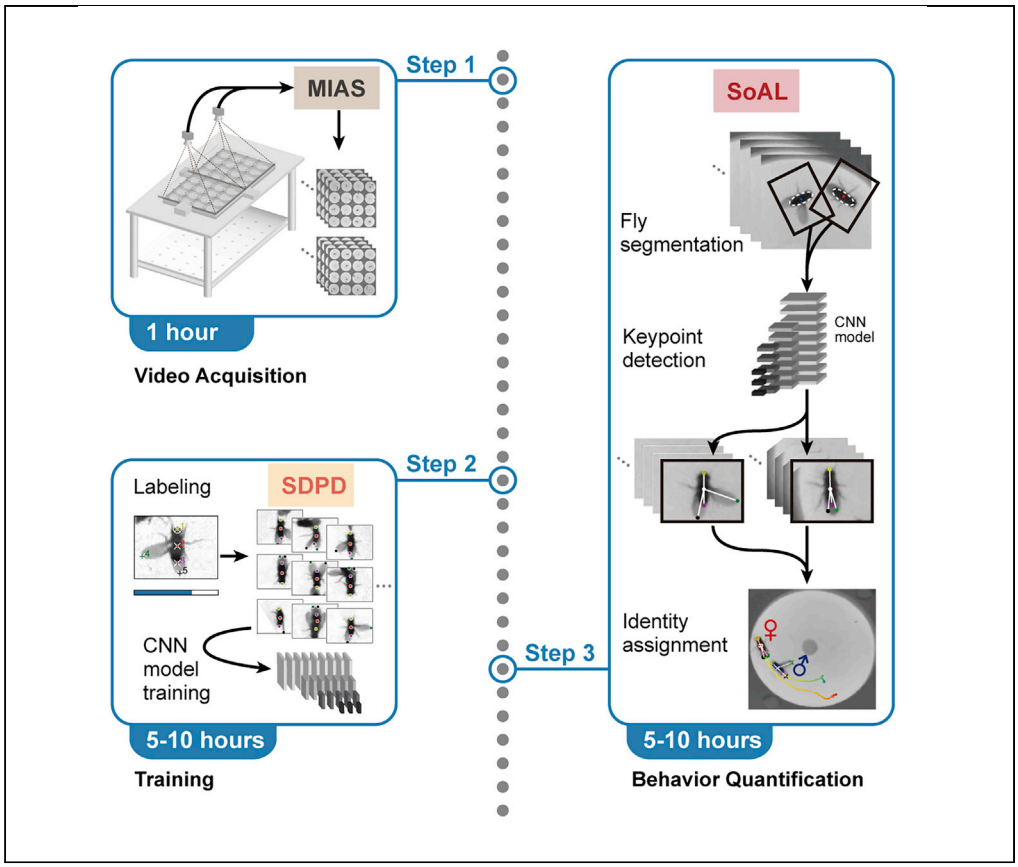
STAR Protocols

CellPress
OPEN ACCESS

Protocol

Protocol for quantitative ethology on natural social interactions in *Drosophila*



Quantitative ethology of social interactions between free-moving animals enables precise measurement of behavioral kinematics critical for various disciplines such as neuroscience. Here, we describe a set of tools for quantitative ethology of social interactions including the analysis pipeline SoAL, the training dataset SDPD, and the camera control software MIAS, along with experimental details. These tools are directly applicable for courtship behavior in Drosophila and can be used for other social interactions in other species after modification.

Publisher's note: Undertaking any experimental protocol requires adherence to local institutional guidelines for laboratory safety and ethics.

Jing Ning, Zhou Li, Xingjiang Zhang, ..., Qiong Liu, Yefeng Shao, Yi Sun

sunyi@westlake.edu.cn

Highlights

A protocol for quantification of natural social interactions in flies

Automatic pose estimation and behavior annotation of interacting flies

A unified image acquisition system for synchronized recording with multiple cameras

A labeled dataset of socially interacting flies along with manual labeling tool

Ning et al., STAR Protocols 3, 101621
September 16, 2022
https://doi.org/10.1016/j.xpro.2022.101621

STAR Protocols

CellPress
OPEN ACCESS

Protocol

Protocol for quantitative ethology on natural social interactions in *Drosophila*

Jing Ning,[1,2,3,4] Zhou Li,[1,2,3,4] Xingjiang Zhang,[1,2,3,4] Junlong Wang,[1,2,3,4] Dandan Chen,[2,3,4] Qiong Liu,[2,3,4] Yefeng Shao,[2,3,4] and Yi Sun[1,2,3,4,5,6,*]

[1]College of Life Sciences, Zhejiang University, Hangzhou, China

[2]Westlake Laboratory of Life Sciences and Biomedicine, Hangzhou, China

[3]Key Laboratory of Growth Regulation and Translational Research of Zhejiang Province, School of Life Sciences, Westlake University, Hangzhou, China

[4]Institute of Basic Medical Sciences, Westlake Institute for Advanced Study, Hangzhou, China

[5]Technical contact

[6]Lead contact

*Correspondence: sunyi@westlake.edu.cn
https://doi.org/10.1016/j.xpro.2022.101621

SUMMARY

Quantitative ethology of social interactions between free-moving animals enables precise measurement of behavioral kinematics critical for various disciplines such as neuroscience. Here, we describe a set of tools for quantitative ethology of social interactions including the analysis pipeline SoAL, the training dataset SDPD, and the camera control software MIAS, along with experimental details. These tools are directly applicable for courtship behavior in Drosophila and can be used for other social interactions in other species after modification. For complete details on the use and execution of this protocol, please refer to Ning et al. (2022).

BEFORE YOU BEGIN

Here, we explain in detail our experimental and analytical procedures of courtship behavior in *Drosophila*. This protocol enables precise ethological measurement of the behavioral dynamics of interacting pairs of flies.

Chamber fabrication

Timing: hours to days

The behavioral chambers are machined by laser. We used a 60 W CO_2 laser which is necessary for efficient processing of acrylic parts with 2–5 mm thickness. One array of chambers is made up of the following parts: a bottom plate, a chamber wall, a top plate, separators, and cover plates (Figure 1A). All the parts are machined with acrylic sheets except the separators, which are made of paper.

1. Acrylic parts.

Size of the transparent bottom plate is 96 mm by 96 mm. The center holes are 2 mm in diameter and spaced 22 mm apart, forming a 4 by 4 array.

Note: Since the side walls could impede the movements of the flies when they were interacting, we kept the females in the centers of the chambers by providing food, and the females were starved beforehand.

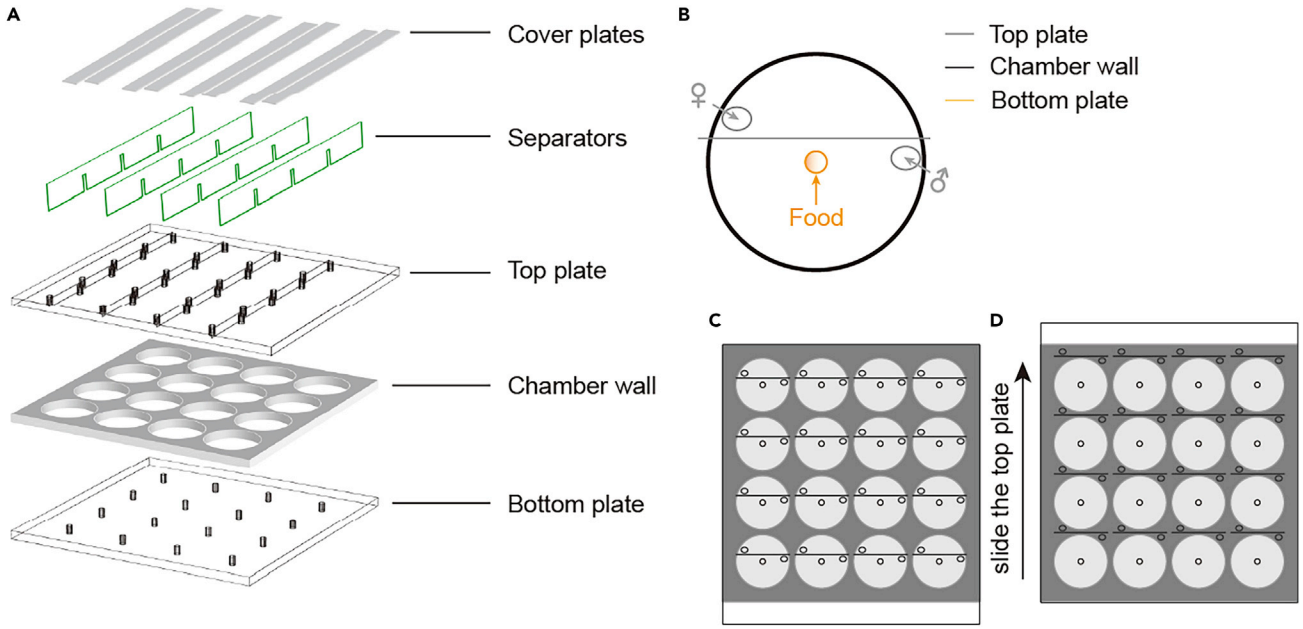


Figure 1. The chamber parts

(A) 5 parts of the chamber array (green: paper parts, other: acrylic parts).

(B) Top view of one chamber, composed of 3 layers (color-coded). The center hole for food is on the bottom plate. The two holes for introducing flies are on the top plate.

(C and D) Top view of the assembled chamber array. The state for introducing flies (C) and the state for recording (D).

The chamber wall is 96 mm by 96 mm in area and 3 mm in height. Each chamber is 20 mm in diameter and spaced 2 mm apart, also forming a 4 by 4 array.

⚠ CRITICAL: To avoid visual interference between adjacent chambers, the wall should be opaque.

Note: The number of chambers can be customized based on the field of view of the camera. The chamber diameter should not be less than 20 mm, otherwise the male will contact the chamber wall when courting the female at the center of the chamber.

The transparent top plate is 96 mm by 104.5 mm (Figures 1A and 1C). For each chamber, cut a slit on the top plate for the insertion of the separator (Figure 1B). The slit separates the chamber into two spaces and keeps the food hole to one side (Figure 1B). Cut two holes on each side for introducing flies.

⚠ CRITICAL: The top plate is designed longer than the chamber wall to form two working states (Figures 1C and 1D). The first state (Figure 1C) is used for introducing flies and the second state (Figure 1D) prevents the flies from fleeing during video recording.

