DeepLabCut tutorial

Society of Integrative and Comparative Biology
Phoenix, Arizona

NOTE: some videos have been removed from this pdf version!

Alexander Mathis
Swiss Federal Institute of Technology Lausanne /
École Polytechnique Fédérale de Lausanne (EPFL)

Jan 7th, 2022

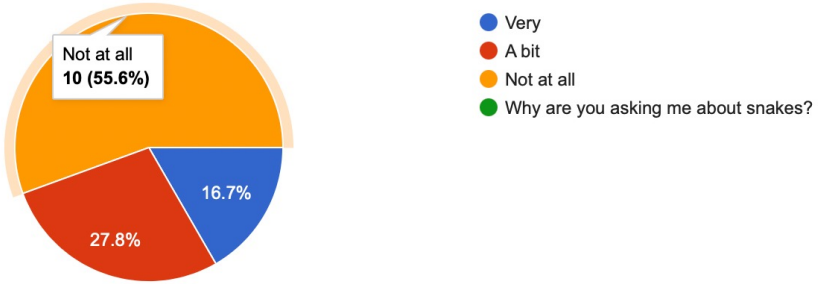
campus biotech

Mathis Group
computational neuroscience & AI

Your feedback

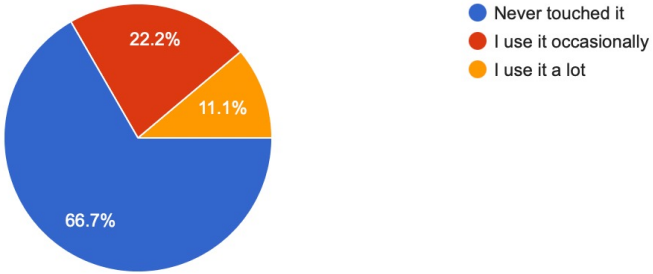
How experienced are you with Python?

18 responses



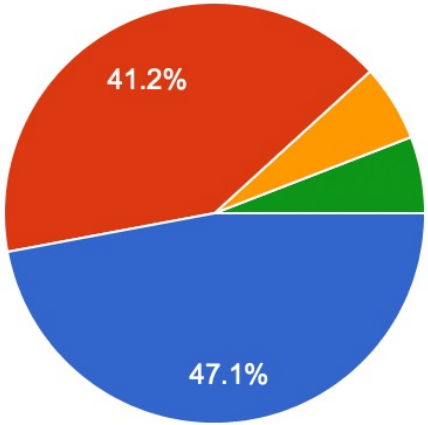
How experienced are you with DeepLabCut?

18 responses



What is your main application for DeepLabCut?

17 responses



- Single animal
- Multi animal (2.2+ -> https://www.biorxiv.org/content/10.1101/2021.04.30.442096v1)
- Real-time feedback (DeepLabCut live -> https://elifesciences.org/articles/61909)
- Both multi and single anim

Your goals …

- I would like to get a basic understanding of how DeepLabCut works
- 3D implementation
- How to run DLC successfully and get animal tracking data from it
- Basic functions/introduction
- How to train it for very high accurate digitization.
- How to prepare files and train dataset
- Going through the entire process of setting up, training, collecting data
- Basics!
- Basic run-through of steps and especially how to fix poor training- do you need to start all over if it doesn't track well?
- General workflow, tips for beginners (if suitable for students). Also, suggestions on hardware
- Things I haven't tried yet, like real time and 3D
- How to use this software to track small, low-contrast animals in low-light settings.
- How to track/quantify specific movements and behaviors of a single animal
- How to use COLAB, options when tracking two different organisms, best approach when trying to train multiple models from recordings that all have one organism in common but a different second interacting organism
- ….

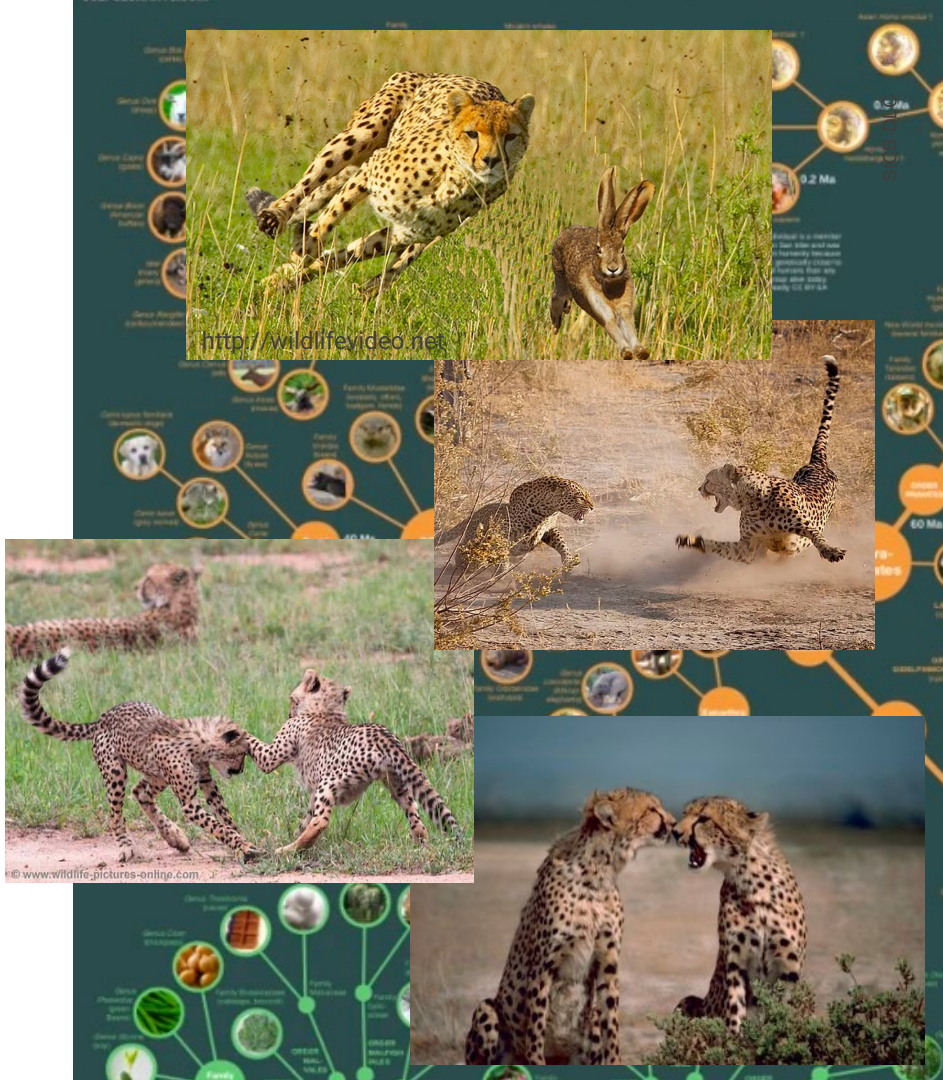
Your goals …

- I would like to get a basic understanding of how DeepLabCut works
- 3D implementation
- How to run DLC successfully and get animal tracking data from it
- Basic functions/introduction
- How to train it for very high accurate digitization
- How to prepare files and train dataset
- Going through the entire process of setting up, training, collecting data
- Basics!
- Basic run-through of steps and especially how to fix poor training- do you need to start all over if it doesn't track well?
- General workflow, tips for beginners (if suitable for students). Also, suggestions on hardware
- Things I haven't tried yet, like real time and 3D
- How to use this software to track small, low-contrast animals in low-light settings
- How to track/quantify specific movements and behaviors of a single animal
- How to use COLAB, options when tracking two different organisms, best approach when tying to train multiple models from recordings that all have one organism in common but a different second interacting organism
- ….

Massive diversity of animals

Behavior is highly diverse

We need to reduce the problem to meaningful quantifications



Measuring behavior: Pose estimation

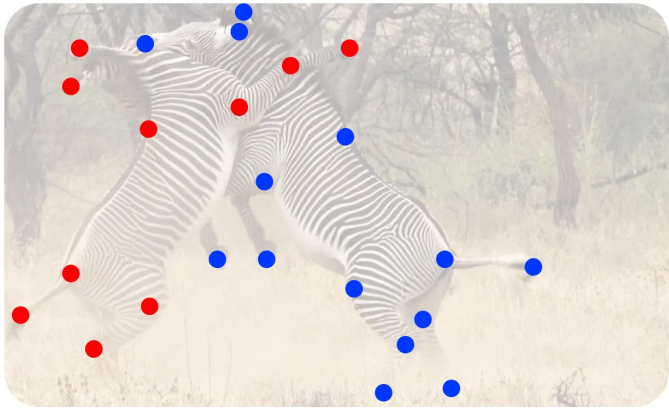
Pixel Representation

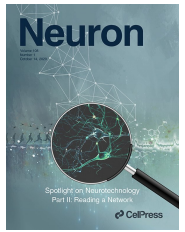
Pose Estimation Algorithm

Subject 1 Keypoints

Subject 2 Keypoints

Keypoint Representation

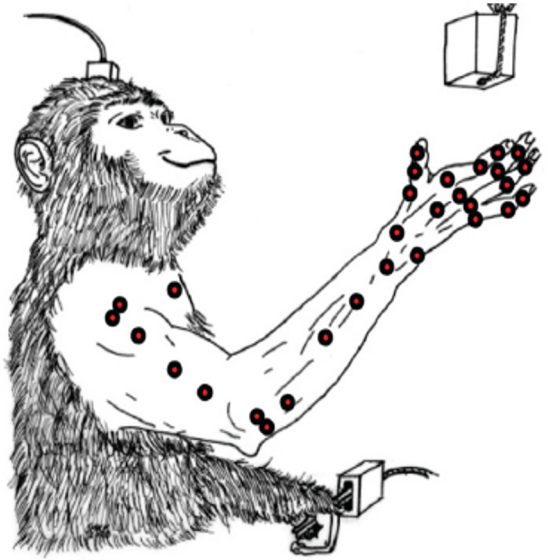




Reviewed in A. Mathis, Schneider, Lauer, M. Mathis, 2020, Neuron

Marker-based pose estimation

Bernstein, 1920



Vargas-Irwin et al. 2010

Reviewed in A. Mathis, Schneider, Lauer, M. Mathis, 2020, Neuron

Deep learning for human pose estimation

MoDeep – *starting 2014*
DeepPose
Conv. PoseMachines
…
DeeperCut
OpenPose
…
> 10,000 papers on human pose estimation with Deep Learning

Appealing properties:
- Work “in the wild”
- Robust (backgrounds, individuals, etc)
- Relatively fast
- No body model
- No (manual) parameter tuning

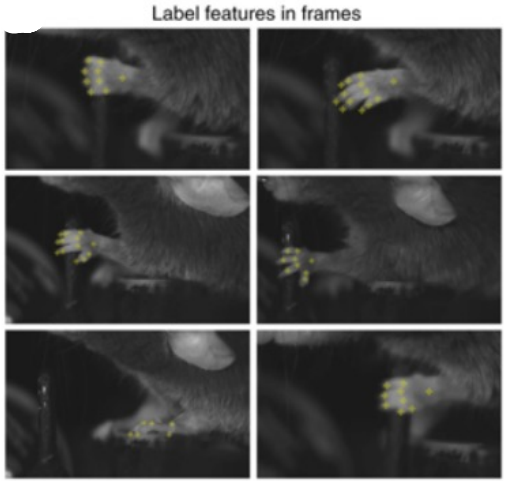
deep neural networks

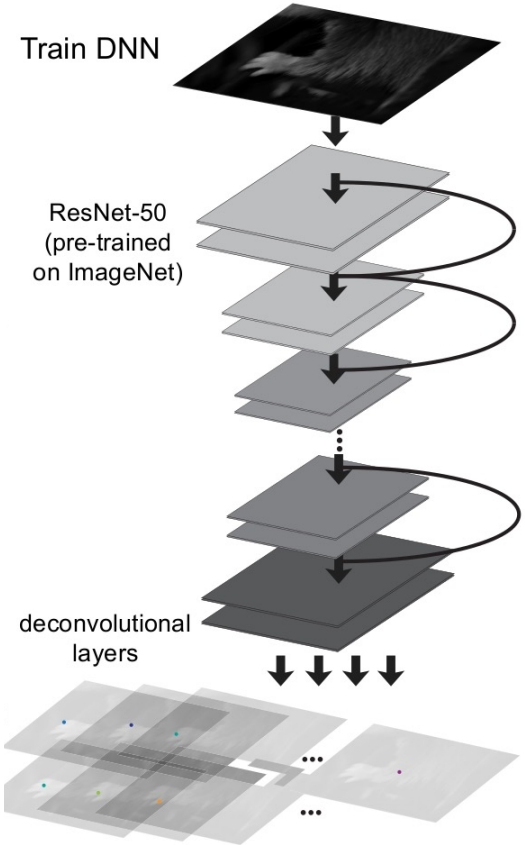
image → Predictor → pose

train

A lot of labeled images (>10^6 joints!)

DeepLabCut: a toolbox for markerless pose estimation via **transfer learning**

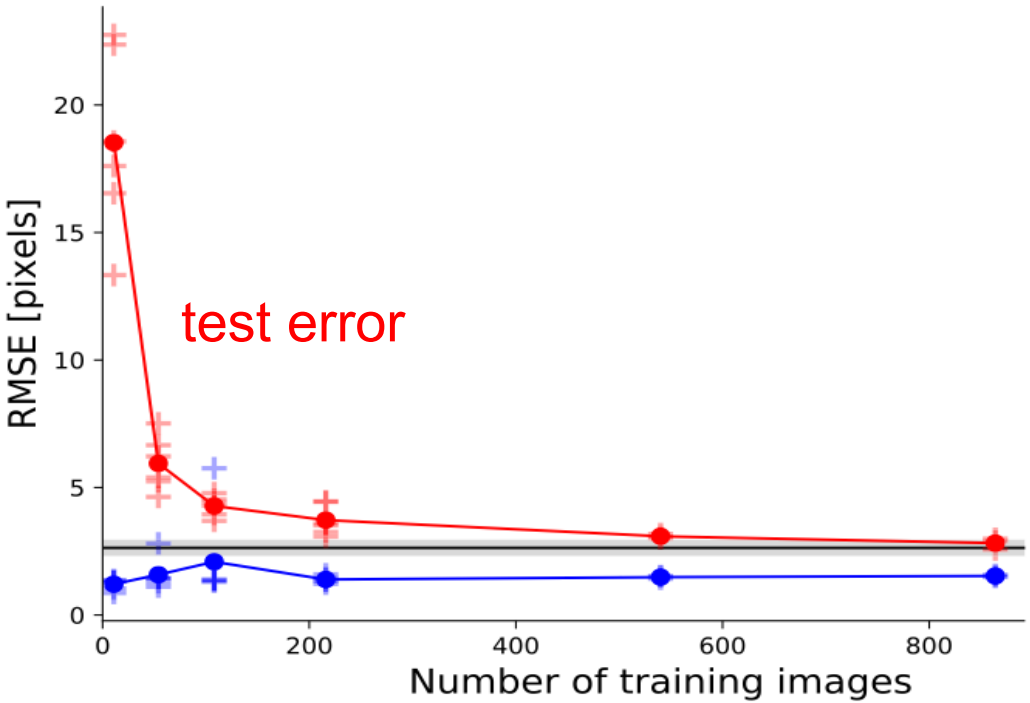


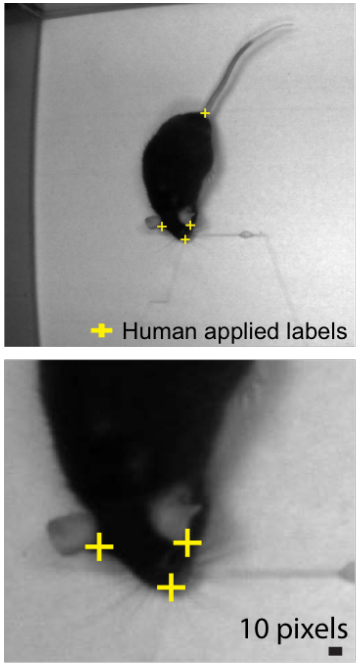


141 labeled frames

A. Mathis …. M.W. Mathis* and Bethge*
Nature Neurosci, 2018

How little training data is required?

