Pose Estimation in NWB

ndx-pose

Neurodata Extensions Catalog

bit.ly/nwb-cosyne-2023

Pose Estimation in NWB

NWB Currently supports writing from two popular pose estimation frameworks

- SLEAP
- DeepLabCut (DLC)

We do not have time to fully explain the details of how these packages work…

…so we will simply explain the input-output (I/O) workflow

Input: Storing natural behavior in NWB

- The NWB data type for videos called the `ImageSeries`

- It has two modes
  - Internal
    - ➢ each frame of the video is extracted and written as a Dataset
    - ➢ same as any other array-valued series you might find in an NWB file

```
nwbfile.acquisition
```

```
{'Video: Rat10-20140708-01-prerun': Video: Rat10-20140708-01-prerun pynwb.image.ImageSeries
 Fields:
   data: <HDF5 dataset "data": shape (5773, 360, 640, 3), type "|u1">
   description: Video recorded by camera.
   timestamps: <HDF5 dataset "timestamps": shape (5773,), type "<f8">
   timestamps_unit: seconds
   unit: Frames
```

Input: Storing natural behavior in NWB

- The NWB data type for videos called the `ImageSeries`

- It has two modes
  - Internal
    - ➢ each frame of the video is extracted and written as a Dataset
    - ➢ same as any other array-valued series you might find in an NWB file
  - External
    - ➢ data type instead contains a *path* to the video file(s) outside the NWB file

```
'OriginalVideoBodyCamera': OriginalVideoBodyCamera pynwb.image.ImageSeries
Fields:
  data: <HDF5 dataset "data": shape (0, 0, 0), type "|u1">
  description: The original video each pose was estimated from.
  external_file: <StrDataset for HDF5 dataset "external_file": shape (1,), type "|O">
  format: external
```

Input: Storing natural behavior in NWB

- The NWB data type for videos called the `ImageSeries`

- It has two modes
  - Internal
    - ➢ each frame of the video is extracted and written as a Dataset
    - ➢ same as any other array-valued series you might find in an NWB file
  - External
    - ➢ data type instead contains a *path* to the video file(s) outside the NWB file

```
nwbfile.acquisition["OriginalVideoBodyCamera"].external_file[:]
```

```
array(['./sub-CSHL051_ses-ecb5520d-1358-434c-95ec-93687ecd1396_behavior+ecephys+image/sub-CSHL051_ses-ecb5520d-1358-434c-95ec-93687ecd1396_OriginalVideoBodyCamera.mp4'],
      dtype=object)
```

Input: Storing natural behavior in NWB

- The NWB data type for videos called the `ImageSeries`

- It has two modes
  - Internal
    - ➢ each frame of the video is extracted and written as a Dataset
    - ➢ same as any other array-valued series you might find in an NWB file
  - External
    - ➢ data type instead contains a *path* to the video file(s) outside the NWB file

- Best Practices
  - Internal mode
    - ➢ videos of physiological recordings (such as optical imaging)
    - ➢ stimulus presentations
  - External mode
    - ➢ videos of natural behavior

Note: Reasoning for Best Practices

- Reasoning for these Best Practices is a bit nuanced…
- For some history of decision, see…
  - NWB:#1647, DANDI:#769 and DANDI-helpdesk:#30
- DANDI
  - strongly recommends these practices for storage
  - will automatically remap all paths and organize folder storage at time of upload

Step 1: Install SLEAP or DLC

- SLEAP - Talmo Lab - instructions - GitHub repo
- DLC - Mathis Lab - instructions - GitHub repo

- This step can take a while, but just be patient and persistent

- If anything goes differently from instructions, when in doubt ask
  - Google
  - ChatGPT
  - or raise an issue on their repository or helpdesk