The transparent cover plates (Figure 1A) can be any shape that cover one or a row of holes.

2. Paper parts.

Cut thin paper sheets as separators by laser. We used white recycled printer papers (about 0.1 mm thick).

⚠ CRITICAL: The shape of the separators must fit the slit of the chamber (Figure 1B). Insertion of the separator into the slit divides the chamber into two parts, one with food and the

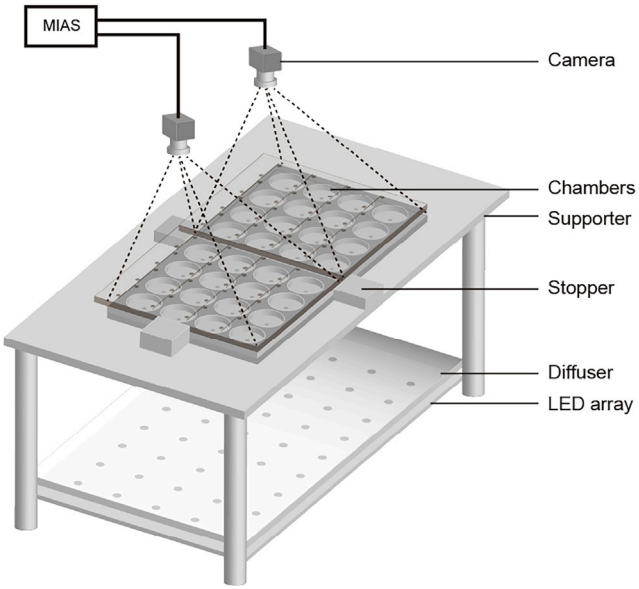


Figure 2. Experimental setup
Two chamber arrays (refer to Figure 1) are positioned on the supporter and fixed by the stopper. The LED array on the bottom illuminates the chambers. The diffuser softens the light. Each camera on the top captures one chamber array via the video acquisition software MIAS.

other without. The separator can be inserted into the four slits of each row on the top plate to avoid unintended interaction of the two flies.

Note: Paper parts become unsmooth after a long time of use. Replace the paper parts when necessary.

3. Food preparation.

Add 3.4 g (68 g/L) sucrose, 1 g (20 g/L) agarose, and 0.7 g (14 g/L) yeast into distilled water to prepare a 50 mL mixture. Heat the mixture by microwave until dissolution. Filter out undissolved yeast chunks.

Note: To avoid interference on fly body detection, the food filled in the center holes needs to be light-colored.

Experimental setup

Timing: hours to days

This step builds the experimental platform for acquiring videos of interacting flies.

4. Experimental platform.
   a. Set up the experimental platform on a stable board.
   b. Support a 300 mm by 150 mm by 5 mm white acrylic plate with four legs (Figure 2).
   c. Place two chamber arrays at the center of the supporter.
   d. Stick three acrylic blocks as stoppers on the supporter plate to fix the positions of the chamber arrays.

Note: The stopper on the left is taller than the top plate to align the left edges. The other two stoppers are as tall as the chamber wall to allow the sliding of the top plate.

Note: The entire platform should be readily moved into the incubator.

5. Illumination.
   a. Place a white-colored light emitting diode (LED) array (12 cm by 12 cm, 6 by 7 bulbs, 10 W total power) under the supporter (Figure 2).
   b. Set a white plastic sheet on the top of the LED array as a diffuser.
   c. Adjust the distance between the LED, the diffuser, and the chambers, so that all chambers are uniformly illuminated and the temperature inside the chambers is not affected by the heat emitted from the LED.

Optional: The white-colored LED can be replaced with an infrared LED array to perform experiments in the dark.

6. Video acquisition.

The videos are acquired with two cameras via the acquisition software MIAS. We describe below the preparation for video acquisition using FLIR (Teledyne FLIR) cameras as an example.

Note: We developed, MIAS, a Windows-based application written in C++, for simultaneous video acquisition using multiple cameras. Compared with other recording programs, MIAS is designed specifically for high throughput ethological experiments. First, MIAS supports cameras from various vendors, including some less supported ones. Second, MIAS provides unified control of multiple cameras from different vendors operating simultaneously. Third, MIAS records time stamps for each frame, facilitating synchronization.

   a. Download and install the camera driver. The camera driver corresponding to the executable MIAS on FLIR cameras is Spinnaker SDK v2.6.0.157.

⚠ CRITICAL: Install the Spinnaker SDK in "Application Developer" and with "Visual Studio 2015 Runtime Files" selected.

   b. Download and unzip the executable MIAS package (https://doi.org/10.6084/m9.figshare.17630585, based on Windows 10), or compile the source code (https://github.com/SunLabWestlake/MIAS, see the webpages for details) with Visual Studio 2017 (https://visualstudio.microsoft.com).

Note: The unzipped MIAS directory is like this:

```
- MIAS
    - CameraDrivers
        [The drivers of the four supported camera models]
    config.json
        [The configuration file of MIAS]
    MIAS_All.exe
    MIAS_FLIR.exe
    MIAS_HIKVISION.exe
    MIAS_DaHeng.exe
    MIAS_Basler.exe
    [Other "dll" files called by MIAS]
```

The folder "CameraDrivers" contains the installation packages of the four supported camera models. The "exe" files are MIAS executables for the four camera models respectively.

⚠ CRITICAL: For running MIAS, we recommend high clock speed CPU with at least 8 GB of RAM. The following hardware configurations have been tested: Intel Core i7 8700 CPU (3.2 GHz), 32 GB RAM.

Optional: Edit the configuration file of MIAS ("config.json" in the same directory of the "exe" file) to change the acquisition settings. Here is an example:

```
{
  "need_record": 0,
  "show_fps": 10,
  "fourcc": "DIVX",
  "cameras": {
    "KG0170070217": {
      "fps": 200,
      "name": "A"
    }
  }
}
```

Change the encoding format ("fourcc", "DIVX" and "MJPG" have been tested), the frame rate for previewing ("show_fps", use 0 to disable preview), the frame rate of the video ("fps"), and the camera name ("name") by modifying this file. "need_record" indicates whether to start the recording when opening MIAS (if it is set to 0, no recording will happen during preview). "fps" and "name" are subordinated to each camera configuration with the serial number ("KG0170070217", can be found by the camera configuration program included in the camera driver) as the key. The content must follow the "json" file format. If not so, an error message will appear when opening MIAS, and the default configuration will be used.

c. Install two cameras over the chambers (Figure 2).
d. Open the camera configuration program included in the camera driver (e.g., "SpinView.exe" for FLIR cameras). Set appropriate exposure time and acquisition frame rate.
e. Adjust the positions of the cameras to ensure that all the chambers fit within the field of view of the corresponding camera.
f. Execute MIAS_FLIR.exe to test the recording (Figure 3). Press "c" and "Enter" in the console window to start the recording. Press "q" and "Enter" to stop the recording and quit. Inspect the result videos in the folder "video" under the MIAS directory.

⚠ CRITICAL: If there are many "f" characters appearing in the console during acquisition, then the program dropped frames frequently. Reduce the acquisition frame rate with the camera configuration program, and close other nonessential processes. If the problem persists, upgrade the computer hardware.

Optional: Lens distortion should be corrected for lenses with short focal lengths. This is achieved by calibration and anti-distortion. Through calibration, we obtain the intrinsic and extrinsic parameters of the camera, which are used for computing the ideal coordinate of

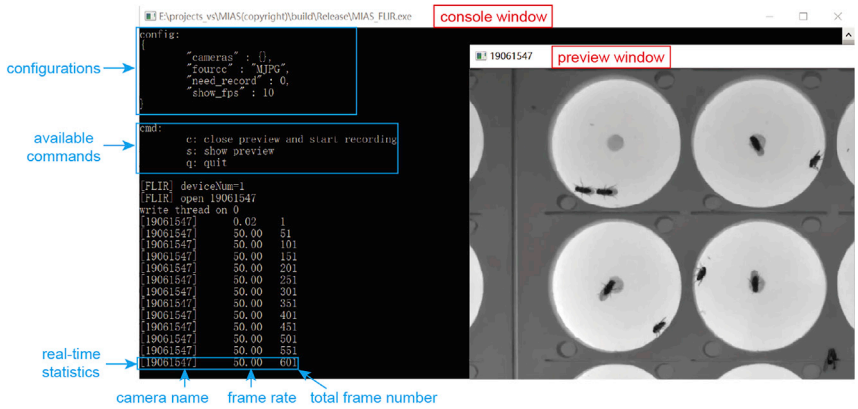


Figure 3. MIAS user interface for video acquisition with a FLIR camera

This is a screenshot when MIAS is running with a FLIR camera (in the experiments taken on the setup shown in Figure 2, there are two preview windows for the two cameras respectively). The main window is a Windows console. The preview window shows the real-time image of the camera (one for each camera). If the preview window is not showing, press "s" and "Enter" in the console window to show the preview window. Press "c" and "Enter" in the console window to close the preview window and start recording. Press "q" and "Enter" to quit MIAS.

the detected keypoints later (refer to step-by-step method details step 13). We have provided a script for anti-distortion calibration ("tools/calib/calib.py"). First, print a checkerboard picture and lay the picture flat on the chamber array. Second, capture an image of the checkerboard. Third, run the script "calib.py" to generate the calibration information (e.g., "tools/calib/calib_1_info.pickle", "1" is the camera name). This file will be used after keypoint detection ("SoAL_ID_Anno.py", refer to step-by-step method details step 15). Note that the setup should not be changed after calibration.

Note: Cover the LEDs on the camera models with black tapes to prevent light interference.

Fly husbandry

Timing: 3 weeks

This step describes the preparation of flies for experiments.

7. Preparing single-housing food.
   a. Fill 1.5 mL microcentrifuge tube with approximately 0.5 mL standard medium (about half of the height).
   b. Close the microcentrifuge tube and drill a small hole on the lid.
   c. Make separated grids with pieces of cardboard. Each grid accommodates one tube.

Note: Separate each tube with cardboard to prevent the flies from seeing each other, thus creating identical visual environments. Single-housing enhances the courtship arousal of the males during experiments.

⚠ CRITICAL: The hole on the lid should not be larger than the size of the fly body.

8. Collecting and single housing males.
   a. Rear fruit flies (*Drosophila melanogaster*) in the vials with standard cornmeal medium.
   b. Keep the rearing incubator at 22°C with 50% humidity, on a 12 h /12 h light/dark cycle.
   c. Prepare vials with small groups of the two sexes for breeding offspring.
   d. After 10–12 days, collect male pupae when it is easier to identify them by the sex combs.

e. Single-house the pupae in the microcentrifuge tubes with food for 7–10 days.

