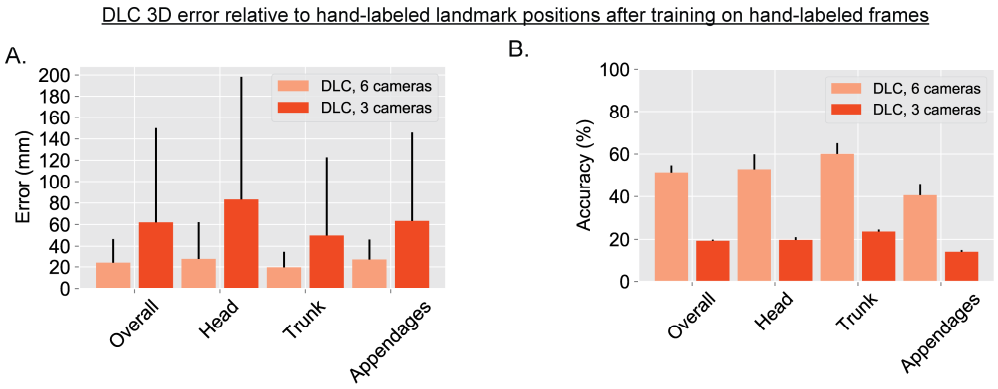
Supplementary Figures

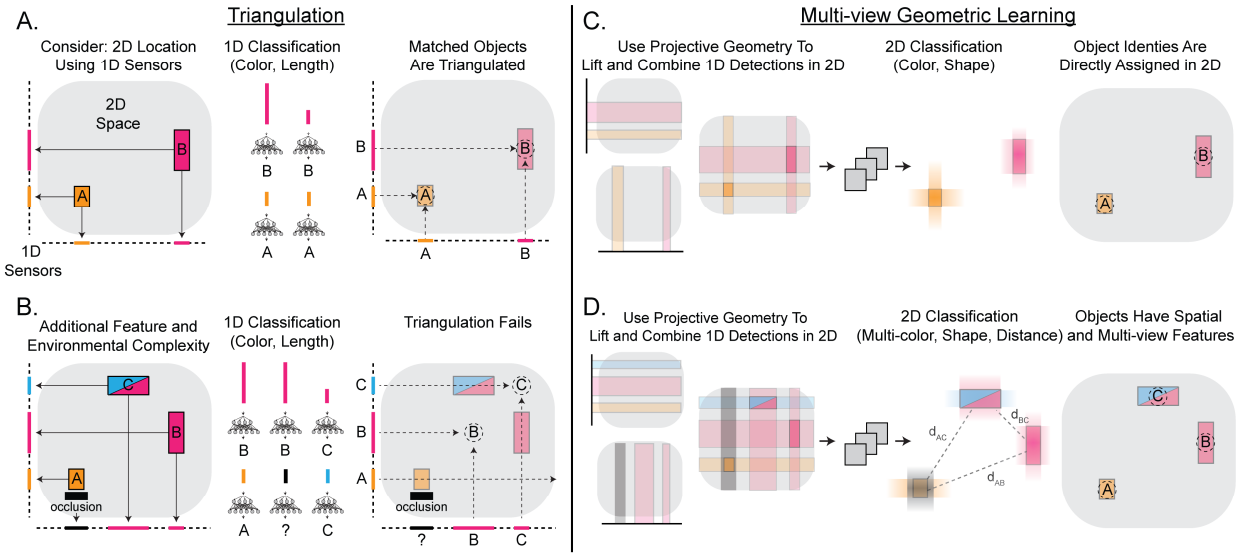
Supplementary Figure 1 | Error metrics for DeepLabCut fine-tuned with hand-labeled frames.



A. Mean DLC Euclidean error for trunk, head, and appendage landmarks. *N* = 3 animals. *N* = 894 overall landmarks, *N* = 193 head, *N* = 371 trunk, *N* = 330 appendages. Error bars show the standard deviation.

B. Bar plots of DLC accuracy for trunk, head, and appendage landmarks using the same data as in (**A**). Accuracy is computed using an 18 mm (the average distance between two forelimb markers) threshold to binarize predictions. Error bars show 95% confidence intervals.

Supplementary Figure 2 | Illustration of triangulation vs. volumetric concepts.



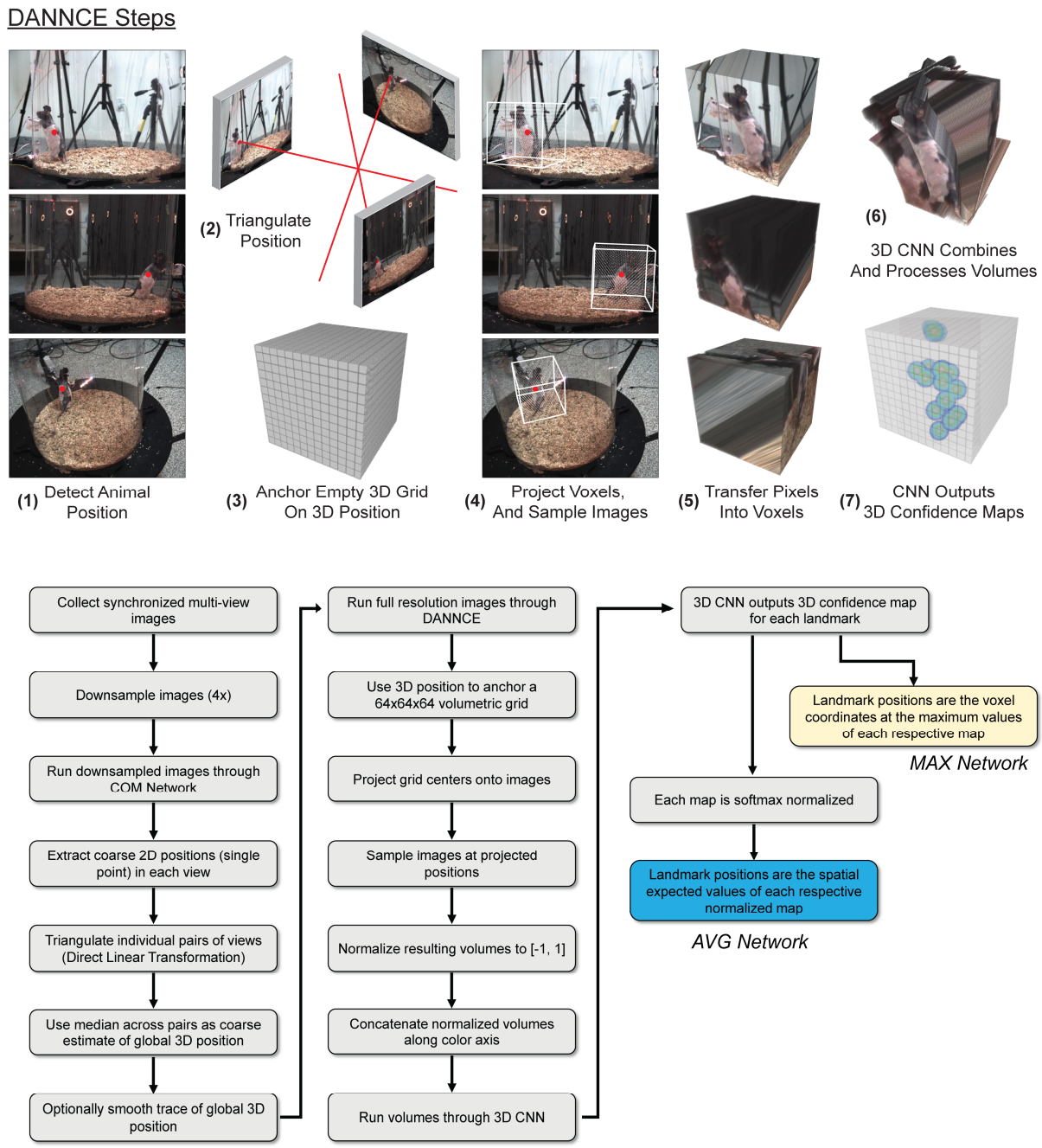
A. *I*n an analogous scenario where measurements using 1D sensors are used to reconstruct the 2D positions of specific objects, 1D classifiers are sufficient when the projections of each object onto the sensors are perfectly separated and distinguishable.

B. When the features associated with each object become more natural and complex (e.g. colors that look different depending on viewing angle), and objects become occluded, it is challenging to classify the detections on each 1D sensor separately and unambiguously.

C. Rather than classify 1D detections separately in 1D using 1D classifiers, an alternative is to combine the 1D detections in a 2D space, using knowledge of where the 1D detections would intersect based on the orientation of the sensors relative to this space. A 2D, rather than a 1D, classifier takes in the entire 2D space as input and can use overlapping color and shape to assign identities to each object.