Step 2: Launch GUI

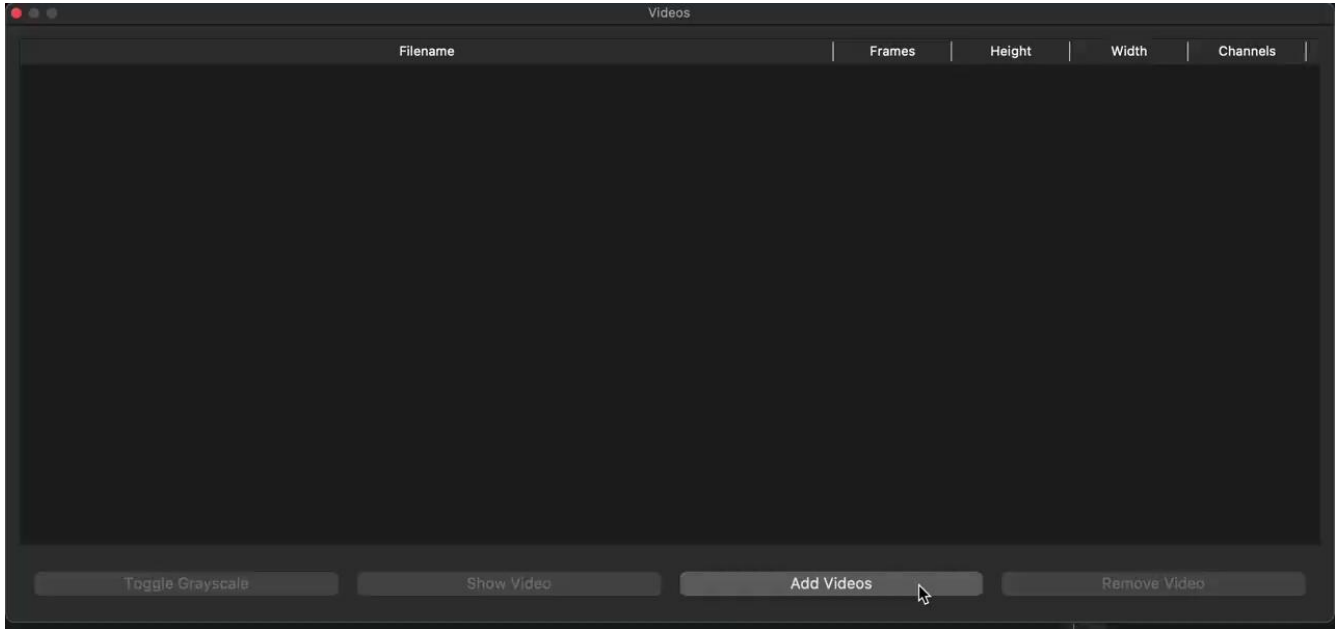
- These can be run headless, but for your first time I recommend using the **G**raphical **U**ser **I**nterface (GUI) to help walk you through it

- Both are able to load in the original videos (`.mpg`, `.mp4`, `.avi`, etc…)

- Due to the previous discussion regarding storing `ImageSeries` of natural behavior, it is recommended to import directly from the raw video files when using either software package