Note: Experiments will be performed 5–10 days after eclosion. Males of this age perform courtship robustly.

9. Group housing females.
   a. At the same time when preparing vials for breeding male pupae, prepare at least two times of vials for providing females (in the same incubator as the males).
   b. Discard the parents after 2–4 days.
   c. After 10–12 days (the same time as the male pupae eclosion), collect all the adult flies and put them together into another vial for group-housing. We put 10–15 females and males respectively in each vial.

Note: Same as single housing, group-housing lasts for 7–10 days. This ensures that the age of the interacting flies are similar.

⚠ CRITICAL: Make sure that each vial contains enough flies of both sexes, thus minimizing female receptivity and prolonging the courtship period during experiments.

Analysis setup

Timing: 2–3 days in total. Time for dataset collection and manual labeling varies depending on the dataset size. Training time depends on both the dataset size and the hardware performance

Timing: For step 11d, about 1 s for one keypoint

Timing: For step 12c, 5–10 h for SDPD-15k on the *Basic Hardware Configuration* mentioned above

Timing: For step 12d, less than 1 min on the *Basic Hardware Configuration*

This step describes SoAL installation and demonstrates network model training with either the provided SDPD-15k dataset or custom datasets (Figure 4 left part).

10. Software installation.
    a. Install PyTorch, follow the official instructions (https://pytorch.org/).

⚠ CRITICAL: SoAL requires GPUs to accelerate the training and keypoint detection. Thus CUDA compatible GPU needs to be installed. The following hardware configurations have been tested:

Basic Hardware Configuration: Intel Core i7 8700 (3.2 GHz) CPU, 32 GB RAM, nVIDIA GeForce GTX 1060 GPU.
Advanced Hardware Configuration: dual Intel Xeon E5-2678 v3 (2.5 GHz) CPUs, 128 GB RAM, dual nVIDIA GeForce RTX 2060 GPUs, 250 GB SSD.
   b. Clone the source code of SoAL (https://github.com/SunLabWestlake/SoAL).

Note: SoAL has been successfully tested on both Windows 10 and Ubuntu 18.04 operating systems.

Note: We employed the deep convolution network "HRNet"(Sun et al., 2019) for keypoints detection. The folder "hrnet" in the source repository is a clone of HRNet code

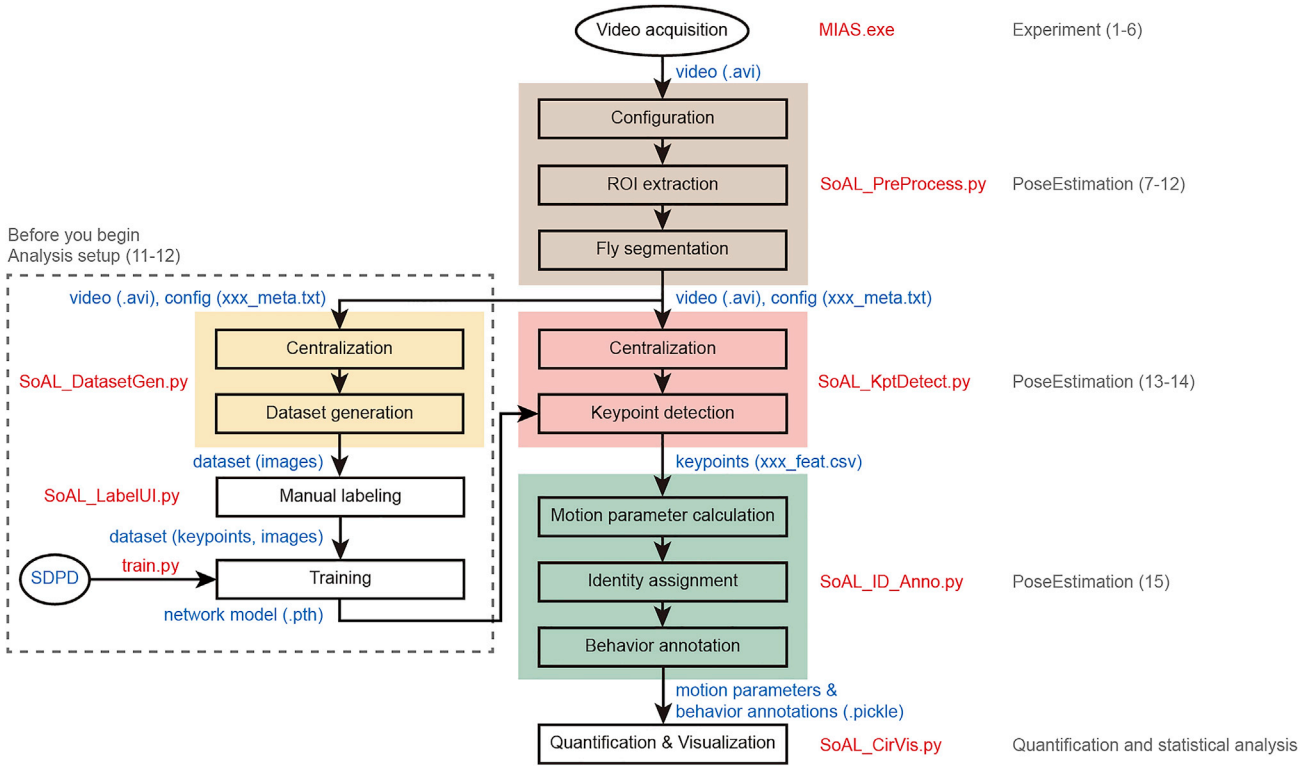


Figure 4. The workflow of the behavior analysis

The main workflow includes preprocessing (beige shade), keypoint detection (red shade), identity assignment and behavior annotation (green shade). The left part is the training process to provide a network model for keypoint detection. Red texts specify the programs and scripts to run. Blue texts are the files generated after processing. Gray texts show the corresponding steps described in this protocol.

(https://github.com/leoxiaobin/deep-high-resolution-net.pytorch). We modified the code to support different number of keypoints.

c. Install the required Python packages. Open a terminal in the root directory of the SoAL source code and execute the following command in the console:

```
>pip install -r requirements.txt
```

d. If the quality and lighting conditions of the acquired images are similar to those in our experiments (Figure 3. preview window, Figure 7), download the HRNet model trained on SDPD (hrnet_w32_SDPD-15k.pth, included in the tutorial data package: https://doi.org/10.6084/m9.figshare.19711729, under the folder "hrnet_model". If this model is used, the following steps ("11. Preparing dataset" and "12. Training") should be skipped. Also, utilizing the tutorial data, the behavior analysis workflow is demonstrated in Methods videos S1, S2, S3, and S4.

Optional: If you want to train your own model, follow steps 11a and 12.

11. Preparing dataset (Methods video S5).

This step prepares the dataset for training HRNet model.

Note: The keypoints detection is based on centralized animals (Figures 6A and 6B), which means the image input to the model keeps the fly body vertical (Figure 5A). Thus, the dataset includes a set of centralized images and an annotation file that stores the keypoint positions.

a. If the dataset SDPD-15k is suitable for the keypoints detection of your data, download SDPD-15k (https://doi.org/10.6084/m9.figshare.17624888) and unzip it to the folder "dataset" (under SoAL code directory). If SDPD-15k is used, skip the following steps and jump to step "12. Training". Otherwise, follow the steps below ("b" to "e") to generate custom datasets.

Note: The dataset SDPD-15k contains 15560 images (64 by 48 pixels) selected from 77 pairs of interacting flies (Figure 5B). We manually labeled 5 keypoints as head, center, tail, left wing tip, and right wing tip (Figure 5A).

b. Collect videos for generating dataset. Follow steps 1–7 of the step-by-step method details to acquire videos of interacting flies and generate configurations.
c. Sample and centralize the images from videos for inclusion into the dataset. Open a terminal in the root directory of the SoAL source code. Execute the following command in the console to sample some images and generate a dataset without keypoints.

```
>python tools/SoAL_DatasetGen.py DSNAME 10
```

The first parameter "DSNAME" is the name of the generated dataset, and the second parameter "10" specifies the number of images extracted from each fly (For each fly in each ROI, sample 10 frames uniformly within a video. The total number of images will be 10 × 2 × ROI number × video number).

Optional: Edit the following lines in the file "tools/SoAL_DatasetUtils.py" beforehand to specify the input and output path:

```
VIDEO_PATH = "video/"

DATASET_PATH = "dataset/"
```

VIDEO_PATH is the parent path containing the video folders to sample from (here use the folder "video" under SoAL code directory); DATASET_PATH is the path where the dataset will be stored (here use the folder "dataset" under SoAL code directory).

Note: This script creates a dataset consistent with the COCO dataset format (Lin et al., 2014). The images are stored in "dataset/images". And the keypoint positions (empty for now) are stored in "dataset/annotations/person_keypoints_DSNAME.json".

```
- dataset
   - annotations
     person_keypoints_DSNAME.json
   - images
     - DSNAME
         [all the images]
```

d. Label the dataset.
   i. Execute the following command to show the manual labeling user interface (Figure 5A).

```
>python tools/SoAL_LabelUI.py DSNAME
```

Note: This script can also be used for inspecting the keypoints of a labeled dataset.

ii. Label 5 keypoints by clicking on the proper locations in the current frame of image 5 times with the order: head, center, tail, left wing tip, right wing tip. To modify one of the keypoints, press number "1" to "5" to select the point, and label the point again by clicking on the more appropriate location.

⚠ CRITICAL: When labeling left and right wings, pay attention to the heading of the fly. For example, the left-wing tip can be either on the left (Figure 5B first row) or right (Figure 5B second row) depending on the heading direction.

iii. Press "Space" to proceed to the next frame and repeat step ii.
iv. After labeling all the frames, press "d" to save the current labeling results. Close the labeling window.

⚠ CRITICAL: All of the images must be labeled, otherwise the following training process will fail.

e. Execute the following command to divide the dataset into two parts: training (80% of the images) and validation (20% of the images).

```
>python tools/SoAL_DatasetGen.py DSNAME divide
```

Note: The divided datasets named "train_DSNAME" and "val_DSNAME" are also stored in the folder "dataset".