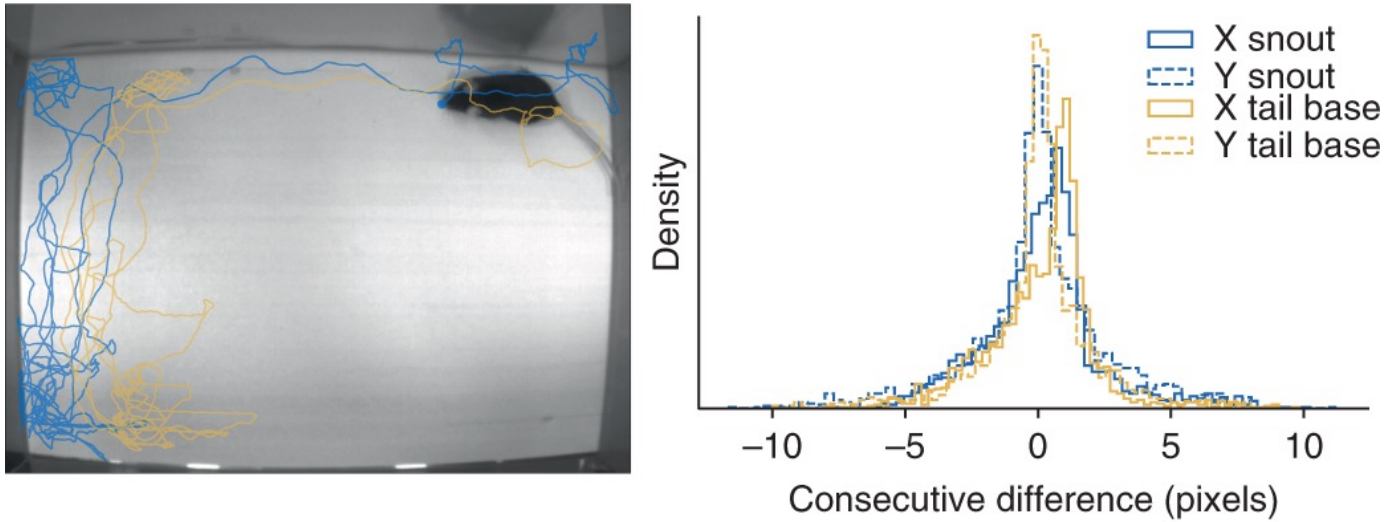




A. Mathis …. M.W. Mathis* and Bethge*
Nature Neurosci, 2018

Generalization to novel mice

Example of generalization: novel mouse tracking



A. Mathis …. M.W. Mathis* and Bethge*
Nature Neurosci, 2018

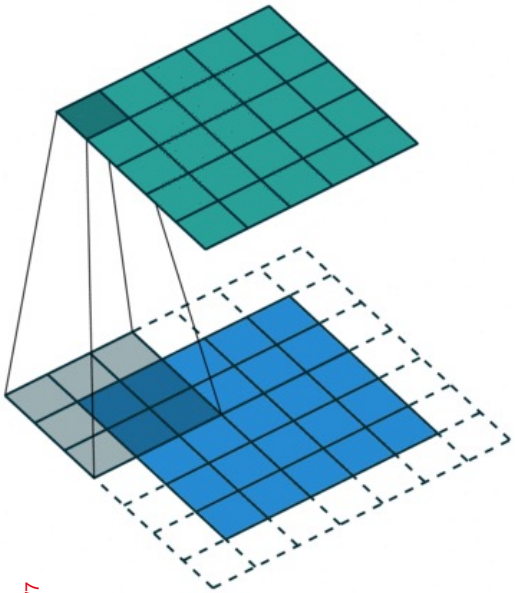
Let's dig into the nitty gritty

Machine learning system:

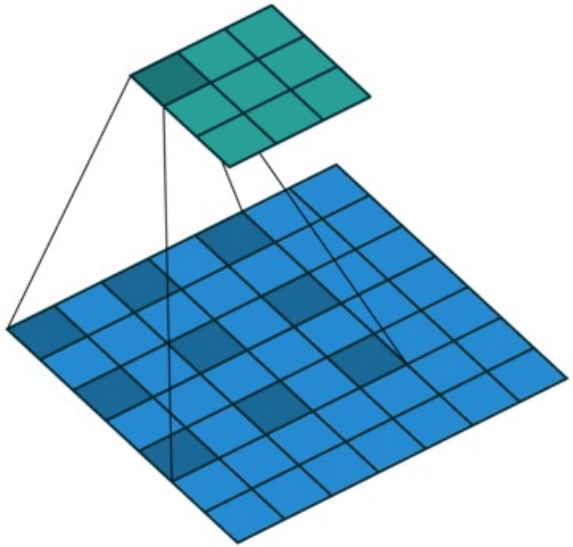
- Architecture
- Objective (task)
- Learning algorithm
- Data

Convolutions

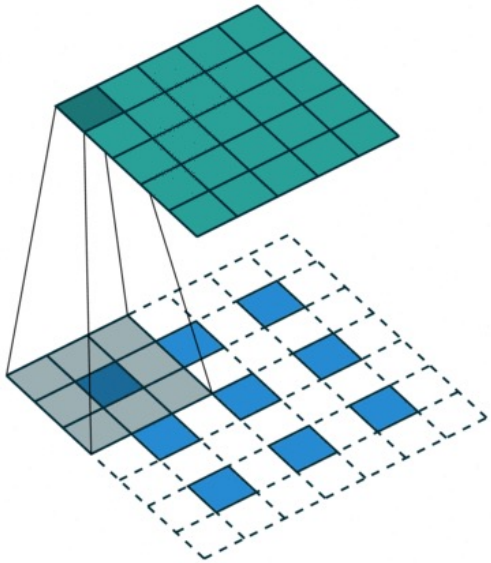
Basic convolution (“same”)



Strided convolution



Strided **de**onvolution

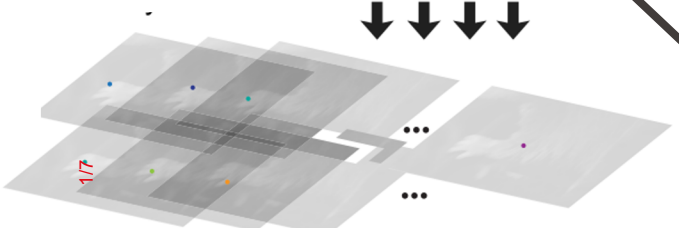


https://github.com/vdumoulin/conv_arithmetic

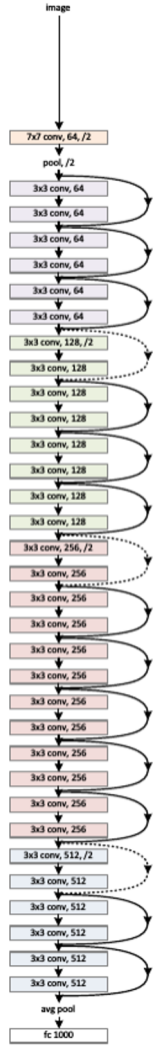
DeepLabCut
Based on
DeeperCut
Insafutdinov et al
CVPR 2017

ResNet
He et al.,
CVPR 2016

Deconvolution Layers (upsample)

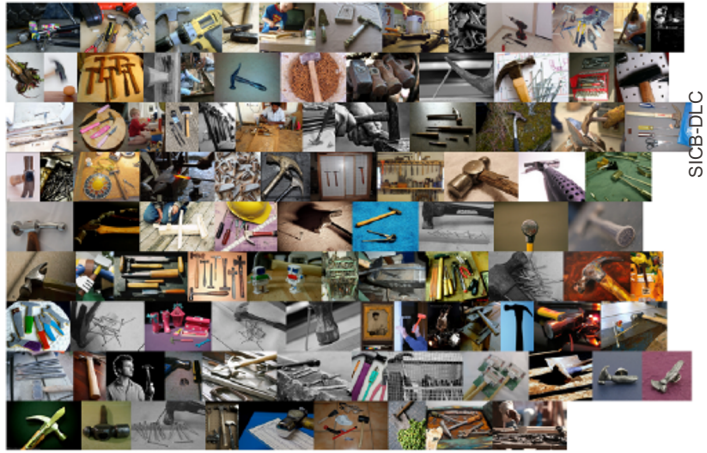


Transfer Learning



ResNet

convolutions



SICB-DLC

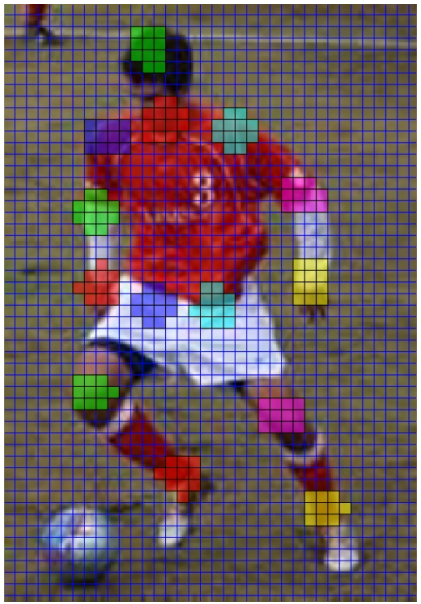
Imagenet: 1.2 million images for 1000 classes

Deng et al. CVPR 2009

Train to predict hammer (for those inputs)
As well as other object classes

Keypoint Detector (Segmentation mask)

- Given: a ground truth coordinate (x_j, y_j) for the body joint j
- Construct a ground truth *scoremap* $s_j(x, y)$

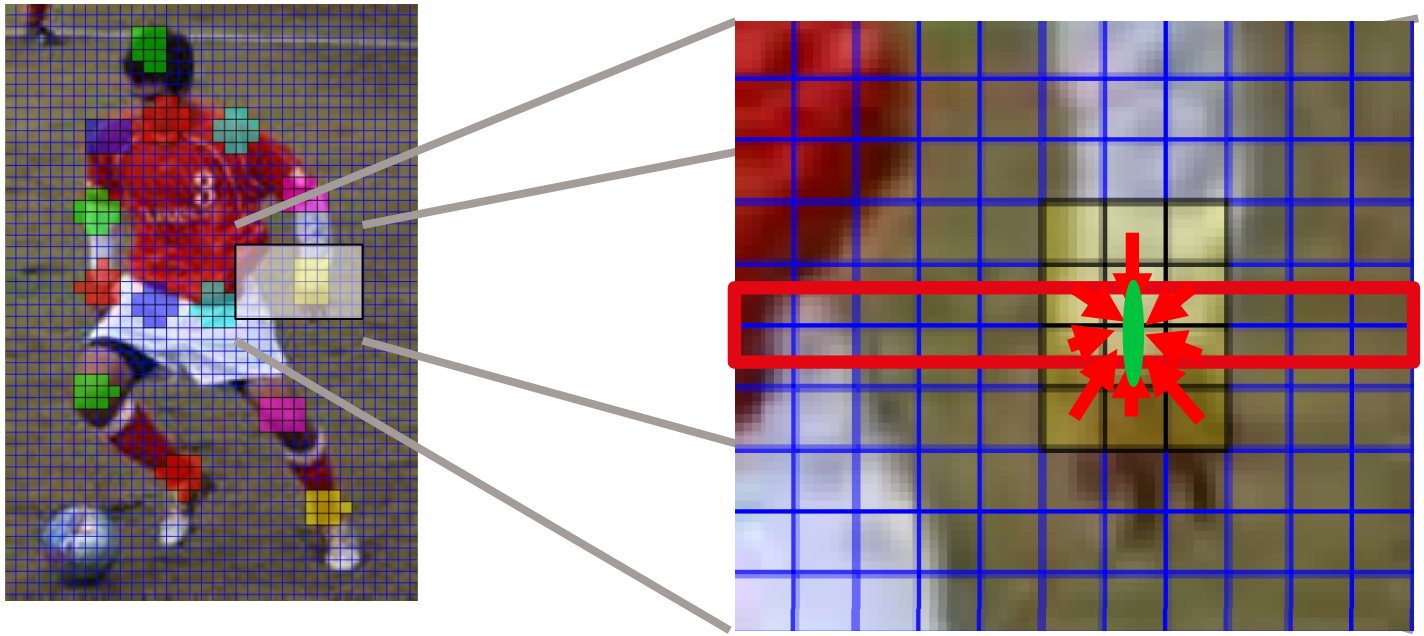


$$s_j(x,y) = \begin{cases} 1, & \text{if } \left\|(x,y) - (x_j, y_j)\right\|_2 \leq R \\ 0, & \text{otherwise} \end{cases}$$

Slide courtesy of Eldar Insafutdinov

Precise Keypoint Localization (Location Refinement)

- Inspired by bounding box regression in Overfeat [Sermanet et al, 2014] and Fast R-CNN [Girshick, 2015]



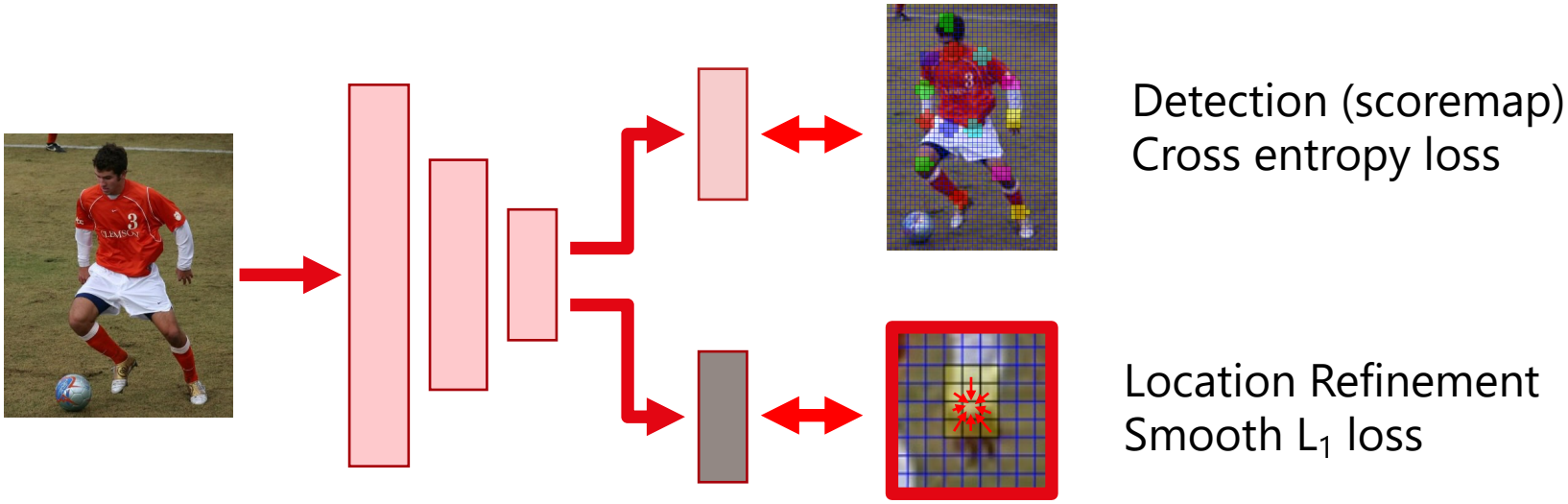
Regress the offsets

$$\Delta x = x_{part} - x$$
$$\Delta y = y_{part} - y$$

Slide courtesy of Eldar Insafutdinov

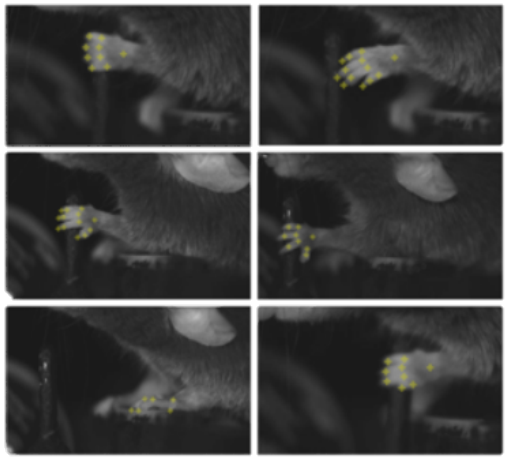
Multi-Task ConvNet

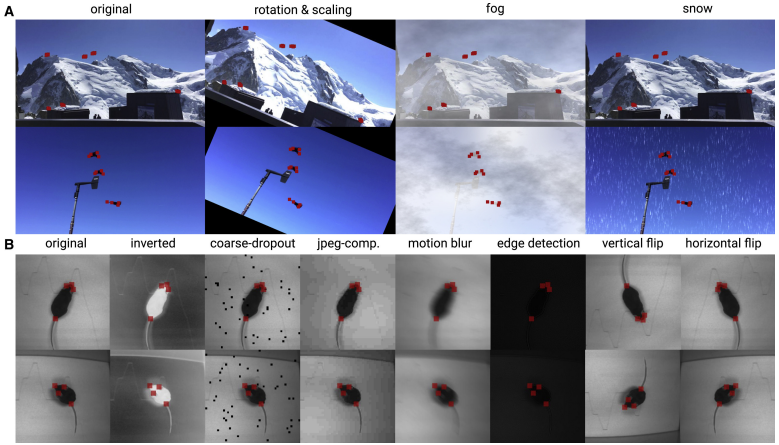
- Minimize the combined loss with SGD (with Momentum)
- Perform augmentation during training

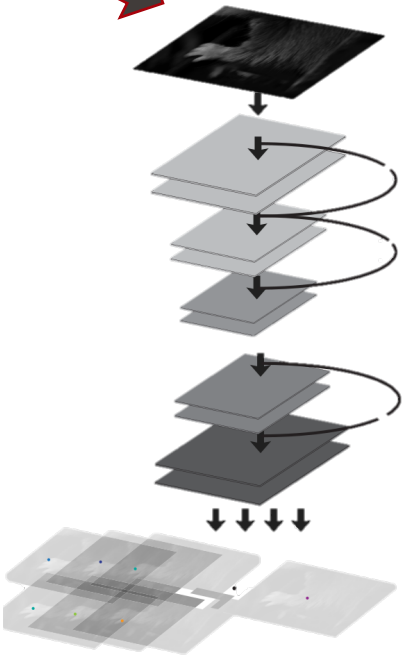


Slide courtesy of Eldar Insafutdinov

How does markerless pose estimation work?