Steps 3-7

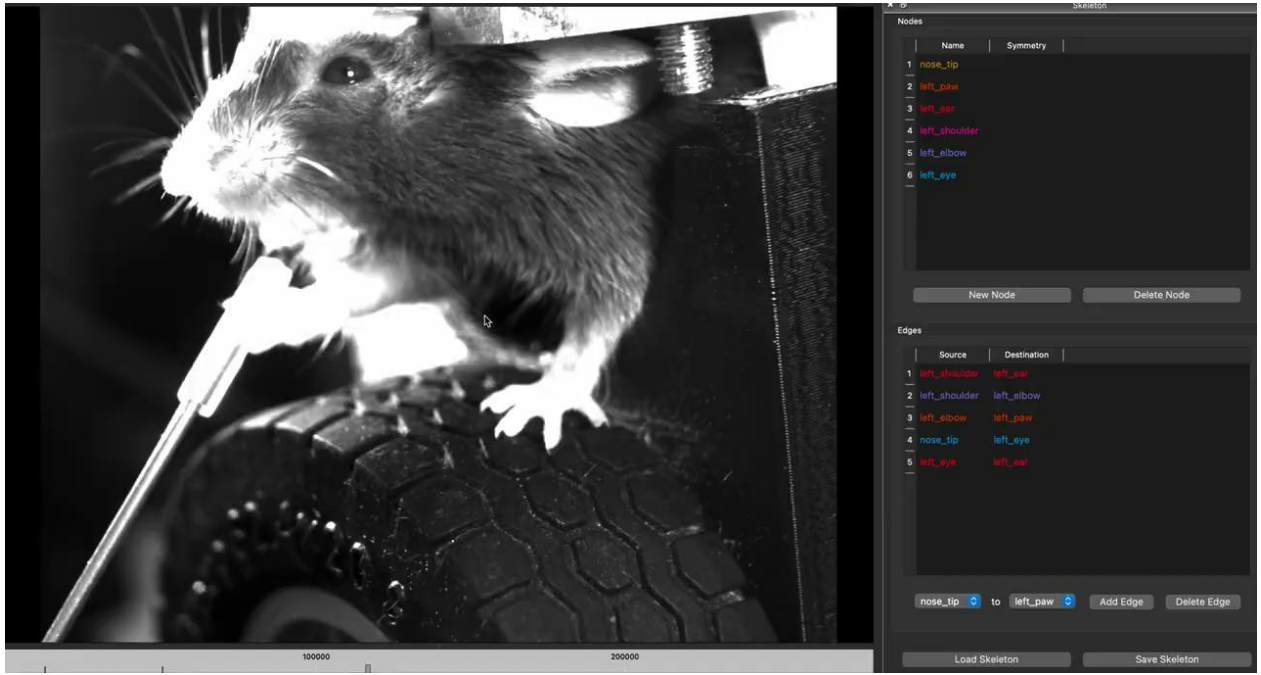
- Step 3: Load videos



bit.ly/nwb-cosyne-2023

Steps 3-7

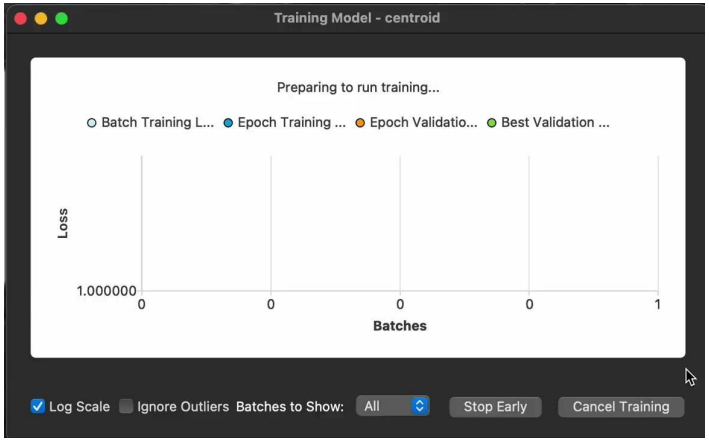
- Step 3: Load videos
- Step 4: Extract frames for training
- Step 5: Label points on the training frames



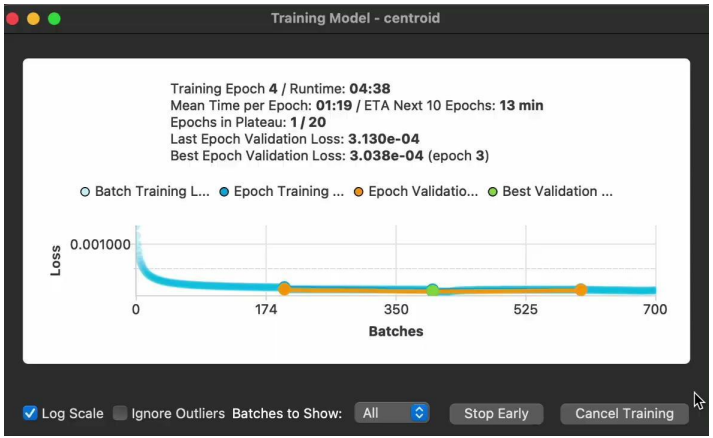
bit.ly/nwb-cosyne-2023

Steps 3-7

- Step 3: Load videos
- Step 4: Extract frames for training
- Step 5: Label points on the training frames
- Step 6: Train the model
- Step 7: Run the prediction



...



bit.ly/nwb-cosyne-2023

Output: Saving results

- The output of the prediction over the entire video can then be saved in either the source `.h5` (DLC) or `.slp` (SLEAP) formats

- These can in turn be converted to NWB format via two utility packages
  - `dlc2nwb` (`.h5` → `.nwb`)
  - `nwb` sub-module of `sleap-io` (`.slp` → `.nwb`; SLEAP also exports NWB via GUI)

- By converting to NWB, downstream software tools and scripts can visualize and analyze the data regardless of which source package was used!

Understanding `PoseEstimation`

- In either case (or for any future integrations with keypoint tracking tools) these are modelled using a common structure in NWB called the `PoseEstimation`

```
nwbfile.processing["behavior"]["PoseEstimationLeftCamera"]
```

```
PoseEstimationLeftCamera abc.PoseEstimation at 0x140450551033632
Fields:
  description: Estimated positions of body parts using DeepLabCut.
  nodes: <StrDataset for HDF5 dataset "nodes": shape (11,), type "|O">
  pose_estimation_series: {
    nose_tip <class 'abc.PoseEstimationSeries'>,
    paw_l <class 'abc.PoseEstimationSeries'>,
    paw_r <class 'abc.PoseEstimationSeries'>,
    pupil_bottom_r <class 'abc.PoseEstimationSeries'>,
    pupil_left_r <class 'abc.PoseEstimationSeries'>,
    pupil_right_r <class 'abc.PoseEstimationSeries'>,
    pupil_top_r <class 'abc.PoseEstimationSeries'>,
    tongue_end_l <class 'abc.PoseEstimationSeries'>,
    tongue_end_r <class 'abc.PoseEstimationSeries'>,
    tube_bottom <class 'abc.PoseEstimationSeries'>,
    tube_top <class 'abc.PoseEstimationSeries'>
  }
  source_software: DeepLabCut
```

