DeepLabCut AI Residency

Day 1 Session 3:
Project creation, labeling, and data management

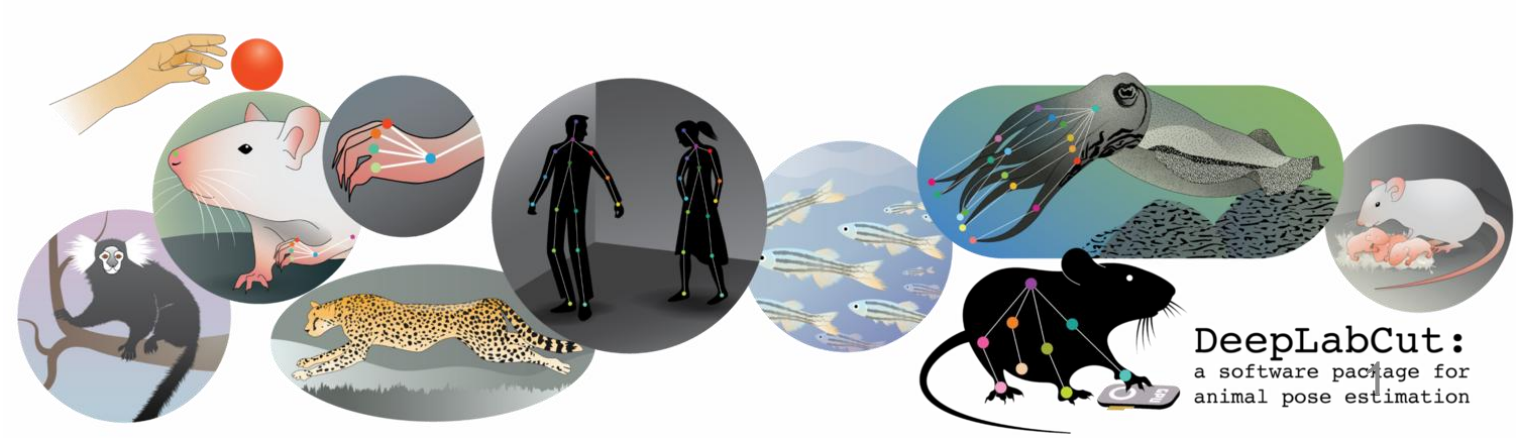
July 30 & August 1, 2025

McGill University, Montreal

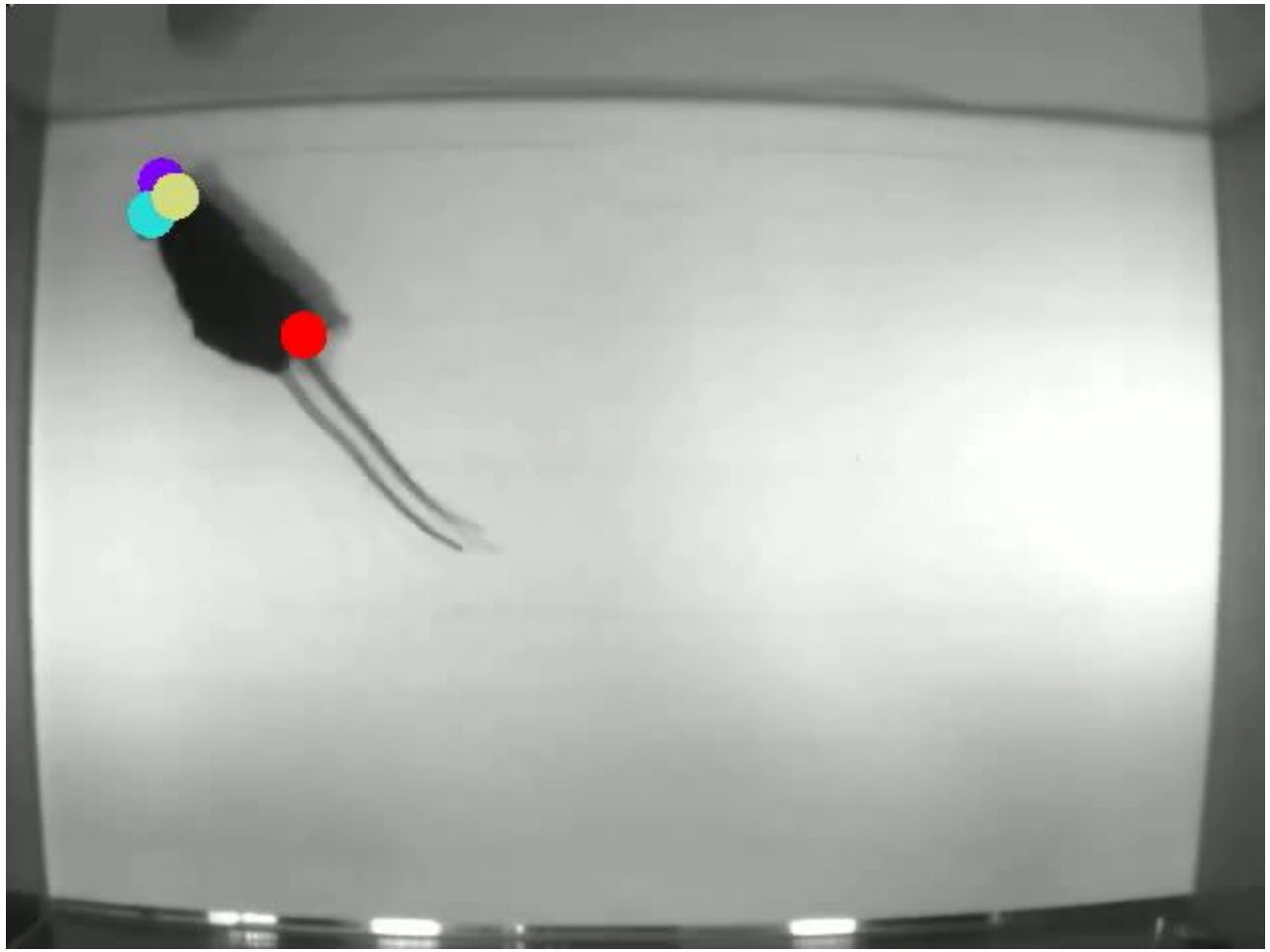
Jiayue Yang

Vic Shao-Chinh Chiang

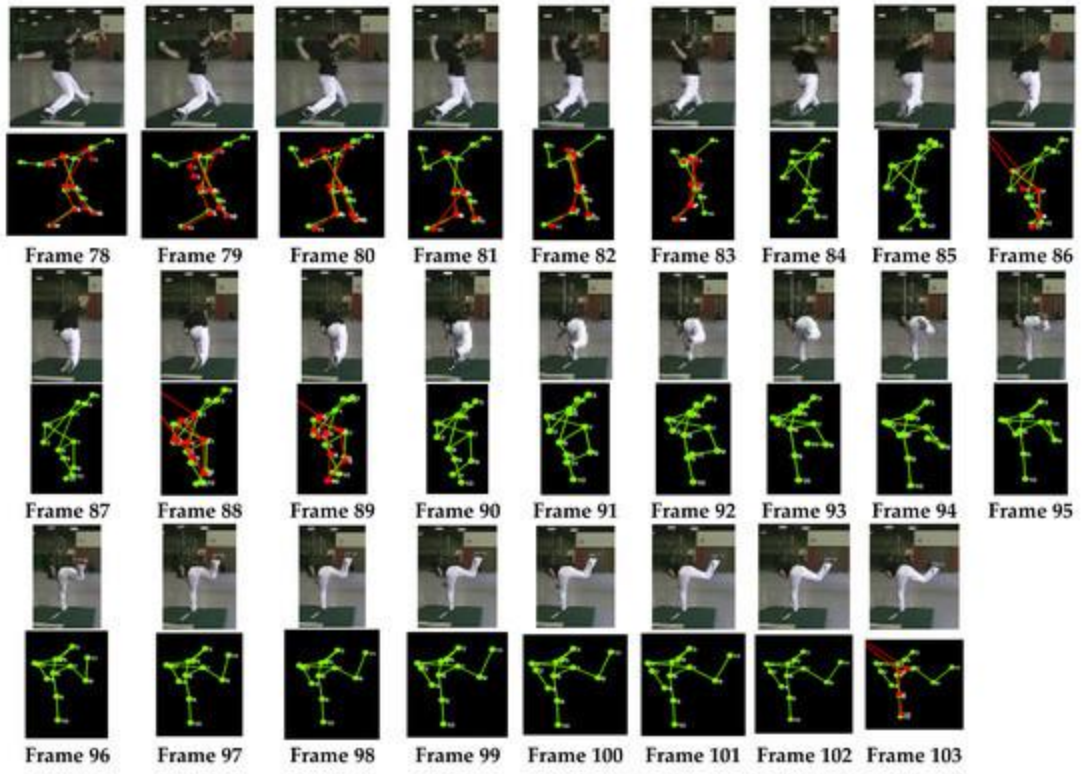
McGill



Example of DeepLabCut

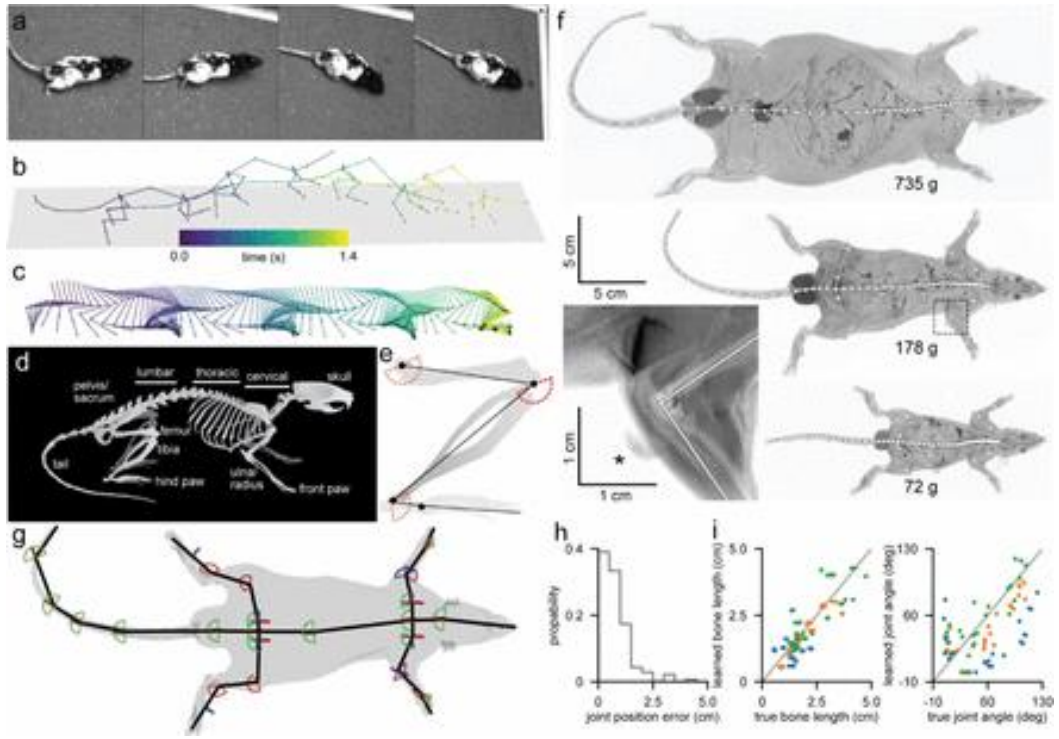


Pose estimation: bodyparts & skeleton



OnePose detection example (humans).

Chung et al. (2022). Future Internet.

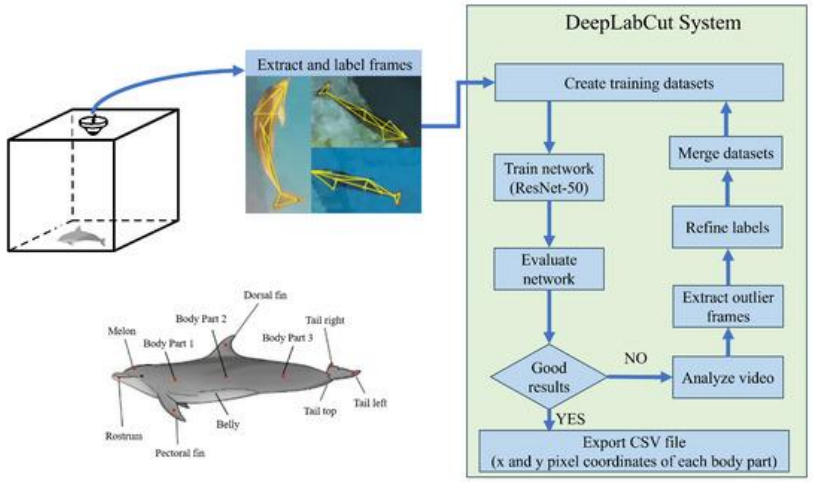


Anatomically constrained skeleton model (ACM) example (mice).

