Machine Learning Methods for Neural Data Analysis

Markerless Pose Tracking

Scott Linderman

STATS 220/320 (NBIO220, CS339N). Winter 2023.

“The brain is worthy of study because it is in charge of behavior”

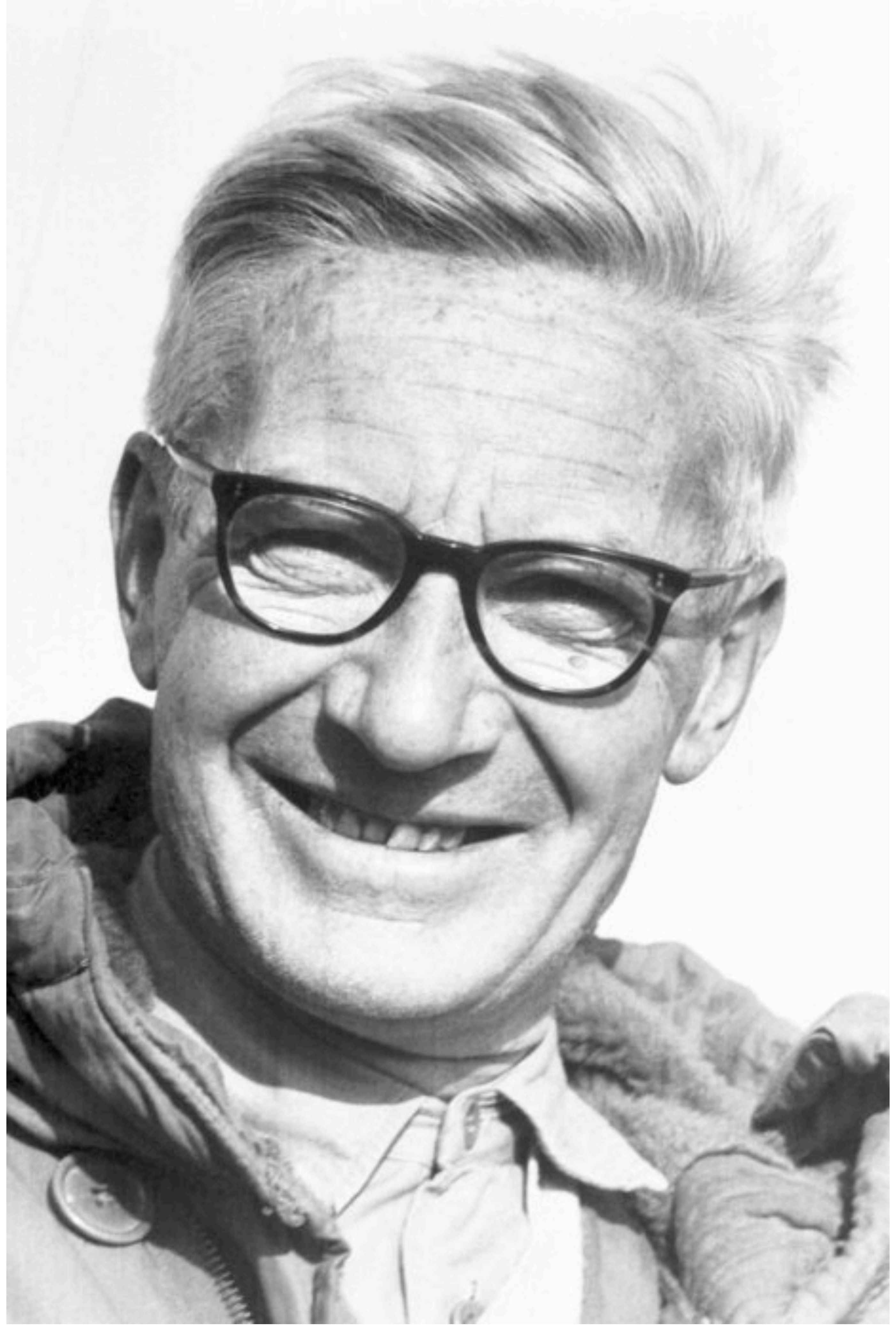
Datta, Anderson, Branson, Perona, and Leifer. Computational Neuroethology: A Call to Action. *Neuron* 2019.

Burgess et al (*Cell Rep.*, 2017)

Ethology

The study of (natural) behavior

- **Hypothesis:** “exposing the structure of behavior…will yield insights into how the brain creates behavior.” Datta et al.
- **Structure:** how behavior in the natural environment is built from components and organized over time in response to ecologically relevant stimuli.
- **Natural behavior:**
    - Exploring new environments
    - Foraging for food
    - Finding shelter
    - Identifying mates
    - …