D. Using the combined 2D strategy, the 2D classifier can combine information across sensors (e.g. the larger pink and blue intersection corresponds to object C), and, critically, it can learn and utilize the distribution of distances between objects. This is especially useful in the presence of occlusions and when landmark positions have consistent spatial relationships, such as across the body of an animal.

Supplementary Figure 3 | Detailed schematic of DANNCE.

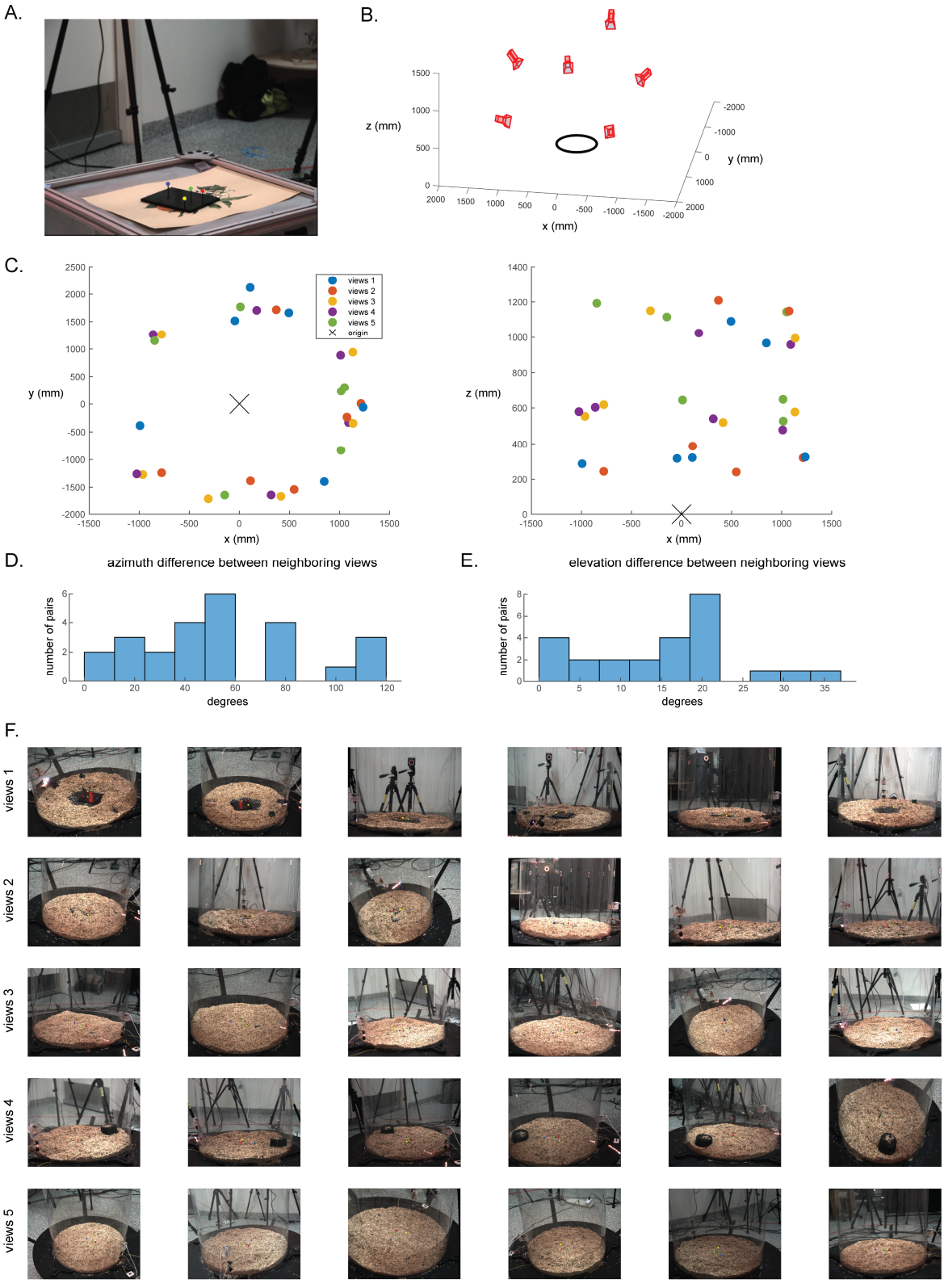


Top. Details of the steps used in DANNCE to predict 3D landmarks, illustrated with real images of a rat. **(1)** To anchor the volume that will contain a 3D representation of the animal, the animal's 2D position in each frame (red dot) is detected using a 2D neural network. **(2)** These 2D positions are triangulated to 3D using the calibrated geometry of the cameras. **(3)** An empty 3D grid is then centered on this triangulated 3D positions, and each voxel is this 3D grid is assigned a real 3D spatial coordinate (in mm). **(4)** The spatial coordinates of each voxel are projected to 2D using the calibrated geometry of the cameras. In these images, the white dots show the projected 2D coordinates of each voxel's 3D coordinate, and the white

outlines denote the 3D border of the grid projected into 2D. **(5)** The RGB image content at each projected voxel 2D location is transferred to the voxel's original 3D position, creating 3D spatially aligned image volumes. **(6)** These volumes are then passed as input to the 3D CNN. The 3D positions of each landmark lie at the intersection of matched image features in these volumes. For this example, rat volumes were manually segmented to illustrate feature convergence. **(7)** Finally, the 3D CNN detects landmark-specific feature convergence and output a 3D spatial confidence map indicating the location of each landmark.

Bottom. Extended flow chart describing all required steps in more detail.

Supplementary Figure 4 | Rat 7M contains a diversity of camera arrangements.



A. Example image of the L-frame target used for calibrating each video and motion capture camera to a single world coordinate system. Positions of markers on the L-frame are overlaid with colored circles.

B. Example positions of a single set of 6 calibrated cameras used for recordings in one rat.

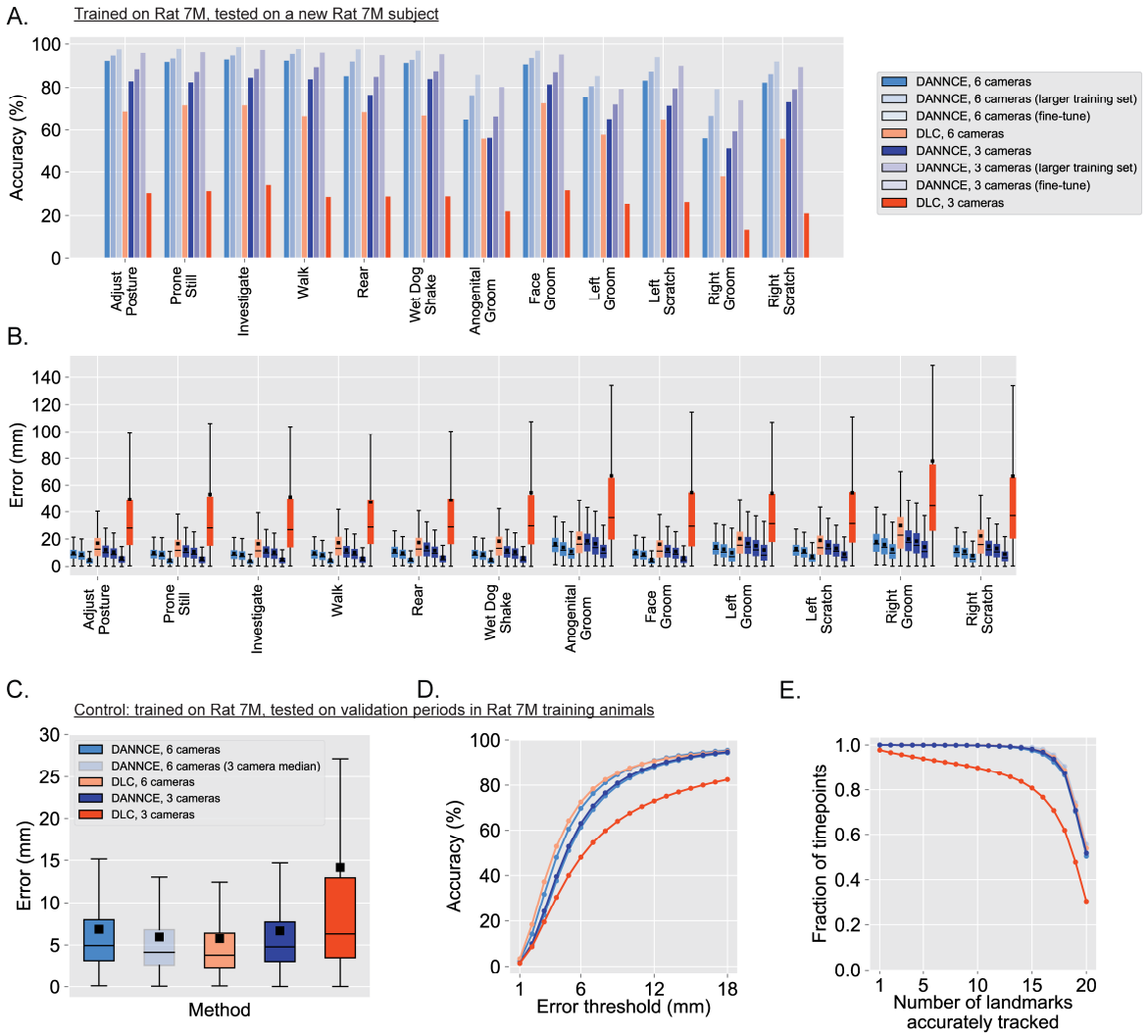
C. All 3D camera positions in Rat 7M, shown on the xy-plane (*left*) and xz-plane (*right*). The different colors represent different recordings.