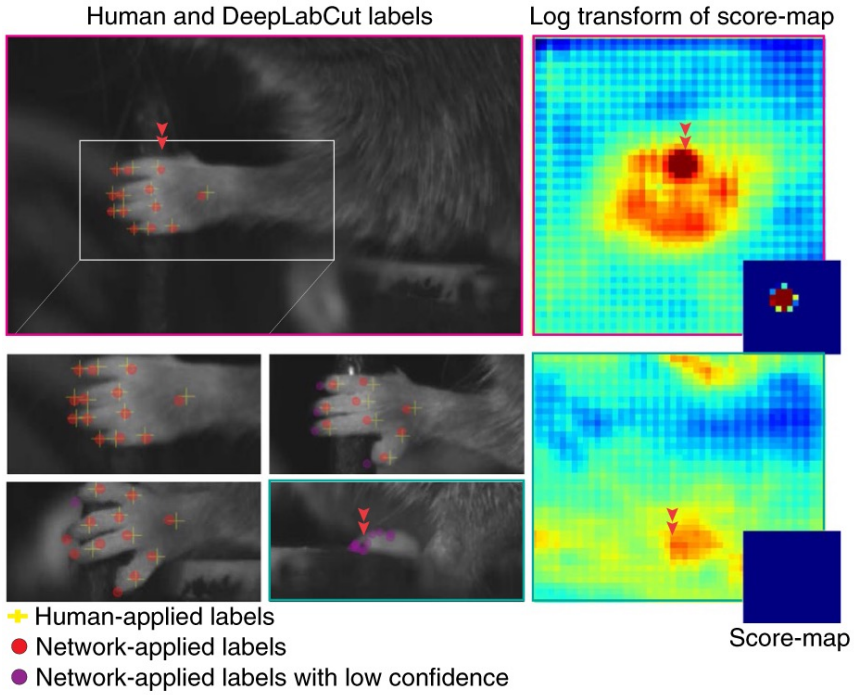


Key Features:

- Data augmentation
- Model architecture
- Optimization

Mathis, Schneider, Lauer & Mathis, 2020 Neuron

Confidence readout (score-maps)



A. Mathis …. M.W. Mathis* and Bethge*
Nature Neurosci, 2018

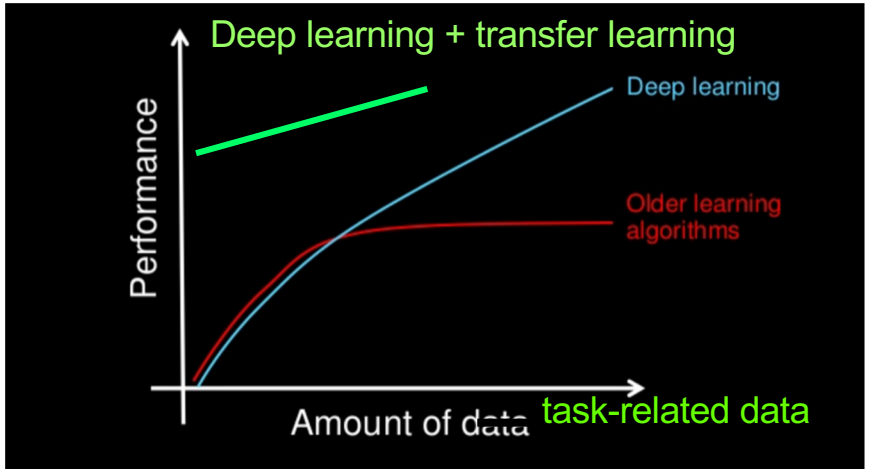
Transfer learning enables pose estimation with less data

Pre-trained! (i.e. on ImageNet)

image → Predictor → pose

train

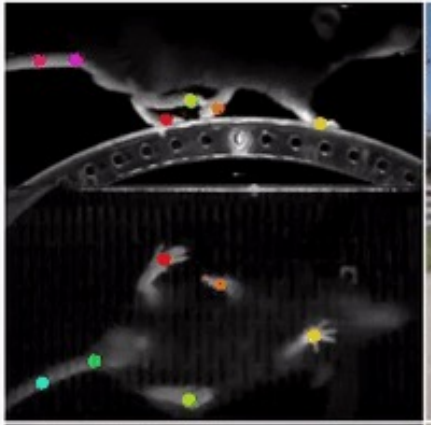
Only a few examples (10-200) for most applications



Andrew Ng

Locomotion
3D tracking of limbs

R. Warren (Sawtell lab)
Columbia University

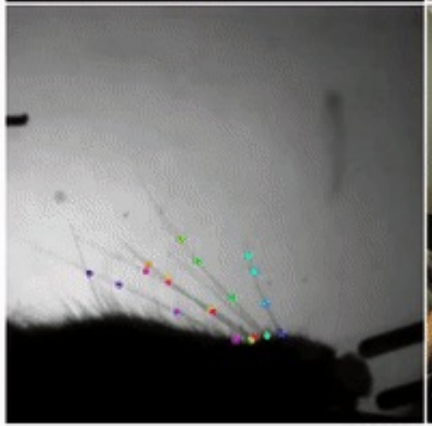


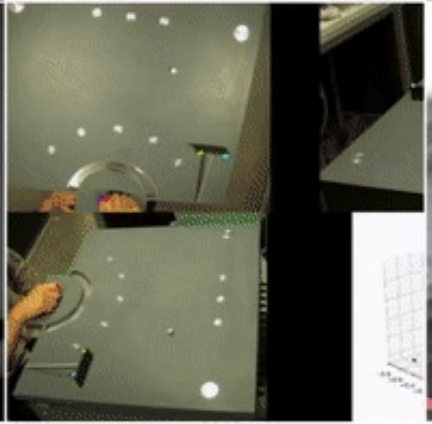
Pellet reaching task

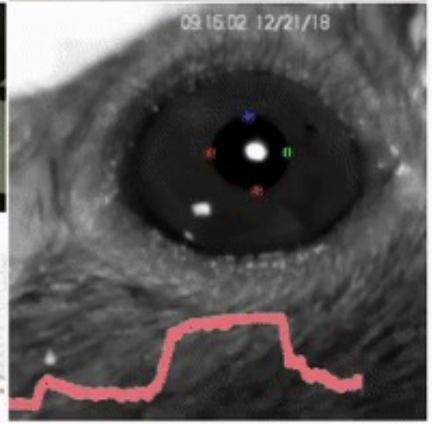
D. Leventhal Lab
University of Michigan

Whisker tracking

A Erskine (Hires lab)
USC



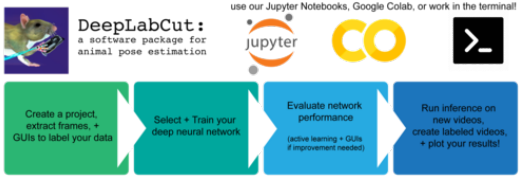




Pupil tracking

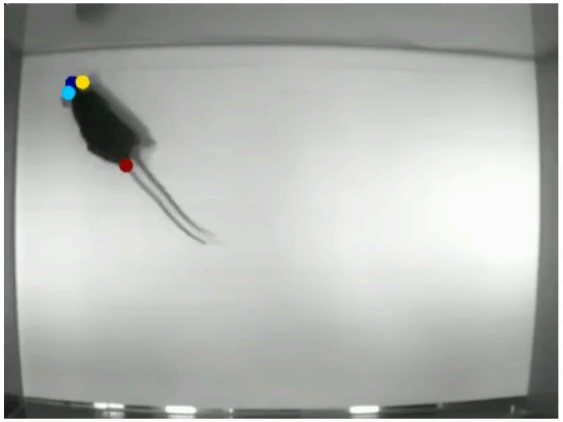
T. Vaissiere
Scripps

J Bonaiuto (Ferrari lab) CNRS

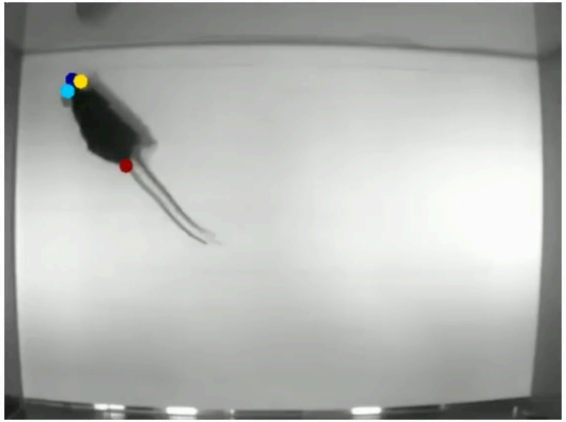


DeepLabCut – Deep Learning toolbox for pose-estimation
Open source & free, > 1,000 labs, >200,000 installations

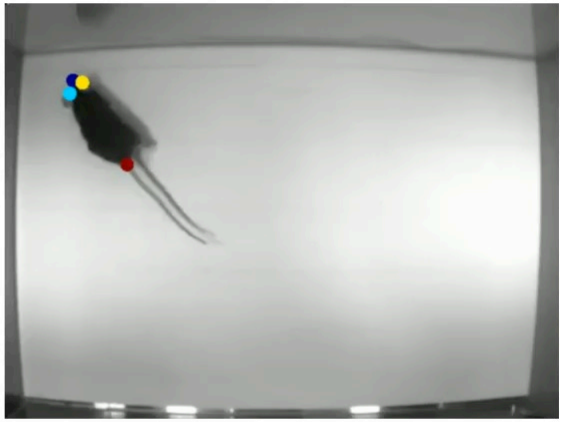
Data Augmentation: how to get the most out of your data!



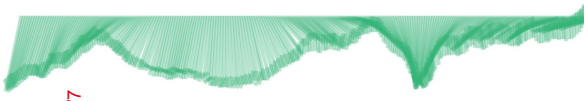
tensorpack

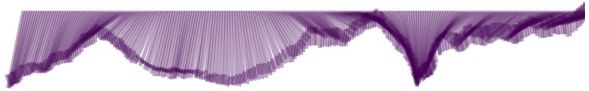


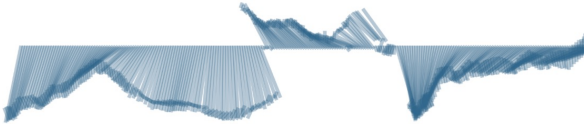
imgaug



scale-crop

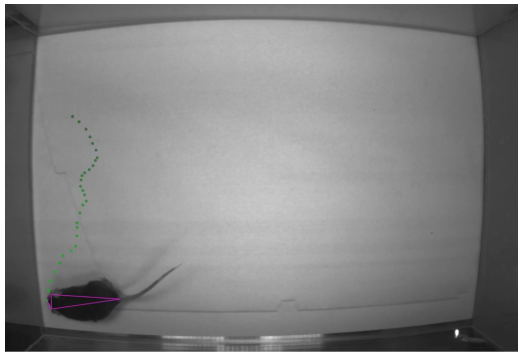


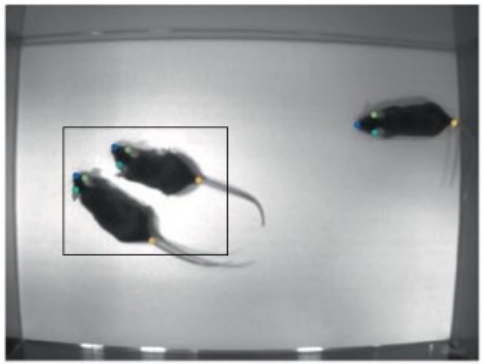


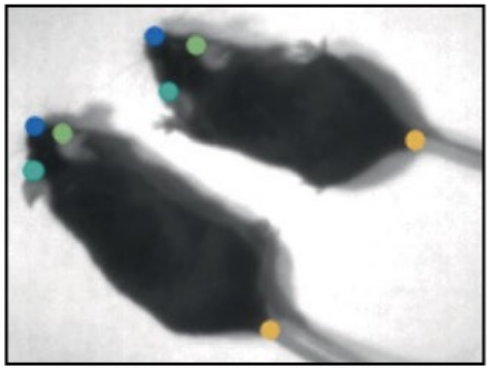


Mathis, Schneider, Lauer & Mathis 2020
Neuron

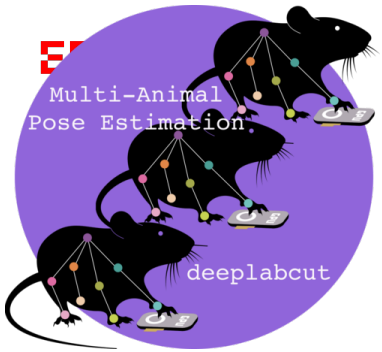
Pose estimation for multiple animals



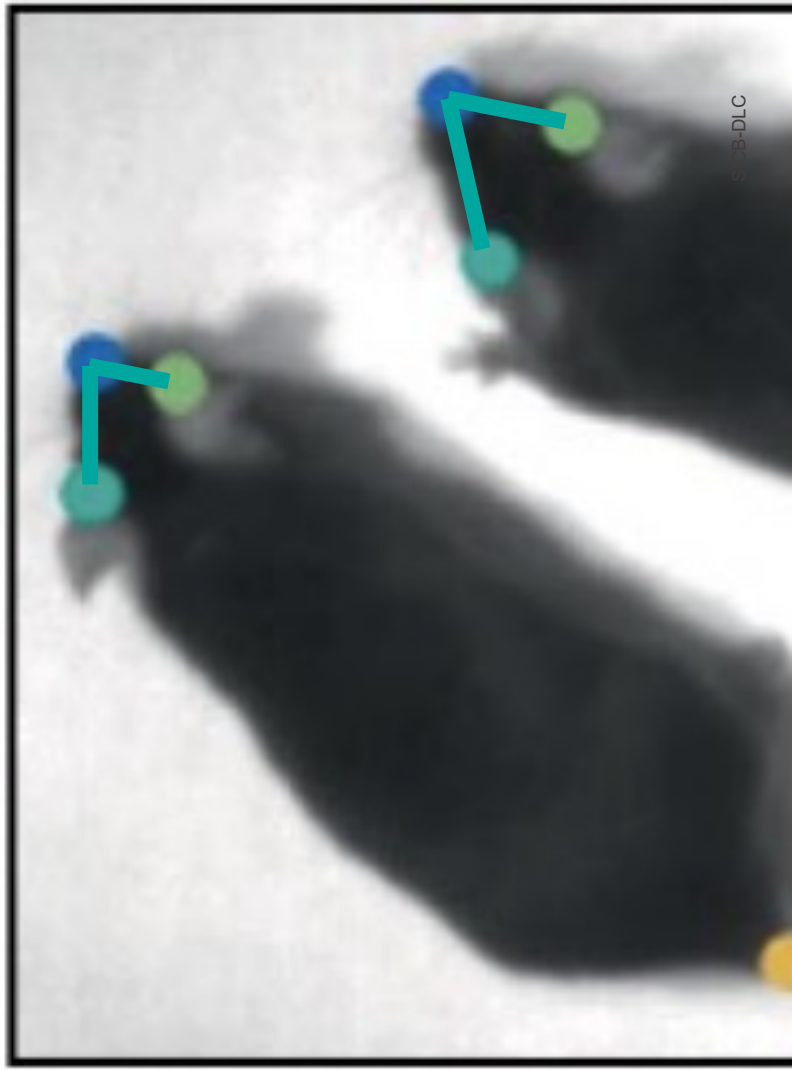




Mathis et al. Nature Neurosci, 2018



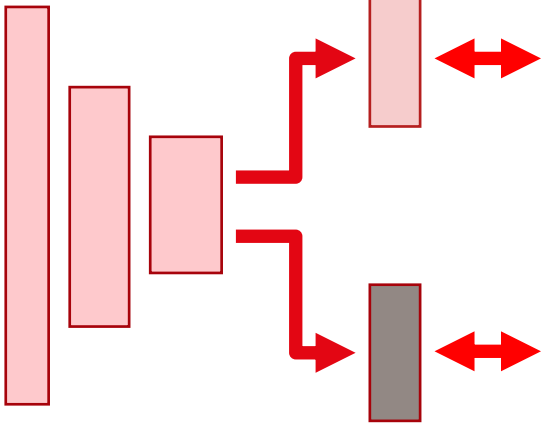
The idea: learn to predict ”limbs”