12. Training (Methods video S6).
    a. You can download HRNet model pretrained on ImageNet (Deng et al., 2009) (https://github.com/leoxiaobin/deep-high-resolution-net.pytorch). We also provided a copy (hrnet_w32-36af842e.pth, https://doi.org/10.6084/m9.figshare.19711729) in the folder "hrnet_model" (under SoAL code directory).

Note: To improve the performance and shorten the training time, the training is based on the models pretrained on ImageNet (Deng et al., 2009).

b. Specify the pre-trained model and the dataset. Modify the configuration file of HRNet model "hrnet/fly_w32.yaml" (can be opened by a text editor such as Sublime Text).

```
DATASET:
    ROOT: 'dataset/'
    TEST_SET: 'val_SDPD-15k'
    TRAIN_SET: 'train_SDPD-15k'
MODEL:
    PRETRAINED: 'hrnet_model/hrnet_w32-36af842e.pth'
```

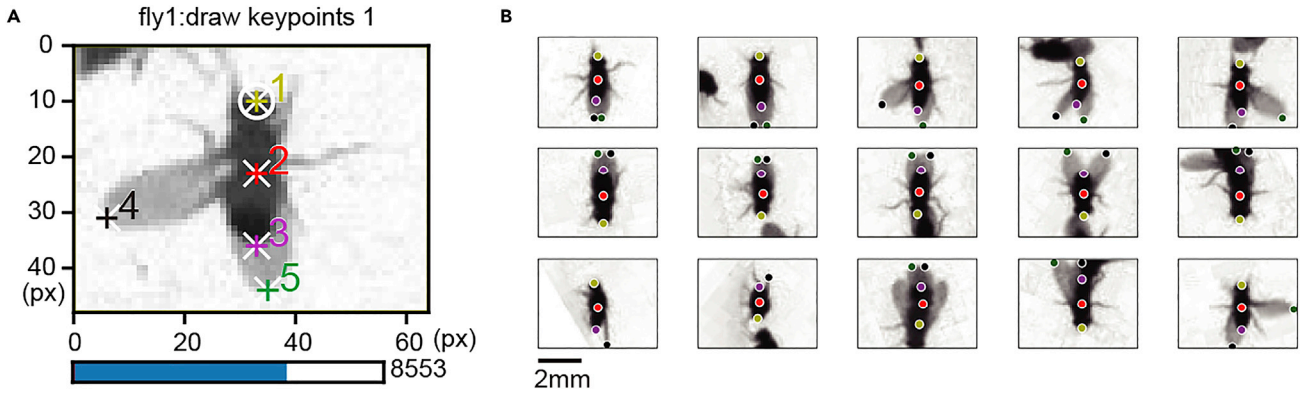


Figure 5. Manual labeling and dataset

(A) The user interface for manual labeling, showing a centralized image with background subtracted (the image and labels are from SDPD-15k). The white crosses and colored digits denote the 5 labeled keypoints: head (yellow, 1), center (red, 2), tail (purple, 3), left wing tip (black, 4), and right wing tip (green, 5). The current selected keypoint is 2 (in white circle). Press "Space" or "z" to navigate to the next and previous frame respectively. Move the slide bar to navigate between frames. Label 5 keypoints by clicking on the proper locations on the image 5 times in the 1–5 sequence. Press number "1" to "5" to change the current selection (denoted by the white circle). The next clicking will assign the new position to selected keypoint. Close this window or press "d" to save the results.

(B) Exemplary labeled images in the dataset SDPD-15k. Here are 15 images with flies exhibiting distinct poses. The color code of the 5 keypoints is the same as (A).

"ROOT" is the root folder of datasets. "TEST_SET" and "TRAIN_SET" specify the two dataset names for testing and training (by default, SDPD-15k is used). "PRETRAINED" is the path of the pretrained model.

c. Run the training script.

Execute the training script:

```
>python hrnet/tools/train.py -cfg hrnet/fly_w32.yaml
```

At the end of each epoch, the script evaluates the model on the test dataset and print AP (average precision, a metric for measuring the model performance). Train until the AP value does not improve (without interruption, training stops when the total epoch number exceeds 200). Stop the training by closing the console. The best model is stored in the output directory: "hrnet_model/coco/pose_hrnet/fly_w32/model_best.pth".

Note: Running time for this script varies depending on the hardware configuration.

Optional: To customize the training parameters, modify the configuration file of HRNet model "hrnet/fly_w32.yaml". Here we list some important items:

```
GPUS: (0,)
OUTPUT_DIR: 'hrnet_model/'
LOG_DIR: 'hrnet_model/log/'
DATASET:
    ROT_FACTOR: 15
    SCALE_FACTOR: 0.35
    BRIGHT_FACTOR: 0.25
TRAIN:
```

```
BATCH_SIZE_PER_GPU: 24

END_EPOCH: 200
```

"GPUS" lists all the GPUs available for training (the index starts with 0). "OUTPUT_DIR" and "LOG_DIR" are the folders for generated network models and log files. The three parameters, "ROT_FACTOR", "SCALE_FACTOR", and "BRIGHT_FACTOR", are used for dataset augmentation (randomly transforming the images to get more training samples). "ROT_FACTOR" specifies the range of rotation transformation (in degrees). "SCALE_FACTOR" specifies the range of scale transformation. "BRIGHT_FACTOR" specifies the range of brightness. "BATCH_SIZE_PER_GPU" is the number of images processed as a batch during training (limited by the VRAM size of the GPU). "END_EPOCH" specifies the total epoch number (one epoch is an iteration of all the images in the training dataset) for the training.

d. Evaluate the model on the test dataset to see the performance.
   i. Modify the configuration item "MODEL_FILE" in "hrnet/fly_w32.yaml" to define which model to test.

```
TEST:

  MODEL_FILE: 'hrnet_model/hrnet_w32_SDPD-15k.pth'
```

By default, the provided model "hrnet_w32_SDPD-15k.pth" is used. If you have trained your own model, change the value to the generated model path "hrnet_model/coco/pose_hrnet/fly_w32/model_best.pth".

   ii. Execute the following script to evaluate the HRNet model on the test dataset and print AP.

```
>python hrnet/tools/test.py -cfg hrnet/fly_w32.yaml
```

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited data		
Social *Drosophila* Pose Dataset (SDPD-15k)	Ning et al. (2022)	https://doi.org/10.6084/m9.figshare.17624888
Executable MIAS package	Ning et al. (2022)	https://doi.org/10.6084/m9.figshare.17630585
Tutorial data	This paper	https://doi.org/10.6084/m9.figshare.19711729 https://doi.org/10.6084/m9.figshare.19711738
Experimental models: Organisms/strains		
Drosophila melanogaster Canton-S (5–10 days old males and females)	Pan Y, Southeast University	N/A
Software and algorithms		
Social Animal Labeler (SoAL)	Ning et al. (2022)	https://github.com/SunLabWestlake/SoAL https://doi.org/10.5281/zenodo.6813122
Multiplexed Image Acquisition System (MIAS)	Ning et al. (2022)	https://github.com/SunLabWestlake/MIAS https://doi.org/10.5281/zenodo.6818402
Python 3.6 (required packages: Cython, EasyDict, opencv-python, shapely, scipy, pandas, pyyaml, json-tricks, scikit-image, scikit-learn, yacs, tensorboardX, pycocotools, matplotlib, tqdm)	Python Software Foundation	https://www.python.org
PyTorch 1.10	PyTorch	https://pytorch.org/
CUDA 11.3	NVIDIA, Inc.	https://developer.nvidia.com/cuda-toolkit

(*Continued on next page*)

Continued		
REAGENT or RESOURCE	SOURCE	IDENTIFIER
HRNet	Sun et al. (2019)	https://github.com/HRNet/deep-high-resolution-net.pytorch
Visual Studio 2017	Microsoft, Inc.	https://visualstudio.microsoft.com
OpenCV 4.0.1	OpenCV	https://opencv.org/
CMake	CMake	https://cmake.org
Other		
Fly rearing incubator	NBSafe	https://www.nbsafe.com/
Camera BFS-U3-13Y3M-C (with camera driver pinnaker SDK v2.6.0.157)	Teledyne FLIR	https://www.flir.com/
Lens M0824-MPW2	Computar	https://www.computar.com
White-colored LED array (6 by 7 LED chip array in a 12 cm by 12 cm region, 10 W total power)	PinkPurple	N/A
Acrylic sheets (1 m × 1 m, 3 mm or 5 mm thick)	Xintao	https://www.xintaoacrylic.com/
White recycled printer papers (210 mm × 297 mm, 0.1 mm thick)	Asia Symbol	N/A
1.5 mL microcentrifuge tube	ShengHui	N/A
Computer equipped with CUDA compatible GPUs (e.g., Intel Core i7 8700 CPU, 32 GB RAM, nVIDIA GeForce GTX 1060 GPU)	Dell, Inc.	https://www.dell.com/
GPU nVIDIA GeForce GTX 1060	NVIDIA, Inc.	https://www.nvidia.cn/
GPU nVIDIA GeForce RTX 2060	NVIDIA, Inc.	https://www.nvidia.cn/

MATERIALS AND EQUIPMENT

Fly food for experiments		
Reagent	Final concentration	Amount
Sucrose	68 g/L	3.4 g
Agarose	20 g/L	1.0 g
Yeast	14 g/L	0.7 g
ddH_2O		50 mL
Store under 4°C for less than 6 days.		

STEP-BY-STEP METHOD DETAILS

We first describe how to acquire videos with MIAS. We then demonstrate the analysis workflow utilizing SoAL with a series of graphical user interfaces and scripts (Figure 4, right part).

Experiment

Timing: 2–4 days for daily experiments. 18–24 h for starvation. 0.5–1.5 h for fly loading and adaptation. 1 h for video acquisition

Carry out the experiments in the incubator at 22°C and 50% humidity, within 0–4 h after the beginning of the day cycle.

Note: We have provided a 10 min example video acquired during the experiment. The example video is included in our tutorial data (https://doi.org/10.6084/m9.figshare.19711738, containing an example video along with all the analysis results after "Pose estimation"). If you want to skip the experiment and practice the pose estimation (refer to step-by-step method details "Pose estimation"), download this tutorial data and perform the pose estimation (refer to step-by-step method details step 7).

Note: We combined single-housed males with group-housed females to enhance courtship, as single-housing promotes male courtship while group-housing suppresses female receptivity.