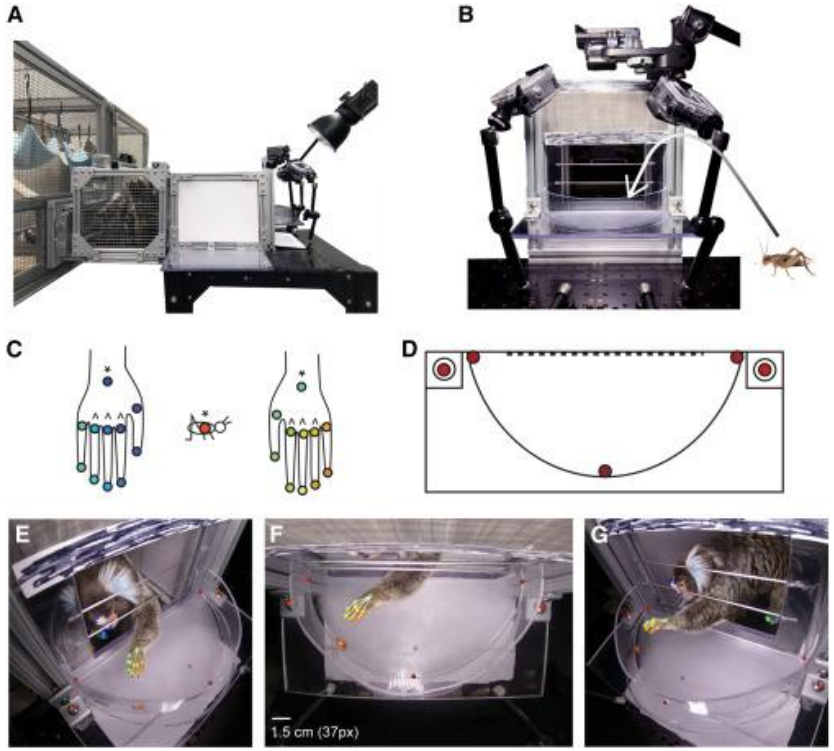
Monsees et al. (2021). Biorxiv.

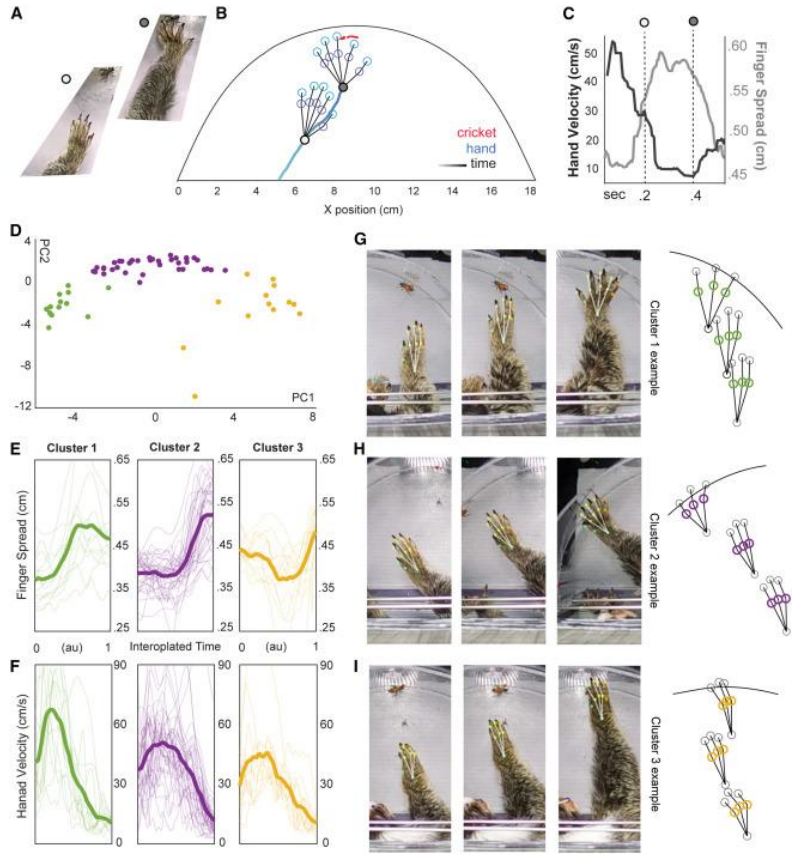
Which bodyparts can be detected?

- Anything you want to focus on your animal(s), joints, hands, fingers, snout/tail base, ears, whiskers…
- But **doesn’t** need to be a moving object (i.e. light, box, toy, etc. also works)

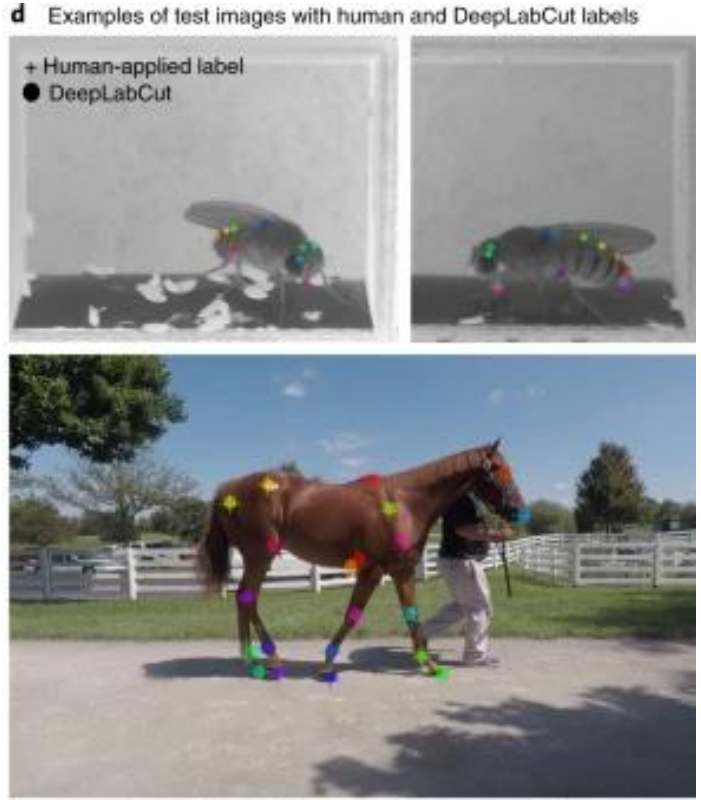


Tseng, et al. (2024). J. Mar. Sci. Eng.

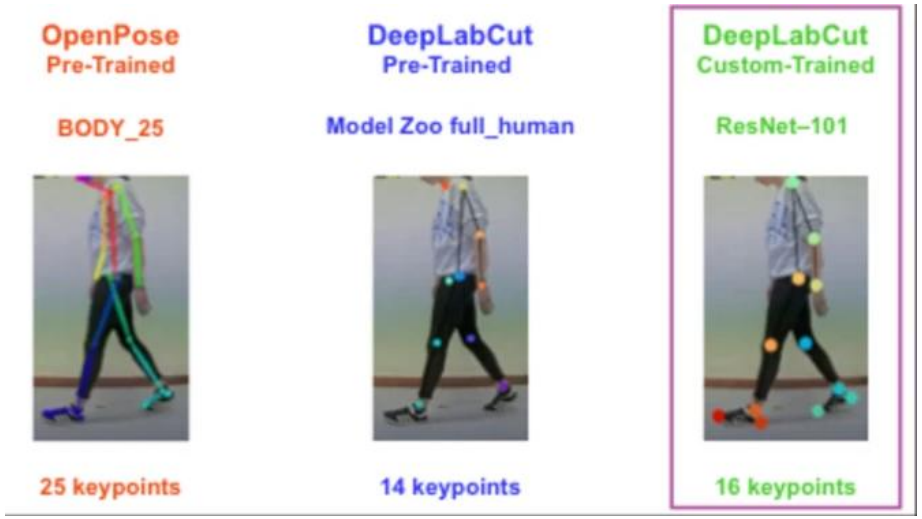




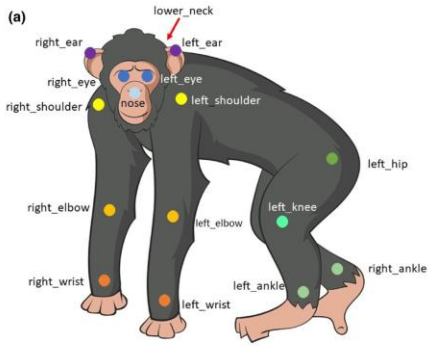
Shaw, et al. (2023). Current Biology.

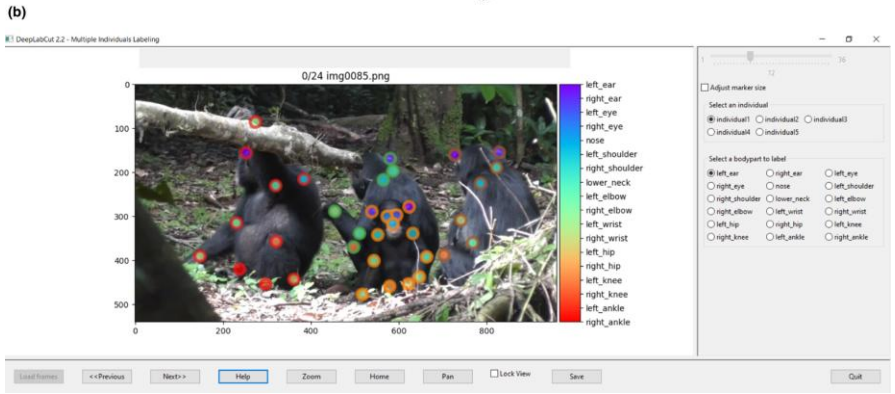


Nath, et al. (2019). Nature Protocols.



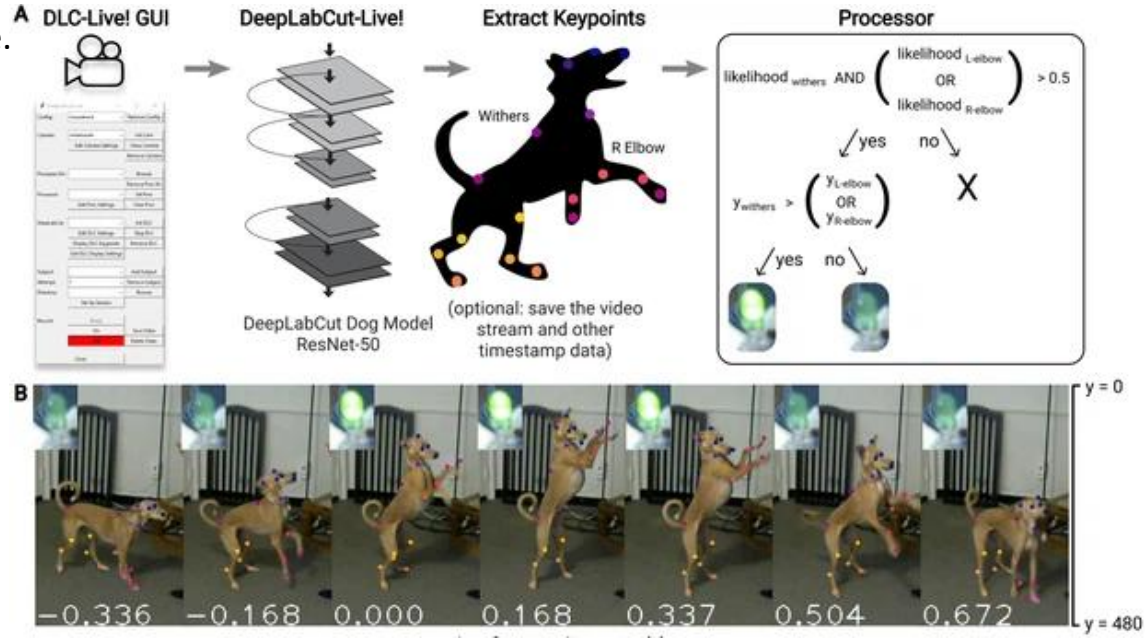
Panconi et al. (2025). Scientific Reports.

