	https://github.com/lvrgb777/STPoseNet

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources should be directed to and will be fulfilled by the lead contact, Xiang Liao (xiang.liao@cqu.edu.cn).

Materials availability

This study did not generate new unique reagents.

Data and code availability

- The data that support the findings of this study are available on request from the lead contact.
- All original code has been deposited at GitHub and is publicly available as of the date of publication. The URL is listed in key resources table.

EXPERIMENTAL MODEL AND STUDY PARTICIPANT DETAILS

All the experimental procedures were performed in accordance with institutional animal welfare guidelines and were approved by the Third Military Medical University Animal Care and Use Committee. The ethics approval: No. AMUWEC20210251.

Dataset of open field experiment

In this experiment, adult C57BL/6J mice (male, 6-8 weeks of age) were used for predation habit experiments. In the experiment, we implemented a real-time interactive platform measuring 800 mm in diameter and 300 mm in height (Figure 3C, top). This platform consisted of artificial prey in the form of round neodymium magnets, measuring 40 mm in diameter and 10 mm in height, as well as food pellets (5 × 0.2 mg, Bio-serve, USA) that were magnetized to simulate an attempt to escape from approaching hungry mice. Prior to each trial, the mice were fasted for 12 hours to reduce their body weight to 90-95% of their normal weight. During this fasting period, the mice had free access to water. Following each trial, a recovery period of 24 hours was provided, during which the mice were given food and water. To monitor the positions of the mouse and prey in real-time (30 fps), a webcam was mounted on top of the platform.

For the creation of our training and validation sets, we used 5 experimental videos that had a combined duration of 3 minutes and 40 seconds. From these videos, we extracted a total of 1647 images to form the dataset (training and validation) for 3 key-points and 300 images to establish the dataset (training and validation) for 11 key-points. In the testing dataset, we utilized a 7-second video comprising 221 frames to compare and evaluate the final pose estimation results. The lengths of training and testing datasets were determined based on the course of specific mouse behavior in the open field experiments. The images from the open field experiment dataset exhibited good image quality, with a significant color difference between the background and the mouse, and a clear outline of the mouse's edges. Additionally, since the experiment was conducted in an open field environment, the mice exhibited a strong motivation to move.

Dataset of water maze experiment

In this experiment, adult male mice C57BL/6J (aged 2-3 months at the start of the experiment) were given free access to food and water and lived in a 12-hour light/diurnal cycle (lights turned on at 7 am). In the experiment, we conducted experiments in a circular swimming pool measuring 120 cm in diameter and 40 cm in height. The pool contained opaque water (maintained at 20 ± 1°C) achieved by adding titanium dioxide. The mice were trained to escape onto a hidden platform positioned 30 cm away from the pool wall and 1 cm below the water surface. Visual cues were placed on the pool wall to provide spatial references for the platform's location. Importantly, the platform's position remained fixed throughout all the training trials.[47]

For the creation of our training and validation set, we utilized 6 experimental videos with a combined duration of 3 minutes and 38 seconds. From these videos, we extracted a total of 1392 images to create our dataset (training and validation) for 3 key-points. In the test dataset, we used a 20-second video comprising 500 frames of images to compare and evaluate the final recognition results. The lengths of training and testing datasets were determined based on the course of specific mouse behavior in the water maze experiments.

In the water maze experiment (Figure 3C, bottom), we encountered challenges due to poor lighting conditions and the maze partially obstructing the mouse's body. Specifically, when the mouse turned, its head became completely immersed in water. These factors posed difficulties for testing the reliability and accuracy of our pose estimation method.

Dataset annotation

To ensure accurate annotations, two experienced annotators independently labeled the key-points of the mouse (nose, neck, and tail etc) in each image. Their labeled results were then compared, and a final consensus was reached to produce the ground truth for the dataset.