1. Female starvation.

Note: Females need to be starved 18–24 h before the beginning of the experiment.

   a. Shift the group-housed flies (refer to before you begin step 9) into an empty vial with a sheet of wet paper.
   b. Put the empty vial back into the rearing incubator.
2. Fly loading.
   a. Switch the incubator on and set the temperature to 22°C and the humidity to 50%.
   b. Assemble the chambers.
      i. Remove the chamber walls and top plates in Figure 2.
      ii. Heat the fly food (refer to before you begin step 3) by microwave until melted.
      iii. Pour the food into the holes at the center of each chamber on the bottom plate (Figure 1B) with a pipette.
      iv. Stack up the chamber walls and top plates with both the top and left edges aligned (Figures 1C and 2).
      v. Insert the four separators into the slits by rows.
   c. Introduce males into the parts with food (Figure 1B, the hole below the separation line) guided by a soft tube. Cover the holes on the top plate with the cover plates immediately after the fly is guided into the chambers.

Note: To make an introducing tube, cut the two ends of a 2 mL pipette, and make sure that one end fits the microcentrifuge tube and the other end fits the introducing hole. Quickly open the lid of the microcentrifuge tube when the fly stays away from the lid, while aligning the end of the tube with the microcentrifuge tube. After the fly moves into the soft tube, pinch the big end to prevent escaping. Place the small end above the introducing hole and wait until the fly moves into the chamber.

Note: Anesthetization affects the neural activity of the flies that may not recover immediately after waking up. We therefore recommend not to anesthetize the males to maintain the best condition of the animals at the cost of experiment difficulty.

   d. Anesthetize the group-housed flies by ice.
      i. Pick the females with tweezers and gently moved them into the parts without food (Figure 1B, the introducing hole above the separation line), thus preventing them from eating before the experiment.
      ii. Cover the holes on the top plate with the cover plates.
3. Adaptation.

Leave the flies in the chamber for 20–30 min for recovery and adaptation.

4. Closing chamber.
   a. Just before the video acquisition, remove the separators and slide the top plate up to cover the introducing holes (Figure 1D).
   b. Remove the cover plates to clear the view.

Note: This operation should finish within 2 min to reduce the interaction time before video acquisition.

⚠ CRITICAL: When sliding the top plate, take care not to crush the flies with the introducing hole. When there are flies in the introducing hole, wait a short time until all the flies leave the holes.

5. Video acquisition.
   a. Run the video acquisition software (e.g., MIAS_FLIR.exe for FLIR cameras). This program shows a black console (Figure 3) and two preview windows.
   b. Inspect the preview window to ensure the field of view is fine.
   c. Press "c" and "Enter" in the console to start the recording.
   d. Press "q" and "Enter" in the console to stop the recording and quit MIAS.

Note: In our example, the video is recorded with a resolution of 1280 by 1024 pixels and a fixed acquisition rate of 66.6 frames/s.

Note: The videos acquired are stored in the folders named "video" under MIAS directory, with the folder structure as follows:

```
- MIAS
    config.json
    MIAS_All.exe
    MIAS_FLIR.exe
    [Other files]
    -video
       -20200703_141812_1
          20200703_141812_1.avi
          20200703_141812_1.log
       -20200703_141812_2
          20200703_141812_2.avi
          20200703_141812_2.log
```

For each camera, MIAS generates two files ("xxx.avi" is the recorded video. "xxx.log" stores the timestamps of each frame) in the same folder. The name (e.g., "20200703_141812_2") is composed of date, time, and the camera name, separated by "_".

6. Chamber cleaning.

After each experiment, anesthetize the flies by ice or CO_2 and discard them. Rinse the surface of the supporters and all the parts of the chambers with deionized water.

Pose estimation

Timing: hours to days

Timing: For keypoint detection (step 13), about 2 h for the example video on the *Basic Hardware Configuration*

The following steps use the tutorial data to demonstrate the analysis pipeline. The analysis result includes the time series of 5 keypoint positions.

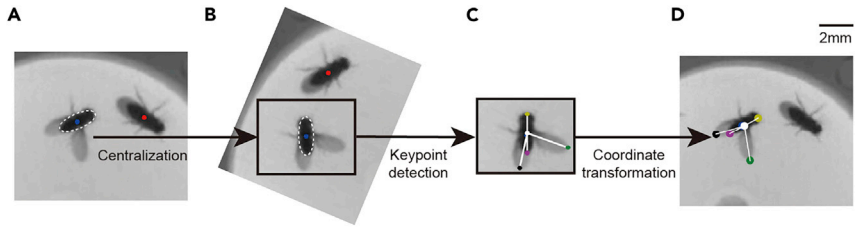


Figure 6. The top-down approach for keypoints detection

(A and B) Segment a local region around each fly, and centralize by transforming the segmented image to ensure that the fly body appears vertically at the center of a 64 by 48 pixels image (B).

(C) Detect the 5 keypoints of a fly.

(D) Transform the keypoints back to the original coordinate.

Note: To perform pose estimation, we used a top-down approach, applying ROI extraction, body detection, and keypoints prediction on the video frames in parallel (Figure 6).

7. Download tutorial data.

Download and unzip the tutorial data (https://doi.org/10.6084/m9.figshare.19711729).

Note: The folder "video" under the SoAL code directory corresponds to the "video" folder under the MIAS directory (refer to step-by-step method details step 5).

Note: As a reference, another package of tutorial data (https://doi.org/10.6084/m9.figshare.19711738) already contain configuration files and analysis results. The files will be replaced, after performing the following steps.

8. Configuration (for steps 8–12 see demonstration in Methods video S1).

Execute the following command to call the configuration UI.

```
>python SoAL_PreProcess.py \
video/20200703_141812_2/20200703_141812_2.avi
```

Note: The command line parameter "video/20200703_141812_2/20200703_141812_2.avi" is the path of the video file to be configured (the symbol "\" is a line continuation, do not input this symbol and the following line break). This configuration shows 3 dialog boxes in series (described in the following steps 9–11) to set the parameters for keypoint detection and behavior annotation.

9. Setting the scale.
   a. The first dialog shows a random sampled frame from the video for setting the scale, defined as the number of pixels occupying a 1-mm length.

   Note: The scale factor is calculated by automatic detection of the chamber diameter (Figure 7A blue line). Alternatively, the chamber diameter can also be specified manually (Figures 7B and 7C).

   b. Input the real diameter (in mm) of the chamber in the textbox "Diameter (mm)". The textbox "Scale (px/mm)" then shows the calculated scale factor.
   c. Click "Confirm" to proceed to the next step.

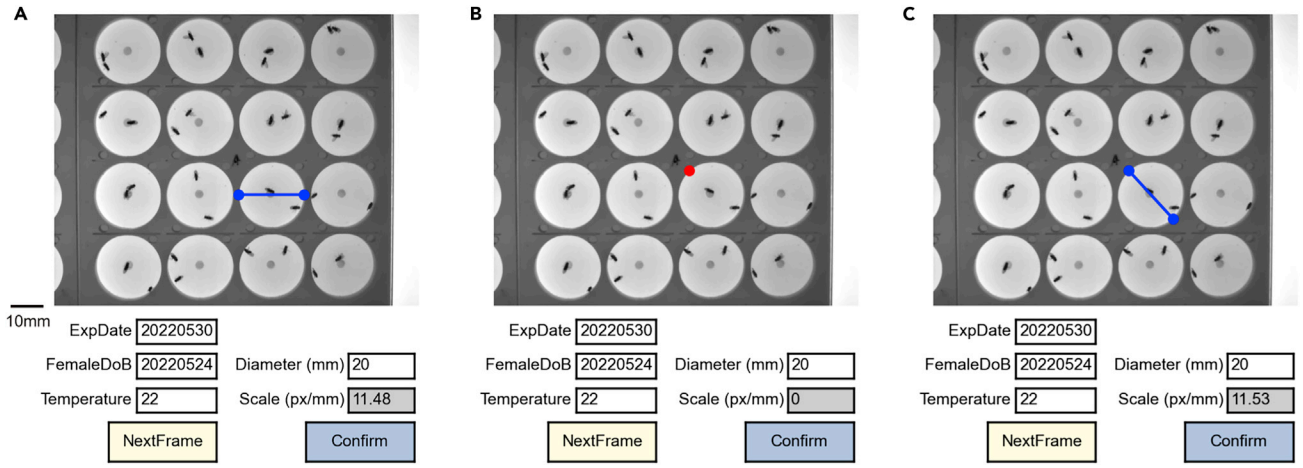


Figure 7. UI for setting the scale factor

(A) This UI shows the image of a frame with an auto-detected chamber diameter (blue line). Optionally, modify the textboxes below the image to change the experiment date ("ExpDate"), female eclosion date ("FemaleDoB"), and temperature ("Temperature"). The textbox "Diameter (mm)" is for specifying the real length of the chamber diameter. "Scale (px/mm)" is updated automatically when the diameter and the real length are both specified. Click button "Random" to change a frame to show.

(B and C) If the detected diameter is incorrect, label two points on the image by clicking to indicate a diameter. First, click the start point (B, red point). Second, click the end point (C).

10. ROI extraction.
    a. Now another dialog shows an image superimposed with 16 blue circles indicating the auto-detected chamber regions (Figure 8A).

    Optional: If the detection is incorrect, change the parameters for circle detection (the textbox "CircleParam") or use the manual mode (see problems 10 and 14 for the details).

    b. Some of the regions should not be analyzed due to the unsuccessful preparation. Click in any circle to exclude the region from being ROI (the circle will change to red, Figure 8B).
    c. Click "Confirm" to proceed to the next step.
11. Background subtraction.

Wait several seconds of computing until the appearance of the next dialog box.

Note: The program computes the background image by averaging 100 frames that are uniformly sampled in time throughout the video.

12. Fly segmentation.

Note: The program segments the flies in the images by detecting connected regions in binarized images.

    a. The dialog shows the binarized ROI regions (Figure 9, 15 ROIs are shown because one ROI is excluded).
    b. Slide the bar at the bottom to change the binarization threshold.

⚠ CRITICAL: A rule of thumb for choosing a good threshold is to make sure that the segmented fly body as previewed in Figure 9C is complete and does not include the wings.