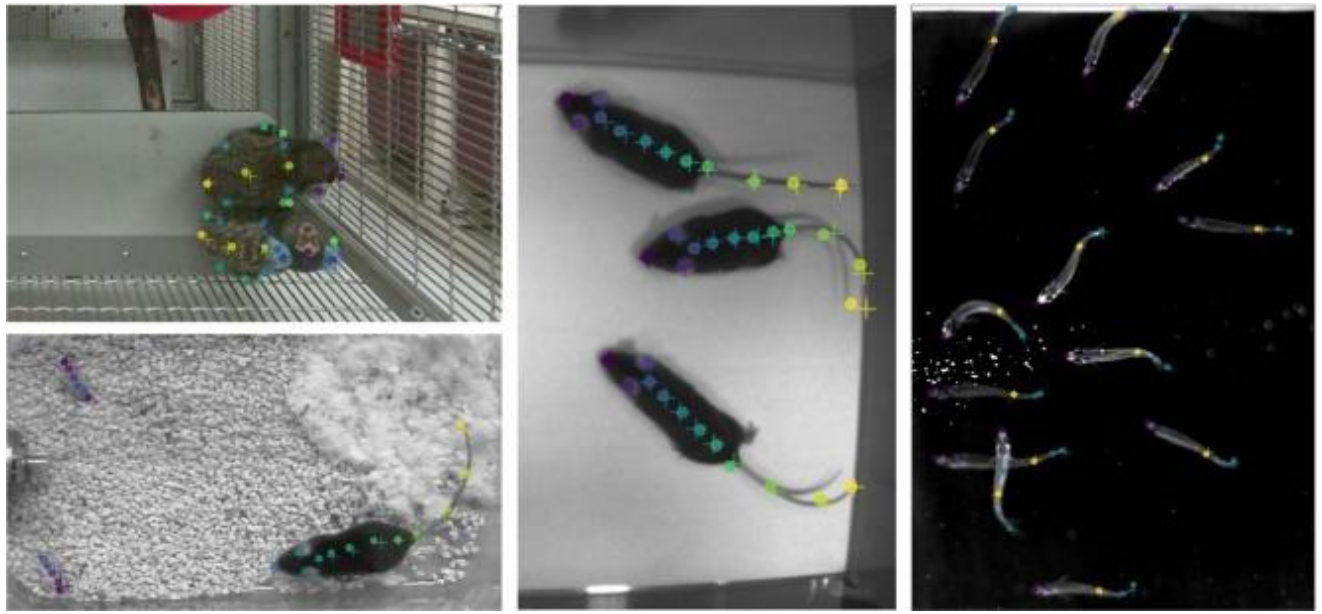




Wiltshire, et al. (2023). Journal of Animal Ecology.

Kane, et al. (2020). eLife.



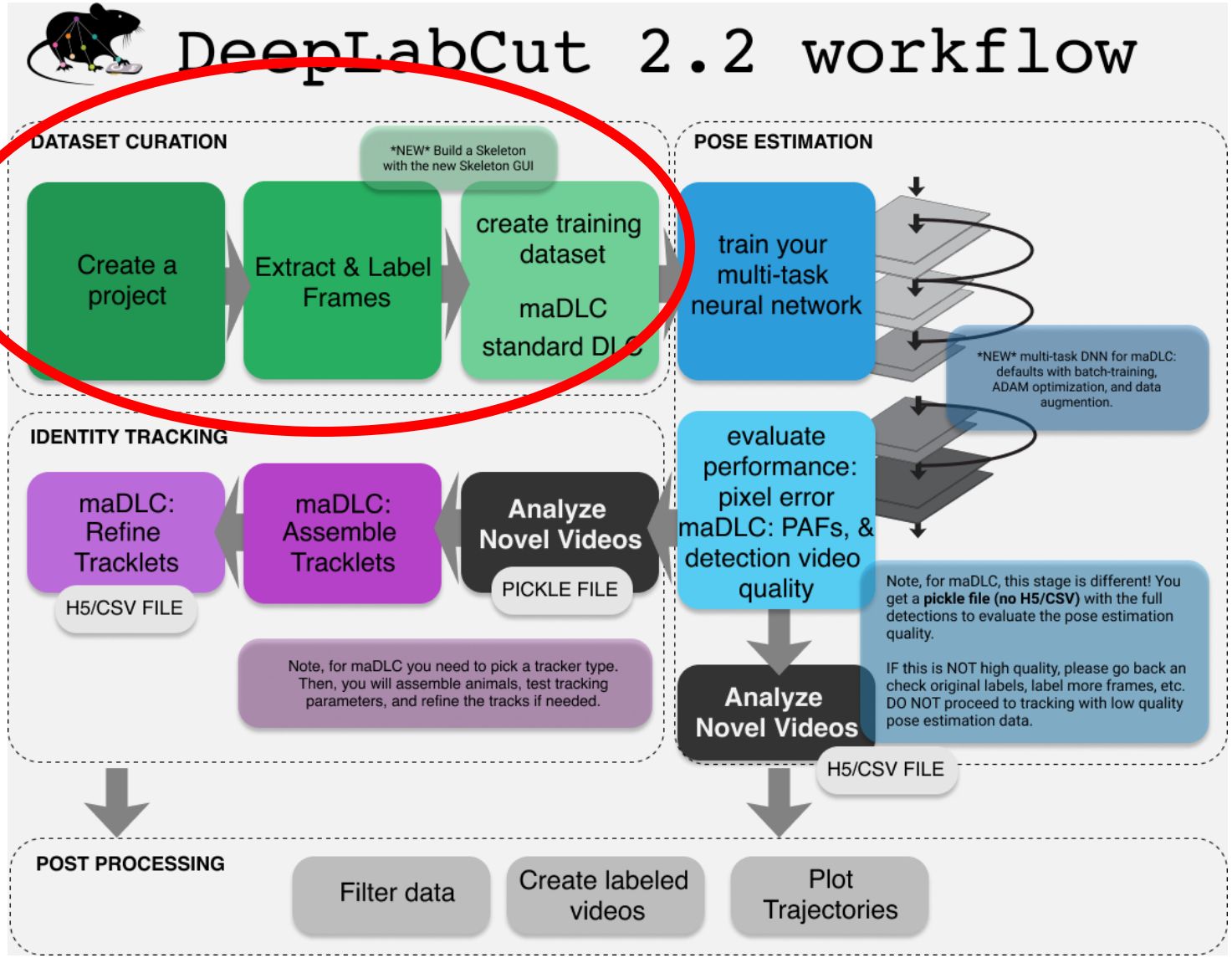


Lauer, et al. (2022). Nature Methods.

DLC workflow

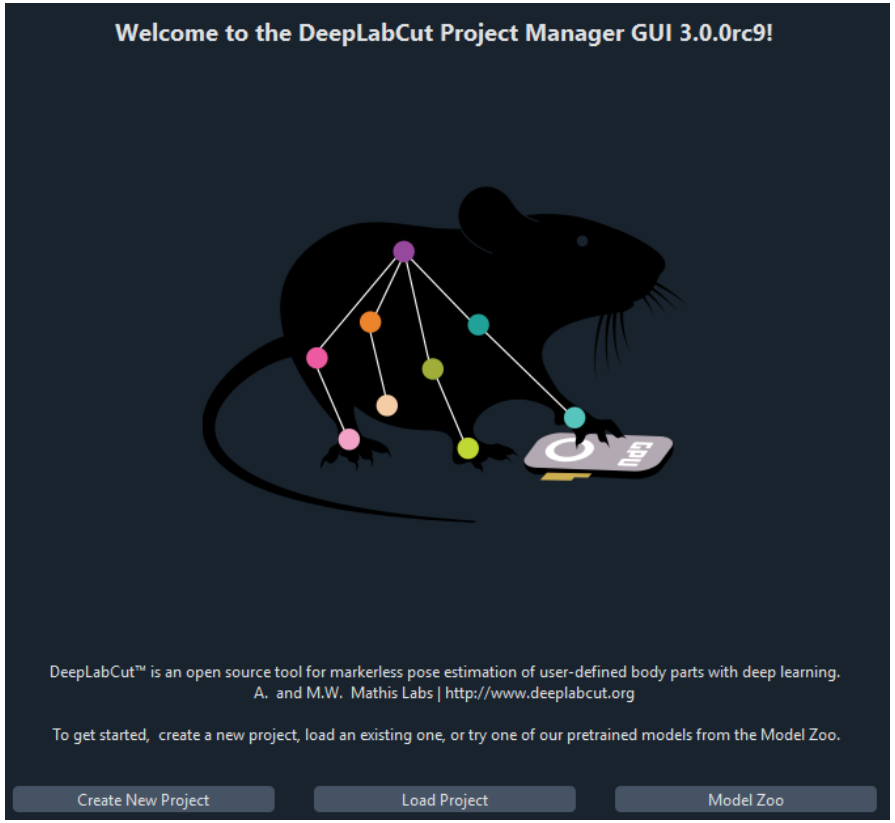
1. Opening a python session,
2. Importing deeplabcut,
3. Creating a project,
    - single animal vs. multiple animals
4. Selecting frames,
5. Labeling frames,
6. Then training a network…

DLC workflow – integration of annotation, training and inference



Lauer et al. (2022). Nature Methods.
Nath et al. (2019). Nature Protocols.

DLC GUI vs Google colab



- Simple user interface
- All analysis steps could be done here
- Could also use Prompt commands

- Access to powerful GPUs on Google's servers for faster trainingSession timeout
- Larger datasets or more complex models

But…

- Session timeout
- No Persistent Environment

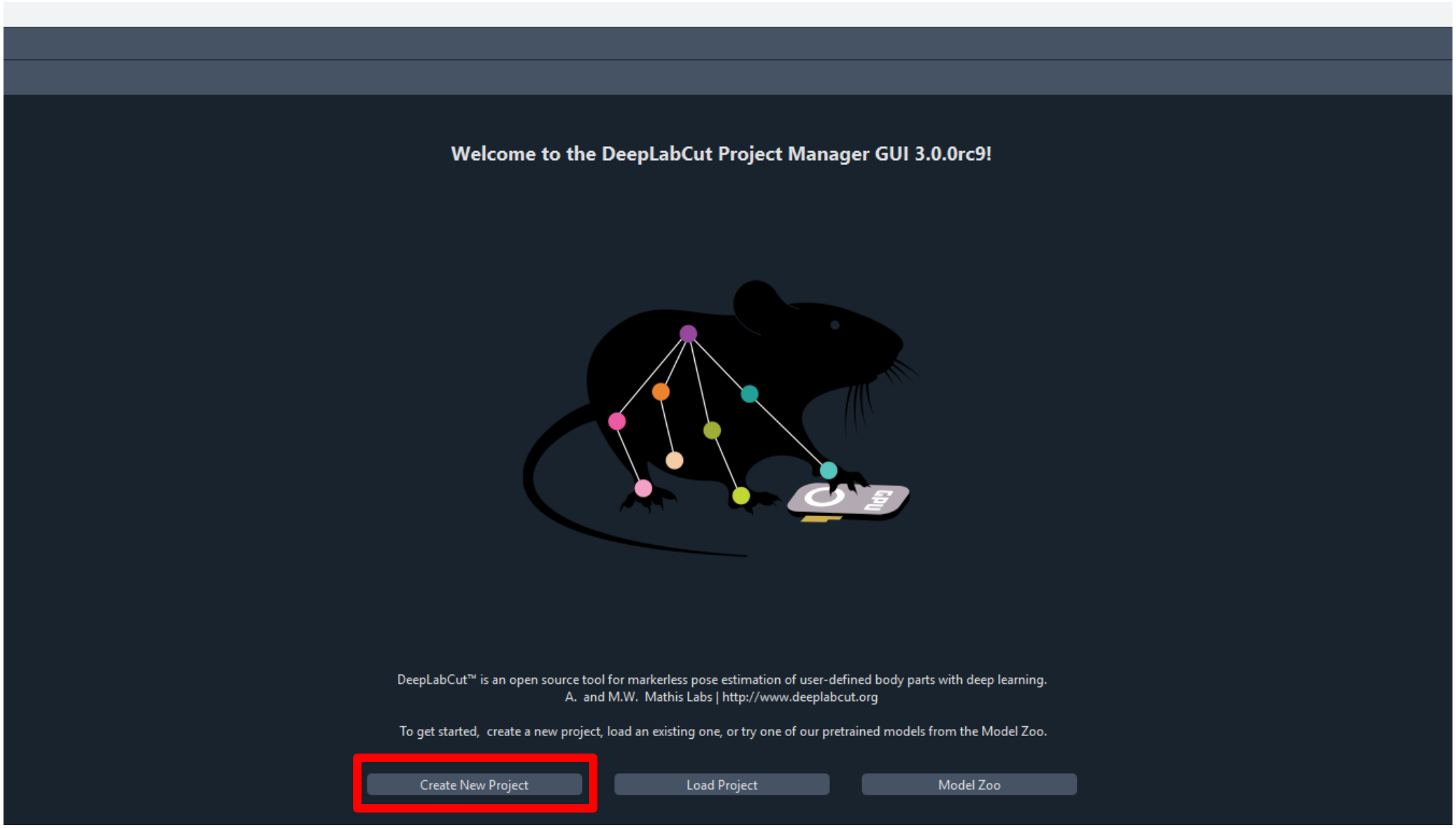
Summary of DLC commands

Table 1 | Summary of commands

Operation	Command
Open IPython and import DeepLabCut (Step 1)	`ipython` `import deeplabcut`
Create a new project (Step 2)	`deeplabcut.create_new_project('project_name','experimenter',['path of video 1','path of video2',...])`
Set a `config_path` variable for ease of use (Step 3)	`config_path = '/yourdirectory/project_name/config.yaml'`
Extract frames (Step 4)	`deeplabcut.extract_frames(config_path)`
Label frames (Steps 5 and 6)	`deeplabcut.label_frames(config_path)`
Check labels (optional)(Step 7)	`deeplabcut.check_labels(config_path)`
Create training dataset (Step 8)	`deeplabcut.create_training_dataset(config_path)`
Train the network (Step 9)	`deeplabcut.train_network(config_path)`
Evaluate the trained network (Step 11)	`deeplabcut.evaluate_network(config_path)`
Video analysis and plotting results (Step 11)	`deeplabcut.analyze_videos(config_path,['path of video 1 or folder','path of video2',…])`
Video analysis and plotting results (Step 12)	`deeplabcut.plot_trajectories(config_path,['path of video 1','path of video2',...])`
Video analysis and plotting results (Step 13)	`deeplabcut.create_labeled_video(config_path,['path of video 1','path of video2',...])`
Refinement: extract outlier frames (Step 14)	`deeplabcut.extract_outlier_frames(config_path,['path of video 1','path of video 2'])`
Refine labels (Step 15)	`deeplabcut.refine_labels(config_path)`
Combine datasets (Step 16)	`deeplabcut.merge_datasets(config_path)`