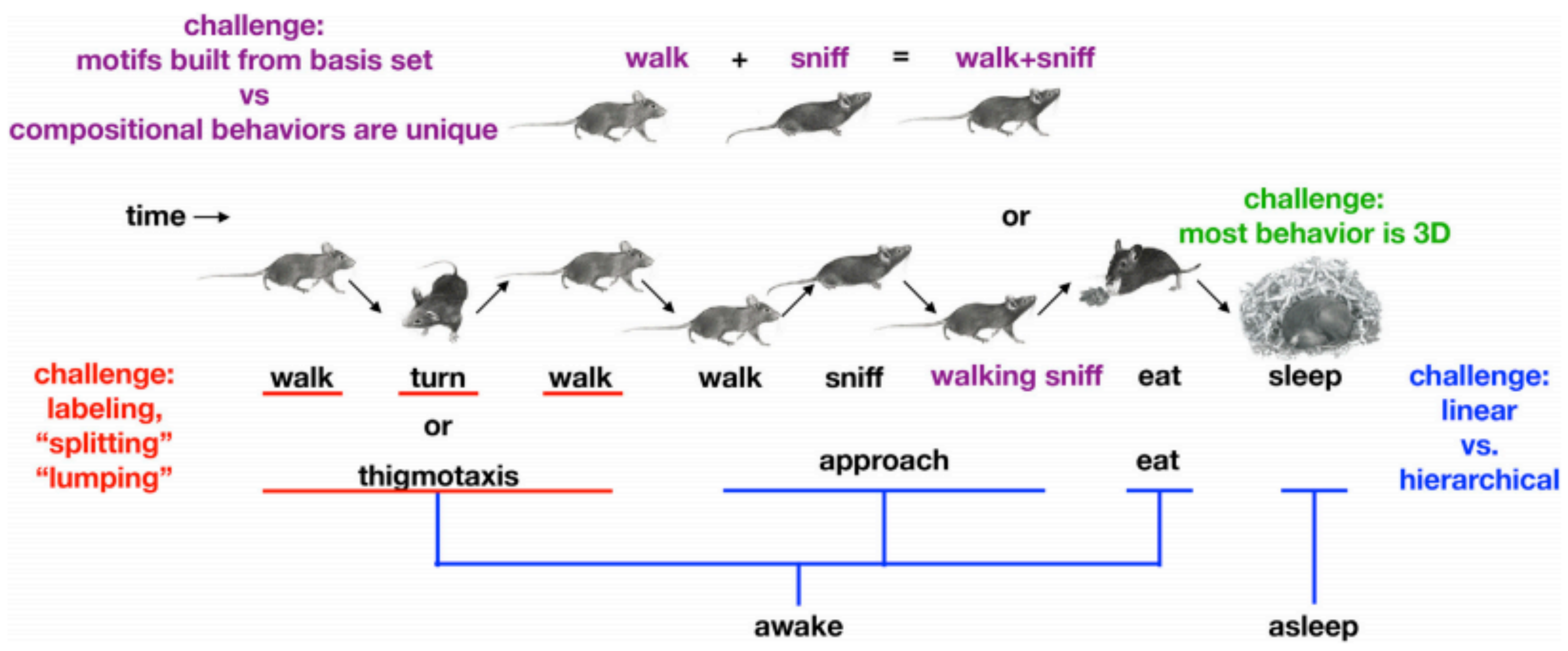
Nikolaas Tinbergen
Nobel Prize in Physiology or Medicine 1973

Yilmaz and Meister (*Curr. Bio.*, 2013)