The length of the videos in the training dataset was determined according to the specifications of the animal behavior experiments, such as when the mouse captured the target in the open field pursuit experiment or when the mouse located the platform in the water maze experiment. There was no human intervention with the animals during these experiments. We utilized the napari module in DeepLabCut for annotating the key-points of the mouse and python-labelme for annotating the object bounding boxes. To create 3 key-points on the mouse body, we initially labeled the nose, neck, and tail (Figure 3A). Subsequently, we extended the key-point labeling to include the nose, left and right ears, five positions evenly distributed along the mouse spine, as well as the left and right hind legs, and the tail (Figure 3B). Subsequently, we converted the dataset into the format required by each model, while ensuring the consistency of the dataset. The training and testing datasets are derived from different videos, which ensures the authenticity of the model evaluation.

METHOD DETAILS

Neural network architecture

The STPoseNet model is designed based on YOLOv8-pose and added two new modules, one is the TCM and another is Kalman filter. The TCM is used in estimation with data enhancement matching for model training. STPoseNet is an end-to-end model in the process of model training and video processing, which ensures the stability in real applications.

In the architecture of STPoseNet, we added the proposed modules based on YOLOv8 and adjusted model parameters. The backbone and head of YOLOv8 were not adjusted and used its default parameters (4.37 million parameters).[48] The backbone part passes through the five-layer convolutional network, and then the head part connects to the TCM, and the TCM outputs the final prediction result (Figure 1A). We set batch size as 8, max epoch as 400, intersection over union (IoU) as 0.7 in our model training.

Overall process of the tracking-cropping module

The tracking-cropping module (TCM) is an approach we proposed to leverage time-domain information in mouse behavior videos. It achieves the process by enlarging and cropping the image area around the detected mouse, thereby enhancing the prediction capability, especially in extreme cases. During the video recognition process by the model, the TCM module coordinates and controls the recognized images and recognition strategies (Figure 1).

When a recognition task begins, or when the cropped image (Figure 1A, right image) fails to recognize the mouse (determined by a likelihood lower than a predefined value), the original image is used for pose estimation (Figure 1B, green line). After obtaining the recognition results (bounding box and key-points) through the Backbone and Head modules, these results are marked and generated (Figure 1B, frame output). Simultaneously, the coordinates are forwarded to the next frame for the annotation and image cropping (Figure 1B, cropping image). The cropped image is then passed to the Backbone and Head model for key-point detection (Figure 1B, blue line). The obtained coordinates need to be converted back to the coordinates of the original image, and the next frame is marked and cropped as well. Moreover, when the mouse is not recognized in the original image, Kalman filtering is employed to predict key-point locations (indicated by red lines). Overall, the TCM module plays a crucial role in optimizing the recognition process, ensuring improved performance in mouse behavior analysis.

Tracking and cropping process for mouse detection

Our main strategy for improving recognition accuracy is to track the cropped mouse region (Figure 1B, blue line). In many behavioral experimental scenarios, the mouse only occupies a small portion of the image, resulting in a small target with a large background. One challenge in improving accuracy lies in the fact that deep learning models require a fixed input size (in our model, 640 × 640). As a result, general behavioral recognition images need to be scaled down for the input, which reduces the target (i.e., the mouse itself) that is relatively small. Additionally, when the identified targets and key-points are generated, they need to be scaled back up, which amplifies errors in equal proportion. Hence it is a major obstacle in improving accuracy in current pose estimation methods.

To address this, we leverage the short interval time between adjacent frames and the limited displacement speed of the mouse in reality. Based on the mouse's position and range in the previous frame (Figure 2A, top image), we can obtain the possible range of the mouse in the next frame (Figure 2B, middle image) by expanding the range (currently, doubling it). Using this range, we crop the next frame image to obtain a new input image with a larger target and smaller background (Figure 2B, bottom image). The new image is then fed into the model for recognition. The obtained key-point coordinates are converted to actual coordinates (Equations 1, 2, and 3), completing a complete round of tracking and cropping recognition (Figure 2C). This approach not only avoids the information loss caused by scaling down the image before input (i.e., fewer image pixels occupied by the mouse), but also mitigates error amplification caused by scaling up.