D. Histogram of azimuth angle differences between neighboring camera pairs across the Rat 7M dataset. A neighboring camera pair is a given camera and its closest neighbor. All cameras and their closest neighbors are included, without repeating pairs.

E. Histogram of the elevation angle differences between neighboring camera pairs.

F. Example video frames from each of the 30 views included in the Rat 7M dataset. Each view shows the L-frame targets with marker positions overlaid with colored circles.

Supplementary Figure 5 | Extended performance metrics for DLC and DANNCE.



A. Landmark prediction accuracy (% predictions with error < 18 mm relative to ground truth) on specific behavioral categories for the same data and methods as in **Fig. 3A**. *N* = 199 instances of each behavior, *N* = 1980 markers for each 6 camera method per behavior, *N* = 39,600 markers for each 3 camera method per behavior. DLC predictions come from the "median" triangulation condition.

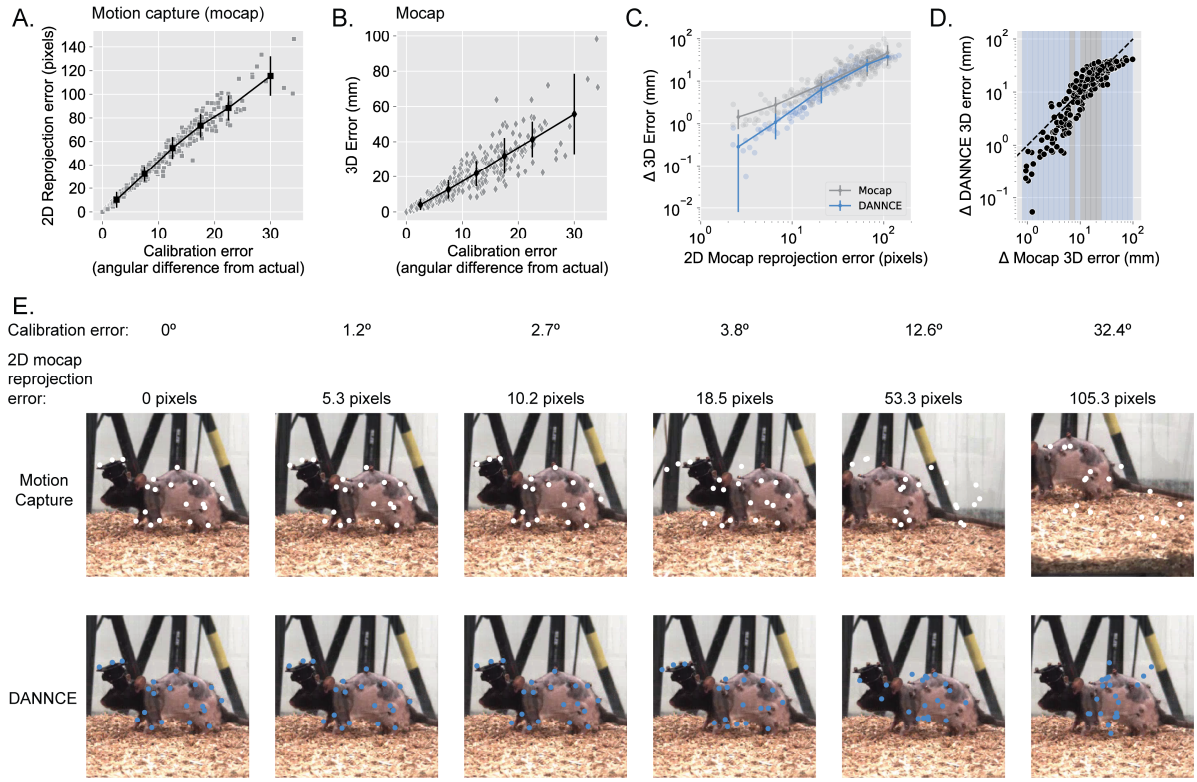
B. Box plots of Euclidean error for the same data and behaviors as in (**A**). Box plots in (**B-C**) show median with IQR and whiskers extending to 1.5x IQR. Black squares in the box plots are arithmetic means.

C. Box plots of Euclidean error for the indicated methods, over withheld data in *N* = 4 training animals. *N* = 80,000 markers. "DANNCE, 6 cameras (3 camera median)" computes final pose output as the median over predictions from all 20 combinations of 3 cameras.

D. Landmark prediction accuracy as a function of error threshold, for the same data and methods as in (**C**). Line colors use the same legend as in (**C**).

E. Fraction of timepoints with the indicated number of markers accurately tracked (error < 18 mm). Line colors use the same legend as in (**C**).

Supplementary Figure 6 | DANNCE is more robust than triangulation to small calibration errors.



A. Simulation of error in triangulated ground truth motion capture measurements with camera calibration noise. Random 3D rotations were applied to each of 6 cameras' extrinsic rotation matrices and the error relative to ground truth motion capture was computed after triangulating and reprojecting using the noisy calibration matrix. Shown is a scatter plot of the reprojection error (averaged over 6 cameras) vs. the angular error (gray squares, averaged over 6 cameras). The black squares and error bars reflect the mean and standard deviation of the reprojection error over bins in angular calibration error. Error was calculated in a single Rat 7M subject, over the same $N = 3000$ 2D landmarks for each simulation ($N = 300$ simulations).

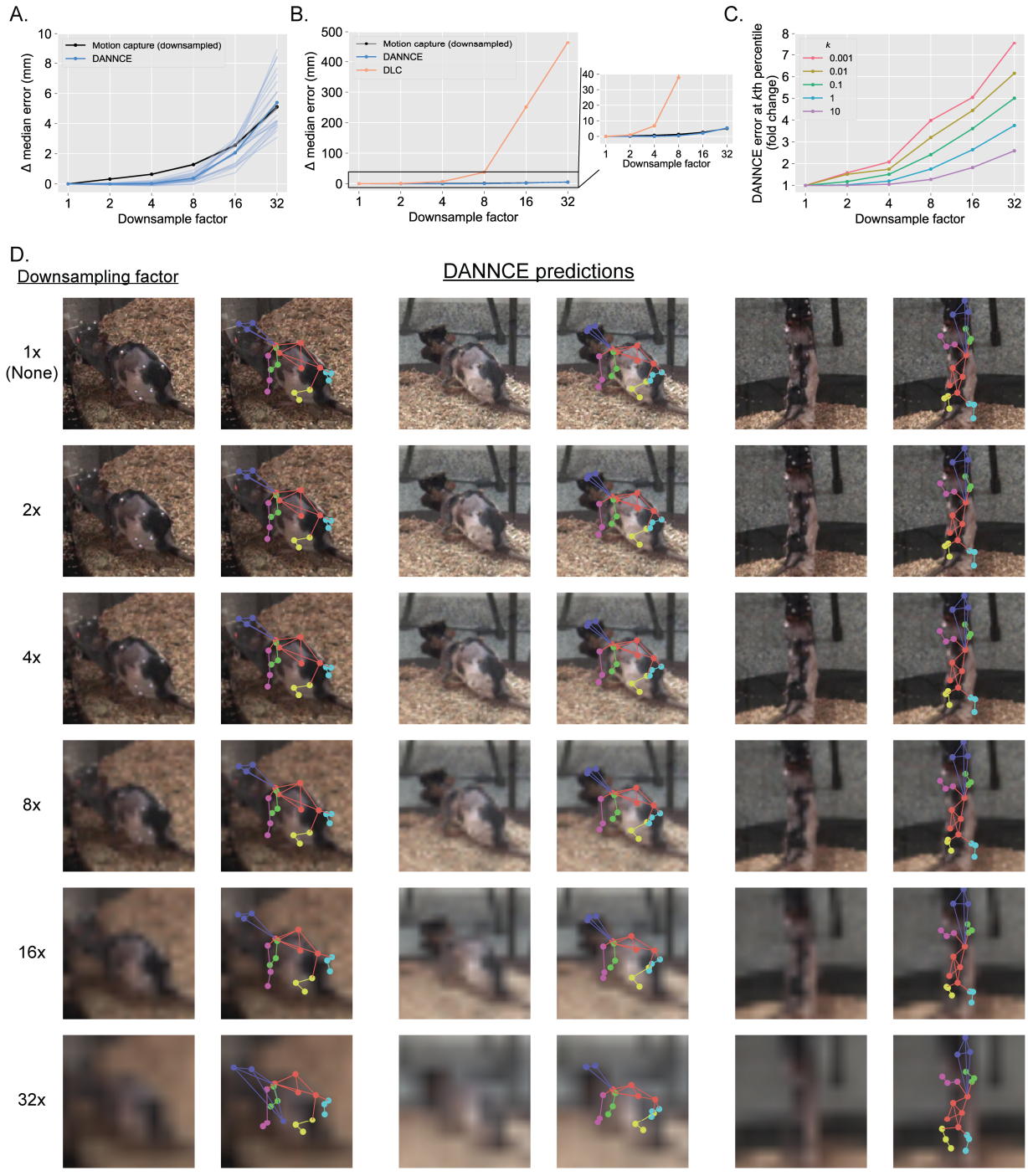
B. Scatter plot of 3D error relative to ground truth for the same calibration errors and samples as in (**A**). For each simulation, 3D points were calculated via triangulation, using the shifted rotation matrices, of ground truth 2D reprojections. $N = 500$ 3D landmarks for each of $N = 300$ simulations. Black squares and error bars reflect the mean and standard deviation of the 3D error over bins in angular calibration error.