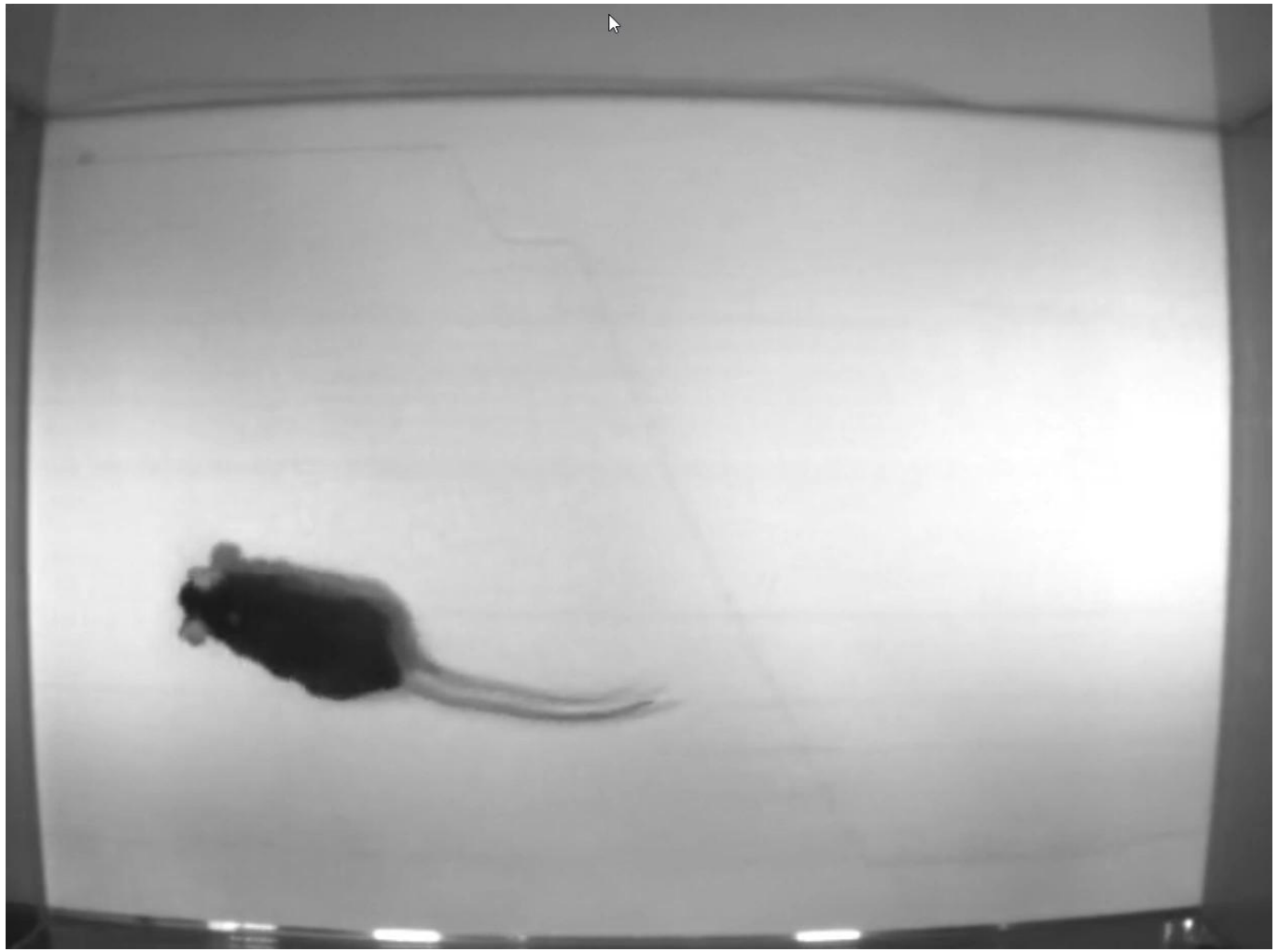
Nath et al. (2019). Nature Protocols.
https://www.nature.com/articles/s41596-019-0176-0/tables/1

Have a play with DeepLabCut GUI

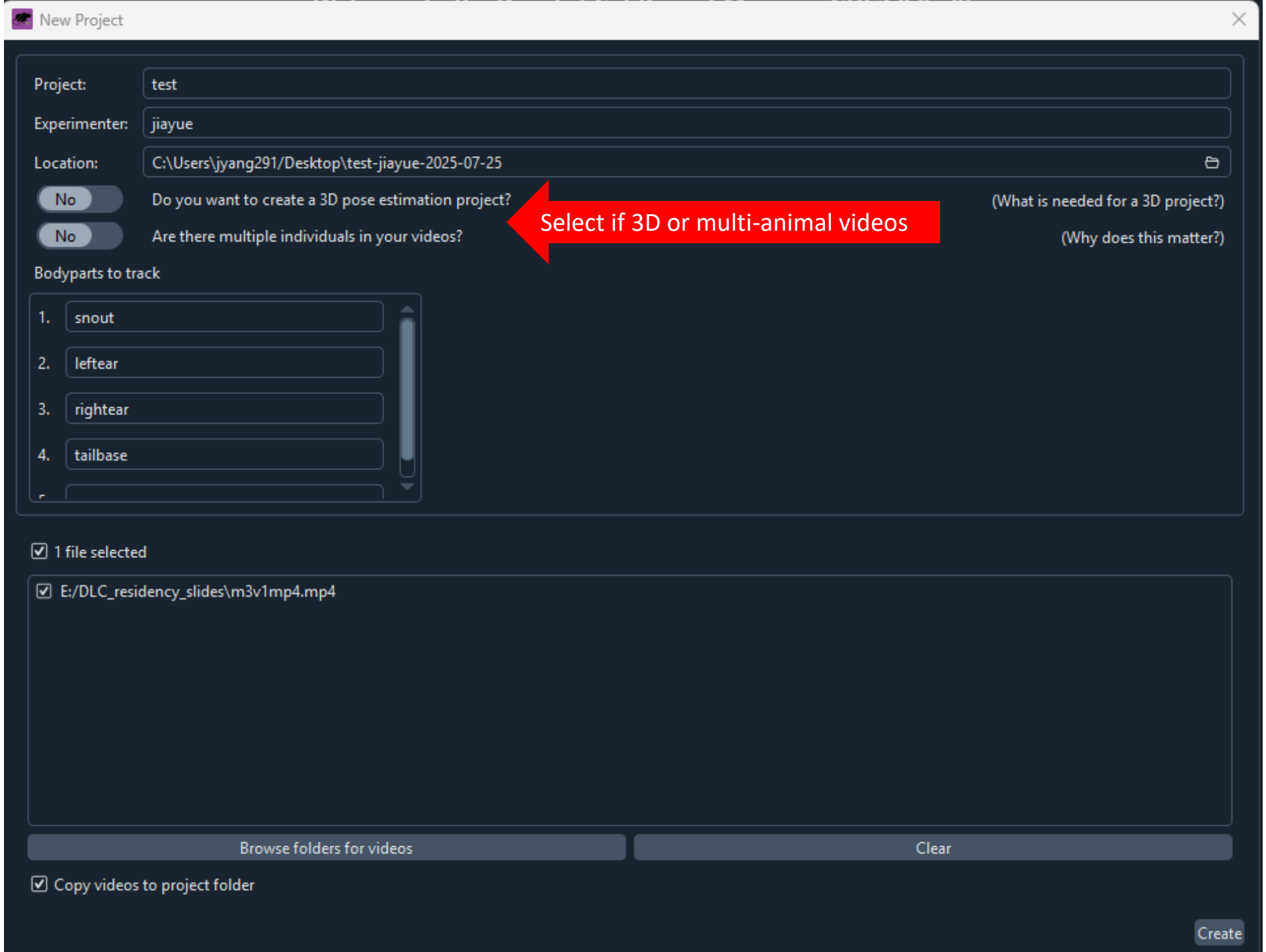


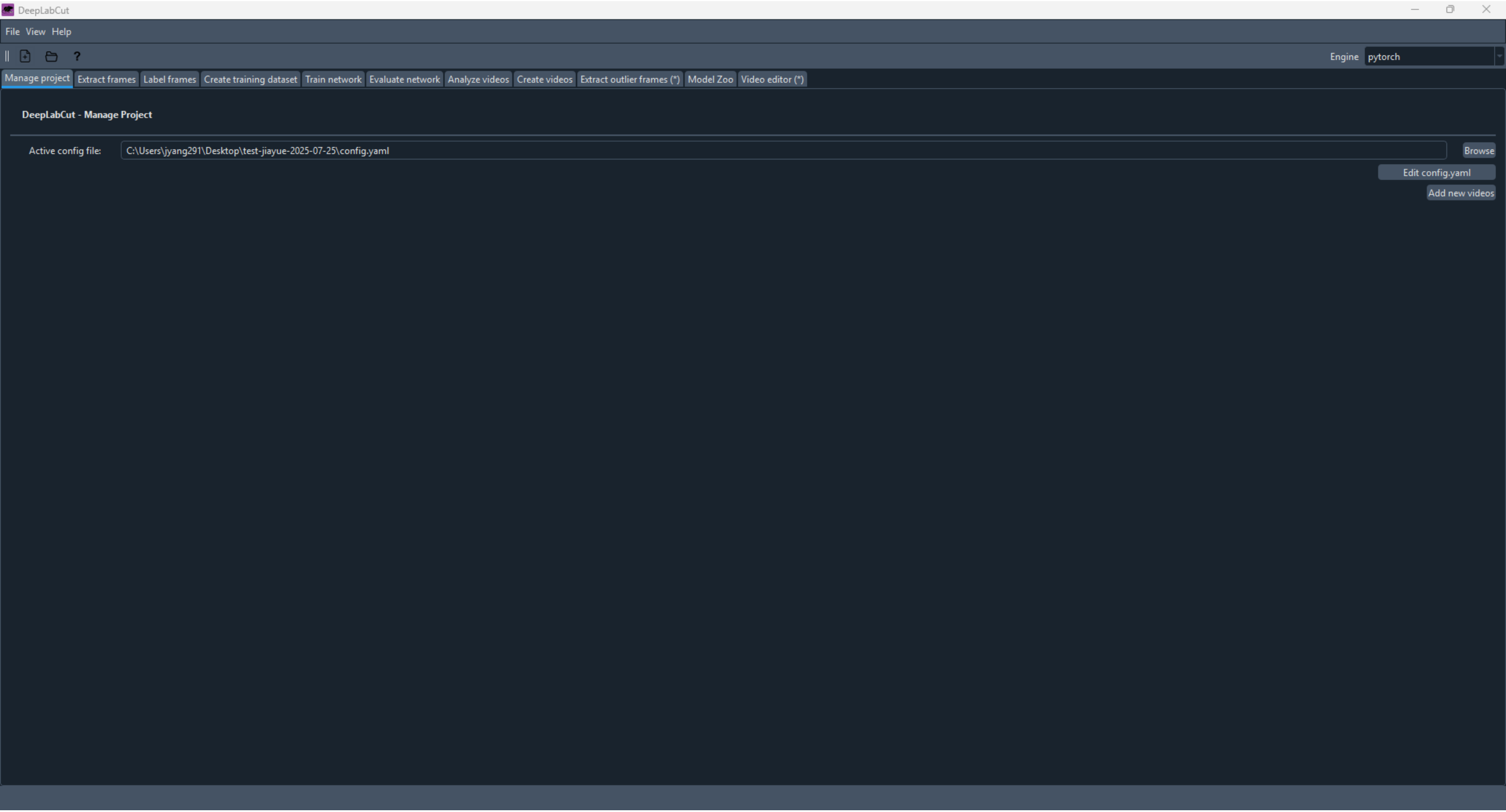
Test video

(reference: https://github.com/DeepLabCut/DeepLabCut/tree/main/examples/openfield-Pranav-2018-10-30



Mathis et al. (2018). Nature Neuroscience.
Nath, Mathis et al. (2019). Nature Protocols.
Lauer et al. (2022). Nature Methods.
Ye et al. (2024). Nature Communications.