Computational (Neuro)Ethology

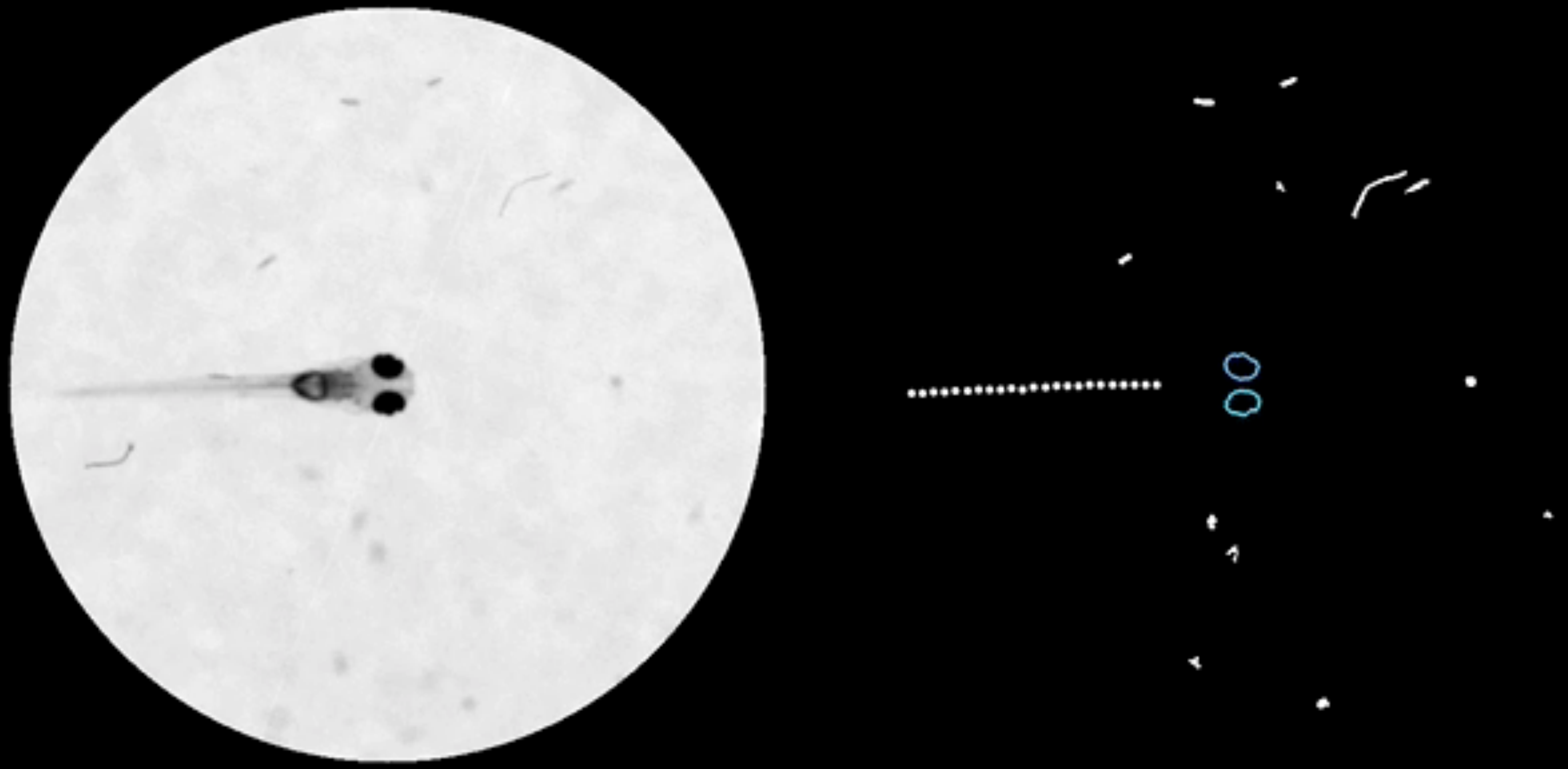
Quantifying natural behavior (and relating it to neural activity)

- Leveraging advances in **computer vision** and **machine learning** to extract behavioral features of interest from raw data.
- Modeling the dynamics of 3D pose as a function of sensory input and internal state.
- Decomposing behavior into stereotyped components and behavioral motifs.
- Correlating behavioral motifs with large scale neural recordings.
- Identifying causal relationships between neural activity and motor output.



Datta et al (*Neuron*, 2019)

Johnson et al (*Curr. Bio.* 2020)