Data enhancement for TCM

Due to the varying proportions of images occupied by the mouse after tracking and cropping, it is crucial to adapt to this situation by expanding the dataset. This expansion allows the model to effectively recognize images with large targets and small backgrounds. To achieve this, we perform a specific operation of cropping and transforming the original data's annotated images and labels according to the predetermined tracking crop magnification ratio. By performing this operation, we can ensure that the model is exposed to a more diverse range of training examples, facilitating its ability to accurately identify and classify images with different target-to-background ratios. This approach contributes to the model's robustness and overall performance in recognizing and tracking mouse behavior. The transformation formula is:

$$Img_{cut}^{i+1} = Img_{orig}^{i+1}\left[Box_{mouse_x}{}_{orig}^{i} - Tf_x{}_{orig}^{i} : Box_{mouse_x}{}_{orig}^{i} + Tf_x{}_{orig}^{i}, Box_{mouse_y}{}_{orig}^{i} - Tf_y{}_{orig}^{i} : Box_{mouse_y}{}_{orig}^{i} + Tf_y{}_{orig}^{i}\right] \quad \text{(Equation 1)}$$

$$Tf_x{}_{orig}^{i} = Box_{mouse_x}{}_{orig}^{i} - \gamma * Box_{mouse_{width}}{}_{orig}^{i} \quad \text{(Equation 2)}$$

$$Tf_y{}_{orig}^{i} = Box_{mouse_y}{}_{orig}^{i} - \gamma * Box_{mouse_{length}}{}_{orig}^{i} \quad \text{(Equation 3)}$$

where Img is image information, Box_{mouse_x}, Box_{mouse_y}, $Box_{mouse_{width}}$, $Box_{mouse_{length}}$ are the central coordinates (x and y) of the box and the width and length of the box, respectively.

Estimation of error ratio

As illustrated in Figure 2, the original frame's image size is 640 × 480. The size of the mouse itself is approximately 50 × 40, and the size of the original image is around 100 × 80. Since the input of model has a fixed size of 640 × 640 pixels, images of varying sizes are resized to fit this dimension and then restored after the recognition process is completed. Without the TCM, the conversion ratio for the input image obtained during the recognition process can be calculated as k1 = (Area1)/(Area2), where Area1 represents the image area size of the model input, and Area2 denotes the area size of the original image. This ratio accounts for the transformation necessary to map the original image to the fixed model size. By considering this conversion ratio, the model can effectively analyze the resized image and make accurate recognition, ensuring reliable tracking of the mouse behavior. Assuming that the identification error is μ_1, the final error is k1 × μ_1. When passing through the TCM, the conversion ratio of the input image obtained by the recognition process is k2 = (Area1)/(Area3), where Area3 is the cropped area size that is generally 2 × 2 times of the size of the mouse target frame. Assuming the error is still μ_2 (the error will be less than μ_1 in the real case), the final error size is μ_2/k2. When the difference of error size between μ_1 and μ_2 is ignored, the optimization magnification is about k2/k1(Area2/Area3).

Therefore, the specific error optimization ratio K is:

$$K = \frac{S1}{S2} \quad \text{(Equation 4)}$$

where S1 is the original image size and S2 is the cropped size is generally 2 × 2 times the size of the mouse target frame.

The optimization effect becomes more pronounced as the proportion of mouse in the original image decreases. This observation was particularly evident when we compared the final results of the open field and water maze experiments.