New Project
Project:
test
Experimenter:
jiayue
Location:
C:\Users\jyang291/Desktop\test-jiayue-2025-07-25
No
Do you want to create a 3D pose estimation project?
(What is needed for a 3D project?)
No
Are there multiple individuals in your videos?
(Why does this matter?)
Select if 3D or multi-animal videos
Bodyparts to track
1. snout
2. leftear
3. rightear
4. tailbase
1 file selected
E:/DLC_residency_slides\m3v1mp4.mp4
Browse folders for videos
Clear
Copy videos to project folder
Create


DeepLabCut
File View Help
Engine pytorch
Manage project
Extract frames
Label frames
Create training dataset
Train network
Evaluate network
Analyze videos
Create videos
Extract outlier frames (*)
Model Zoo
Video editor (*)
DeepLabCut - Manage Project
Active config file:
C:\Users\jyang291\Desktop\test-jiayue-2025-07-25\config.yaml
Browse
Edit config.yaml
Add new videos

Box 1 | Glossary of parameters in the project configuration file (*config.yaml*)

The *config.yaml* file sets the various parameters for generation of the training set file and evaluation of results. The meaning of these parameters is defined here, as well as referenced in the relevant steps.

Parameters set during the project creation

- `task`: Name of the project (e.g., mouse-reaching). (Set in Step 1; do not edit.)
- `scorer`: Name of the experimenter (set in Step 1; do not edit).
- `date`: Date of creation of the project. (Set in Step 1; do not edit).
- `project_path`: Full path of the project, which is set in Step 1; edit this if you need to move the project to a cluster/server/another computer or a different directory on your computer.
- `video_sets`: A dictionary with the keys as the full path of the video file and the values, `crop` as the cropping parameters used during frame extraction. (Step 1; use the function `add_new_videos` to add more videos to the project; if necessary, the paths can be edited manually, and the `crop` can be edited manually).

Important parameters to edit after project creation

- `bodyparts`: List containing names of the points to be tracked. The default is set to `bodypart1, bodypart2, bodypart3, objectA`. Do not change after labeling frames (and saving labels). You can add additional labels later, if needed.
- `numframes2pick`: This is an integer that specifies the number of frames to be extracted from a video or a segment of video. The default is set to 20.
- `colormap`: This specifies the colormap used for plotting the labels in images or videos in many steps. Matplotlib colormaps are possible (https://matplotlib.org/examples/color/colormaps_reference.html).
- `dotsize`: Specifies the marker size when plotting the labels in images or videos. The default is set to 12.
- `alphavalue`: Specifies the transparency of the plotted labels. The default is set to 0.5.
- `iteration`: This keeps the count of the number of refinement iterations used to create the training dataset. The first iteration starts with 0 and thus the default value is 0. This will auto-increment once you merge a dataset (after the optional refinement stage).

If you are extracting frames from long videos

- `start`: Start point of interval to sample frames from when extracting frames. Value in relative terms of video length, i.e., `[start=0,stop=1]` represents the full video. The default is 0.
- `stop`: Same as `start`, but specifies the end of the interval. Default is 1.

Related to the Neural Network Training

- `TrainingFraction`: This is a two-digit floating-point number in the range [0–1] used to split the dataset into training and testing datasets. The default is 0.95.
- `resnet`: This specifies which pre-trained model to use. The default is 50 (user can choose 50 or 101; see also Mathis et al.[12]).

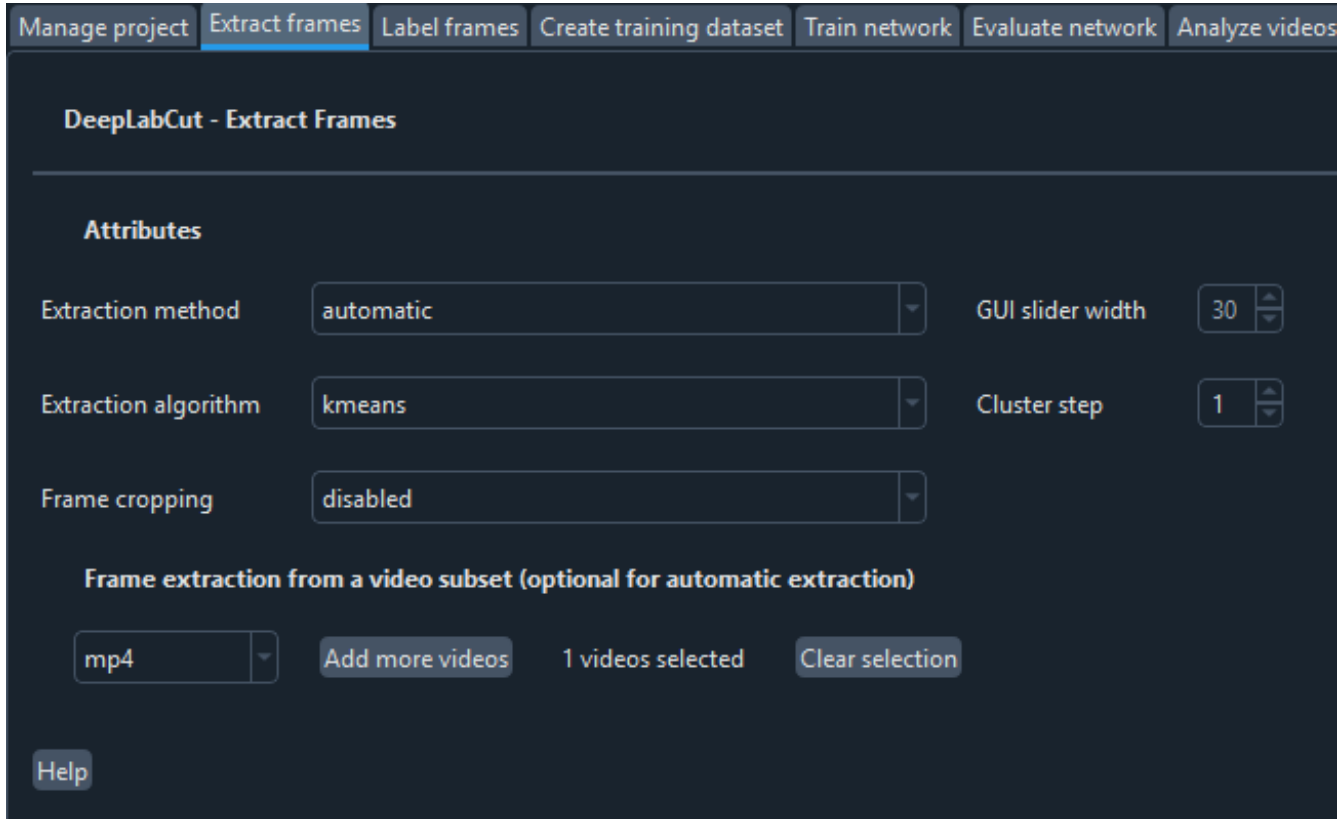
Used during video analysis (Step 13)

- `batch_size`: This specifies how many frames to process at once during inference (for tuning of this parameter, see Mathis and Warren[27]).
- `snapshotindex`: This specifies which checkpoint to use to evaluate the network. The default is −1. Use `all` to evaluate all the checkpoints. Snapshots refer to the stored TensorFlow configuration, which holds the weights of the feature detectors.
- `pcutoff`: This specifies the threshold of the likelihood and helps to distinguish likely body parts from uncertain ones. The default is 0.1.
- `cropping`: This specifies whether the analysis video needs to be cropped (in Step 13). The default is `False`.
- `x1,x2,y1,y2:` These are the cropping parameters used for cropping novel video(s). The default is set to the frame size of the video.

Used during refinement steps

- `move2corner`: In some (rare) cases, the predictions from DeepLabCut will be outside of the image (because of the location refinement shifts). This binary parameter ensures that those points are mapped to a user-defined point within the image so that the label can be manually moved to the correct location. The default is `True`.
- `corner2move2`: This is the target location, if `move2corner` is `True`. The default is set to `(50,50)`.

Nath et al. (2019). Nature Protocols.
https://www.nature.com/articles/s41596-019-0176-0

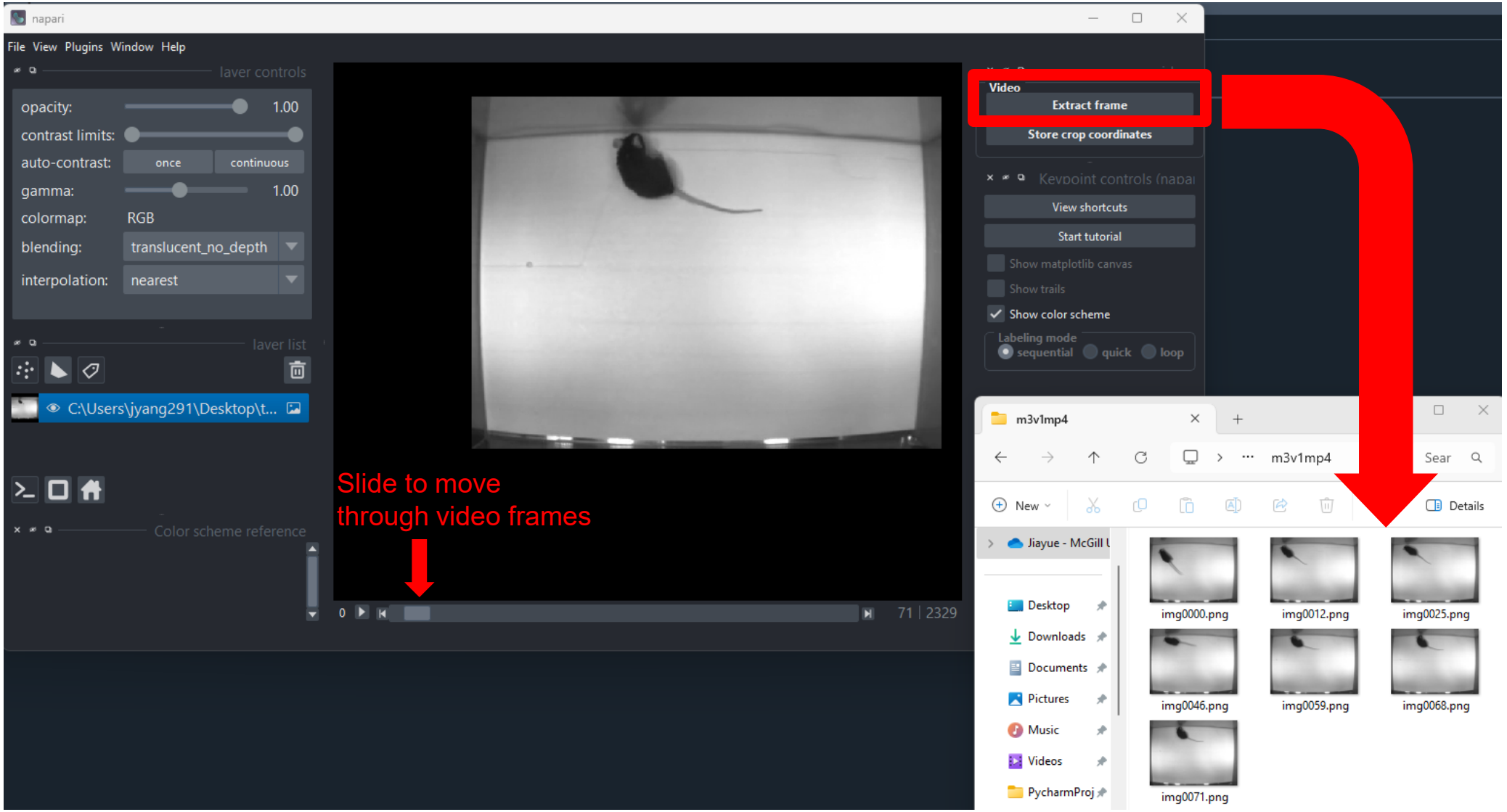


With **automatic** extraction:

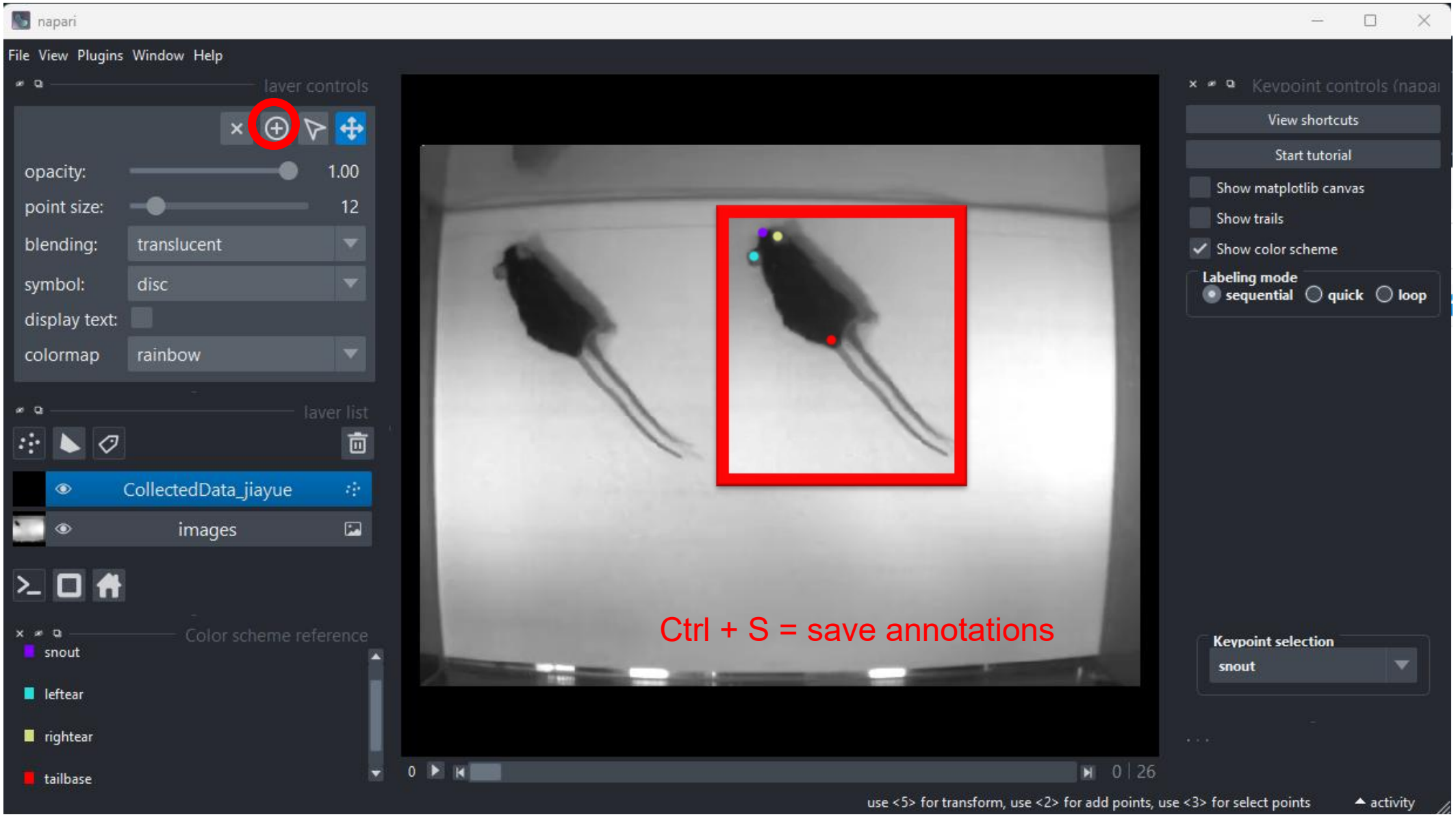
→ directly proceed to the frame labeling

→ Console: “successfully extracted frames.”

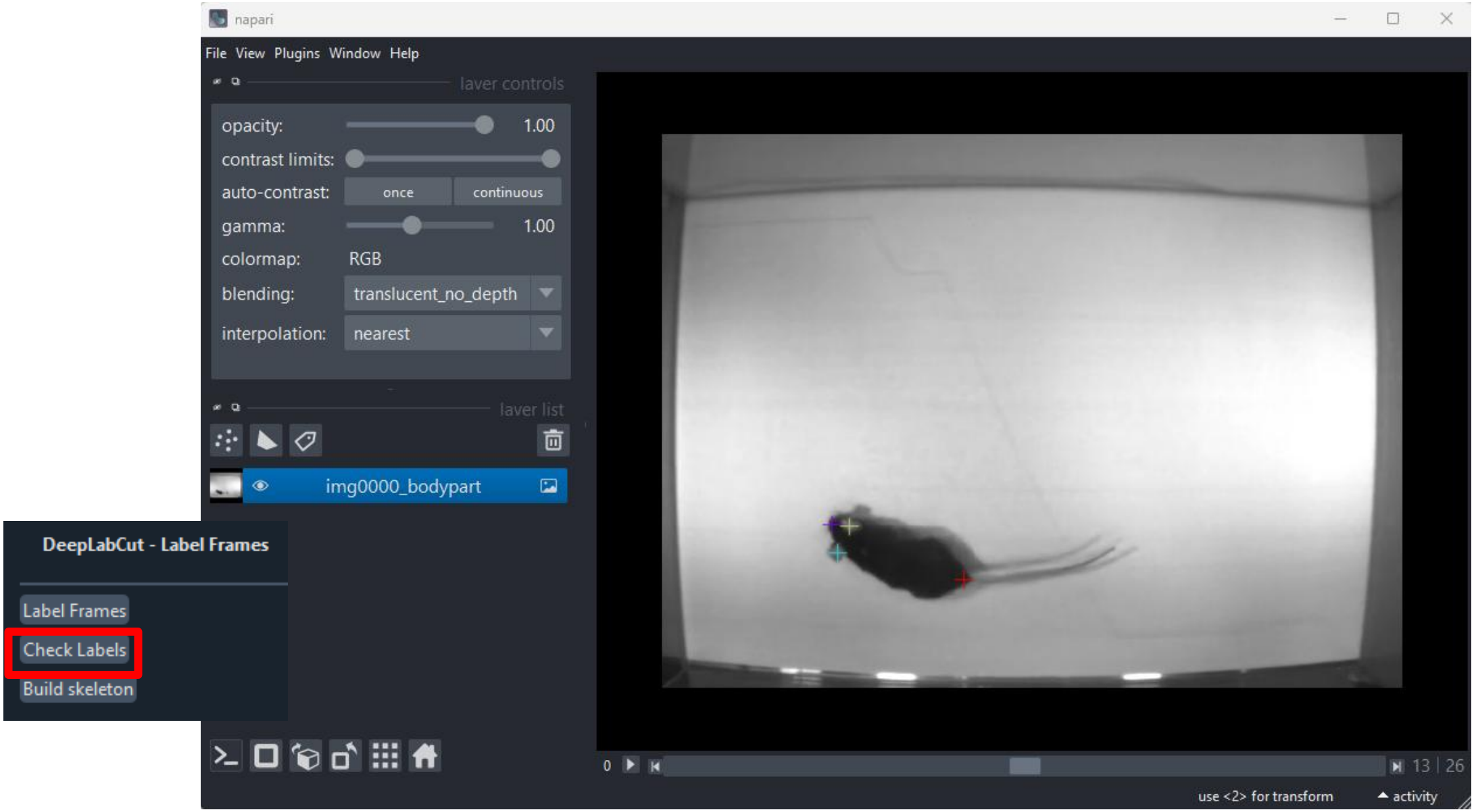
If manual extraction:



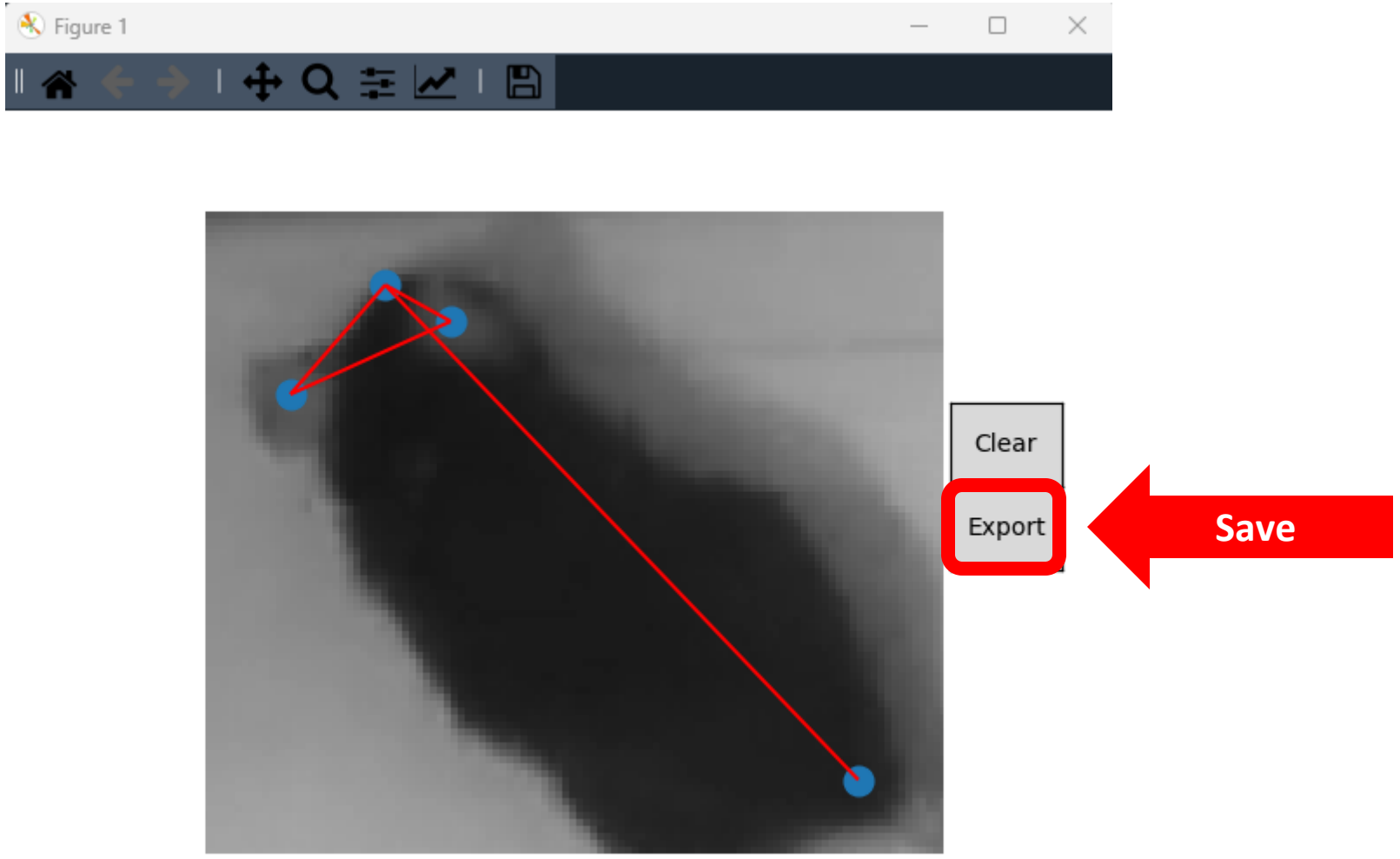
Label frames

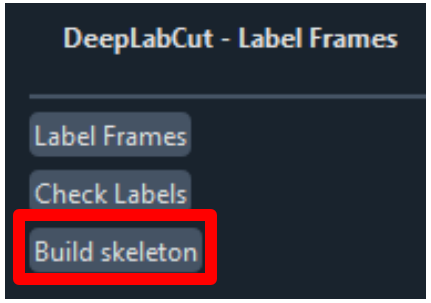


Label frames

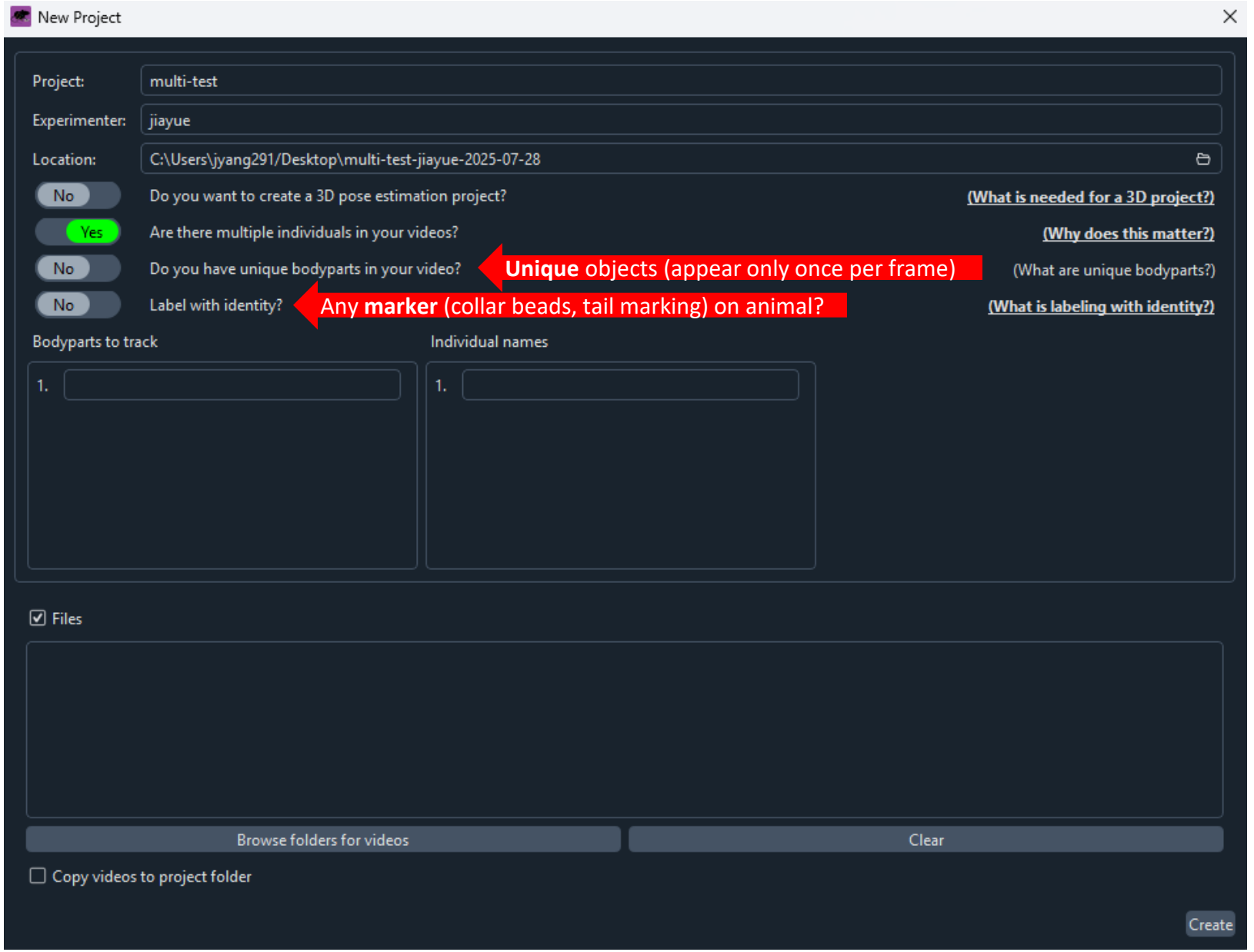


Label frames

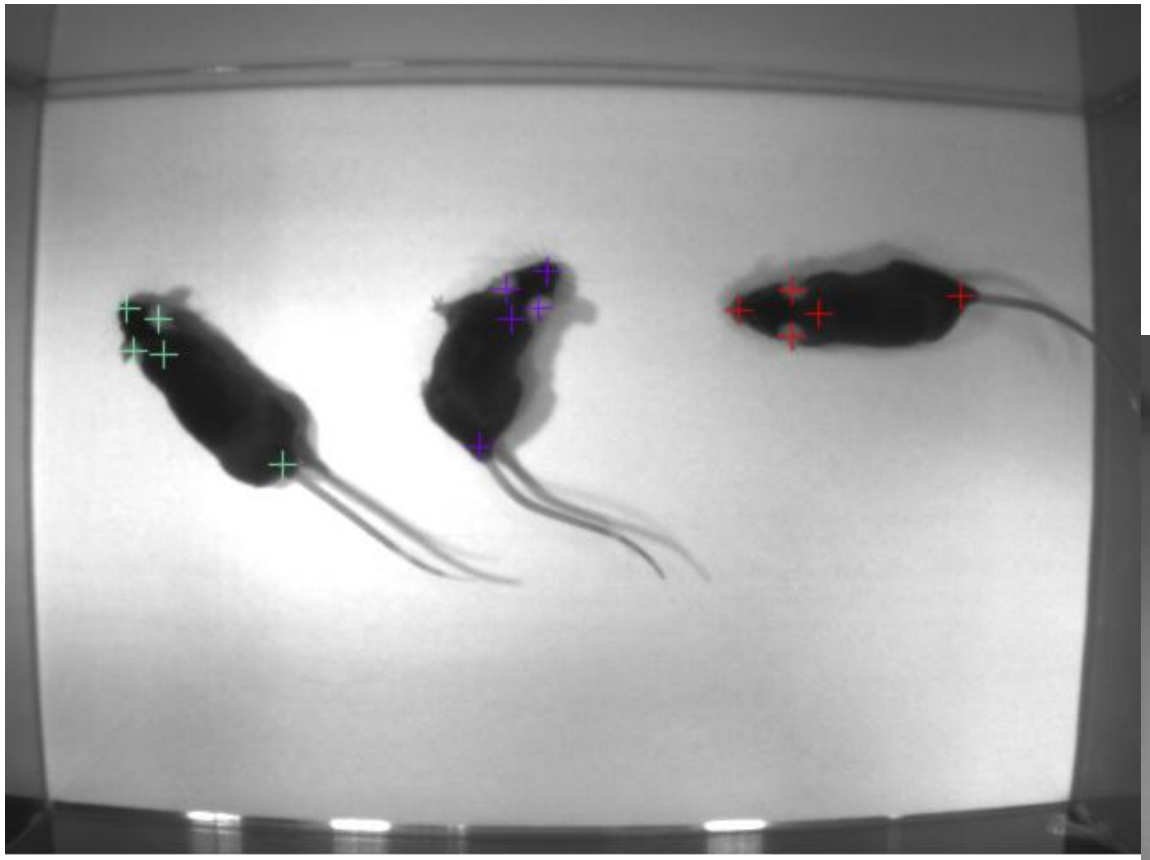


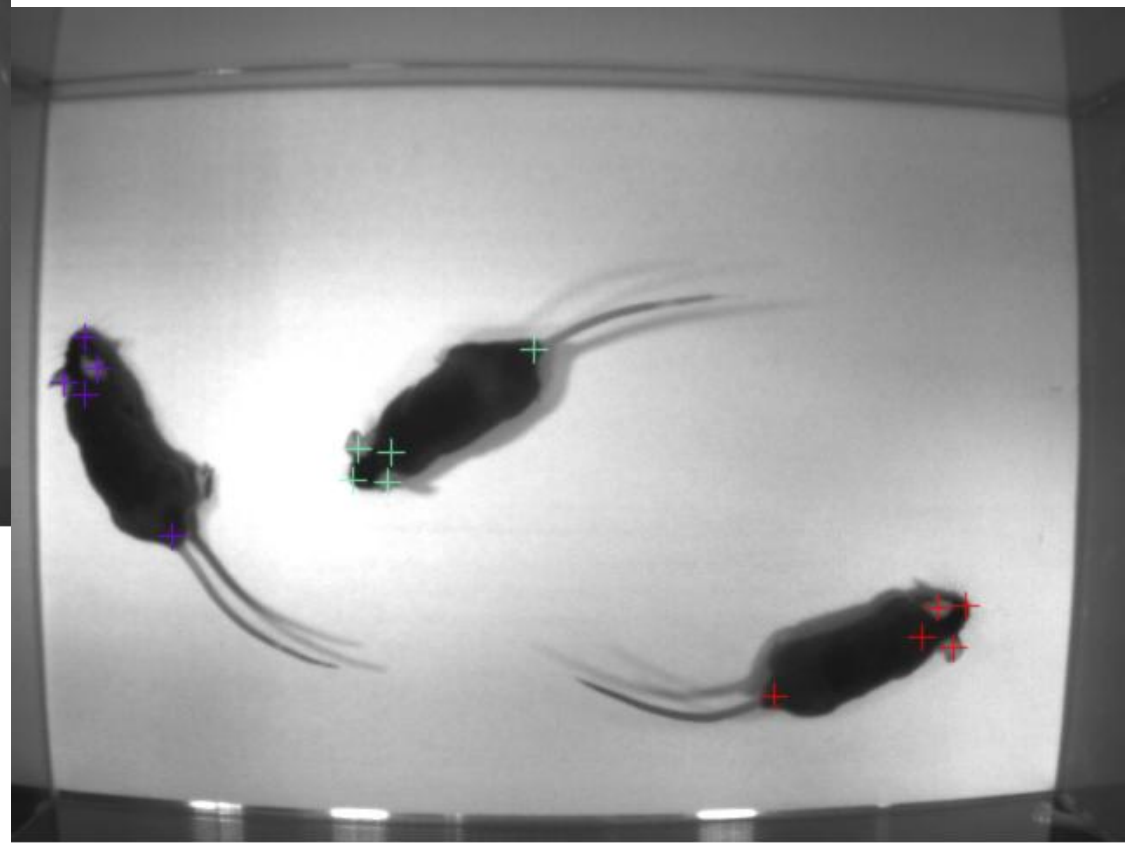


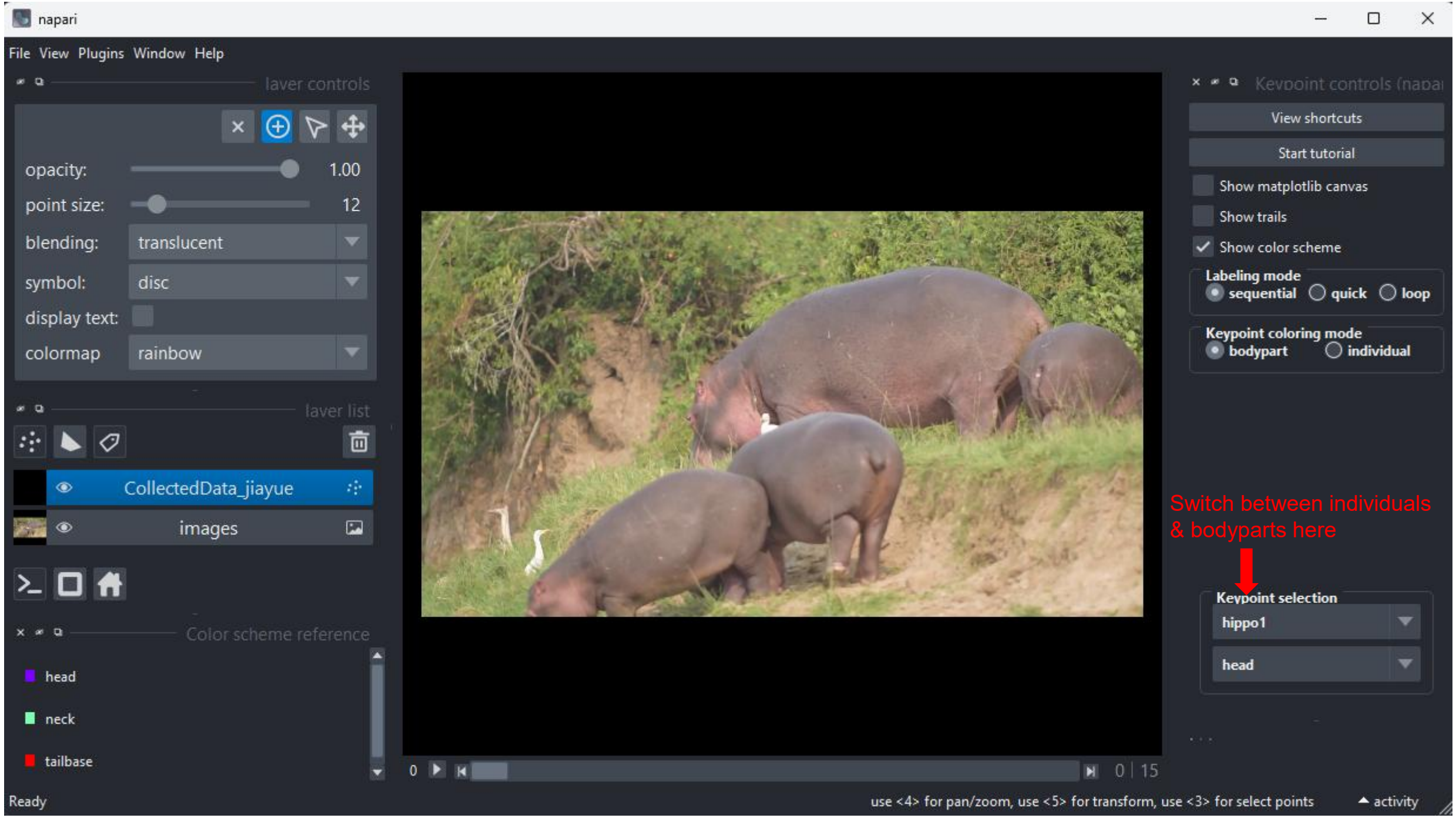
What if multiple animals?