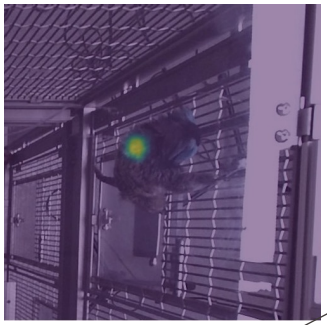
Inspired by “Part Affinity Fields” in Open Pose
Cao et al. CVPR 2017

Multi-Task Deep Convolutional Network

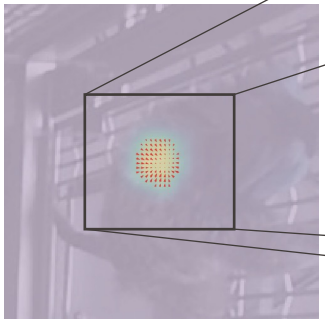
- Minimize the combined loss; segmentation + vector field
- Transfer learning (ConvNet pre-trained on object recognition)
- Perform augmentation during training

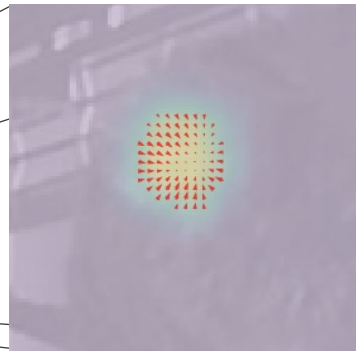


deconvolution layers (3 per bodypart)



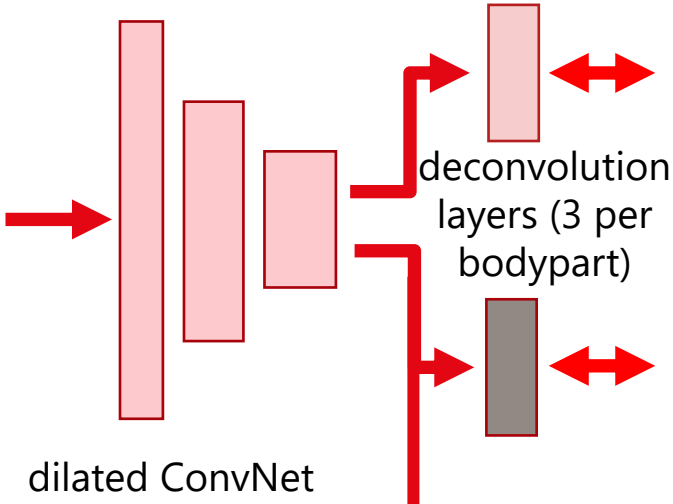
Detection (scoremap)
Cross entropy loss





Location Refinement
Smooth L_1 loss

Multi-Task DCN (DLC 2.2)



dilated ConvNet

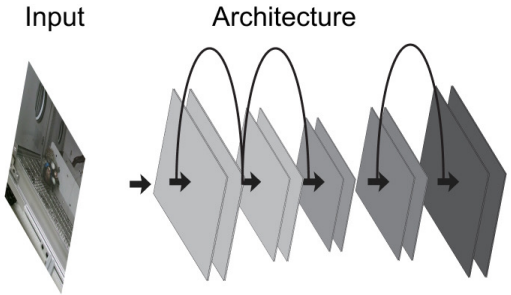
deconvolution layers (3 per bodypart)

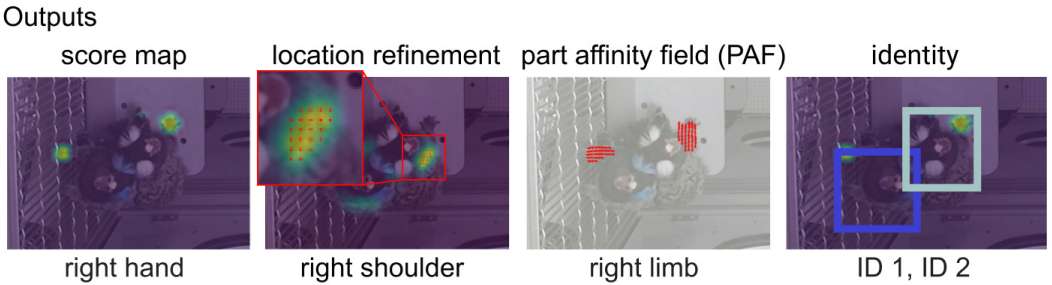
Location Refinement
Smooth L_1 loss

deconvolution layers (2 per limb)

- Lauer et al. BioRxiv 2021

Multi-animal pose estimation & tracking with DeepLabCut





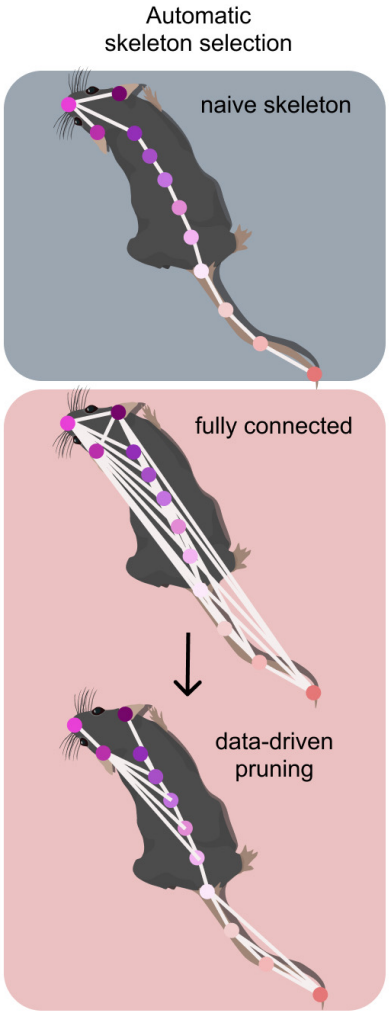
Lauer et al. BioRxiv 2021

Bodyplan agnostic graph optimization

Learn to predict:

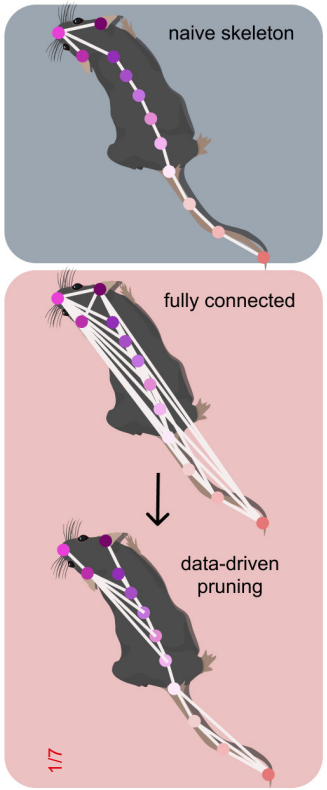
- Bodyparts
- Limbs (= costs for associations)

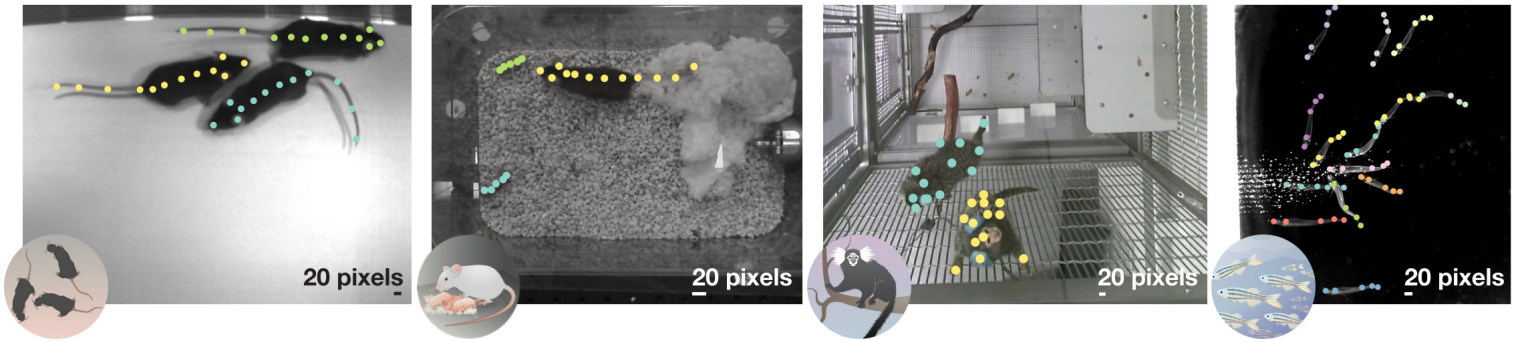
Which limbs should be learned?

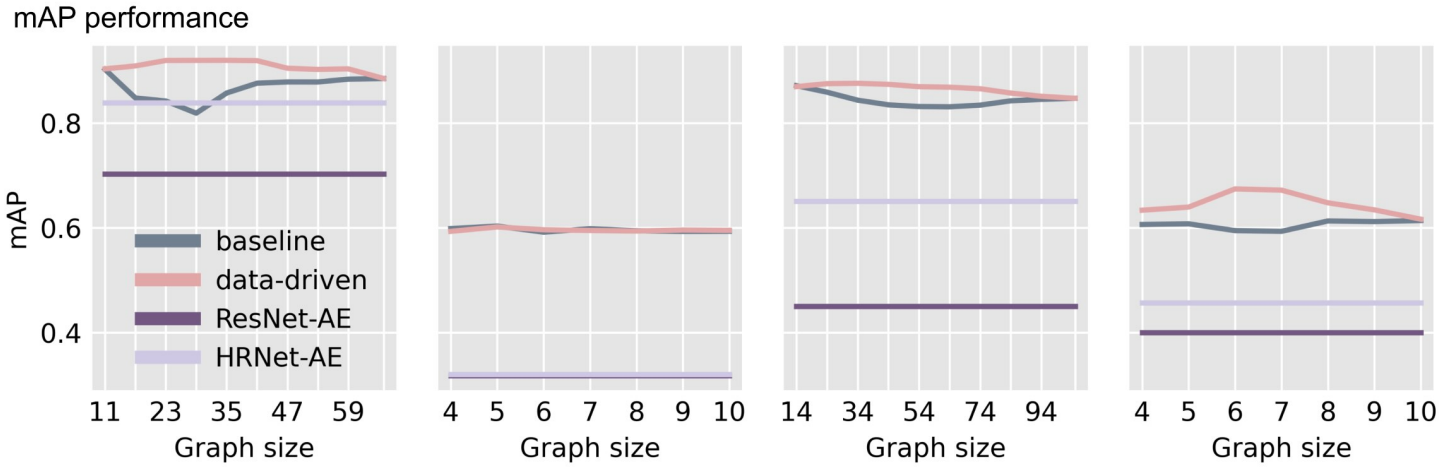


Lauer et al. BioRxiv 2021

EPFL Novel data driven method outperforms baseline methods (giving SOTA perf)



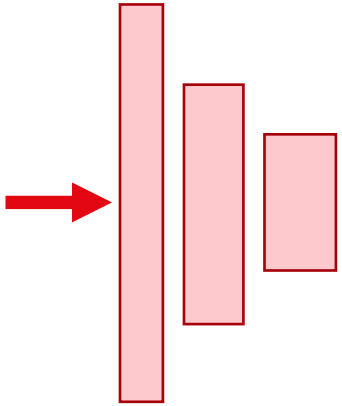




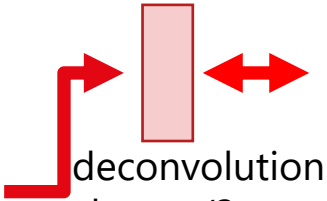
Lauer et al. BioRxiv 2021

ICB-DLC

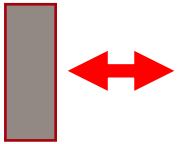
Multi-Task DCN (DLC 2.2)



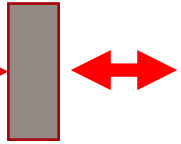
dilated ConvNet



deconvolution layers (3 per bodypart)

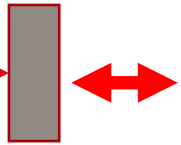


Location Refinement
Smooth L_1 loss



Partaffinity field
Smooth L_1 loss

deconvolution layers (2 per limb)



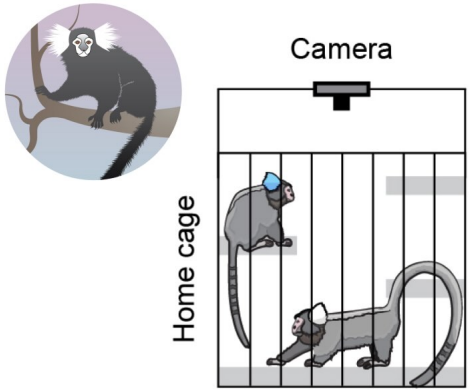
Detection (scoremap)
Cross entropy loss

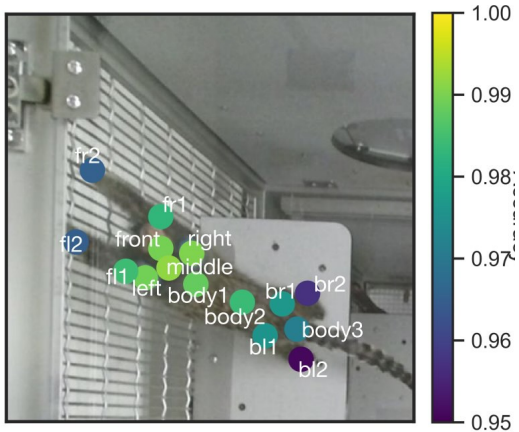
deconvolution layers (1 per individual)

- Lauer et al. BioRxiv 2021

Identification of individuals

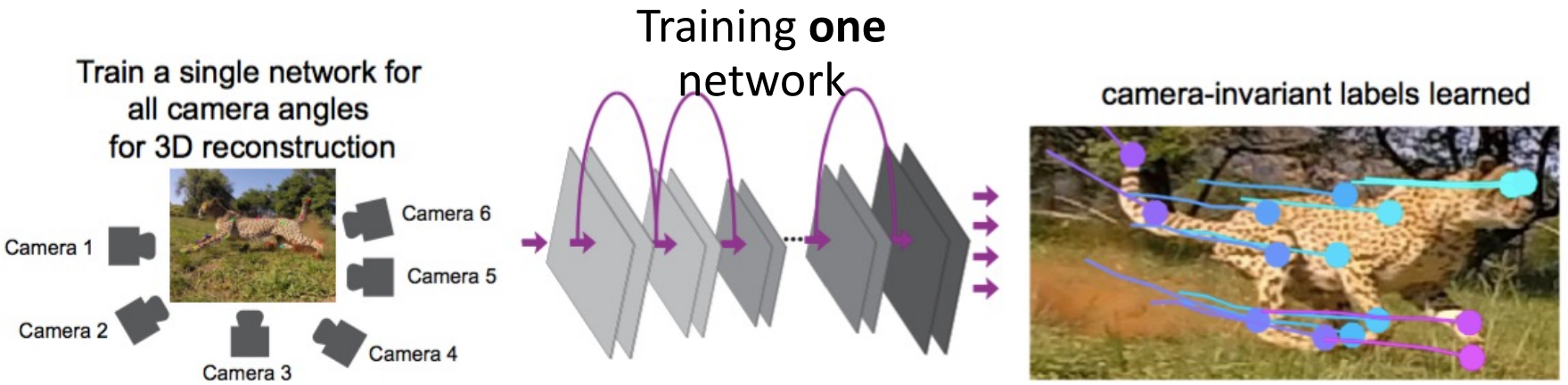
Individual identification performanc





3D pose estimation?

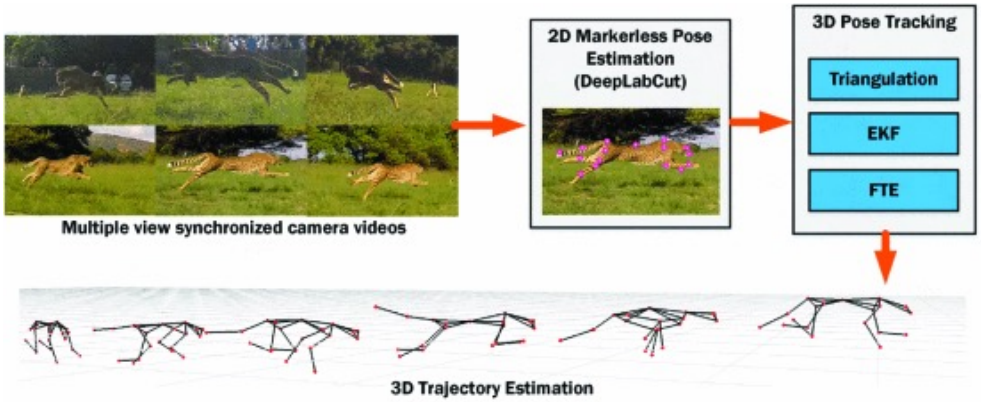
3D-pose estimation of Cheetahs in the “wild”



- Nath*, Mathis* et al. Nature Protocols 2019 and BioRxiv 2018

Data: Amir Patel (U of Cape Town)

AcinoSet: A 3D Pose Estimation Dataset and Baseline Models for Cheetahs in the Wild



Pose Type	Metric	TRI	EKF	FTE
Run	RMSE	28.24	3.40	2.76
	SEM	0.26	0.03	0.03
	NRMSE	0.17	0.02	0.02
Dive	RMSE	76.35	39.40	38.44
	SEM	0.70	0.36	0.35
	NRMSE	0.56	0.29	0.28

EKF: extended Kalman filter
FTE: full trajectory estimation

Joska et al., ICRA 2021

Can DeepLabCut be applied in real-time?