Prediction module for missing pose data

In addition to the TCM module, we also account for various extreme situations that may occur during the actual experiments. These situations can lead to ineffective image recognition, resulting in missing tracking results. To prevent this, we employ a Kalman filter to calculate the characteristic parameters of the recognition results from several previous frames. When the confidence of the recognition results is extremely low (indicating a low confidence in the recognition box or key-points), it signifies that the recognition results are unacceptable, then a Kalman filter is applied:

$$x_n = F_n x_{n-1} + v_n \quad \text{(Equation 5)}$$

$$y_n = H_n x_n + w_n \quad \text{(Equation 6)}$$

where F_n is the $D \times D$ state-transition matrix, v_n is a $D \times 1$ Gaussian random state noise vector with zero mean and covariance matrix Q_n, H_n is the $M \times D$ measurement matrix, and w_n is a $M \times 1$ Gaussian random measurement noise vector with zero mean and covariance matrix R_n. The state space model (Equation 5) describes a dynamical system in which the state evolution and measurement processes are both linear and Gaussian. Using the time-domain information to identify the key-point position change before, the mouse position of this frame is identified to prevent missing data. By implementing the Kalman filter, we aim to ensure that such unreliable recognition results are detected and appropriately handled.[49,50] This helps to improve the overall accuracy and reliability of the tracking process, allowing for a more robust analysis of mouse behavior.

Performance evaluation

Previous studies have often used likelihood of recognition and the ratio of recognition pixel error to mouse body length as evaluation metrics for key-point recognition of animal postures. However, there are no precise criteria for comparing the size of the error, nor do they compare the final results of pose estimation, such as the head orientation of the mouse.

In our study, we address these limitations by defining the nose, neck, and tail beginning of the mouse as key-points for recognition. We introduce three measurement indicators to evaluate the accuracy of the pose estimation. The evaluation metrics include: (1) distance error (Equations 7 and 8), the linear distance between the three marked points (Figures 3A, A-B-C) and the identified key-points (Figure 3A, A′-B′-C′); (2) head deviation angle α (Figure 3A ∠A′B′M′) is the angle between nose and neck, and tail deviation angle β (Figure 3A, ∠C′B′N′) is the angel between neck and tail; (3) likelihood of calculating the three key-points of mouse.

$$\Delta = \Delta_{head} + \Delta_{nose} + \Delta_{tail} \quad \text{(Equation 7)}$$

$$\Delta_i = \sqrt{(x_{label} - x_{result})^2 + (y_{label} - y_{result})^2} \quad \text{(Equation 8)}$$

SLEAP employed an error threshold of approximately 3% for insect detection. However, considering the flexible nature of mice bodies where their body lengths can vary significantly, with the maximum length potentially exceeding twice the minimum length (Figure S4), we established 8% of the maximum body length as the distance error threshold. Adhering to this criterion, we derived the angle threshold by bisecting a mouse. The tangent value of the difference between head orientation and body orientation angles was set at double the value of 0.08, resulting in arctan0.16 (around 10°). Hence, we utilized a 10% threshold for angle error determination.

The formula for calculating RMSE (Root Mean Square Error):

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - f(x_i))^2} \quad \text{(Equation 9)}$$

To measure likelihood, mAP50 and mAP50-95 were calculated. The mAP is the average of AP over all detected classes, and AP is the area under the precision-recall curve. The mAP50 is the mean average precision calculated at an IoU threshold of 0.5. The mAP50-95 is the average mAP over different IoU thresholds, from 0.5 to 0.95, step 0.05.

QUANTIFICATION AND STATISTICAL ANALYSIS

To compare data among the different methods, we used one-way ANOVA to calculate the statistical significance. To compare data between two groups, we conducted two-sided Z-test to determine the statistical significance, with a threshold of $p < 0.05$ for the differences observed. In the figures, the violin plot represents the median (central mark), the entire range of the data (dashed line boundary), and the whiskers extend to the minimum and maximum values excluding outliers. The generation of violin plots was conducted using Matplotlib v3.4.3 software. All data processing in this article is done using scripts written in python.

For further statistical information regarding the experiments, please refer to the figure legends.