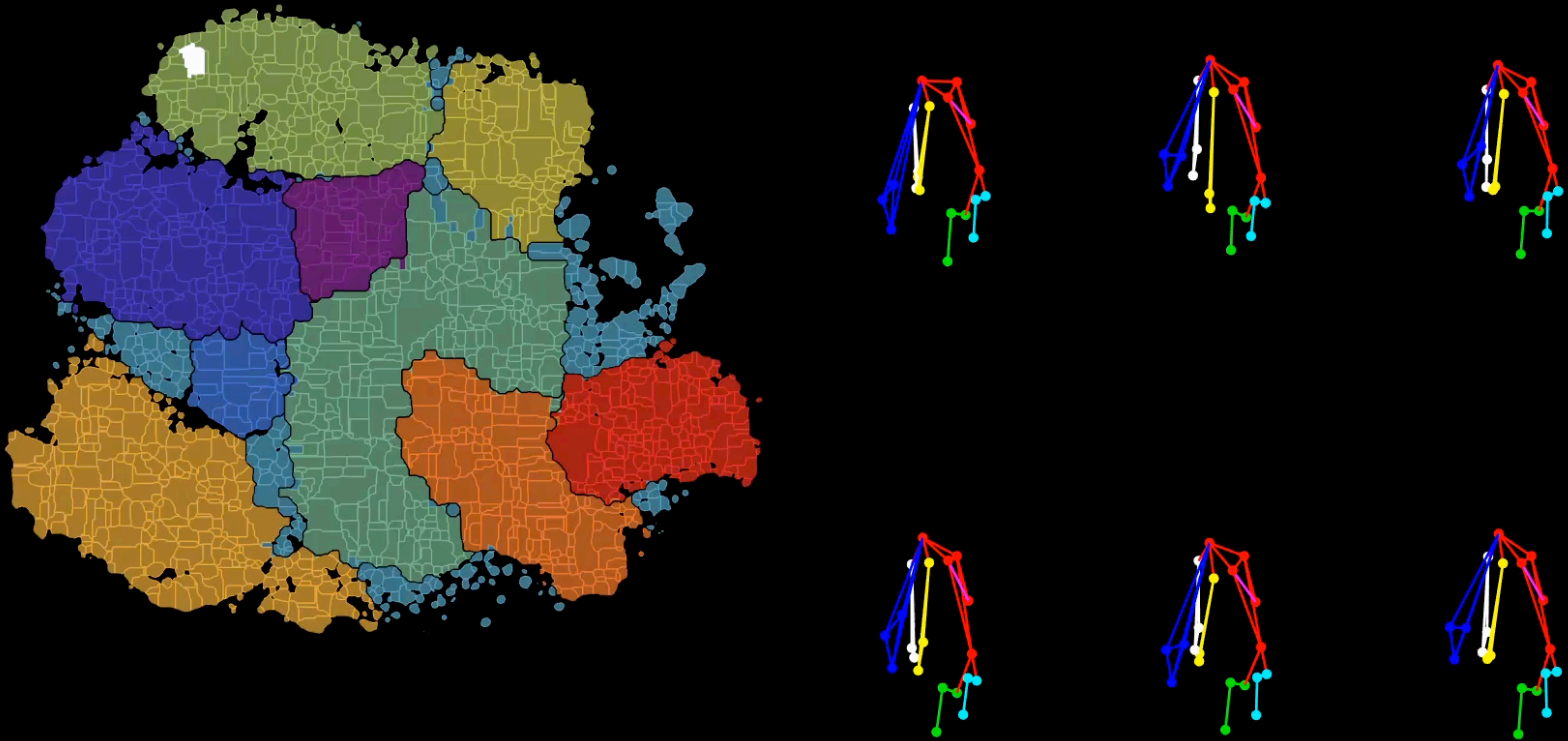


Johnson et al (*Curr. Bio.* 2020)

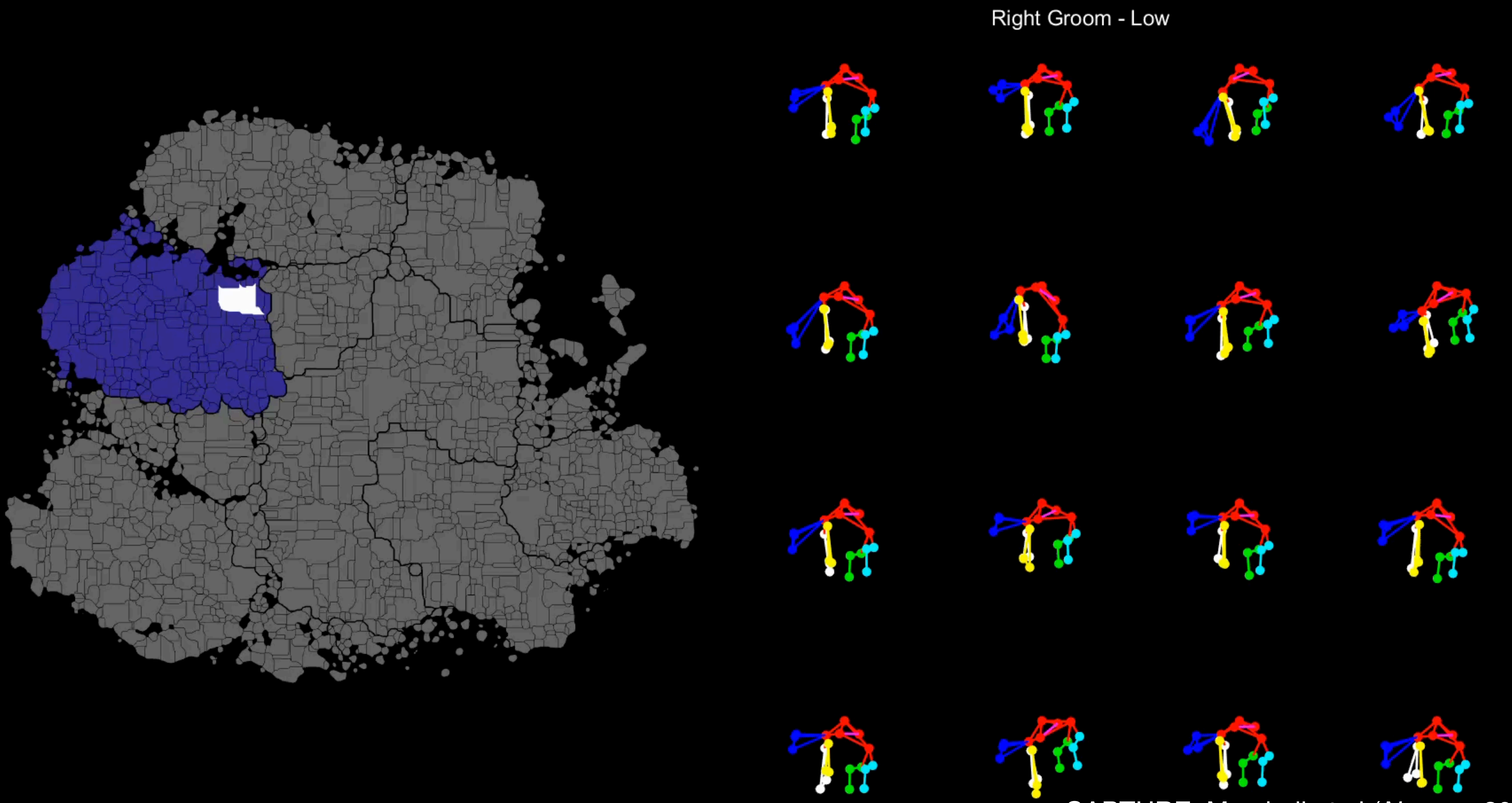
CAPTURE: Marshall et al (*Neuron*, 2020)