C. Scatter plot of the change in 3D error for triangulation and for DANNCE predictions as calibration noise increases, for the same simulations as in (**A**) and (**B**). $N = 500$ 3D and 3000 2D landmarks for each of $N = 300$ simulations for each condition.

D. Scatter plot showing the change in 3D error for DANNCE vs. triangulation across all simulations. The colors indicate regions where either DANNCE (blue) or motion capture (gray) are more robust to calibration error, in terms of the number of simulations achieving lower 3D error (i.e., asymmetries over the identity line, dashed black).

E. Example (1 of 6 views) showing the effect of calibration error on triangulation vs. DANNCE reprojections.

Supplementary Figure 7 | DANNCE performance is more robust than triangulation to decreases in image resolution.



A. Line plot showing the change in median Euclidean error for DANNCE predictions (6 cameras) and triangulation as input image resolution is decreased. Triangulation is adjusted by rounding 2D landmark coordinates, here computed using reprojections of motion capture, to the nearest multiple of the indicated downsampling factor and then retriangulating. Light lines are the median error for each of the 20 marker types, and the dark line for each condition is the mean of the condition's marker-specific traces.

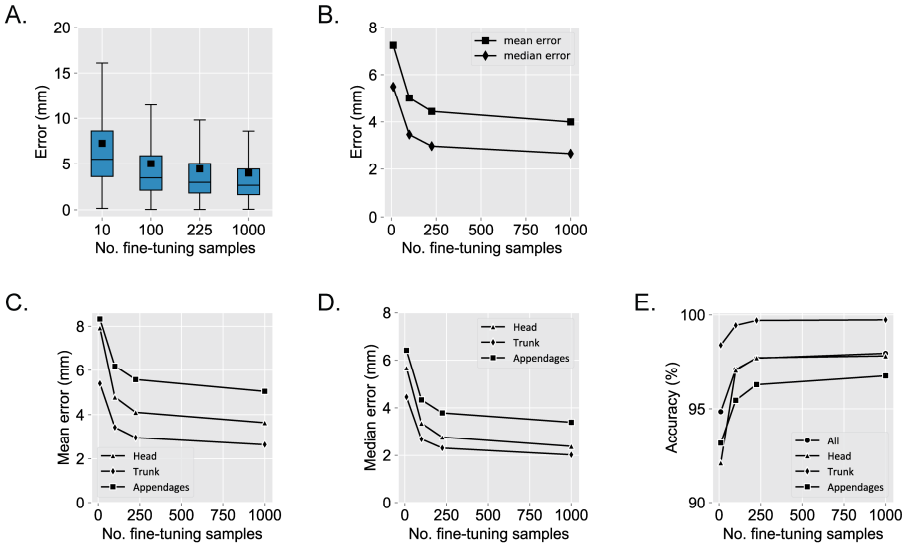
All data are from the Rat 7M validation subject. DANNCE predictions using the "DANNCE, 6 cameras (fine-tune)" model. *N* = 400,000 markers for each downsample factor.

B. Line plot of the mean lines from (**A**), with the addition of the change in median error for 6-camera DLC (median triangulation method). Inset shows a zoom-in of the indicated region.

C. Line plots showing the fold-change in DANNCE Euclidean error at increasing error percentiles, for the indicated image downsampling factors, using the same DANNCE data as in (**A**).

E. Examples of DANNCE prediction reprojections for three different Rat 7M samples at decreasing image resolutions. Each pair of columns is a different sample, with each pair of columns showing 1 of 6 total views. Images were downsampled using the mean of pixel values over non-overlapping image patches, followed by bilinear upscaling back to the original image size.

Supplementary Figure 8 | DANNCE performance increases with the size of the training dataset when fine-tuning.



A. Box plots of Euclidean error for 6-camera DANNCE predictions on the validation subject after fine-tuning the network with an increasing number of frames from Rat 7M. *N* = 33,200 validation markers for each training condition. Plots show median with IQR and whiskers extending to 1.5x IQR. Black squares are arithmetic means.

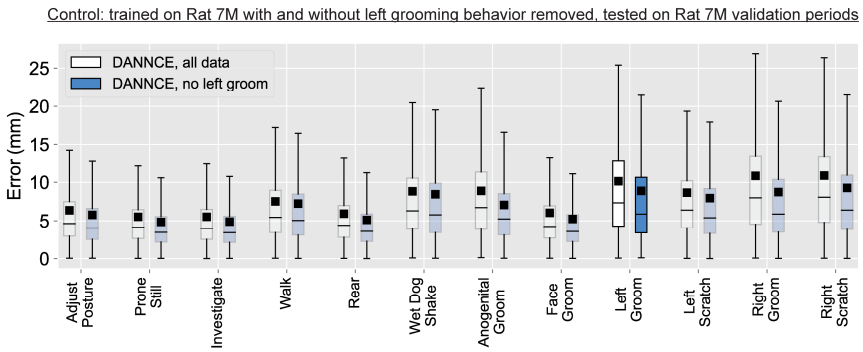
B. Plots showing the mean and median for the data in (**A**).

C. The mean error for each training condition, broken down by marker type. *N* = 4980 head markers, *N* = 11,620 trunk, *N* = 16,600 appendages.

D. The median error for each training condition, broken down by marker type, for the same data partitions as in (**C**).

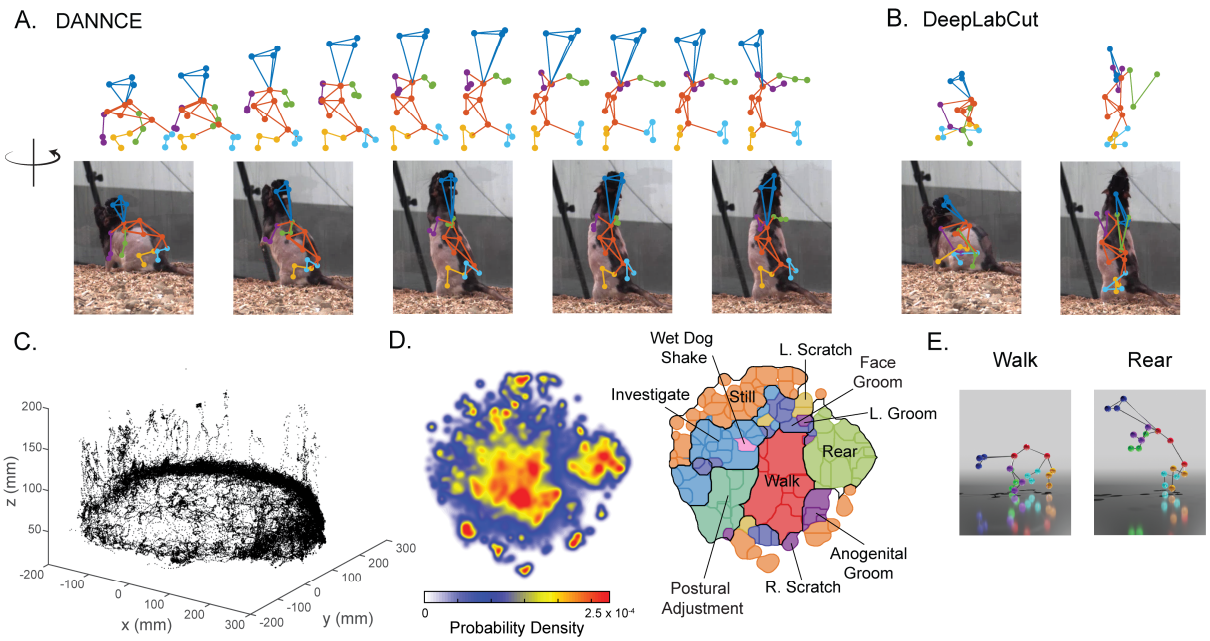
E. The accuracy (18 mm threshold) for each training condition, broken down by marker type, for the same data partitions as in (**C**).