Providing real-time feedback in experiments

SICB-DLC

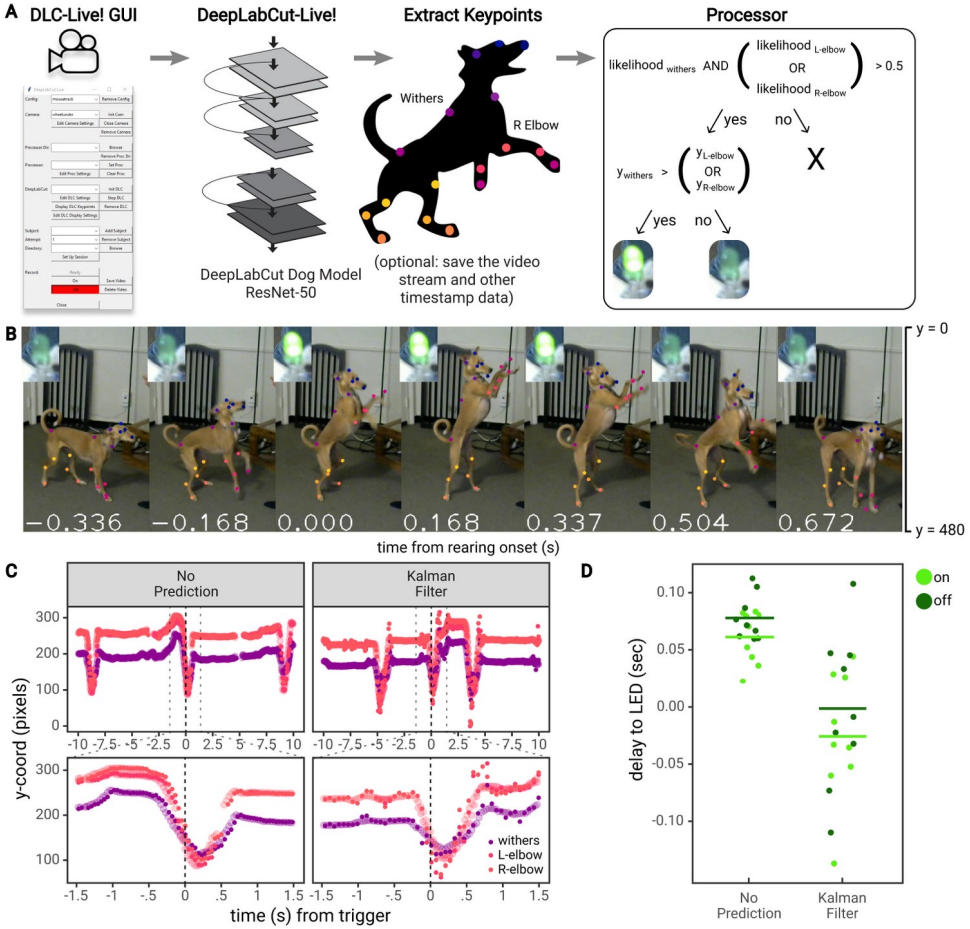
- https://github.com/DeepLabCut/DeepLabCut-live

#Initialize model:
from dlclive import DLCLive
my_live_ob ject = DLCLive(”/exportedmodel/directory”)

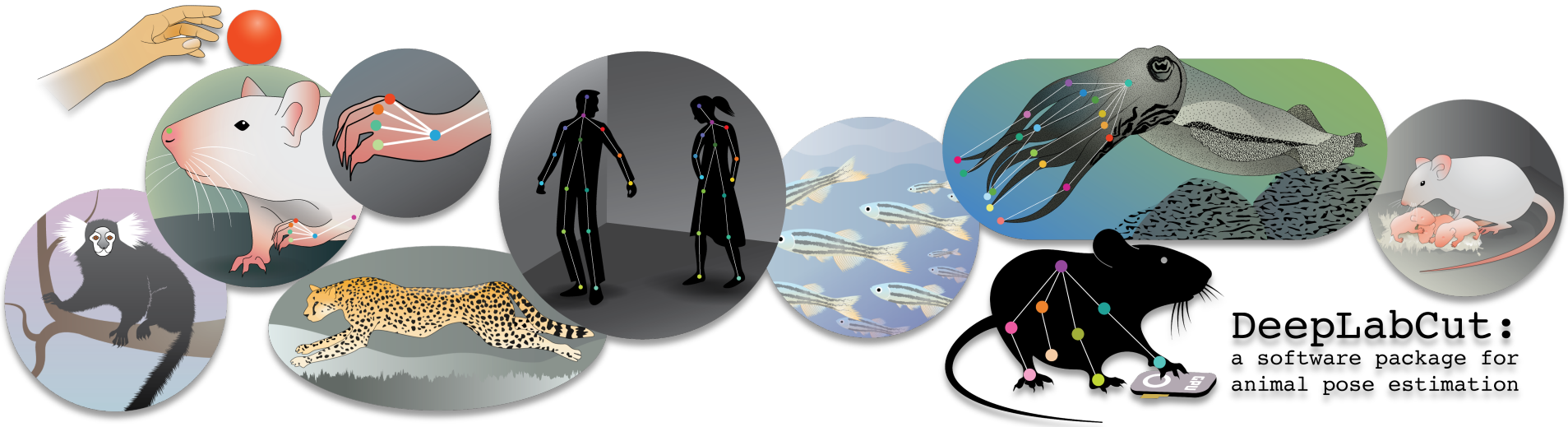
# run inference:
my_live_object.init_inference(my_image)
pose = my_live_object.get_pose(my_image)

Kane et al. eLife 2020

Example experiment



Kane et al. eLife 2020



modelzoo.deeplabcut.org

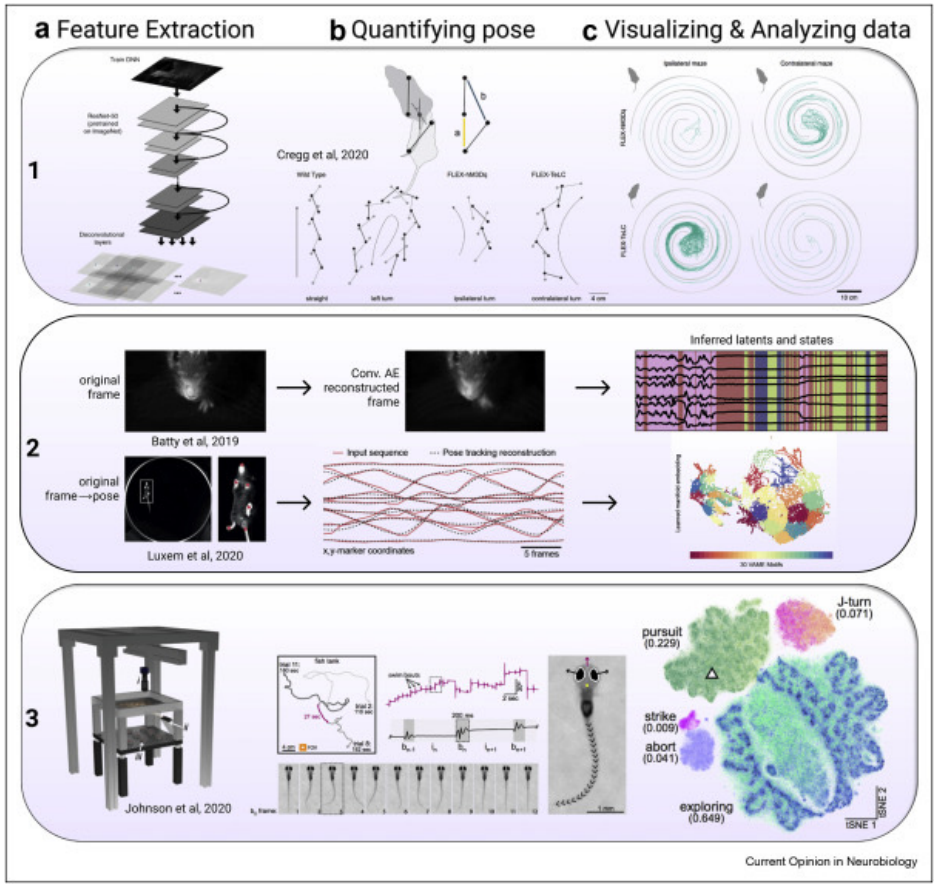
Example model in the DLC model zoo

(added 3 days after the pre-print!)

Labuguen et al. Biorxiv, August 1st, 2020

modelzoo.deeplabcut.org

What to do with DLC output?



Hausmann*, Marin-Vargas*, Mathis, Mathis arxiv, Curr Op. in Neuro

How to use DeepLabCut?

DeepLabCut workflow

Train DNN

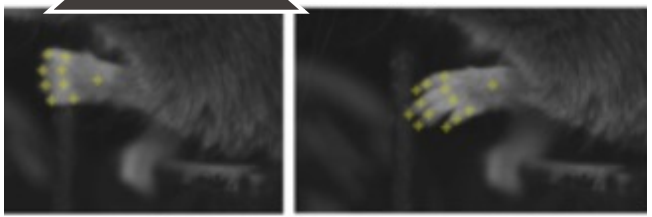
Create a project, extract frames, + GUIs to label your data

Select + Train your deep neural network

Evaluate network performance

(active learning + GUIs if improvement needed)

Run inference on new videos, create labeled videos, + plot your results!



refine?

Steps 7 and 8

GUI: labeling

Unique label 1
Unique label 2
Unique label 3

Step 9: check labels!

Frames extracted from video
(any species can be used)

Steps 1–5

Create project

Step 6

Extract frames

Label frames (GUI)

Step 10

Create training datasets

Step 11

Train network

Step 12

Evaluate network

Good results

No

Yes

Need more training data?

No

Yes

Analyze video

Stop

dlc-models

Labeled-data

Training-datasets

Videos

Config.yaml

Evaluation-results

Directory created

File created

Process

Merge datasets

Refine labels (GUI)

Extract outlier frames

Analyze video

Steps 16–18

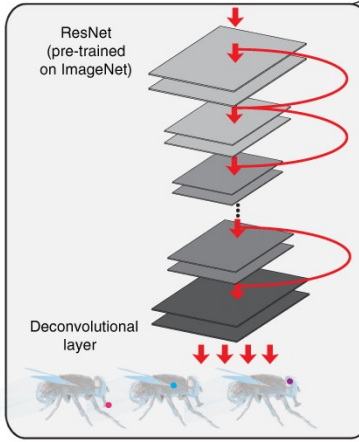
GUI: refinement

Unique label 1
Unique label 2
Unique label 3

Train network

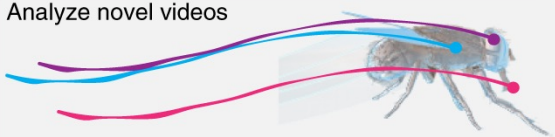
ResNet (pre-trained on ImageNet)

Deconvolutional layer



Steps 13–15

Analyze novel videos



SICB-DLC

nature protocols

PROTOCOL

https://doi.org/10.1038/s41596-019-0176-0

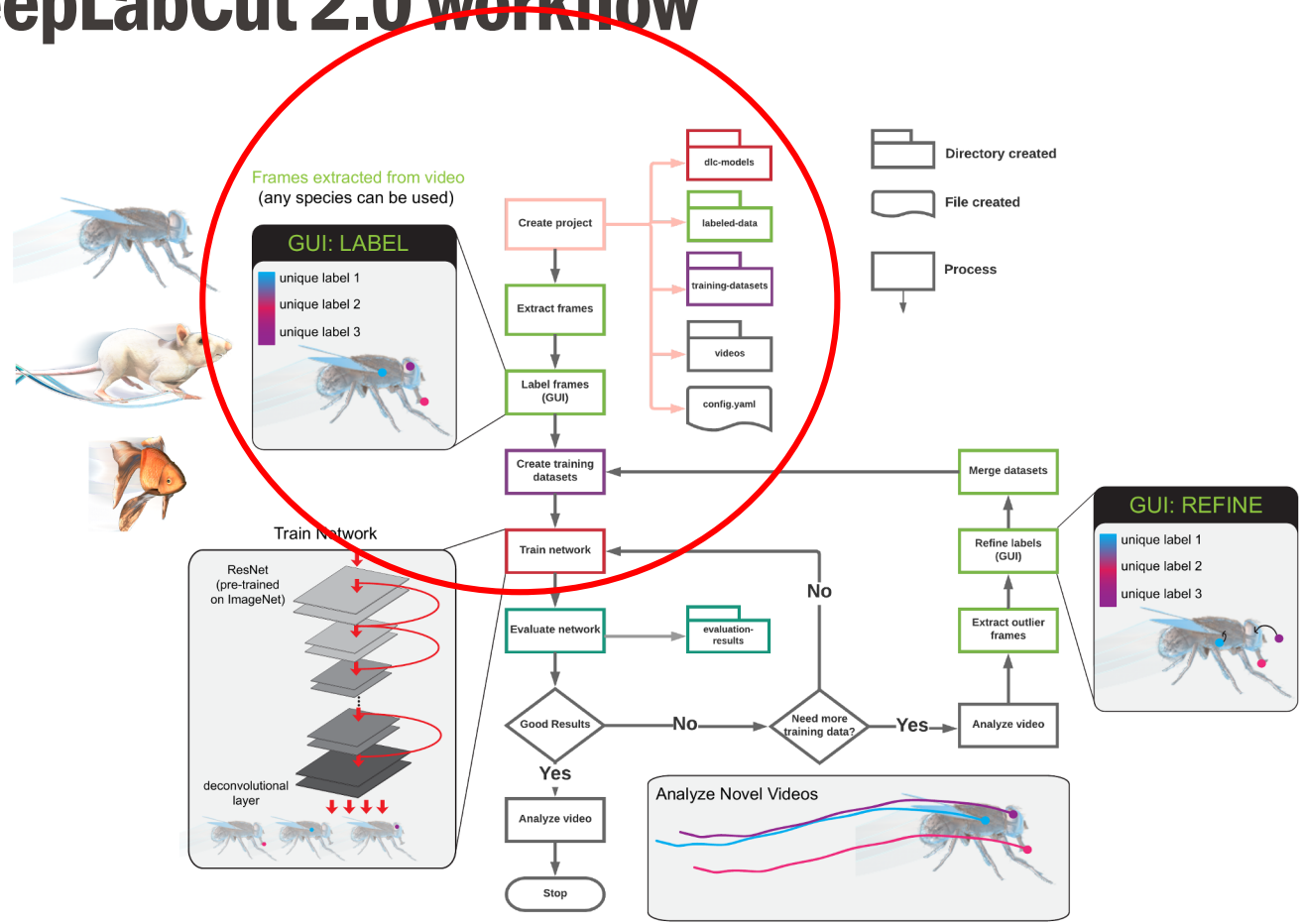
Using DeepLabCut for 3D markerless pose estimation across species and behaviors

Key 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',...])`

Nath*, Mathis* et al. Nature Protocols

DeepLabCut 2.0 workflow



Nath*, Mathis* Nature Methods 2019

2D Project Folders

▶ training data,
trained networks,
`config.yaml` file

This is the master folder created when you create a project (Step 1)

New (2D) videos for analysis

▶ batch process videos
▶ analyzed data

you can place new videos here, and then run:

```
deeplabcut.analyze_videos(config_path,
    folderpath, videotype='.mp4')
```

Your 2D or 3D project “entry point” is through the `config.yaml` file
When you want to work on your project:

```
activate DLCenvName
ipython
import deeplabcut
config_path ='/home/yourprojectfolder/config.yaml'
```

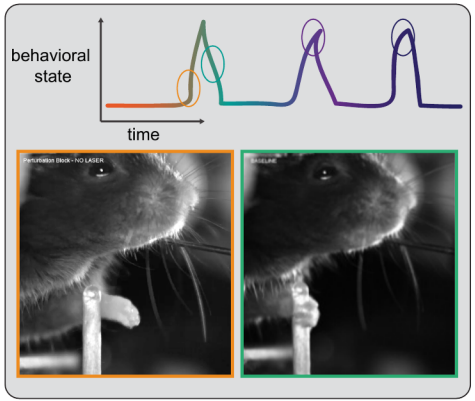
Frame extraction: deeplabcut.extract_frames(..)