CAPTURE: Marshall et al (*Neuron*, 2020)

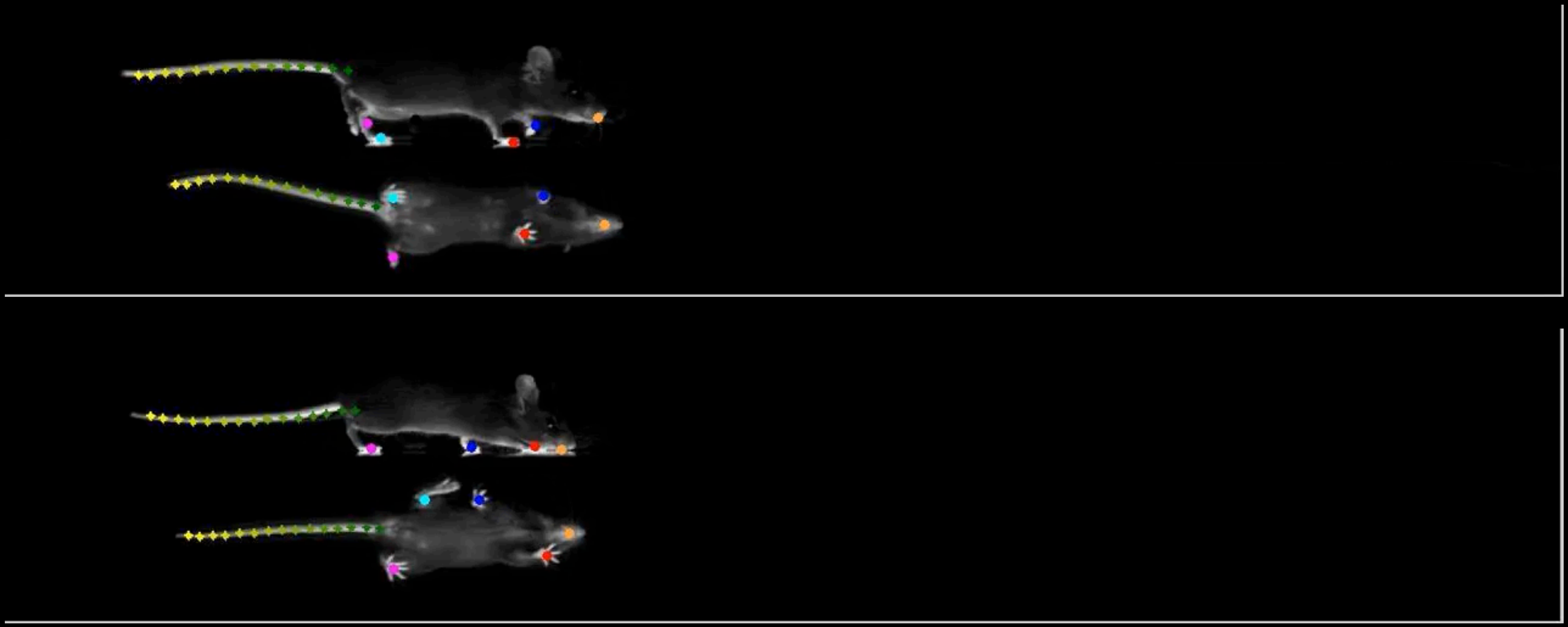
Left Groom



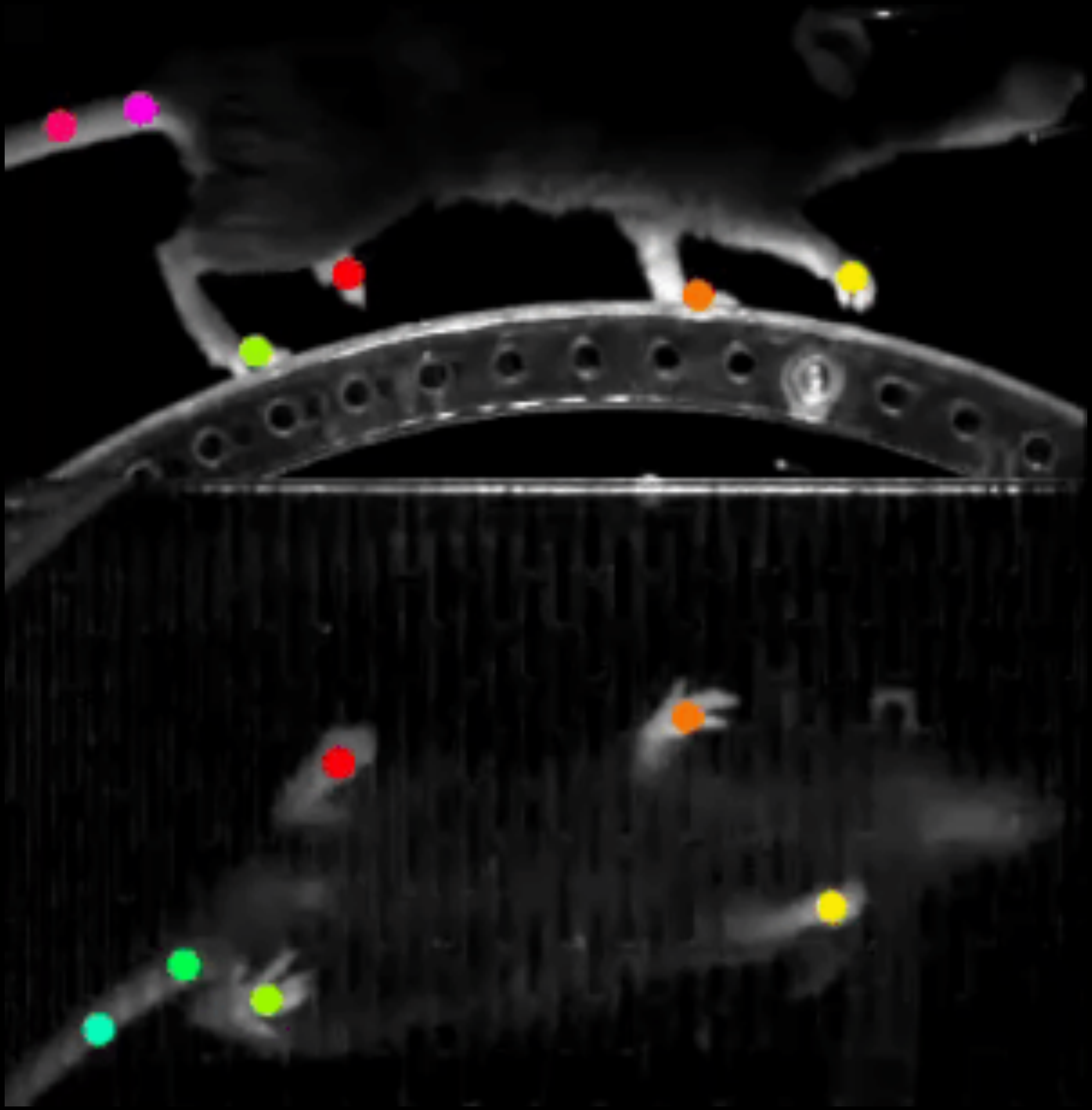