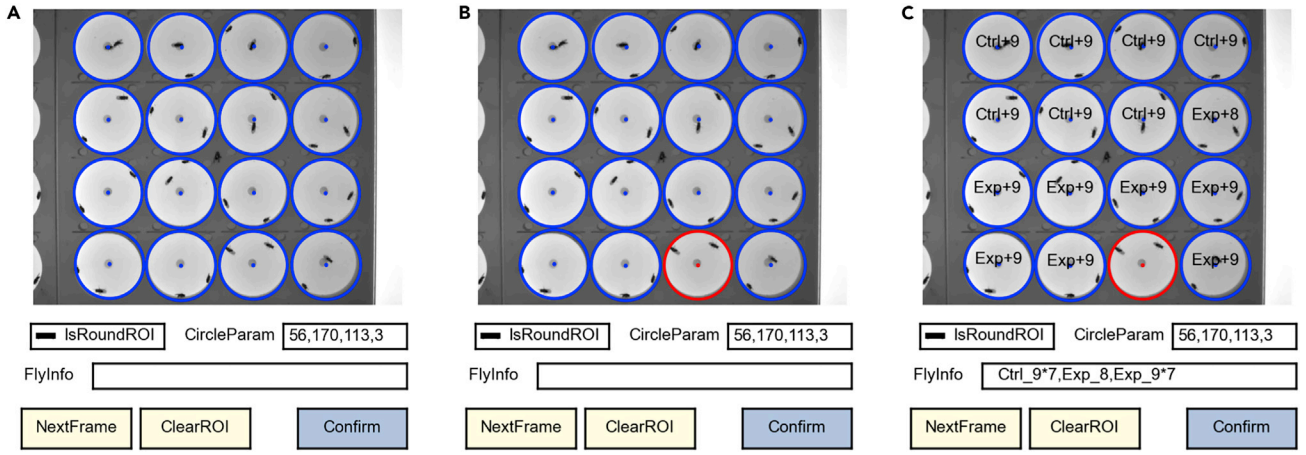


Figure 8. UI for ROI extraction

(A) This UI shows the image of a frame with auto-detected ROIs (blue circles).

(B) Deselect ROIs as needed by clicking in the circle (red circle, the third one of the fourth row, which includes a dead fly). Click again to re-include the circle as a ROI.

(C) Input fly information for each ROI in the textbox "FlyInfo". Specify all ROIs according to the order of top to bottom and left to right (skipping the excluded ROIs with red circles). The text for one ROI is composed of a group name and an age number, separate by "_". The adjacent ROIs with the same group and age can be merged and represented by adding "*N" (N is the number of the same ROIs). Different ROIs are separated by ",". For example, the text "Ctrl_9*7,Exp_8,Exp_9*7" (textbox "FlyInfo") means the first 7 males are from "Ctrl" group and are 9 d old, the eighth male is from "Exp" group and are 8 d old, and the last 7 males are from "Exp" group and are 9 d old. The red one (the third one of the fourth row) is excluded.

Optional: For different videos with the same illumination condition, the same threshold can be used (the default value of the threshold can be changed in the source file "SoAL_Constants.py" named "DEFAULT_GRAY_THRESHOLD").

Optional: Change the preview mode by clicking the buttons at the bottom (Figure 9).

c. Click "Confirm" to finish the configuration.

Note: The configuration results are stored in two files, a text file "xxx_config.json" and a background image "xxx.bmp" ("xxx" is the video name), both in the same directory as the video file.

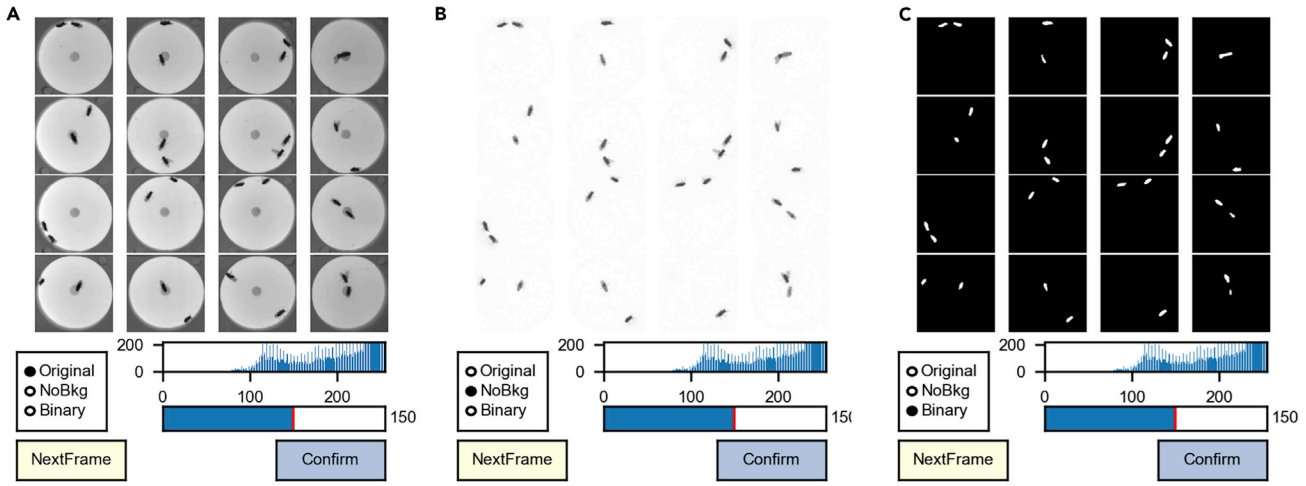


Figure 9. UI for fly segmentation

(A–C) The UI shows the images of 15 extracted ROIs. The histogram under the images is the gray scale histogram of current image. Slide the bottom bar to change the binarization threshold (the value is shown on the right, "150"). Switch the radio buttons at the left bottom to change the viewing mode: original (A), background-subtracted (B), and binary (C). The default viewing mode is binary (C).

13. Running keypoint detection (Methods video S2).
    a. Choosing the model.
    Modify the configuration item "MODEL_FILE" (under the item "TEST") in "hrnet/fly_w32.yaml" to specify the model to use for keypoint detection (same as the step "before you begin, step 12b"). By default, the provided model "hrnet_w32_SDPD-15k.pth" is used.
    b. Execute the script to automatically analyze all the videos in the directory.

```
>python SoAL_KptDetect.py all
```

The program will print the processing progress. Wait until the message "[Main]: all finished" is printed. Close the console window to quit.

Note: The directory is specified by the item "VIDEO_TODO_DIR" in "SoAL_Constants.py", and the default path is "video/". The videos are automatically processed in parallel based on the hardware configuration.

Note: To determine whether a video has been successfully processed, inspect the text file ".state" in each video folder to see if "finish" is shown.

Note: Processing speed is about 150 centralized images per second on the *Basic Hardware Configuration*, and 400 centralized images per second on the *Advanced Hardware Configuration* with at least two videos processing simultaneously.

Optional: Before running keypoint detection, edit the following items in the file "SoAL_Constants.py" to modify some parameters.

```
MODEL_CONFIG = "hrnet/fly_w32.yaml"

VIDEO_TODO_DIR = "video/"

MAX_TASK = 2

BATCH_SIZE = 60

MODEL_SHAPE = 64, 48

FLY_NUM = 2
```

"MODEL_CONFIG" provides the path of the model configuration file. "VIDEO_TODO_DIR" is the parent folder of the video to be analyzed. "MAX_TASK" is the number of videos processed in parallel (limited by the clock speed of the CPU and the RAM), which is 2 with the *Basic Hardware Configuration*. "BATCH_SIZE" is the number of centralized images processed in a batch on the GPU (limited by the VRAM capacity of the GPU), which is 60 with the *Basic Hardware Configuration* and 200 with the *Advanced Hardware Configuration*. "MODEL_SHAPE" defines the size (width and height, in pixels) of the centralized image as input to the HRNet model. "FLY_NUM" defines the number of flies in one ROI, this number is 2 in cases of male-female courtship.

14. Inspecting keypoints (Methods video S3).

Note: Here is an example of part of the folder structure after the keypoint detection:

```
- video

    - 20200703_141812_2
```

```
20200703_141812_2.avi
20200703_141812_2.bmp
20200703_141812_2_config.json
  - 0
    20200703_141812_2_0_kpt.csv
    20200703_141812_2_0_config.json
  - 1
    20200703_141812_2_1_kpt.csv
    20200703_141812_2_1_config.json
  - 2
    20200703_141812_2_2_kpt.csv
    20200703_141812_2_2_config.json
```

The digital folders include the keypoint information files "xxx_kpt.csv" of each ROI (the last number in the name indicates the ROI number). The file "xxx_config.json" contains the configuration information of this ROI.

a. Inspect the keypoint information "xxx_kpt.csv" by executing the following script (Figure 10).

```
>python tools/SoAL_ViewKptUI.py \
video/20200703_141812_2/0/20200703_141812_2_0_kpt.csv
```

Note: The command line parameter "video/20200703_141812_2/0/20200703_141812_2_0_kpt.csv" is the keypoint information file to inspect. This UI shows the image of the specific ROI of the current frame, the 5 keypoints of each fly, and two centralized flies superimposed with the keypoints and the skeletons.

b. Navigate to desired frames by the bottom slide bar.

Note: The "SoAL_ViewKptUI" command can be executed alongside "SoAL_KptDetect" simultaneously. This is useful for inspecting the partial results quickly to find the problems. If the results are not satisfactory, you can stop the keypoint detection and make adjustments before running the "SoAL_KptDetect.py" command again.

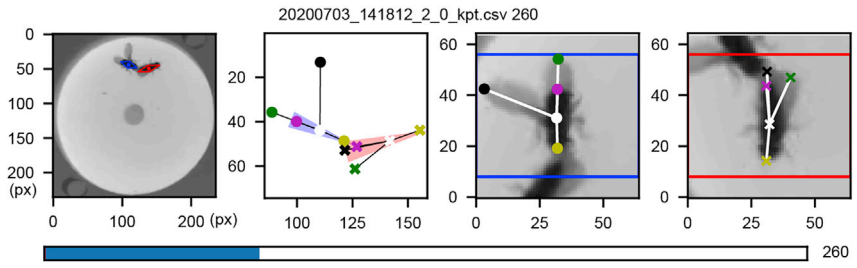


Figure 10. UI for Inspecting detected keypoints

This UI shows 4 diagrams of a frame: the image of the ROI, the keypoints and heading directions (triangles, two flies are color-coded in blue and red), and two centralized flies with the keypoints. Move the slide bar to change the showing frame. Press "Left arrow" or "Right arrow" to navigate to the next and previous frame respectively. Directly input the frame number and press "Enter" to navigate to the requested frame.