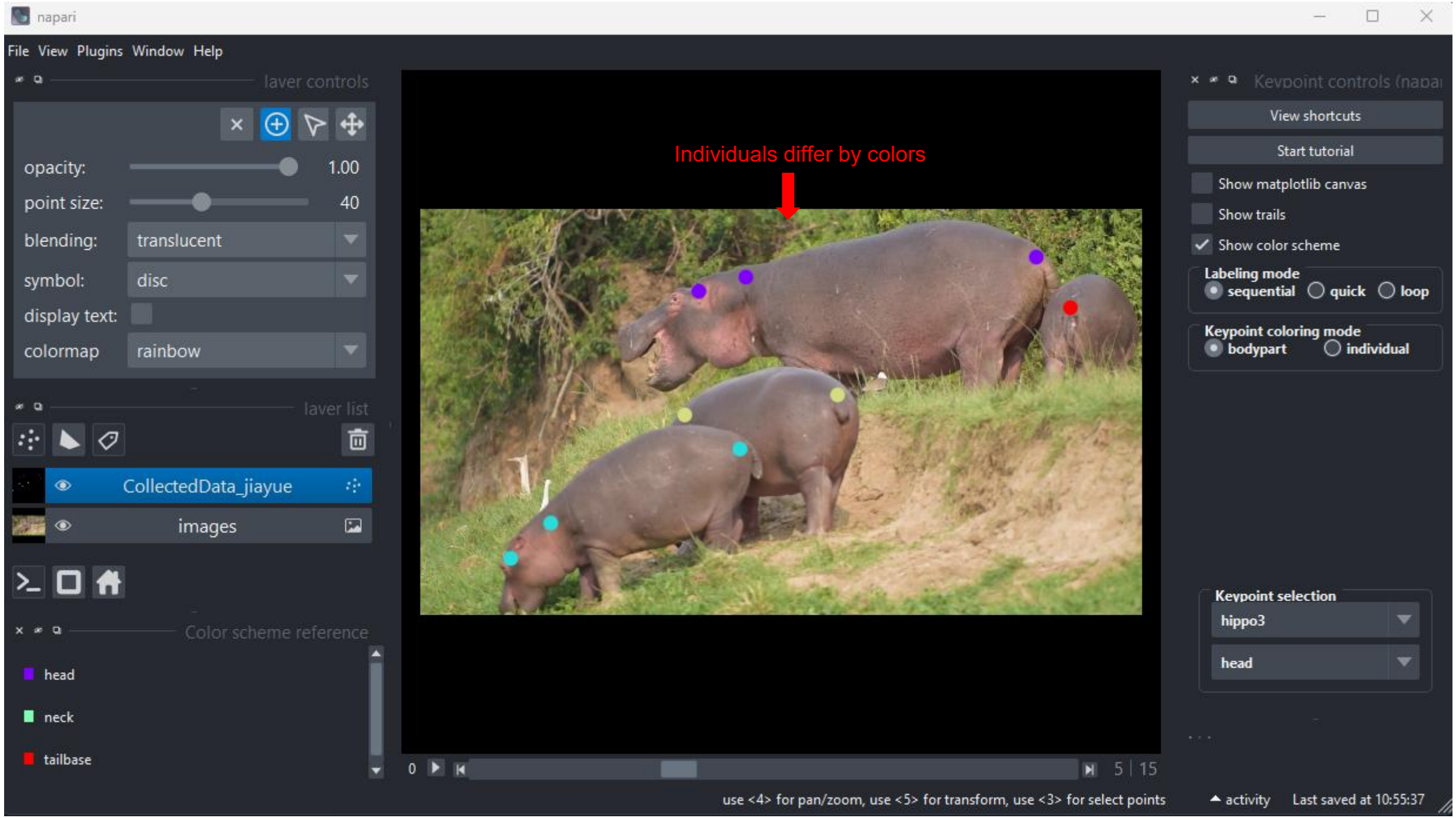


Example: mice





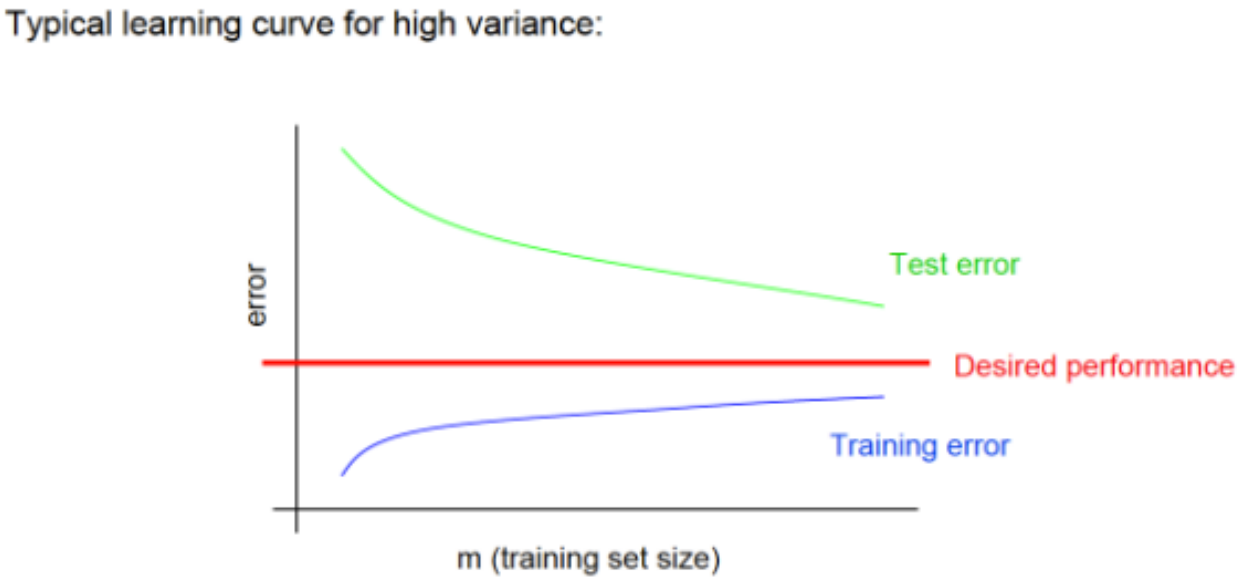

napari
File View Plugins Window Help
layer controls
opacity: 1.00
point size: 12
blending: translucent
symbol: disc
display text:
colormap rainbow
layer list
CollectedData_jiayue
images
Color scheme reference
head
neck
tailbase
0
0 | 15
Keypoint controls (napa
View shortcuts
Start tutorial
Show matplotlib canvas
Show trails
Show color scheme
Labeling mode
sequential quick loop
Keypoint coloring mode
bodypart individual
Switch between individuals & bodyparts here
Keypoint selection
hippo1
head
Ready
use <4> for pan/zoom, use <5> for transform, use <3> for select points
activity


napari
File View Plugins Window Help
layer controls
opacity: 1.00
point size: 40
blending: translucent
symbol: disc
display text:
colormap rainbow
layer list
CollectedData_jiayue
images
Color scheme reference
head
neck
tailbase
Individuals differ by colors
0
5 | 15
Keypoint controls (napa
View shortcuts
Start tutorial
Show matplotlib canvas
Show trails
Show color scheme
Labeling mode
sequential quick loop
Keypoint coloring mode
bodypart individual
Keypoint selection
hippo3
head
use <4> for pan/zoom, use <5> for transform, use <3> for select points
activity
Last saved at 10:55:37

Data management:
How should a training dataset look like?

A good training set should look like:

- Covers the range of **variability** expected in your videos, with **clear and consistently** labeled frames showing **diverse poses, conditions, and subjects**.



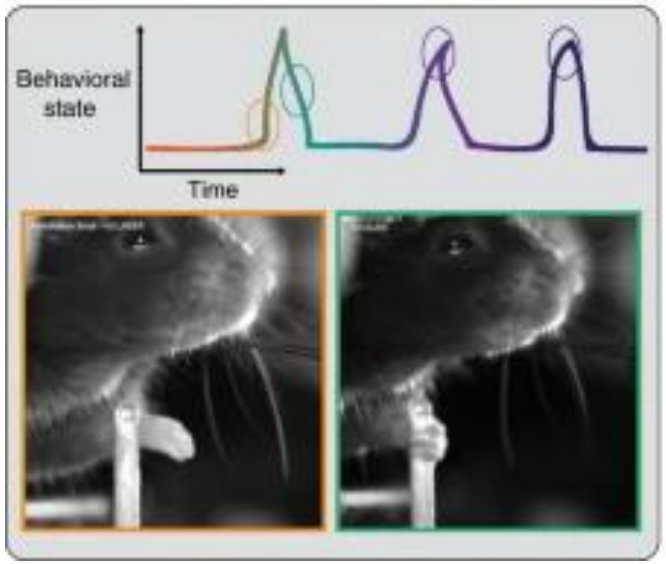
Frame extraction

Select videos from which to grab frames:

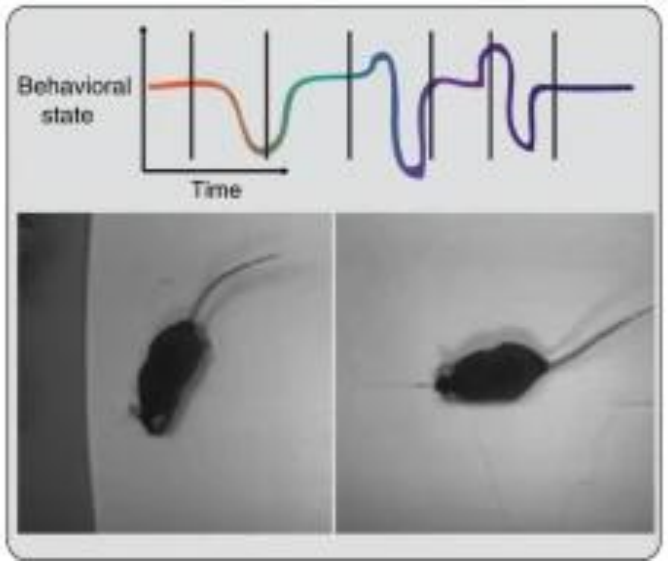
Use videos with images from
-Different sessions reflecting (if the case) varying light conditions, backgrounds, setups, and camera angles
-Different individuals, especially if they look different (i.e., brown and black mice)

3 methods for frame extraction to create a labeled train/test set

Image-based clustering (*k*-means)



Random temporal sampling (uniform)

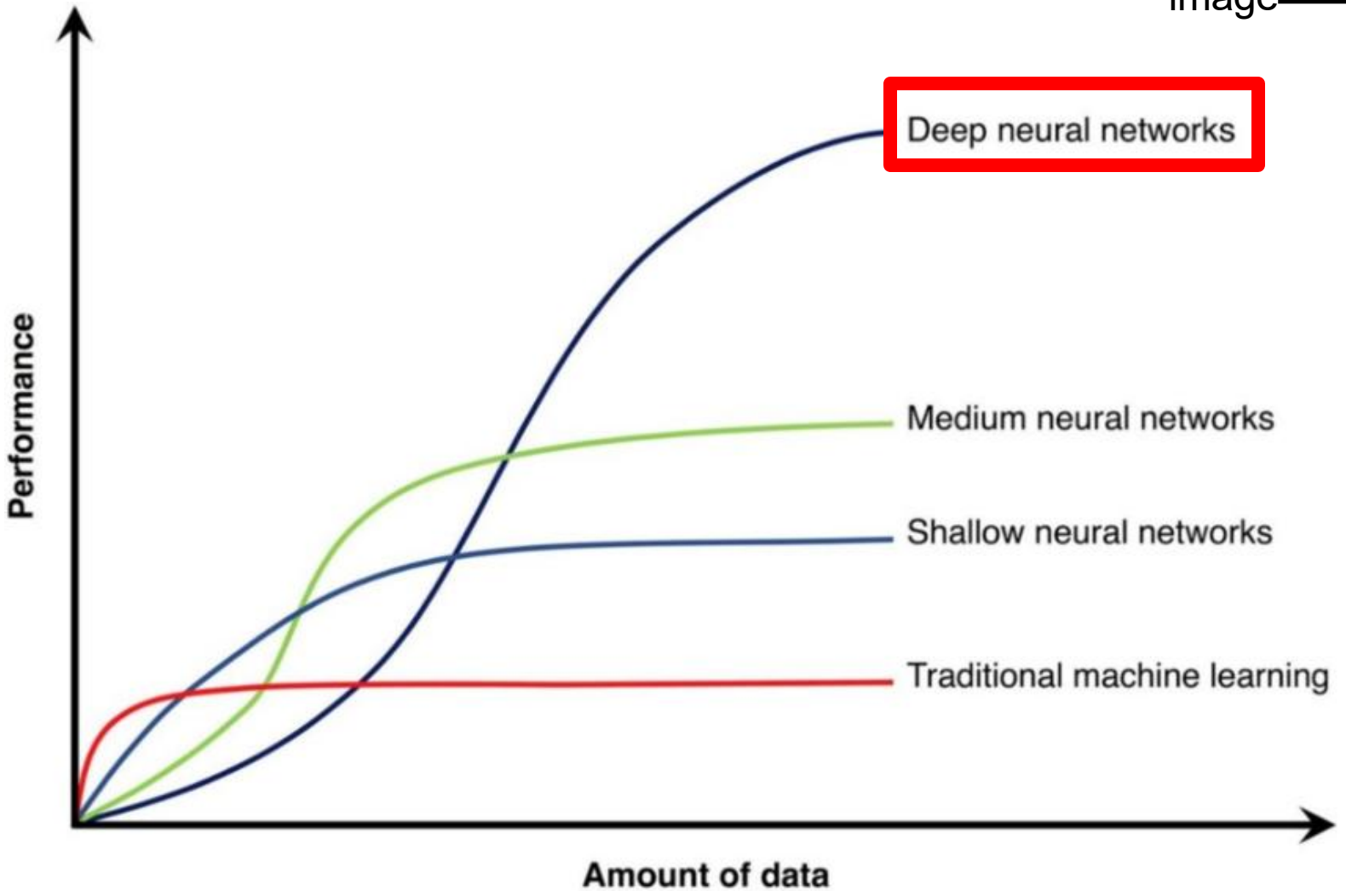


GUI for manual frame grabbing

Nath et al. (2019). Nature Protocols.

Data amount vs. performance

image → Predictor → pose



Labeling on GUI (napari):

- Zooming
- Only need to label visible features/bodyparts
- The bodypart label (names) are defined in the config file

  - could be edited or added (if need to add extra)
- The labels and their meanings could be defined by users

Labeling on GUI (napari):

- **Consistent** labeling
    - **Same locations** across frames
- Please avoid label error, especially:
    - If **small number** of images
    - **Left** and **right** labels, avoid flipping when labeling
- **Check labels** by plotting them on all figures (before training)
- Can reopen GUI to **correct labels** if mistakes