Select videos to grab frames:

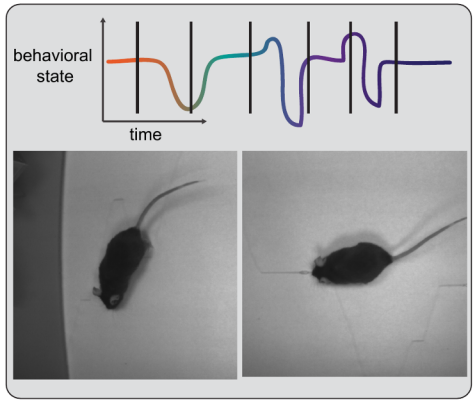
Use videos with image from:
- different sessions reflecting (if the case) varying light conditions, backgrounds, setups, and camera angles (etc).
- different individuals, especially if they look different (i.e. brown + black mice)

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

Image based clustering (k-means)



Random temporal sampling (uniform)



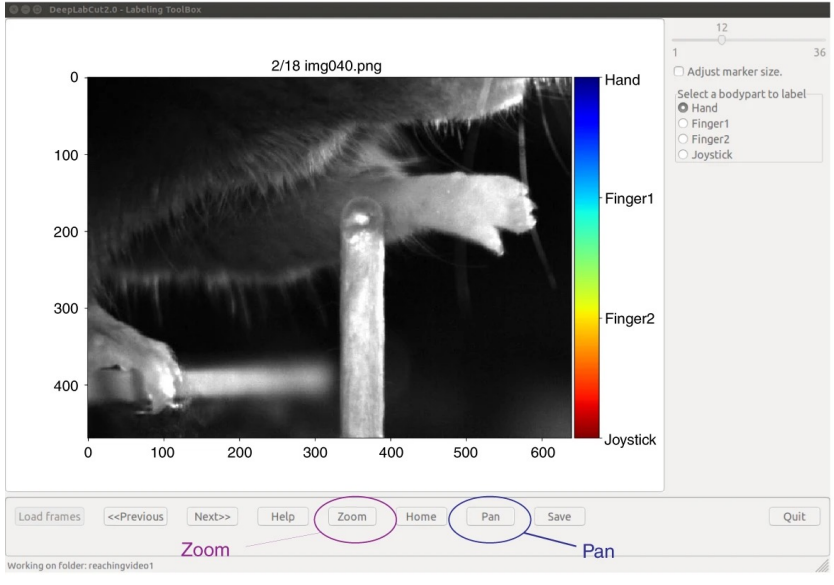
GUI for manual frame grabbing

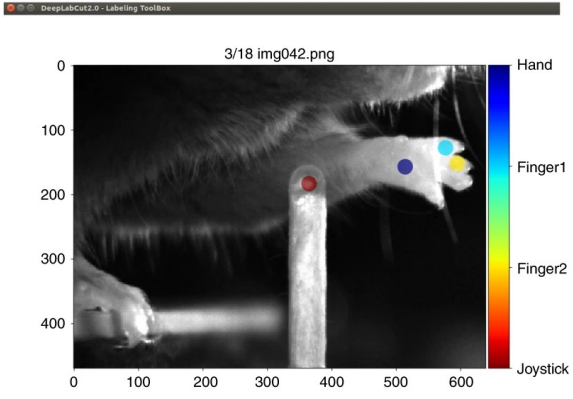
Nath*, Mathis* BioRxiv 2018

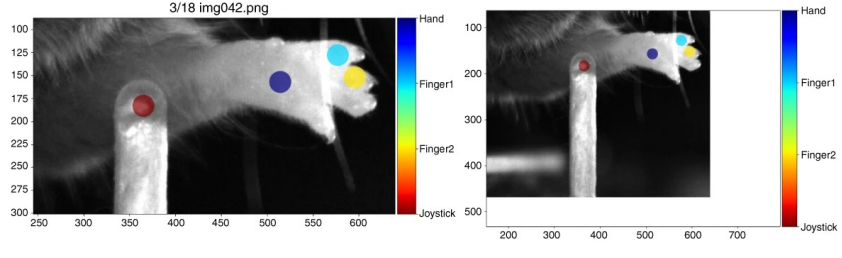
Label frames: deeplabcut.label_frames(..)

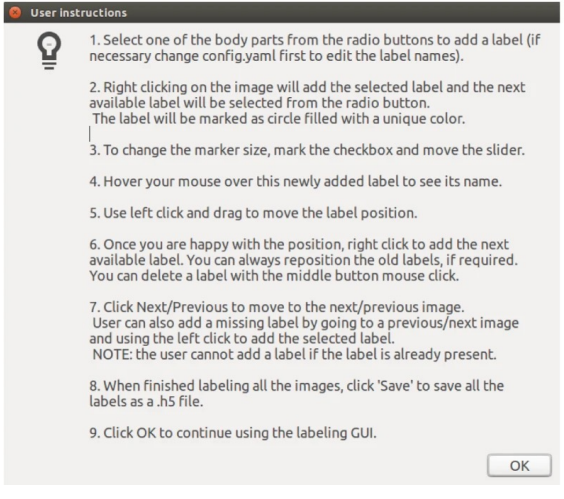
Label frames using the interactive GUI:

```
deeplabcut.label_frames(config_path)
```





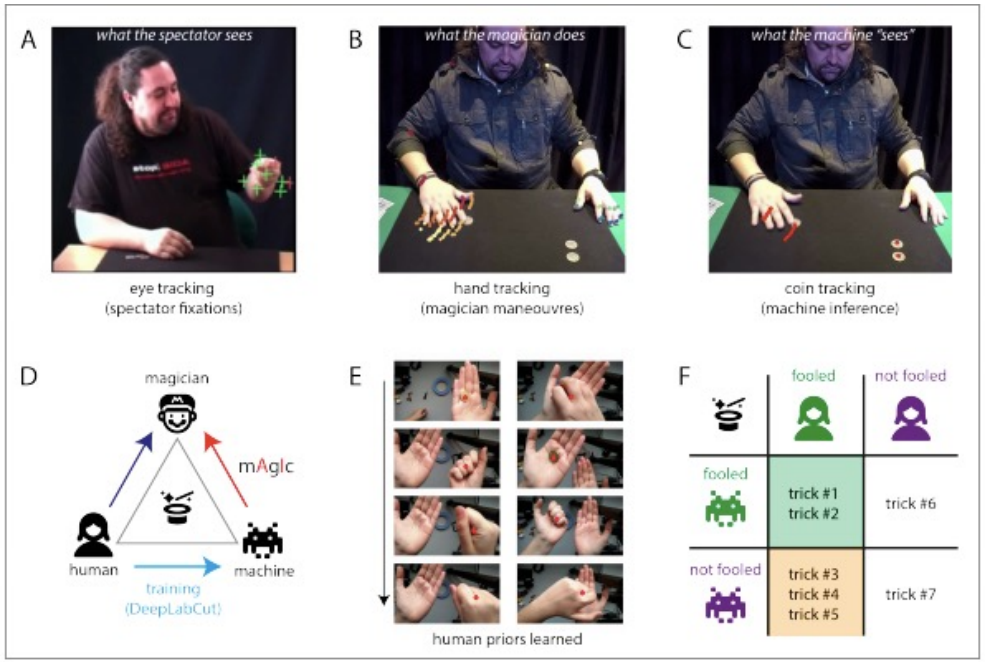




What bodyparts can be detected?

DLC has very large receptive fields (due to deep ResNet), so many visual features + geometric layout can be co

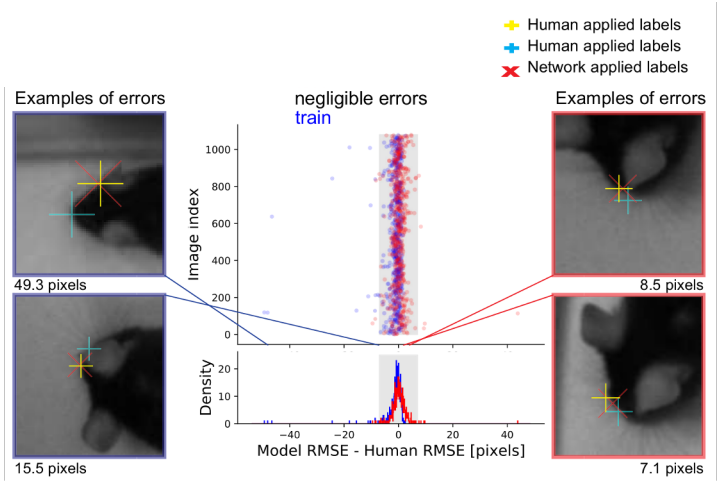
- Salient points (e.g. snout/tail base)
- Texture of objects
- Can teach the network to guess
- Geometric relationship
  - (e.g. left/right ear; whiskers!)



Playing magic tricks to deep neural netw untangles human deception
by Zaghi-Lara et al. arxiv 2019

Labeling considerations

- Label (feature locations) **consistently**
- Errors are bad! (especially when only a few images are labeled)
- Left vs. right body parts can be dist. if full animal visible (due to large rec. fields), but flipped labels are problematic



- You can check labels by plotting (do this before training!)
- You can correct wrong labels manually with the labeling GUI

What network & augmentation method?

Network architecture:

- Network backbones: MobileNets V2, *ResNets, DLC-Rnet* & EfficientNets
- Scale (level of deconvolution) -> decrease stride for closely interacting animals and and when you want high resolution
- Differences: **Inference & training speed, resolution**

Pre-training (Weights):

- *ImageNet pretrained (the generalist) -> see* Mathis et al. 2021 WACV
- MPII-pose pretrained
- DeepLabCut model zoo

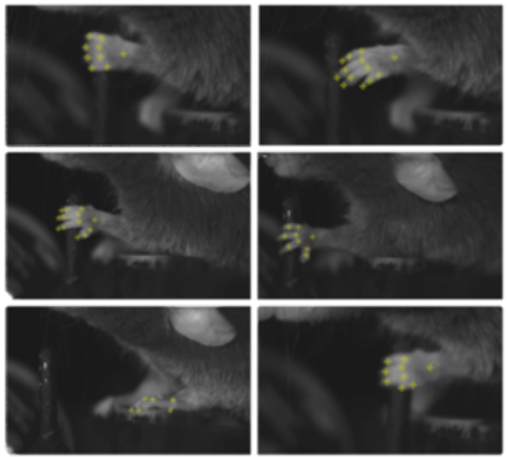
Training method:

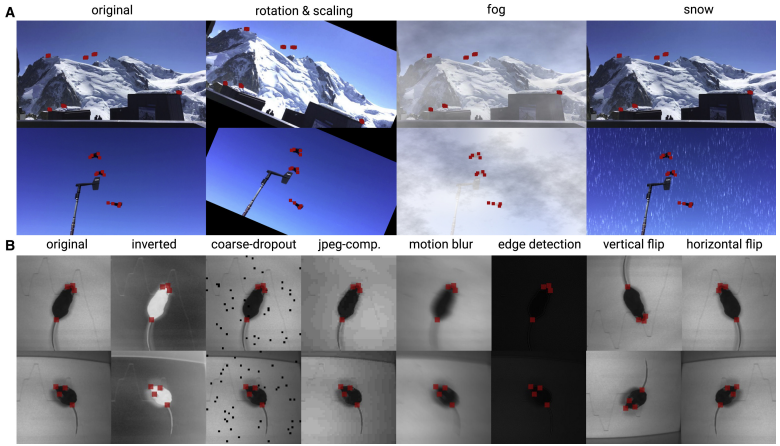
- Learning rate & optimizer (Adam/SGD, batch_size, …)
- Augmentation transformations (rotation, scaling, …)

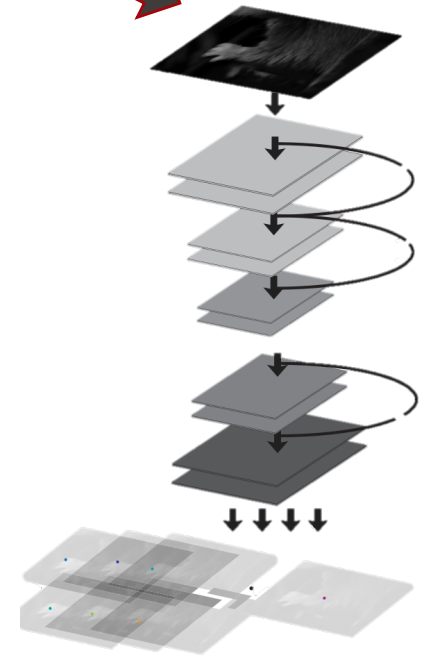
Online demo … let's start training

DEMO:
https://github.com/DeepLabCut/DeepLabCut/blob/master/examples/COLAB/COLAB_DEMO_mouse_openfield.ipynb

Augmentation

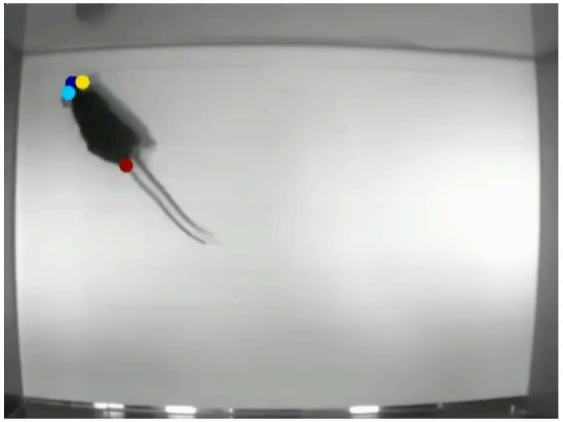




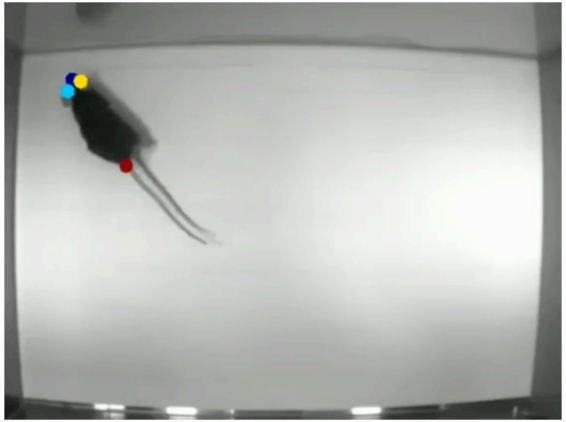


Mathis, Schneider,
Lauer & Mathis, 2020
Neuron

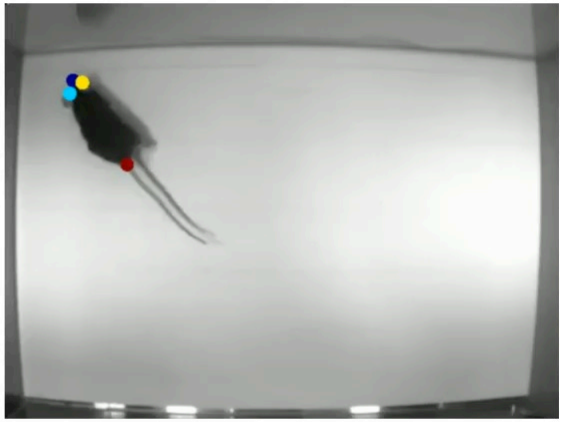
Data Augmentation: how to get the most out of your data!