Supplementary Figure 9 | DANNCE generalizes to unseen behavioral categories.



Box plots of Euclidean error of the 6-camera DANNCE network on individual behavioral types after withholding a single behavior ("Left Groom") from the training set in temporal validation periods of the *N* = 4 training animals. Box plots of error after training on the entire dataset are shown for comparison. *N* = 1,642 – 72,070 instances of each behavior (*N* = 32,840 – 1,441,400 markers). For left groom, *N* = 7,520 behavioral instances, *N* = 150,400 markers. Plots show median with IQR and whiskers extending to 1.5x IQR. Black squares are arithmetic means.

Supplementary Figure 10 | DANNCE generalizes to markerless rats without additional training data.



A. Example DANNCE predictions over a rearing sequence in a markerless rat (6 total cameras). The *top* row shows the 3D predictions, and the *bottom* row shows every other frame of the top row projected down onto a single 2D camera view.

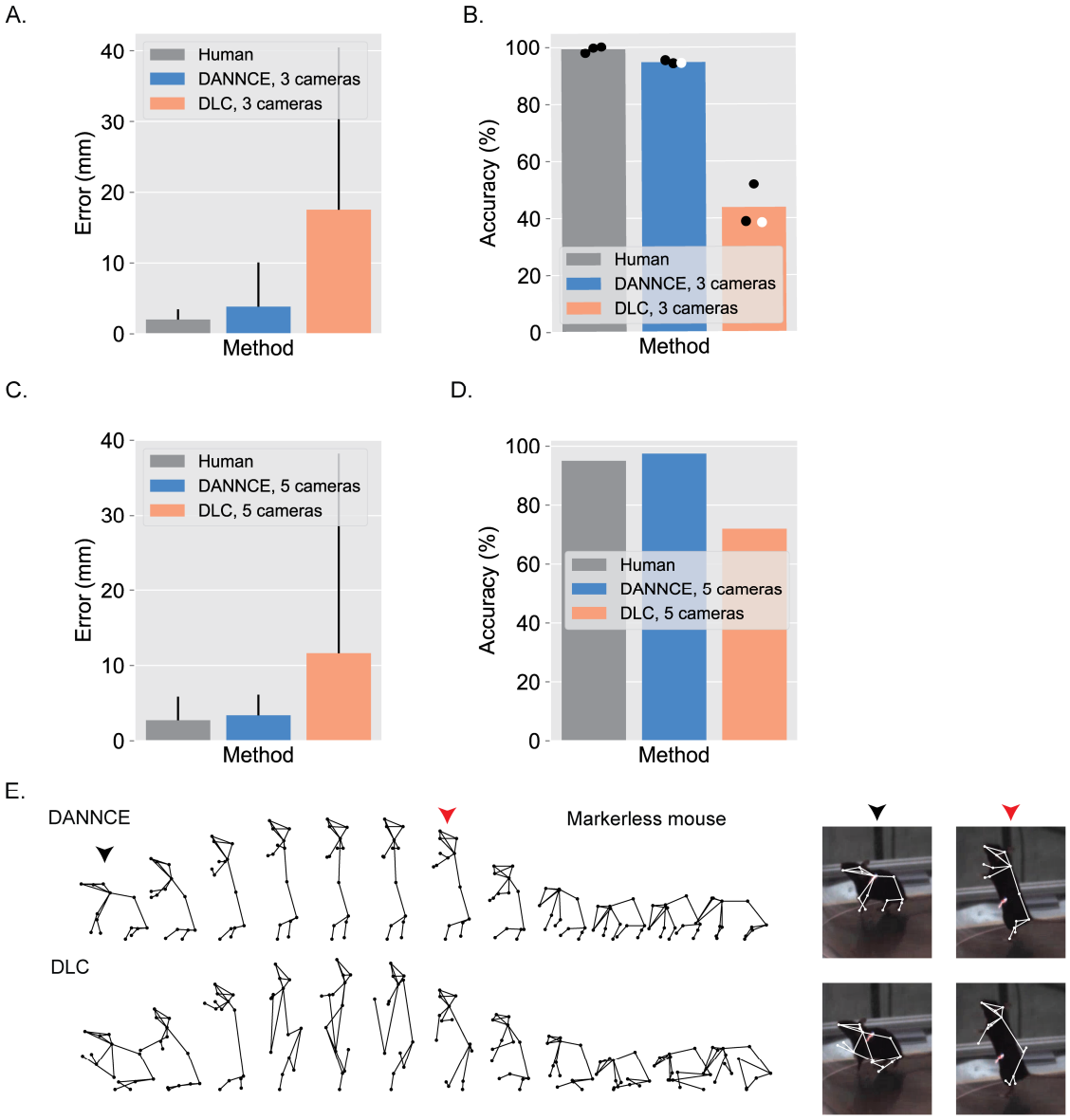
B. Example DLC predictions and projections for two of the frames in (**A**).

C. 3D center of mass position extracted from 1 hour of video recording in a single animal.

D. *Left*, Density map of behavioral space isolated from ~3.5 hours of recording in 3 markerless rats. *Right*, the same behavioral space, segmented into low- and high-level clusters using a watershed transformation. Dark outlines are high-level clusters, light outlines are low-level clusters.

E. 3D renderings of examples from the indicated behavioral categories in (**D**).

Supplementary Figure 11 | DANNCE outperforms DLC in mice.



A. Quantification of mean error relative to triangulated 3D human annotations for DANNCE predictions made using 3 cameras. *N* = 3 animals, *N* = 709 landmarks. Error bars are standard deviation.

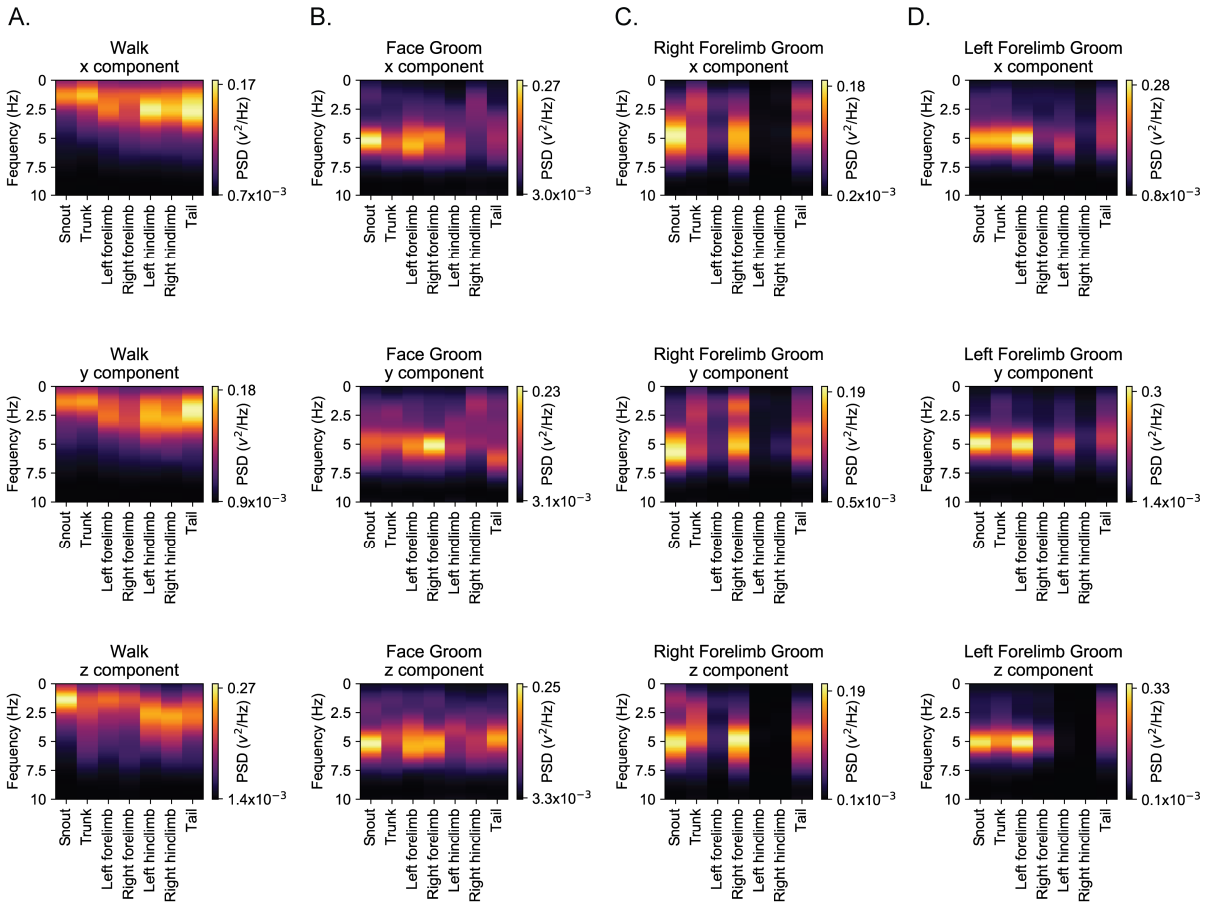
B. Accuracy, as the fraction of landmarks within 9 mm, relative to triangulated landmarks hand-labeled by humans, for the same data and methods as in (**A**). Individual dots show the accuracy for individual animals. Black dots are animals that were used for training, although all samples used for analysis were not used for training. The white dot is a validation animal that was not used for training.