CAPTURE: Marshall et al (*Neuron*, 2020)



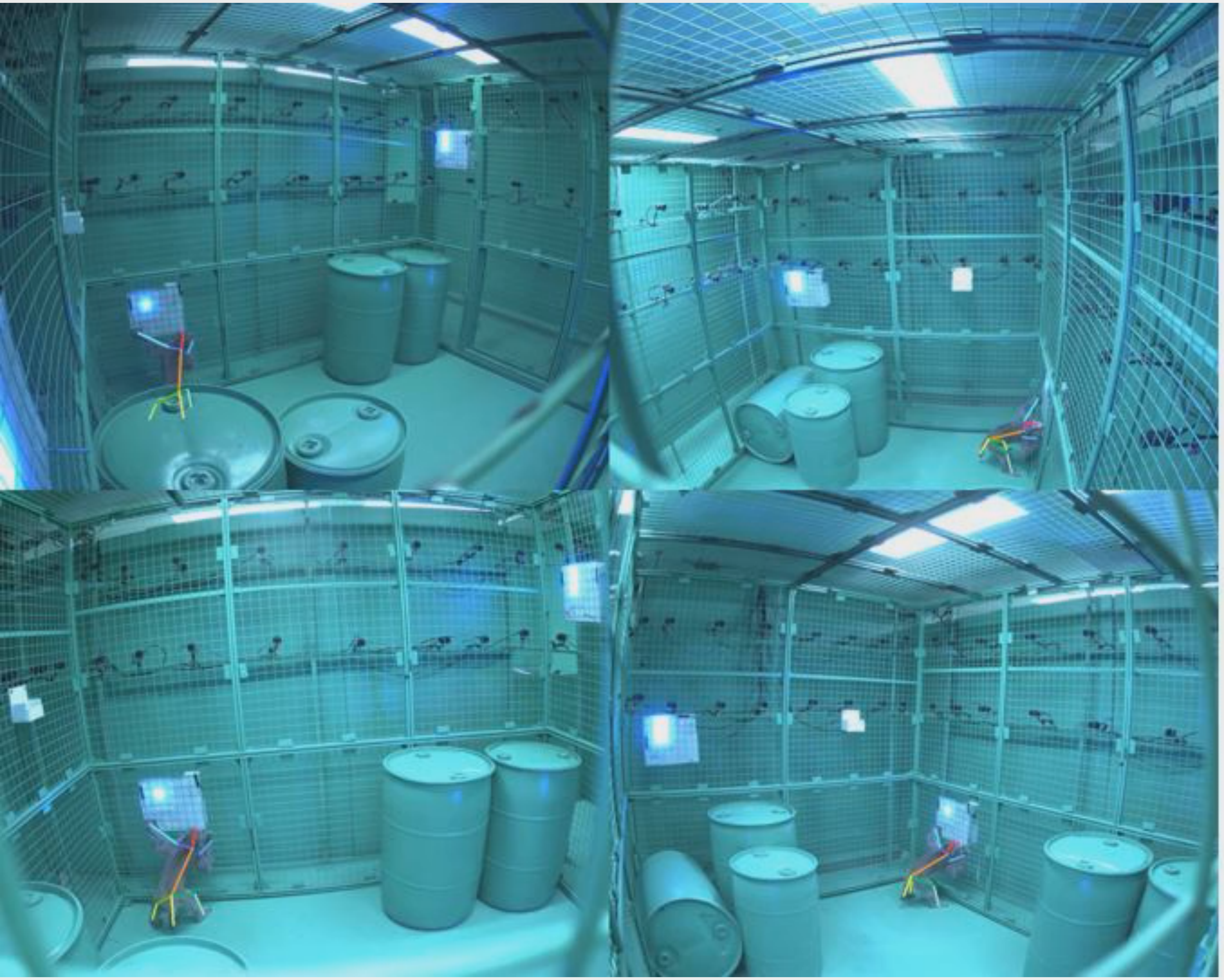
CAPTURE: Marshall et al (*Neuron*, 2020)