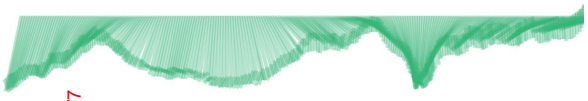
tensorpack

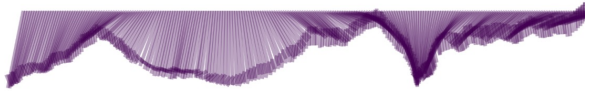


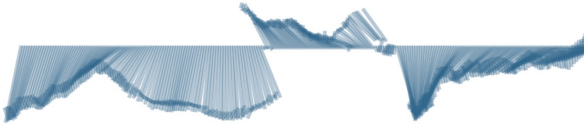
imgaug



scale-crop

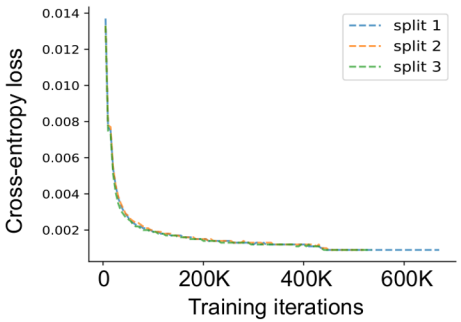


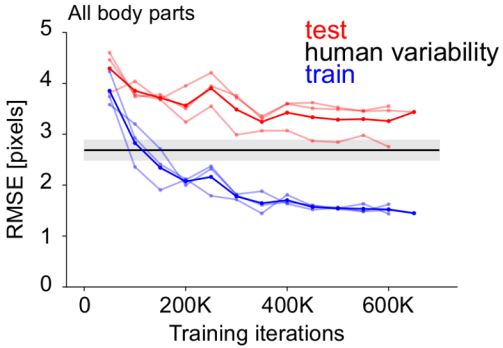


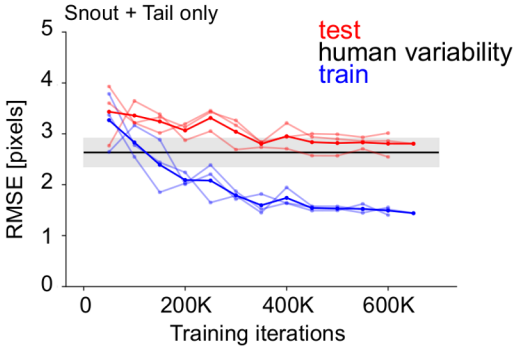


Mathis, Schneider, Lauer & Mathis 2020
Neuron

Evaluation: *Compare test RMSE and ideally your labeling error!*







Mathis et al. 2018, Nat Neuro

Evaluation => look at test images!

a Example of the terminal output

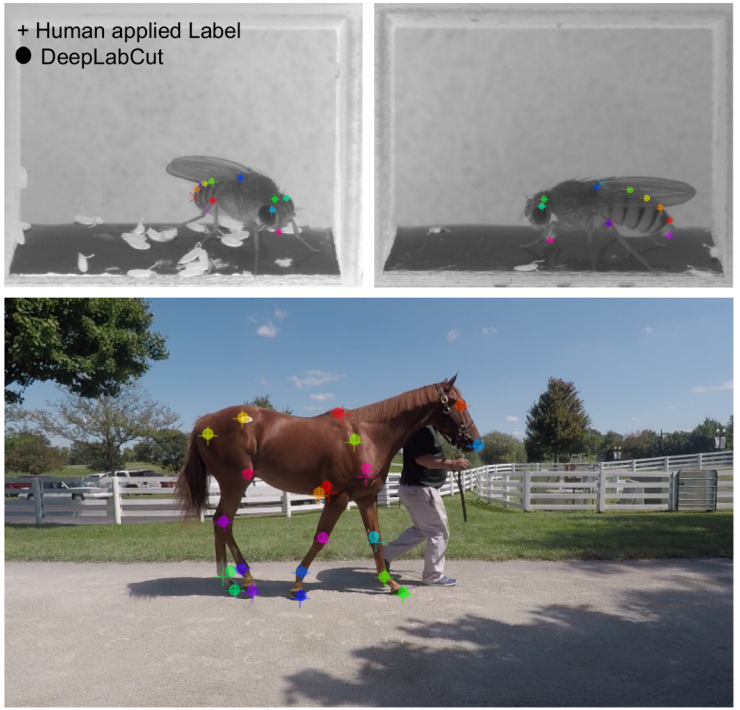
```
deeplabcut.evaluate_network('<path of the proj. config file>', shuffle = [1], plotting=True)
```

```
Assessing accuracy of shuffle #  1  with  99  % training fraction.
Found the following training snapshots:  [(200000, 0)]
You can choose among those for analyis of train/test performance.
Results for 200000 training iterations: 99 1 train error: 4.81 pixels. Test error: 7.02  pixels.
With pcutoff of 0.1  train error: 3.3 pixels. Test error: 5.2 pixels
```

b Examples of **training images** with Human and DeepLabCut labels

c Examples of **test images** with Human and DeepLabCut labels

d Examples of **test images** with Human and DeepLabCut labels



Nath* & Mathis* et al. Nature Protocols 2019

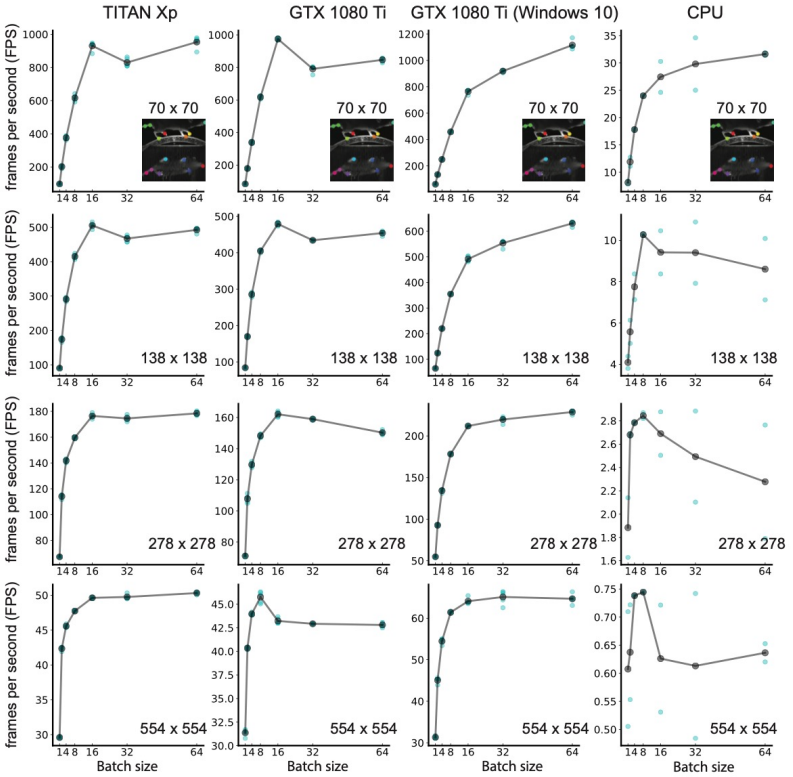
Table 1 | Summary of commands

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)`

Video analysis

New videos need **not** be added to the config.yaml!

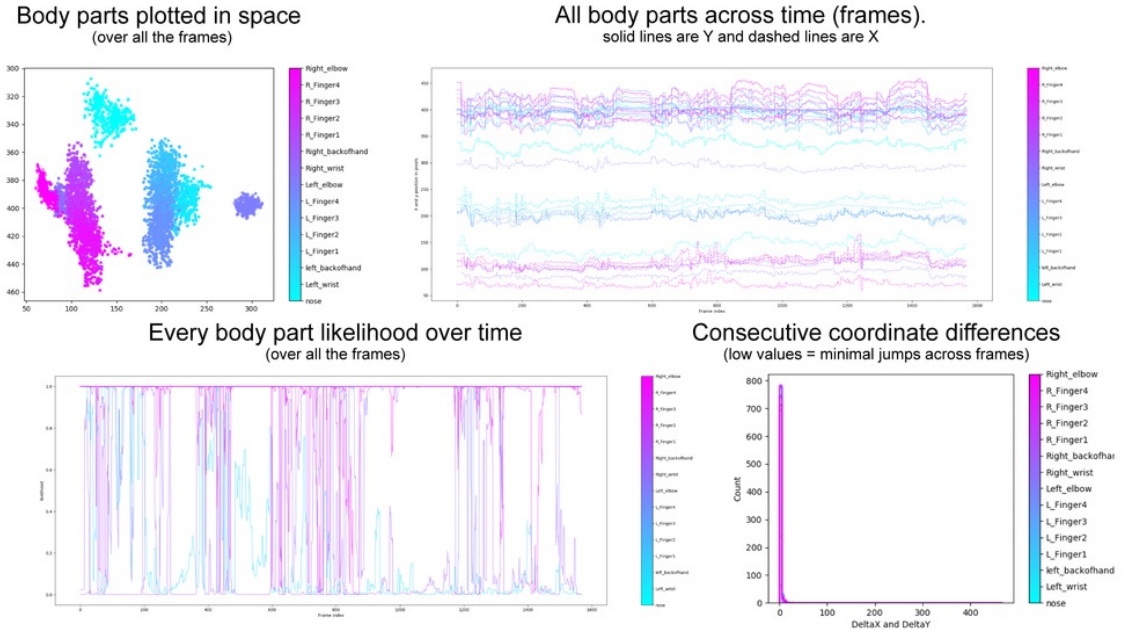
Set *batch_size* in the config.yaml!



Mathis & Warren 2018

Trajectory visualization


```
deeplabcut.plot_trajectories(config_path,['fullpath/analysis/project/videos/reachingvideo1.avi'])
```

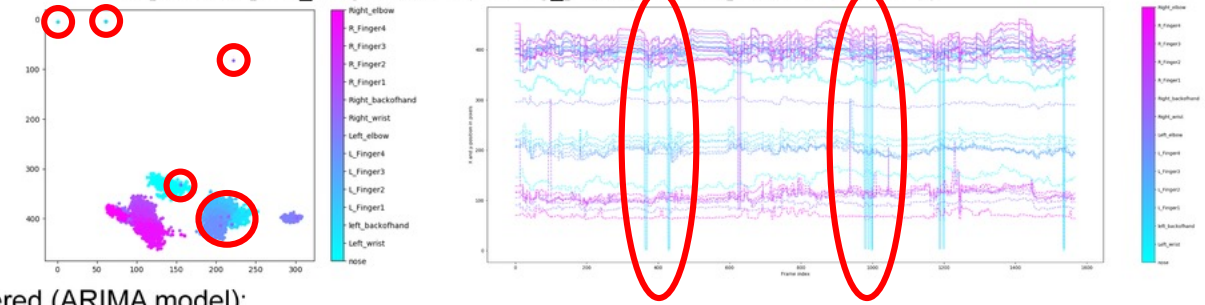


Trajectory filtering

```
deeplabcut.filterpredictions(config_path,['/videopath/mouse.avi'], p_bound=0.1, ARdegree=5, MAdegree=1)
```

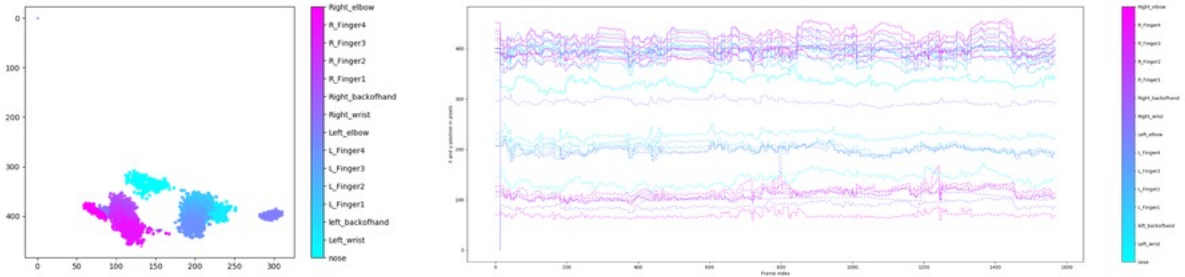
Unfiltered:

```
deeplabcut.plot_trajectories(config_path,['/videopath/mouse.avi'])
```

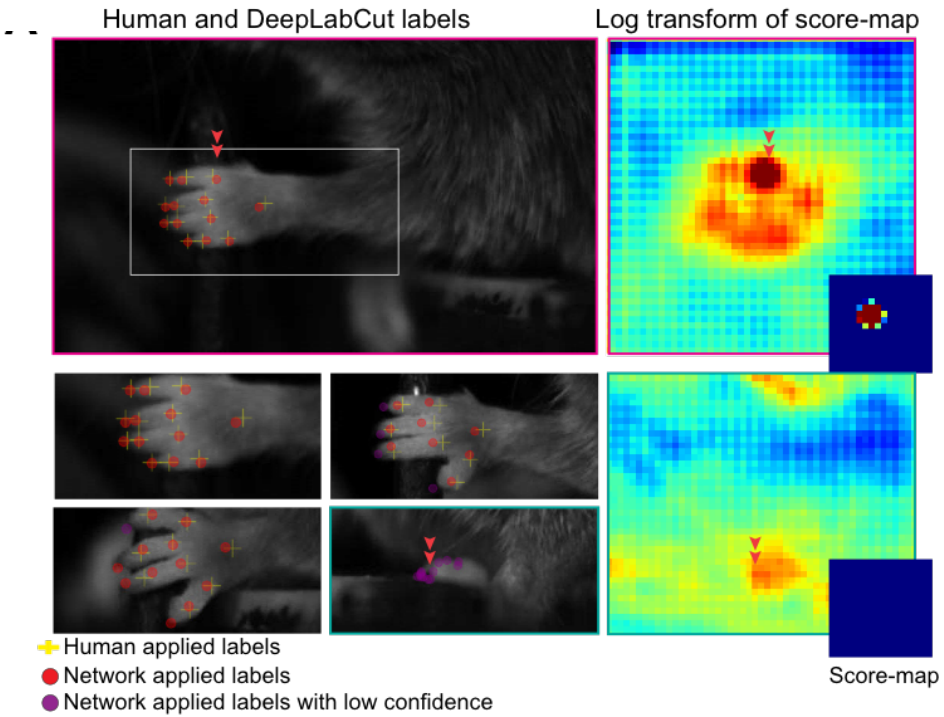


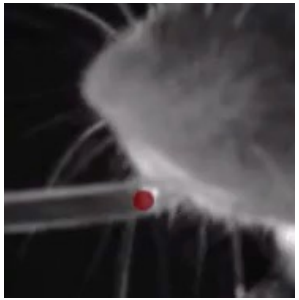
Filtered (ARIMA model):

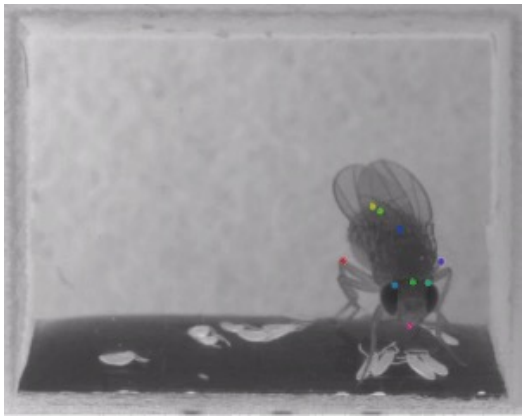
```
deeplabcut.plot_trajectories(config_path,['/videopath/mouse.avi'], filtered=True)
```



Score-maps provide network confidence readout

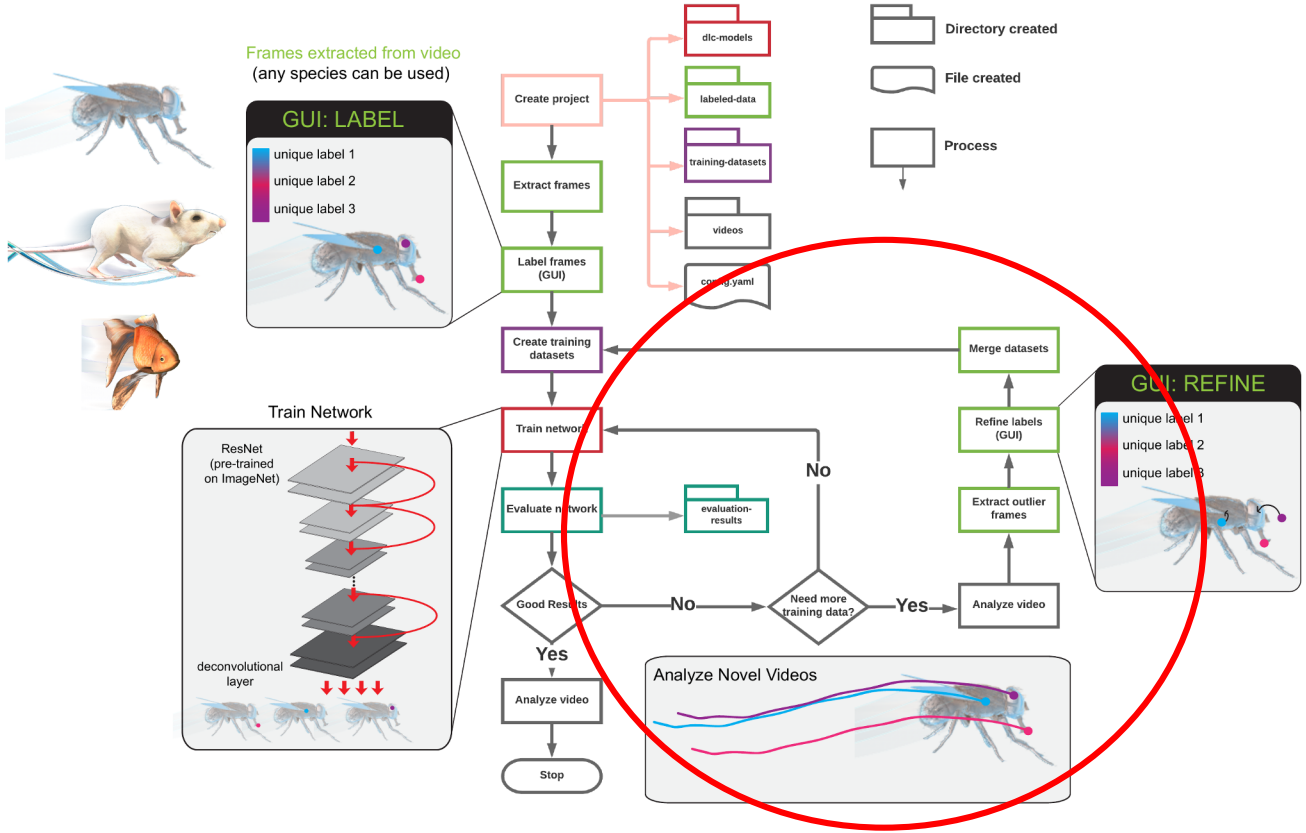






Mathis et al, Nature Neuroscience 2018

DeepLabCut 2.0 workflow



Active learning

- If there are errors in analyzed videos, you can correct those manually and re-train!
- Note you do not need to correct all errors (presumably some are fixed due to generalization)