C. Quantification of mean error relative to triangulated 3D hand labels for DANNCE predictions made using 5 cameras in a single animal. *N* = 287 landmarks. Error bars are standard deviation.

D. Accuracy, as in (**B**), for the same data and methods as in (**C**).

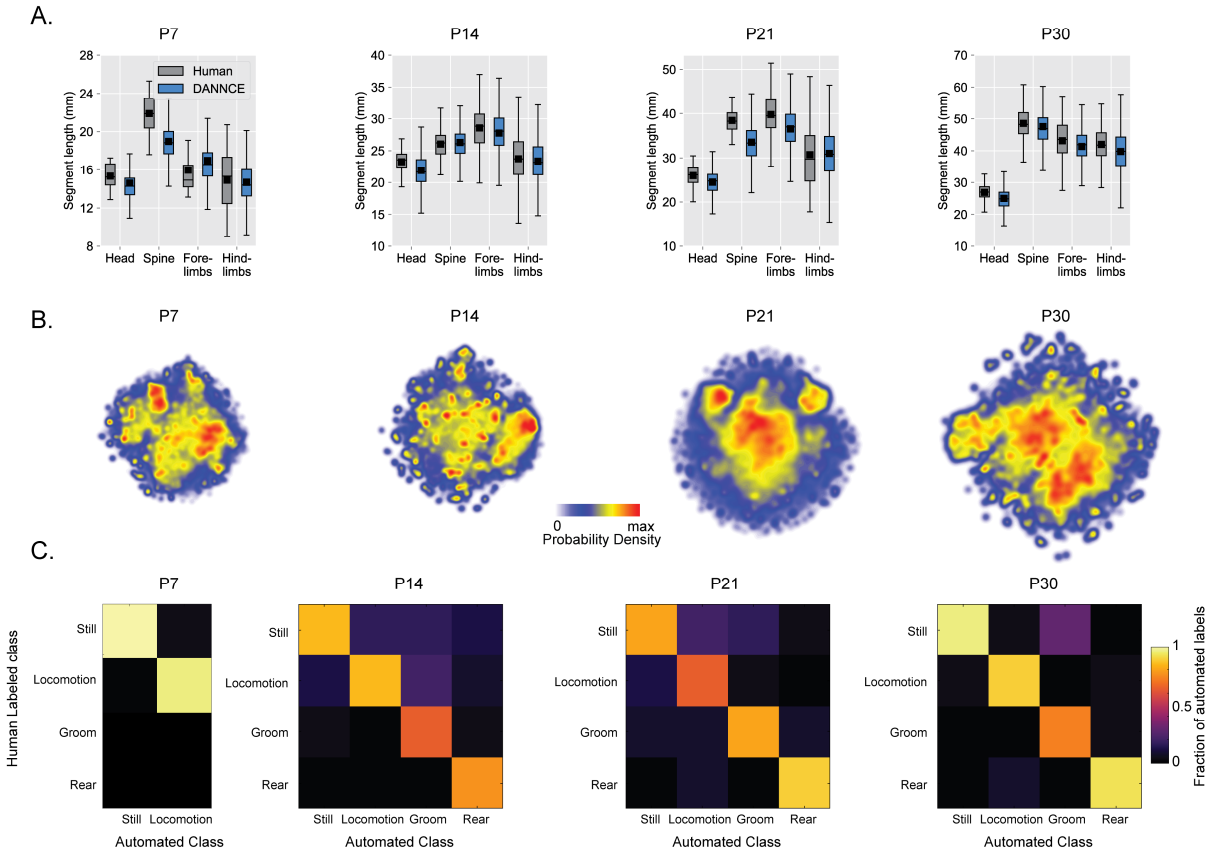
E. Representative example of DANNCE and DLC 3D predictions made using 5 cameras on a rearing sequence. Images to the *right* show reprojections of the samples indicated by the black and red arrowheads.

Supplementary Figure 12 | 3D kinematic analysis of the mouse behavioral repertoire.



A-D. PSDs for individual x-, y-, and z-velocity components for the Walk (**A**), Face Groom (**B**), Right Forelimb Groom (**C**), and Left Forelimb Groom (**D**) clusters in **Fig. 4E-H**. Traces were z-scored individually for each marker before computing PSDs, except for right forelimb (**C**) and left forelimb (**D**) grooming PSDs, in which all limb velocity traces were normalized to the standard deviation of the right and left forelimb traces, respectively.

Supplementary Figure 13 | Validation of the rat pup behavioral clusters discovered by DANNCE.



A. Boxplots of link length distance distributions for both hand-labeled frames and DANNCE predictions. For all developmental timepoints, *N* = 12 animals, *N* = 21–68 segments, hand-labeled, and *N* = 210,612–836,968, DANNCE. Plots show median with IQR and whiskers extending to 1.5x IQR. Black squares are arithmetic means.

B. Density maps that were clustered and annotated to make **Fig. 5D**.

C. Confusion matrices between human-assigned labels and labels derived from DANNCE behavioral mapping in (B), showing 83 ± 11 % agreement between the two labeling methods (mean ± s.d.). Confusion matrices were created from 25 randomly drawn timepoints from each coarse behavioral category by creating videos from [-1,1] s intervals around each timepoint.

Supplementary Note

Hardware and compute benchmarks

In the tables below, we list the system components used for DANNCE on all markerless datasets, along with associated temporal benchmarks. DANNCE runs effectively on many different systems (see hardware specifications per dataset in tables below), making predictions at a maximum of >12 Hz at 64×64×64 voxel resolution on RGB video and >20 Hz at 64×64×64 voxel resolution of monochrome video when using a single GPU. At 32×32×32 voxel resolution, DANNCE achieves a maximum of 44 Hz on RGB video when using a single GPU. DANNCE prediction is also parallelizable, showing linear speed-ups when using additional GPUs (our top speed thus far is 310 Hz for 64×64×64 voxel resolution). Note that Hz is with respect to samples (i.e. timepoints). The total number of video frames processed per second is higher, equal to the sample rate times the number of camera views. DANNCE fine-tuning typically takes less than 12 hours, but it depends on the size of the labeled dataset and the number of epochs run for convergence.

DANNCE configuration parameters

The open-source DANNCE package implements a collection of tunable configuration parameters that can be used to refine DANNCE on new species and datasets. While the default values of these parameters have been set so that DANNCE works “out of the box” on new data, we make the parameters accessible to users as part of a feature-rich system that can in principle be further optimized to address any unexpected new dataset idiosyncrasies. We note that most successful and popular systems, in both software and hardware, provide tunable parameters so that users can maximize the utility of the system via customization.

In this section we define each tunable parameter and describe the expected effect of parameter modifications on DANNCE performance.

Best practices for new users

The lists below can be used when planning hardware purchases and when adapting DANNCE to new environments and species. In general, for new setups, we recommend using at least 3 cameras synchronized with a hardware trigger. Because DANNCE can integrate information from multiple views and landmarks when making predictions, once trained it is robust to decreases in image resolution (Supplementary Fig. 7). We therefore recommend an image resolution that is sufficient to hand label each landmark of interest for fine-tuning.

We recommend running DANNCE on a CUDA-capable GPU with 6 GB memory or more. We provide code for camera calibration, as well as real time compression, on our GitHub page (https://github.com/spoonsso/dannce). DANNCE has been tested and works well on Windows 10, Ubuntu 16.04, and CentOS 7.6. We also recommend at least 16 GB system memory (RAM).

Our parts list for the six-camera mouse setup can serve as a guide for users looking to purchase the necessary components for 3D tracking in a relatively small behavioral space at high frame rates. For larger arenas, additional lighting may be necessary to ensure bright and uniform coverage of the arena. For high frame rate acquisition (100 Hz) at 1 megapixel resolution, an additional GPU with dedicated NVENC capabilities will be needed for real-time compression – we recommend Nvidia Quadro RTX 4000. Slower or lower resolution systems can use the CPU for compression. Our CamPy code repository, which can be accessed from the DANNCE GitHub, provides necessary code for synchronization and real-time video compression for Basler and FLIR cameras.