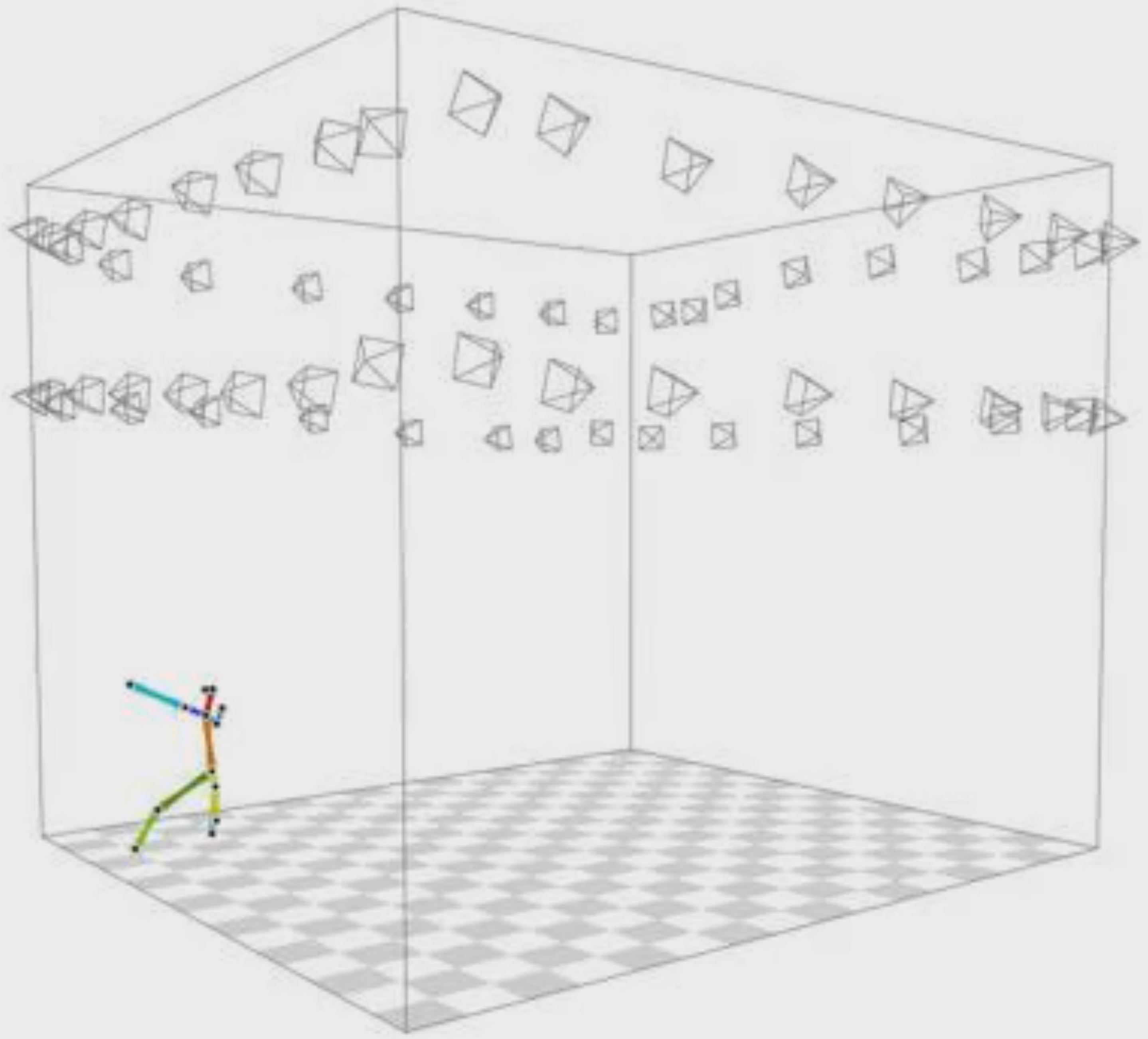


Machado et al (*eLife.* 2015)