Active learning


```
deeplabcut.extract_outlier_frames(config_path,['videofile_path'])
```

- Key arguments

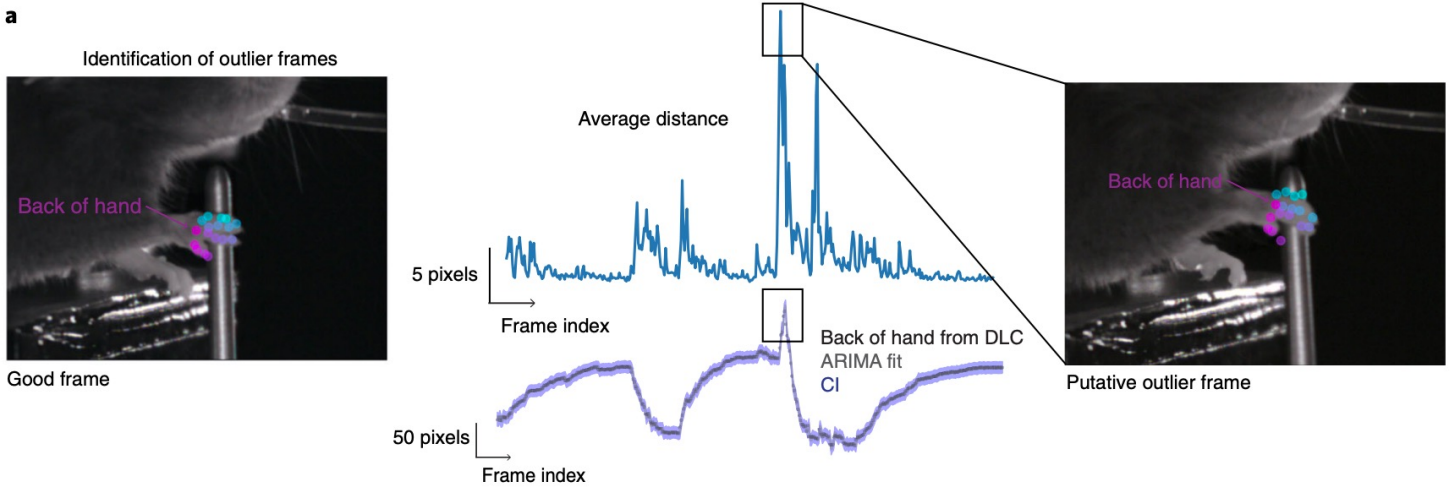
```
outlieralgorithm: 'fitting', 'jump', or 'uncertain'
```

Select frames with jumps between consecutive frames > *epsilon* pixels

Select frames with confidences < *p_bound*

Select frames deviating from
a statistical model fit to the data

Random sampling	`extractionalgorithm='uniform'`
Clustering	`extractionalgorithm='k-means'`


a
Identification of outlier frames
Back of hand
Good frame
Average distance
5 pixels
Frame index
Back of hand from DLC
ARIMA fit
CI
50 pixels
Frame index
Back of hand
Putative outlier frame
b
Refinement of DeepLabCut-applied label location(s)
deeplabcut.refine_labels(config_path)
0/19 img025
Enter the threshold
Select the likelihood threshold
Cancel
OK
Hand
Finger1
Finger2
Joystick
Load labels
<<Previous
Next>>
Help
Zoom
Home
Pan
Save
Quit
ZOOM
4/19 img038 threshold chosen is: 0.10
PAN
5/19 img048 threshold chosen is: 0.10
c
Label removal, i.e., from an occluded point
7/19 img057 threshold chosen is: 0.10
BASELINE
Remove!
Do you want to remove Hand ? You cannot undo this step!
No
Yes
Hand_p=0.06

Multi-animal DLC

DEMO:
https://github.com/DeepLabCut/DeepLabCut/blob/master/examples/COLAB/COLAB_3miceDemo.ipynb

DeepLabCut 2.2+ workflow

DATASET CURATION

Create a project

Extract & Label Frames GUI

create training datasets

maDLC

standard DLC

POSE ESTIMATION

train your multi-task neural network

evaluate performance:

pixel error & detection video quality

QUALITY CHECK 1

IDENTITY AND MULTI-ANMAL TRACKING

Analyze Novel Videos

PICKLE FILE

Assemble & Stitch Tracklets

H5/CSV FILE

Optional GUI: Refine Tracklets

QUALITY CHECK 2

SINGLE ANIMAL

Analyze Novel Videos

H5/CSV FILE

OPTIONAL POST PROCESSING TOOLS

Filter data

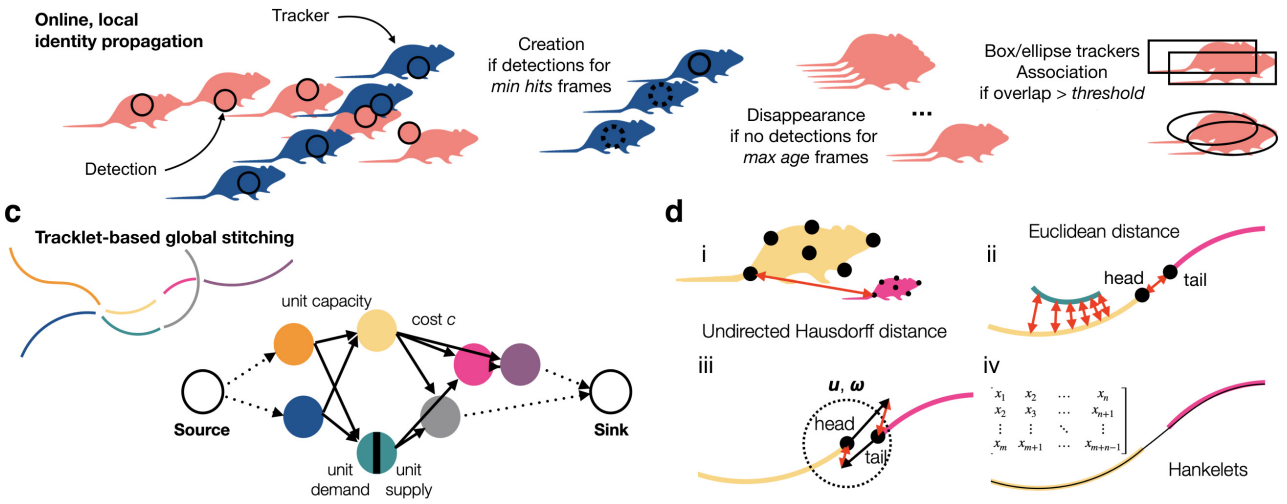
Create labeled videos

Plot Trajectories

3D Triangulation

Tracking

Stitching



Corresponding commands:

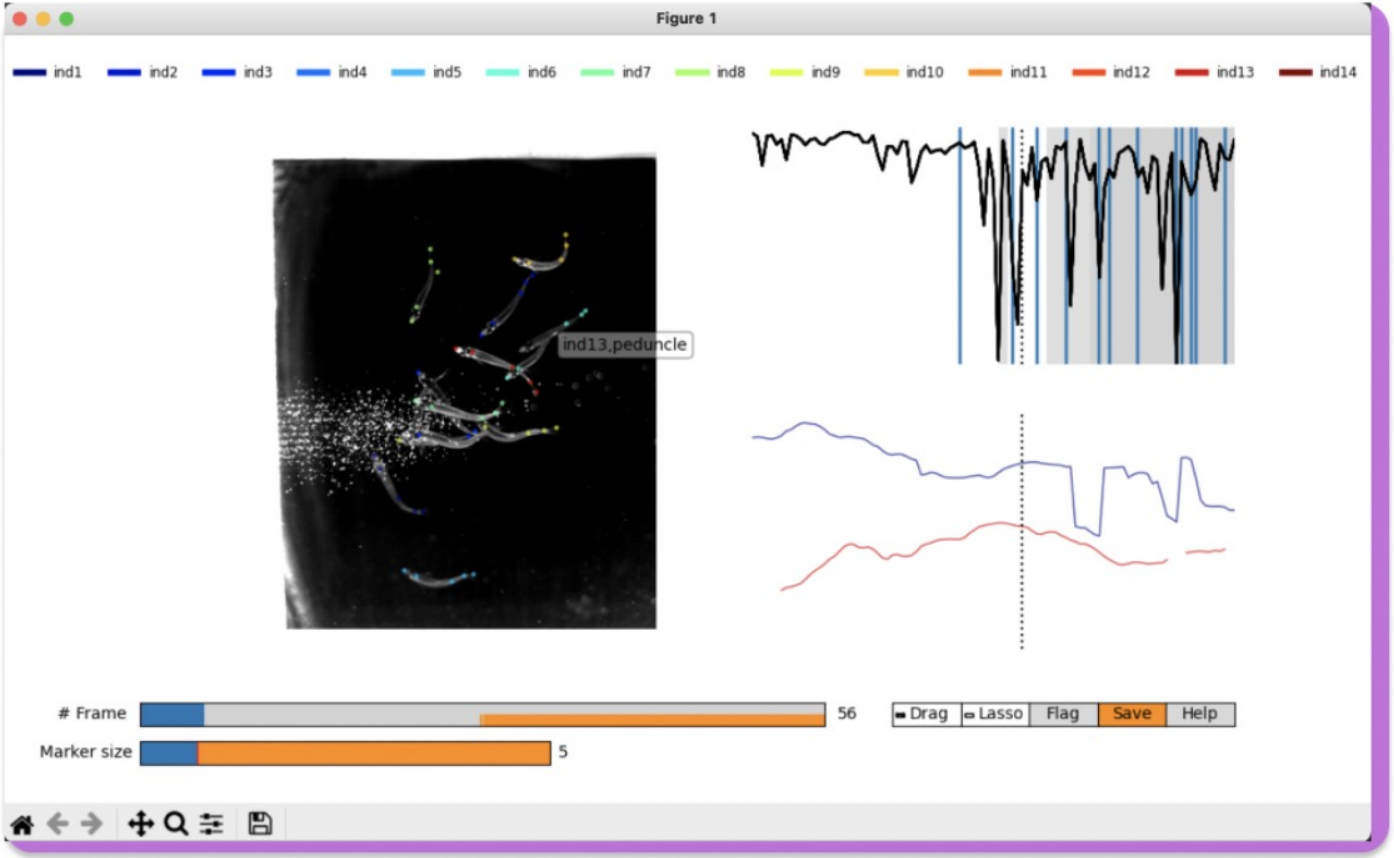
```
dlc.analyze_videos( .. )

dlc.convert_detections2tracklets(…)

dlc.stitch_tracklets(…)
```

DEMO:
https://github.com/DeepLabCut/DeepLabCut/blob/master/examples/COLAB/COLAB_3miceDemo.ipynb

Refine tracklets GUI: deeplabcut.refine_tracklets()



Project GUI demo!

How to use DeepLabCut?

https://github.com/DeepLabCut/DeepLabCut-Workshop-Materials

Your goals …

- I would like to get a basic understanding of how DeepLabCut works
- 3D implementation
- How to run DLC successfully and get animal tracking data from it
- Basic functions/introduction
- How to train it for very high accurate digitization
- How to prepare files and train dataset
- Going through the entire process of setting up, training, collecting data
- Basics!
- Basic run-through of steps and especially how to fix poor training- do you need to start all over if it doesn't track well?
- General workflow, tips for beginners (if suitable for students). Also, suggestions on hardware
- Things I haven't tried yet, like real time and 3D
- How to use this software to track small, low-contrast animals in low-light settings
- How to track/quantify specific movements and behaviors of a single animal
- How to use COLAB, options when tracking two different organisms, best approach when tying to train multiple models from recordings that all have one organism in common but a different second interacting organism
- ….

Acknowledgements

Mathis Group, EPFL:

PhD students: Lucas Stoffl (Poet), Alessandro Marin-Vargas (DeepDraw), Alberto Chiappa, **Mu Zhou (DLC),** Haozhe Qi
Master's Students/interns: Axel Bisi (Deep Draw), Liza Kozlova, Maxime Vidal (Poet)

Chan Zuckerberg Initiative

EPFL

THE KAVLI FOUNDATION

DeepLabCut collaborators:

Mackenzie Mathis, EPFL
Jessy Lauer, EPFL (DLC 2.2 / Mathis labs)
Mu Zhou, EPFL (Mathis group)
Ye Shaokai, EPFL (Mathis lab)
Gary Kane, Harvard (DLC-Live! / M. Mathis lab)
Tanmay Nath, Harvard (DLC 2.0 / M. Mathis lab)
Matthias Bethge, Uni Tübingen
Steffen Schneider, Uni Tübingen (M Mathis/Bethge labs)
Tom Biasi, Harvard (Robustness)
Mert Yüksekgönül, Uni Tübingen (Faster Inference, Robustness)
+ >65 people on GitHub (Thanks!)

Experimental collaborators:

William Menegas, Feng lab, MIT (Marmoset data),
Venkatesh N. Murthy, Harvard (Trail tracking, 3 mouse data)
Catherine Dulac , Harvard (Social behaviors)
Valentina Di Santo, Stockholm University & George Lauder, Harvard (Fish)
Amir Patel, U of Cape Town (Cheetah)

Questions?

Please get in touch!

alexander.mathis@epfl.ch

Open Source Code: https://github.com/DeepLabCut/
Tutorials, Data, Papers etc.: https://deeplabcut.org
User forum: https://forum.image.sc/tag/deeplabcut

Box 2 | Parameters of interest in the network configuration file, *pose_cfg.yaml*

Please note, there are more parameters that typically never need to be adjusted; they can be found in the default *pose_cfg.yaml* file at https://github.com/AlexEMG/DeepLabCut/blob/master/deeplabcut/pose_cfg.yaml.

- `display_iters`: An integer value representing the period with which the loss is displayed (and stored in *log.csv*).
- `save_iters`: An integer value representing the period with which the checkpoints (weights of the network) are saved. Each snapshot has >90 MB, so not too many should be stored.
- `init_weights`: The weights used for training. Default: `<DeepLabCut_path>/Pose_Estimation_Tensorflow/pretrained/resnet_v1_50.ckpt`. For ResNet-50 or 101, -- this will be automatically created. The weights can also be changed to restart from a particular snapshot if training is interrupted, e.g., `<full path>-snapshot-5000` (with no file-type ending added). This would re-start training from the loaded weights (i.e., after 5,000 training iterations, the counter starts from 0).
- `multi_step`: These are the learning rates and number of training iterations to perform at the specified rate. If the users want to stop before 1 million, they can delete a row and/or change the last value to be the desired stop point.
- `max_input_size`: All images larger with size width × height > `max_input_size*max_input_size` are not used in training. The default is 1500 to prevent crashing with an out-of-memory exception for very large images. This will depend on your GPU memory capacity. However, we suggest reducing the pixel size as much as possible; see Mathis and Warren[27].

The following parameters allow one to change the resolution:

- `global_scale`: All images in the dataset will be rescaled by the following scaling factor to be processed by the convolutional neural network. You can select the optimal scale by cross-validation (see discussion in Mathis et al.[12]). Default is 0.8.
- `pos_dist_thresh`: All locations within this distance threshold (measured in pixels) are considered positive training samples for detection (see discussion in Mathis et al.[12]). Default is 17.

The following parameters modulate the data augmentation. During training, each image will be randomly rescaled within the range `[scale_jitter_lo, scale_jitter_up]` to augment training:

- `scale_jitter_lo`: 0.5 (default).
- `scale_jitter_up`: 1.5 (default).
- `mirror`: If the training dataset is symmetric around the vertical axis, this Boolean variable allows random augmentation. Default is `False`.
- `cropping`: Allows automatic cropping of images during training. Default is `True`.
- `cropratio`: Fraction of training samples that are cropped. Default is 40%.
- `minsize`, `leftwidth`, `rightwidth`, `bottomheight`, `topheight`: These define dimensions and limits for auto-cropping.

Nath* & Mathis* et al. Nature Protocols 2019