Mouse setup

Arena: (One) Glass cylinder, 8” diameter × 8” height. Clear acrylic may also be used. We recommend a cylindrical arena to avoid occlusions from the corners of a rectangular prism. Note, for birds, we made the arena large enough to place the cameras inside the arena, thus avoiding this problem. (One) 30” × 30” aluminum breadboard as a base for mounting the setup. (One) PVC sheet for the arena floor. (Five) 1” × 3” × 3/8” mounting bases, for securing the PVC sheet on the breadboard.

Lighting: (Three) Ikan Onyx 240 Bi-Color On-Camera LED light, mounted above the cylinder. 30W (250W equivalent) at full power. IR LEDs can be used for light-sensitive applications. Lights were set to 100% power, camera irises were closed to maximize pixel intensities on the floor just below saturation, and cameras were set to 2 ms exposure time. These values were chosen to maximize contrast and minimize motion blur.

Cameras: (Six) Basler acA1920-150uc, 2/3”, C-mount, 1920×1200, 150 fps, color, USB 3.0, CMOS, global shutter. (Six) 10 meter GP-I/O cables for USB 3.0 cameras. (Six) 3 meter Micro B USB 3.0 cables. (One) PCIe-to-USB 3.0 expansion card (4 ports, 4 host controllers). To achieve 100 Hz frame rate, camera pixel format was set to 8-bit RGB, and resolution was reduced to 1152×1024 pixels.

Lenses and filters: (Six) 8 mm, 2/3”, 5 MP, C-Mount, Ultra Low Distortion lenses. (Six) 30.5 mm Ultra High Contrast Linear Polarizer Filter. Note that the filters reduce glare but are not absolutely necessary.

Camera Mounting hardware: (Six) 1/4-20" Aluminum Mounting Plate. (Six) Rotating clamps for ½” posts, continuously adjustable, 3/16” Hex (ThorLabs). Cameras were mounted onto ½” posts (ThorLabs) at either 90° or ~60° and positioned to achieve even coverage of views around the arena.

We recommend training the COM network and DANNCE with at least 100 samples (the same labels can be used to train both) labeled using Label3D. DANNCE approximates human performance with around 200 samples, but there can be good returns on labeling up to at least 1000 samples (Supplementary Fig. 8). Users should choose the number of points to track based on the requirements of their animal and/or experiment, which generally should capture all independently controllable skeletal segments. Because DANNCE learns geometric relationships between landmarks on the animal skeleton, we expect prediction accuracy to increase with additional labeled points. This intuition is supported by evidence from the human pose literature that prediction accuracy increases with the number of tracked (i.e. labeled) landmarks (Fisch and Clark, 2020, arxiv).

The only parameter that absolutely requires customization before using DANNCE on a new species is **vol_size**, which sets the physical dimensions of the 3D volume encapsulating the subject in the unprojected 3D space. **vol_size** must be large enough to encompass the entire subject and thus should be fit to the actual physical size of the animal.

Hyperparameter (Primary)	Definition	Guidance
vol_size	The length (in mm) of one size of the 3D cube whose center is the animal's 3D position.	This should be big enough to fit the entire animal, with a little wiggle room to accommodate noise in the 3D COM.
net_type	Possible values: `AVG` or `MAX`. This parameter toggles between two different versions of DANNCE. Default: `AVG`.	The `AVG` version can achieve better accuracy but is sometimes harder to train. We recommend starting with `AVG`.
Hyperparameter (Secondary)	**Definition**	**Guidance**
epochs	This is the number of times the full dataset will be looped over during training. Default: 600.	You should need fewer epochs as you increase the number of images in your training dataset. With ~100 timepoints in the training set, 500-1000 epochs is probably sufficient. Inspect your training and validation loss in your tensorboard logs. If using a validation set, you should continue to train if the validation loss has not yet plateaued.
batch_size	This is the number of timepoints that will be included in a batch of samples during training and prediction. This value is constrained by the amount of GPU memory you can access. Default: 4.	For prediction, increasing `batch_size` may increase prediction speed. For training, `batch_size` can be modified as a hyperparameter. The effect of batch size on training is an open area of research in deep learning, although there is agreement that the optimal batch size is problem- and data-specific.
interp	Possible values: `nearest` or `linear`. Sets the interpolation method when building 3D volumes from 2D images. Default: `nearest`.	Nearest neighbor interpolation is faster, but bilinear interpolation can in principle be more accurate. Note that it is best to use the same interpolation method for both training and prediction.
medfilt_window	If not `None`, the 3D COM trace will be smoothed with a median filter with size indicated here. Default: `None`.	Inspect your COM traces after they are generated by the COMnet. You want them to look smooth, without any outliers. If outliers or jumps are sparse, you can try to smooth them away by introducing this

		median filter. The larger the window, the more likely your COM will become inaccurate and not capture the entire animal in the DANNCE 3D volume.
loss	The loss function used for training. Default: `mse_mask`.	DANNCE ships with a small set of built-in loss functions for training the network, in addition to default loss functions provided by keras and tensorflow. We always use our custom `mse_mask`. This is just a mean squared error loss that ignores missing landmarks in labeled samples (from an inability to see the landmark during data labeling).
sigma	When using the MAX network, this value sets the size of the spherical Gaussians (the standard deviation for all 3 dimensions, in mm) used as training targets for each landmark. Default: 10.	This should scale with the size of your species (and volume size) and should approximate the deviation of your labeled landmark positions from the true anatomical position of the landmark.
lr	The learning rate used by the optimizer during training. Default: 0.001.	The learning rate is often included in hyperparameter searches. If not performing a systematic search, you might still try a few different learning rates to see if this improves performance. We do not recommend learning rates larger than 0.01.
n_layers_locked	The number of layers in the pretrained model (including the input layer) whose weights are locked and will not update during training. Default: 2.	After training on a large database, early layers in a CNN have learned to capture features universal to all images (such as edges) and thus do not need to be fine-tuned for another domain. The rule of thumb is that the more data you are using for fine-tuning, the more layers you should unlock. However, in our experiments we find that locking just the first convolutional layer works well,

		even when fine-tuning with just 50 samples.
rotate	Either `True` or `False`. When `True`, during training image volumes are rotated randomly around the z-axis in 90 degree steps. (Default: `True`)	Image augmentation is a commonly used technique for improving neural network performance. For rotation, this generates synthetic examples that increase the diversity of viewpoints in the training data.
augment_brightness	Either `True` or `False`. When `True`, during training image volume brightness is randomly scaled. (Default: `False`)	Brightness augmentation can improve performance and generates synthetic training examples of samples under diverse lighting conditions or exposures.
augment_hue	Either `True` or `False`. When `True`, during training image volume brightness is randomly scaled. (Default: `False`)	Hue augmentation can improve performance and generates synthetic training examples of samples under diverse lighting conditions or differences in camera color balance.
channel_combo	Either `None` or `random`. If random, camera order is shuffled as batches are generated. (Default: `None`)	When the first layers of the CNN are locked during fine-tuning, this parameter has no practical effect. When the first layers are unlocked, setting to `random` makes the network robust to different view configurations.

DANNCE

	Rat7M FT Fig. 3	Mice Fig. 4	Mice Fig. S11A	Mice Fig, S11C	Chickadee Fig. 6	Marmoset Fig. 6
Primary Hyperparameters						
vol_size	240 mm	120 mm	120 mm	120 mm	84 mm	600 mm
net_type	AVG	MAX	MAX	MAX	AVG	MAX
Secondary Hyperparameters						
nvox	64	64	64	64	64	64
lr	0.001	0.001	0.001	0.001	0.001	0.001
loss	mse_mask	mse_mask	mse_mask	mse_mask	mse_mask	mse_mask
epochs	600	1200	1200	817	2000	1696
channel_combo	random	random	random	random	None	random
interp	nearest	nearest	nearest	nearest	nearest	nearest
sigma	n/a	10	10	10	n/a	20
n_layers_locked	2	2	2	2	0	2
batch_size	4	4	4	4	4	4
medfilt_window	None	None	None	None	None	10
augment_brightness	False	False	False	False	True	False
augment_hue	False	False	False	False	False	False
rotate	True	True	True	True	True	True
Performance and Instrumentation						
camera type	FLIR	Basler	FLIR	FLIR	FLIR	FLIR
video size	1280 x 1024	1152 x 1024	1280 x 1024	1280 x 1024	2816 x 1408 or 2816 x 1696	1280 x 1024
animal length (pixels)	170-340	200-400	250-400	150-300	100-500	150-650
mono	False	False	False	False	True	False
acquisition fps	30	100	30	30	60	30
operating system	Ubuntu	Windows	Ubuntu	Ubuntu	Ubuntu & Windows 10	Ubuntu
GPU Type	Titan V 12 GB	Titan RTX 24 GB	Titan V 12 GB	Titan V 12 GB	2080ti 12GB	Titan V 12 GB
training duration	11 hours 6	120 hours	7 hours	10 hours	24hrs	17 hours
prediction speed, 1 GPU	3 cameras: 12.4 Hz 6 cameras: 9.8 Hz	11.1 Hz	12.4 Hz	10.2 Hz	10 Hz	12.4 Hz
no. of views	{3, 6}	6	3	5	6	3
no. of training samples (per subject)	225	82	25	50	70	100

no. training subjects	1	2	2	1	5	1
data rate (GB per camera per hour)	0.3	30 GB	0.3	0.3	15 GB	672 GB