Note: The table file "xxx_kpt.csv" contains the following columns. "frame" is the frame number. The same frame number will appear twice for the two flies. "reg_n" is the number of regions detected. This number is generally 2 for social behavior, but it becomes 1 when the two flies overlap. "area" is the area of one region. "pos:x" and "pos:y" is the position of the region center in the coordinate of the ROI (origin point on the upper left corner). "ori" is the orientation angle. "e_maj" and "e_min" is the major and minor axis of the fitted ellipse. "point:xs" and "point:ys" are the horizontal and vertical positions of the 5 keypoints.

Behavior annotation

Timing: Several minutes depending on the specific case

The following step assigns the correct identities to the detected keypoints and annotates behaviors of each fly. The results include the motion parameters and the behavior bouts.

15. Run identity assignment and behavior annotation (Methods video S4).

Run the following script to assign identities and annotate behaviors of one ROI:

```
>python SoAL_ID_Anno.py \
video/20200703_141812_2/0/20200703_141812_2_0_kpt.csv
```

Note: The command line parameter "video/20200703_141812_2/0/20200703_141812_2_0_kpt.csv" is the keypoint information file to analyze. To analyze an entire folder, pass the video folder as the parameter.

```
>python SoAL_ID_Anno.py video/20200703_141812_2
```

Note: The annotation process loads the keypoint information "xxx_kpt.csv". The motion parameters for each frame are then calculated. Next, identities are assigned according to the wing extension angle. Finally, we detect and annotate specific behavior bouts, such as copulation, wing extension, crabwalking, and circling.

Optional: To enable anti-distortion (refer to before you begin step 6), modify the value of "NEED_UNDISTORT" in the file "SoAL_ID_Anno.py". In this way, executing "SoAL_ID_Anno.py" command also loads the camera calibration information file.

EXPECTED OUTCOMES

Here is an example of the final folder structure after the analysis:

```
- video
   - 20200703_141812_2
      20200703_141812_2.avi
      20200703_141812_2.bmp
      20200703_141812_2_config.json
        - 0
          20200703_141812_2_0_kpt.csv
```

```
20200703_141812_2_0_config.json

20200703_141812_2_0_config_circl.json

20200703_141812_2_0_mot_para0.pickle

20200703_141812_2_0_mot_para1.pickle

20200703_141812_2_0_mot_para2.pickle
```

The three annotation files "xxx_mot_para0.pickle", "xxx_mot_para1.pickle", and "xxx_mot_para2.-pickle" contain all the motion parameters and behavior annotations (see problem 13 to access these files) for shared information of the two flies and specific information for each fly respectively. The file "xxx_config_circl.json" contains the configuration information and the circling bouts annotated.

Execute the following script to visualize per-frame motion parameters and behavior annotations (Figure 11, Methods video S4).

```
>python tools/SoAL_ViewMotParaUI.py \

video/20200703_141812_2/0/20200703_141812_2_0_mot_para0.pickle
```

The command line parameter "video/20200703_141812_2/0/20200703_141812_2_0_mot_para0.-pickle" is the first annotation file.

QUANTIFICATION AND STATISTICAL ANALYSIS

Based on the keypoints, motion parameters, and behavior annotations, we can explore the dynamics of the interacting flies. Here we present visualizations of circling trajectories and wing extension directions based on the tutorial data (The source code for these visualizations can be found in the file "SoAL_CirVis.py").

Using the circling bouts in xxx_config_circl.json, we extracted all the positions and body directions of the two flies during circling. The pooled circling trajectories (Figure 12A) show that the male circle around the female with its body oriented to the female.

We also extracted all the frames during male wing extension. We plotted the positions of the female head in a male-centered coordinate. The color indicates the wing choice of the male when the female head is on that position (Figure 12B).

LIMITATIONS

We have successfully applied SoAL to courtship behavior of *Drosophila*. SoAL can also be used for other types of social interactions such as aggression. However, interactions of more than two animals are not supported yet. SoAL is potentially applicable to other animals, provided that appropriate training dataset is obtained.

TROUBLESHOOTING

Problem 1 (basic)

Failed to install the Python package "pycocotools" (before you begin step 10).

Potential solution

Follow the instructions printed after the error. Generally, downloading and installing the latest version of Microsoft C++ Build Tools solves the problems.

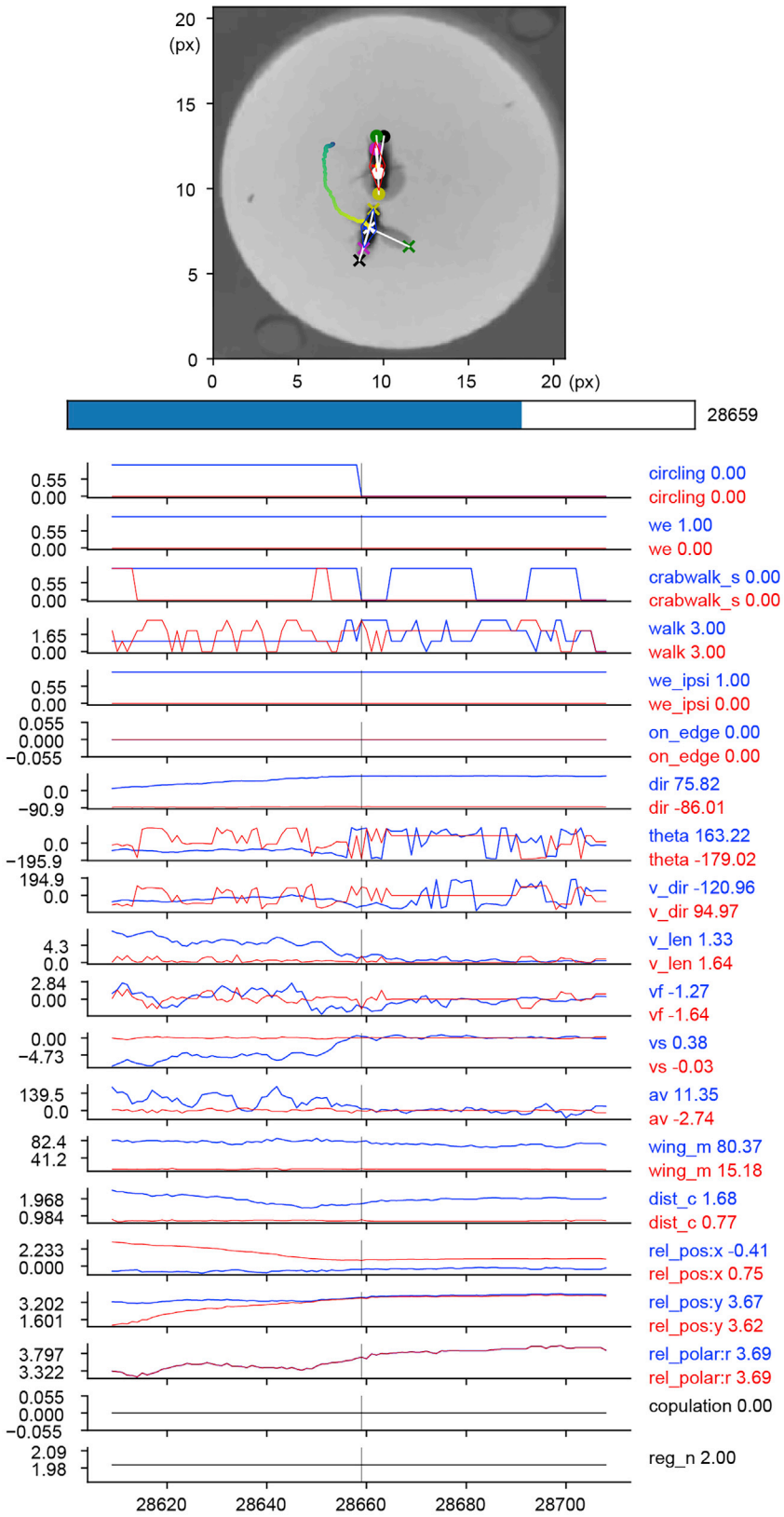

20
(px)
15
10
5
0
0
5
10
15
20 (px)
28659
0.55
0.00
circling 0.00
circling 0.00
0.55
0.00
we 1.00
we 0.00
0.55
0.00
crabwalk_s 0.00
crabwalk_s 0.00
1.65
0.00
walk 3.00
walk 3.00
0.55
0.00
we_ipsi 1.00
we_ipsi 0.00
0.055
0.000
−0.055
on_edge 0.00
on_edge 0.00
0.0
−90.9
dir 75.82
dir -86.01
0.0
−195.9
theta 163.22
theta -179.02
194.9
0.0
v_dir -120.96
v_dir 94.97
4.3
0.0
v_len 1.33
v_len 1.64
2.84
0.00
vf -1.27
vf -1.64
0.00
−4.73
vs 0.38
vs -0.03
139.5
0.0
av 11.35
av -2.74
82.4
41.2
wing_m 80.37
wing_m 15.18
1.968
0.984
dist_c 1.68
dist_c 0.77
2.233
0.000
rel_pos:x -0.41
rel_pos:x 0.75
3.202
1.601
rel_pos:y 3.67
rel_pos:y 3.62
3.797
3.322
rel_polar:r 3.69
rel_polar:r 3.69
0.055
0.000
−0.055
copulation 0.00
2.09
1.98
reg_n 2.00
28620
28640
28660
28680
28700

Figure 11. UI for inspecting results

This UI consists of two panels. The left panel shows the time series of motion parameters of each fly (each color-coded in blue or red), 100 frames around the frame (vertical gray line) under inspection. The right panel shows the image of the ROI with the keypoints and the trajectories of the interacting pair in the recent 3 s. Move the slide bar to change the frame for inspection. Press "Space" to update the left panel. Clicking on the left panel also navigates to the frame at the click position (x axis).

Problem 2 (basic)

Failed to load CUDA when running SoAL (before you begin step 12 and step-by-step method details step 13).

Potential solution

This is often caused by version conflict between CUDA and PyTorch. For the tutorial data, we utilized PyTorch v1.10.2 with CUDA v11.3.

Problem 3 (basic)

CUDA out of VRAM during keypoint detection or training (before you begin step 12 and step-by-step method details step 13).

Potential solution

GPU VRAM is too small for the network configuration. As for training, reduce the "WORKERS" or reduce the "BATCH_SIZE_PER_GPU" under the "TRAIN" item, both are in the "hrnet/fly_w32.yaml". For keypoint detection, reduce the "BATCH_SIZE" in the "SoAL_Constants.py" to a smaller value.