Understanding PoseEstimation

- A single PoseEstimation is a container of multiple PoseEstimationSeries, one for each node (usually a body part) being tracked

```
nwbfile.processing["behavior"]["PoseEstimationLeftCamera"]
```

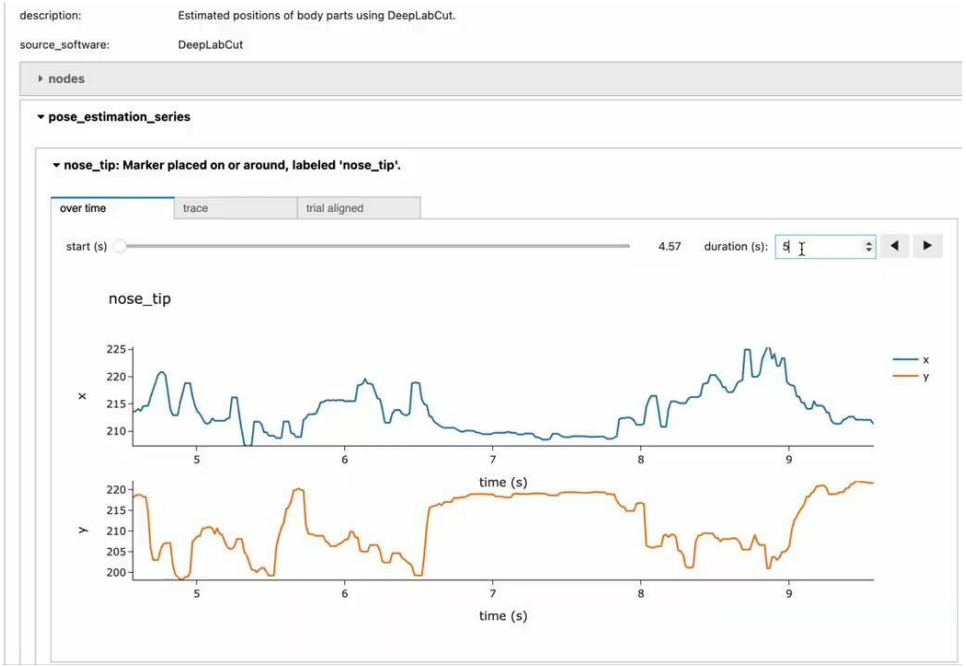
```
PoseEstimationLeftCamera abc.PoseEstimation at 0x140450551033632
Fields:
  description: Estimated positions of body parts using DeepLabCut.
  nodes: <StrDataset for HDF5 dataset "nodes": shape (11,), type "|O">
  pose_estimation_series: {
    nose_tip <class 'abc.PoseEstimationSeries'>,
    paw_l <class 'abc.PoseEstimationSeries'>,
    paw_r <class 'abc.PoseEstimationSeries'>,
    pupil_bottom_r <class 'abc.PoseEstimationSeries'>,
    pupil_left_r <class 'abc.PoseEstimationSeries'>,
    pupil_right_r <class 'abc.PoseEstimationSeries'>,
    pupil_top_r <class 'abc.PoseEstimationSeries'>,
    tongue_end_l <class 'abc.PoseEstimationSeries'>,
    tongue_end_r <class 'abc.PoseEstimationSeries'>,
    tube_bottom <class 'abc.PoseEstimationSeries'>,
    tube_top <class 'abc.PoseEstimationSeries'>
  }
  source_software: DeepLabCut
```

```
nwbfile.processing["behavior"]["PoseEstimationLeftCamera"]["nose_tip"]
```

```
nose_tip abc.PoseEstimationSeries at 0x140450549420416
Fields:
  confidence: <HDF5 dataset "confidence": shape (242446,), type "<f8">
  conversion: 1.0
  data: <HDF5 dataset "data": shape (242446, 2), type "<f8">
  reference_frame: (0,0) corresponds to the upper left corner when using width by height convention.
  timestamps: <HDF5 dataset "timestamps": shape (242446,), type "<f8">
  timestamps_unit: seconds
  unit: px
```

Understanding `PoseEstimation`

- The `data` field of each `PoseEstimationSeries` represents the `x`/`y`/`z` positions being tracked over time



bit.ly/nwb-cosyne-2023