DANNCE Pups

	P7	P14	P21	P30
Primary Hyperparameters				
vol_size	160	160	160	400
net_type	AVG	AVG	AVG	AVG
Secondary Hyperparameters				
nvox	64	64	64	64
lr	0.001	0.001	0.001	0.001
epochs	500	500	500	500
channel_combo	None	None	None	None
interp	linear	linear	linear	nearest
sigma	n/a	n/a	n/a	n/a
n_layers_locked	2	2	2	2
batch_size	4	4	4	4
medfilt_window	30	30	30	30
augment_brightness	True	True	True	False
augment_hue	True	True	True	False
rotate	True	True	True	True
Performance and Instrumentation				
camera type	FLIR	FLIR	FLIR	FLIR
video size	1280 x 1024	1280 x 1024	1280 x 1024	1280 x 1024
animal length (pixels)	170-430	220-610	300-540	360-530
mono	False	False	False	False
operating system	CentOS 7.6	CentOS 7.6	CentOS 7.6	CentOS 7.6
acquisition fps	30	30	30	30
GPU Type	V100 16 GB	V100 16 GB	V100 16 GB	V100 16 GB
training duration	4.7 hrs	22.5 hrs	31.9 hrs	9.2 hrs
prediction speed, 1 GPU	12.4 Hz	12.4 Hz	12.4 Hz	12.4 Hz
prediction speed, Max (no. GPUs)	310 Hz (25)	310 Hz (25)	310 Hz (25)	310 Hz (25)
no. of views	3	3	3	3
no. of training samples (per subject)	30	25	96	35
no. training animals	6	6	6	6
Additional or shared samples	n/a	10 P13, 30 P15, 60 P20, 575 P21	150 P14, 60 P20, 60 P22	105 P40
data rate (per camera per hour)	0.15	0.15	0.15	0.15

DANNCE parameters or properties tested before finalizing models. *Best value is italicized.*

Rat 7M fine-tune (FT). None.

6 camera mouse. Number of training epochs {*1200*, 12000}.

5 camera mouse. sigma {5, *10*}. channel_combo {*None*, random}.

3 camera mouse. None.

Marmoset. Number of training samples {50, *100*}. vol_size {200 mm, *300 mm*}. medfilt_window {0, *10*}. net_type {*MAX*, AVG}. Number of training epochs {*1696*, 2400}.

Bird. vol_size {53, 64, 74, *84 mm*}, n_layers_locked {2, *0*}.

Pups, all ages. net_type {MAX, *AVG*}. nvox {*64*, 80, 96}. lr {5e-5, *1e-3*}. interp {*linear*, nearest}.

Pups, P7. No additional.

Pups, P14. Added additional shared training data from nearby developmental timepoints.

Pups, P21. Added additional shared training data from nearby developmental timepoints.

Pups, P30. vol_size {240, 340, *360,* 400 mm}. Added additional shared training data from nearby developmental timepoints.

COMnet configuration parameters

To anchor the 3D volumes for DANNCE, we provide a training and prediction environment for a 2D CNN that tracks only the rough position (center of mass – COM) of the subject. The COM network also has associated parameters that can be changed by the user. Note that use of the COMnet is not required. If you have good estimates of the animal position via an alternative approach (a different 2D CNN, or a classic computer vision technique), then these estimates can then be plugged into DANNCE. For instance, in bird the COM was tracked with DeepPoseKit.

We consider all the COMnet hyperparameters to be secondary because the COMnet should work well with default values, given enough training data.

Hyperparameter (Secondary)	Definition	Guidance
sigma	This value sets the size of the spherical Gaussians (the standard deviation for all 2 image dimensions, in pixels) used as training targets for the COM. Default: 30.	This should scale with the size of your species (and image size), and should approximate the deviation of your labeled COM positions.
lr	The learning rate used by the optimizer during training. Default: 5e-5.	The learning rate is often included in hyperparameter searches. If not performing a systematic search, you might still try a few different learning rates to see if this improves performance. We do not recommend learning rates larger than 0.01.
epochs	This is the number of times the full dataset will be looped over during training. Default: 100.	You should need fewer epochs as you increase the number of images in your training dataset. With ~100 timepoints in the training set, 100-200 epochs is probably sufficient. Inspect your training and validation loss in your tensorboard logs. If using a validation set, you should continue to train if the validation loss has not yet plateaued.
downfac	This is the factor by which your images will be downsampled before getting passed through the network. Default: 4.	The more you downsample your images (the higher downfac), the faster training and prediction will be. Because you only need a coarse estimate of the animal's position from the COMnet, downsampling does

		not typically degrade performance noticeably
batch_size	This is the number of frames that will be included in a batch of samples during training and prediction. This value is constrained by the amount of GPU memory you can access. Default: 6.	For prediction, increasing `batch_size` may increase prediction speed. For training, `batch_size` can be modified as a hyperparameter. The effect of batch size on training is an open area of research in deep learning, although there is agreement that the optimal batch size is problem- and data-specific. For the COMnet, increases in the `downfac` amount increase the achievable batch size by creating smaller image dimensions. In the same way, increases and decreases in native image resolution also affect the batch size.
augment_brightness, augment_hue, augment_rotation, augment_shear, augment_zoom, augment_shift	We also provide a standard set of 2D image augmentations, as implemented by tensorflow.	Image augmentation is a commonly used technique for improving neural network performance. With a very small labeled dataset, turning on these augmentations could increase COMnet performance.

COMnet

	Rat7M FT Fig. 3	**Mice 6 cam Fig. 4**	**Mice 3 cam Fig. S11A**	**Mice 5 cam Fig. S11C**	**Chickadee Fig. 6**	**Marmoset Fig. 6**
Secondary Hyperparameters						
downfac	4	2	2	2	n/a	4
sigma	10	30	10	10	n/a	20
batch_size	6	2	6	6	n/a	6
lr	5e-5	5e-5	5e-5	5e-5	n/a	5e-5
epochs	30	250	100	55	n/a	1000
augment_brightness	False	False	False	False	n/a	False
augment_hue	False	False	False	False	n/a	False
augment_rotation	False	False	False	False	n/a	False
augment_shear	False	False	False	False	n/a	False
augment_zoom	False	False	False	False	n/a	False

augment_shift	False	False	False	False	n/a	False
Performance and Instrumentation						
training samples added to DANNCE labels (total)	12,946 bootstrapped	0	0	311 bootstrapped	n/a	300 COM-only
training duration	6 hours	1 hour	1 hour	1.5 hours	n/a	8 hours
GPU Type	Titan V	Titan RTX	Titan V	Titan V	n/a	Titan V
operating system	Ubuntu	Windows	Ubuntu	Ubuntu	n/a	Ubuntu
prediction speed, 1 GPU	3 cameras 27 Hz: 6 cameras: 18 Hz	8 Hz	27 Hz	8 Hz	n/a	27 Hz

COMnet Pups

	P7	**P14**	**P21**	**P30**
downfac	2	2	2	2
sigma	18	18	18	18
batch_size	2	2	2	2
lr	5e-5	5e-5	5e-5	5e-5
epochs	200	200	200	200
augment_brightness	False	False	False	False
augment_hue	False	False	False	False
augment_rotation	False	False	False	False
augment_shear	False	False	False	False
augment_zoom	False	False	False	False
augment_shift	False	False	False	False
Performance and Instrumentation				
COM-only training samples added to DANNCE labels	1000 P7	0 P14, 50 P13, 150 P15	450 P21, 150 P22	0 P30
additional COM-only shared samples	0	150 P20, 450 P21	150 P20	0
GPU Type	V100	V100	V100	V100
operating system	CentOS 7.6	CentOS 7.6	CentOS 7.6	CentOS 7.6
prediction speed, 1 GPU	5.7 Hz	5.7 Hz	5.7 Hz	5.7 Hz
prediction speed, Max (No. GPUs)	142.5 Hz (25)	142.5 Hz (25)	142.5 Hz (25)	142.5 Hz (25)

COMnet parameters or properties tested before finalizing models. *Best value is italicized.*

Rat 7M fine-tune (FT). Bootstrapped training samples (see *Methods*).

6 camera mouse**.** None.

5 camera mouse. Bootstrapped training samples (see *Methods*).

3 camera mouse. None.

Marmoset. Added additional COM-only labeled training samples {300}. sigma {10, *20*}.

Pups. Added additional COM-only labeled training samples, except for P30.