Problem 4 (basic)

SoAL continually prints "[Main] check dir..." and no other messages are printed (step-by-step method details step 13).

Potential solution

Check and find errors printed at the beginning. This is often caused by CUDA or VRAM errors mentioned in Problem 2 and 3. Use the solutions above.

Problem 5 (basic)

The fly body is not vertical or the fly body does not appear in the centralized image (Figure 10) when inspecting keypoints (step-by-step method details step 14).

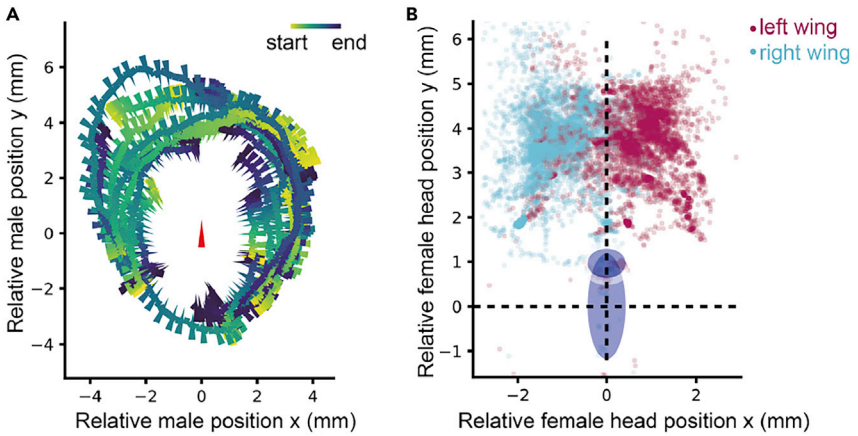


Figure 12. Visualization of circling and wing extension

(A) Pooled circling trajectories of one male, in the female-centered coordinate.

(B) Scatter plot of the female head positions color-coded by male wing extension directions, in the male-centered coordinate.

Potential solution

This is often caused by an inappropriate binarization threshold (Figure 9). To choose a good threshold, the fly body (Figure 9C) should be complete and does not include the wings.

Problem 6 (basic)

Fly body appears in the background image (step-by-step method details step 11).

Potential solution

If the fly keeps immobile for a long time in one position, the pixels of this fly will be treated as the background (e.g., Figure 8B). Just deselect this ROI.

Problem 7 (basic)

The fly body is too small or too large in the centralized image (step-by-step method details step 14).

Potential solution

This is caused by differences in imaging system.

Change the "MODEL_SHAPE" (default value is 64 × 48 pixels, refer to Figure 10, in the two centralized images on the right, the red and blue rectangles depict the 64 × 48 pixels regions) in the file "SoAL_Constants.py" to an appropriate size so that the entire fly body is covered.

Problem 8 (basic)

The accuracy of the keypoint detection is low (step-by-step method details step 14).

Potential solution

When you train a custom model for keypoint detection, make sure that appropriate training is performed. Evaluate the performance of the best model to see the AP. If AP value is low, train for a longer time. Besides, the training dataset may lack some important cases. Labeling and adding some failed frames to the dataset can be helpful.

In addition, during flip augmentation, if you defined a custom skeleton of the keypoints, check whether the value of "FLIP_PAIRS" in "hrnet/lib/dataset/coco.py" is correctly assigned and whether the left-right convention is consistent over the dataset.

Problem 9 (basic)

Identity switching after overlapping (step-by-step method details step 15).

Potential solution

When applying SoAL to male-male interactions or other non-courtship behaviors, switching often occurs. To manually correct the identity, open the inspect UI (SoAL_ViewKptUI.py) and correct on each keypoint file. The identities are already assigned by the closest distance between two adjacent frames. Press "Space" to jump to the next frame when an overlap happened. Press "Ctrl" to swap the identity of the current frame (all the frames afterward are also processed) if the identities are not correct. Close the window to save the result. The result is in the same folder as the original "xxx_kpt.csv" named "xxx_kpt_correct.csv".

Problem 10 (basic)

The chamber regions are not correctly detected (Figure 8, step-by-step method details step 10).

Potential solution

Change the textbox "CircleParam". The 4 parameters are separated by ",", specifying the minimal radius, the maximal radius, the minimal distance between circles, and the extend distance of the circles.

Problem 11 (advanced)
How to estimate the speed and the potential duration of keypoint detection (step-by-step method details step 13)?

Potential solution
See the printed messages start with "[WriteKpt]". The messages show the frame rate of processing, estimation of the remaining time, and the percentage of frames completed.

Problem 12 (advanced)
How to speed up SoAL (step-by-step method details step 13)?

Potential solution
Upgrade the hardware especially the GPUs. Change the "GPUS" in the file "fly_w32.yaml" to assign all available GPUs to the prediction process. Adjust the following items to best use parallelization: "BATCH_SIZE", "MAX_TASK", and "PRED_PROCESS_NUM" (refer to step-by-step method details step 13).

Problem 13 (advanced)
How to access the resulting data generated by behavior annotation?

Potential solution
Use the function "load_dict" in "SoAL_Utils.py" to load "xxx_config_circl.json". The file "xxx_config_circl.json" contains the ROI information and the detected circling and copulation bouts.

Use the function "load_dfs" in "SoAL_Utils.py" to load the three "pickle" files. The "xxx_mot_para0.pickle" stores the per-frame global information. The "xxx_mot_para1.pickle" and "xxx_mot_para2.pickle" store the per-frame parameters of the male and the female respectively (refer to "expected outcomes").

Problem 14 (advanced)
How to deal with non-circular chambers (Figure 8, step-by-step method details step 10)?

Potential solution
Switch the checkbox "IsRoundROI" off to enable manual mode. Click the button "ClearROI" to remove all the ROIs. Click at the top-left and bottom-right points to define a rectangular ROI. Press "Enter" to confirm the shape of this ROI. Repeat the above steps to draw all the ROI regions.

Problem 15 (MIAS)
MIAS cannot find the "xxx.dll" files (refer to before you begin step 6f).

Potential solution
Make sure the right version of camera driver is installed in developer mode (refer to before you begin step 6a).

Problem 16 (MIAS)
Failed to open camera (refer to before you begin step 6f).

Potential solution
Other programs are controlling the camera (e.g., the camera configuration program, refer to before you begin step 6d). Disconnect the camera or close the program occupying the camera.

Problem 17 (MIAS)
Failed to read configuration file (refer to before you begin step 6f).

Potential solution

There is no "config.json" file in the MIAS directory. Alternatively, the syntax of this file is not consistent with the "json" format (refer to before you begin step 6b).

Problem 18 (MIAS)

Nothing happens after inputting commands in MIAS (refer to before you begin step 6f and step-by-step method details step 5).

Potential solution

This is caused by the input method. Before inputting the command, make sure the input system is English.

Problem 19 (MIAS)

The real-time statistics stop to print messages during running MIAS (refer to before you begin step 6f and step-by-step method details step 5).

Potential solution

This is caused by left clicking in the console window which stops running the program. Right click in the console to resume.

Problem 20 (MIAS)

Video file corrupted (refer to step-by-step method details step 5).

Potential solution

This is often caused by incomplete saving. DO NOT click the close button to stop the recording as it interrupts saving. Instead, press "q" and "Enter" and wait for the saving process to finish.

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources and reagents should be directed to and will be fulfilled by the lead contact, Yi Sun (sunyi@westlake.edu.cn).

Materials availability

This study did not generate new unique reagents.

Data and code availability

All SoAL code, including those for pose estimation, behavior annotation, result inspection, quantification and visualization, are available at https://github.com/SunLabWestlake/SoAL, and https://doi.org/10.5281/zenodo.6813122.

The SDPD-15k dataset for training is available at figshare (https://doi.org/10.6084/m9.figshare.17624888).

To better help new users to get familiar with SoAL, we provided two compressed files, one with all the essential files to run SoAL together with SDPD training dataset and a test video, the other one with all expected results in addition to the essential files. Specifically, a compressed file containing all essential files to run SoAL, including SoAL core code and related tools, network model libraries, SDPD training datasets, the HRNet model pretrained on ImageNet, the model further trained on SDPD-15k, and a test video, is available at figshare (https://doi.org/10.6084/m9.figshare.19711729); another compressed file containing all expected results after analyzing the test video and training 1 epoch with SDPD, together with all essential files as mentioned above is available at figshare (https://doi.org/10.6084/m9.figshare.19711738).

The source code of the video acquisition software MIAS is available at https://github.com/SunLabWestlake/MIAS, and https://doi.org/10.5281/zenodo.6818402.

The executable files of MIAS and the corresponding camera drivers together with an example video are available at figshare (https://doi.org/10.6084/m9.figshare.17630585).

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.xpro.2022.101621.

ACKNOWLEDGMENTS

We thank Y. Pan and C. Zhou for providing fly strains; Y. Wang, Y. Guo, and S. Cao from Biomedical Research Core Facilities at Westlake University for fly food preparation; and members of Sun lab for helpful suggestions. This work was supported by National Natural Science Foundation of China (32070991), Westlake Laboratory of Life Sciences and Biomedicine (W101386022101), and Westlake Education Foundation to Y.S.; and Natural Science Foundation of Zhejiang Province (LQ21C090001) to D.C.

AUTHOR CONTRIBUTIONS

J.N. and Y.S. designed the protocol; J.N. implemented the original protocol; J.N., Z.L., X.Z., J.W., D.C., Q.L., and Y.F.S. tested the protocol in various experimental conditions and provided feedback for optimization; J.N. and Y.S. wrote and revised the manuscript with comments from all authors.

DECLARATION OF INTERESTS

J.N. and Y.S. are applying for patents on experimental setup, video acquisition, pose estimation, and identity assignment.

REFERENCES

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Li, F.F. (2009). ImageNet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition, 248–255. https://doi.org/10.1109/CVPR.2009.5206848.

Lin, T.Y., Maire, M., Belongie, S., Hays, J., and Zitnick, C.L. (2014). Microsoft COCO: Common Objects in Context (Springer International Publishing).

Ning, J., Li, Z., Zhang, X., Wang, J., Chen, D., Liu, Q., and Sun, Y. (2022). Behavioral signatures of structured feature detection during courtship in Drosophila. Curr. Biol. *32*, 1211–1231.e7.

Sun, K., Xiao, B., Liu, D., and Wang, J. (2019). Deep High-Resolution Representation Learning for Human Pose Estimation. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 5686–5696. https://doi.org/10.1109/CVPR.2019.00584.