DeepLabCut: Mathis et al. (*Nat Neuro* 2018)





OpenMonkeyStudio: Bala et al (*Nature Comm.*, 2020

DeepFly3D: Gunel et al (*eLife*, 2019)

SLEAP: Pereira et al (2021)

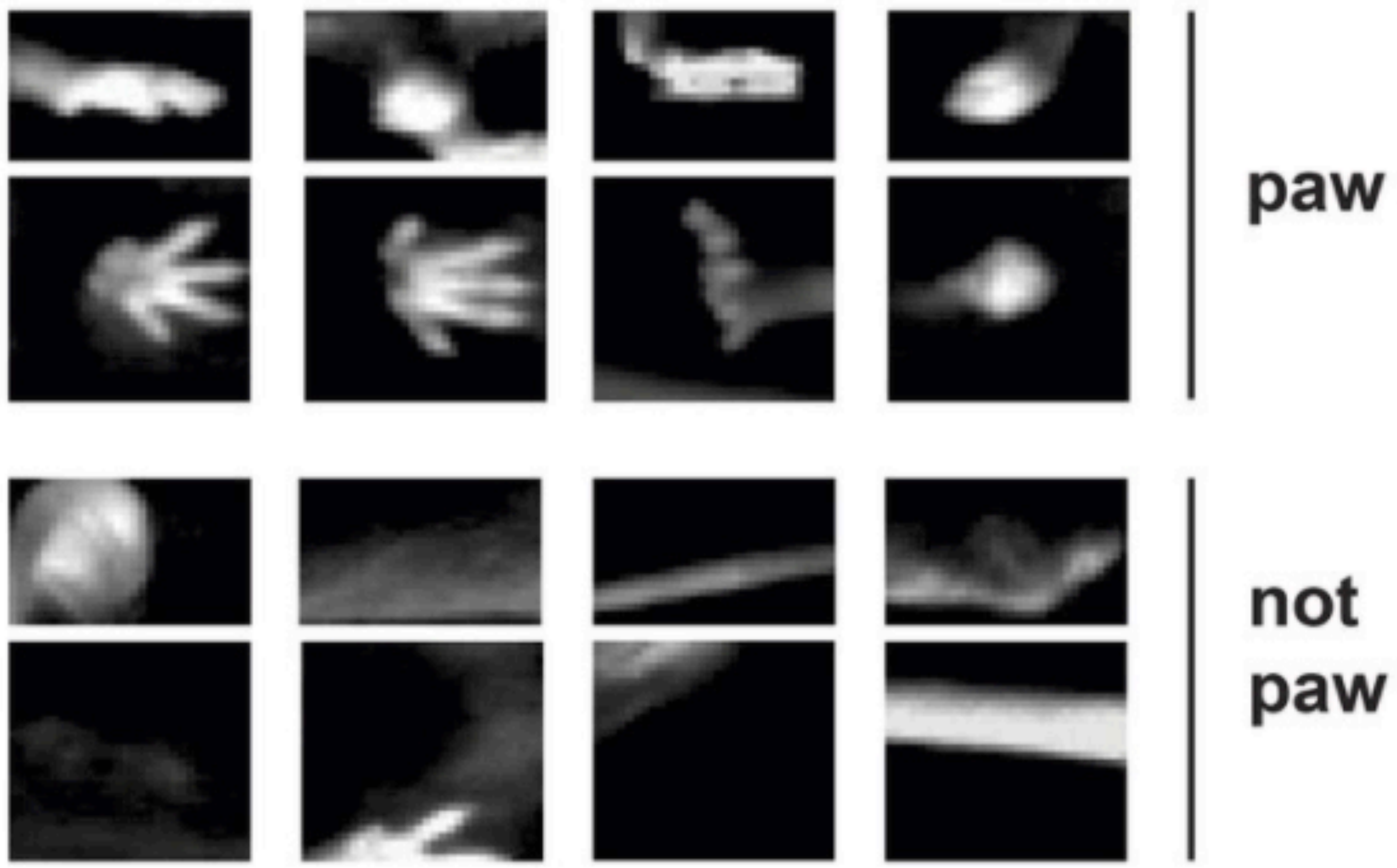
Agenda

1. Basics of markerless pose tracking
2. Pose tracking with CNNs
3. Structured prediction and triangulation

Basic pose tracking

Turn it into a supervised learning problem

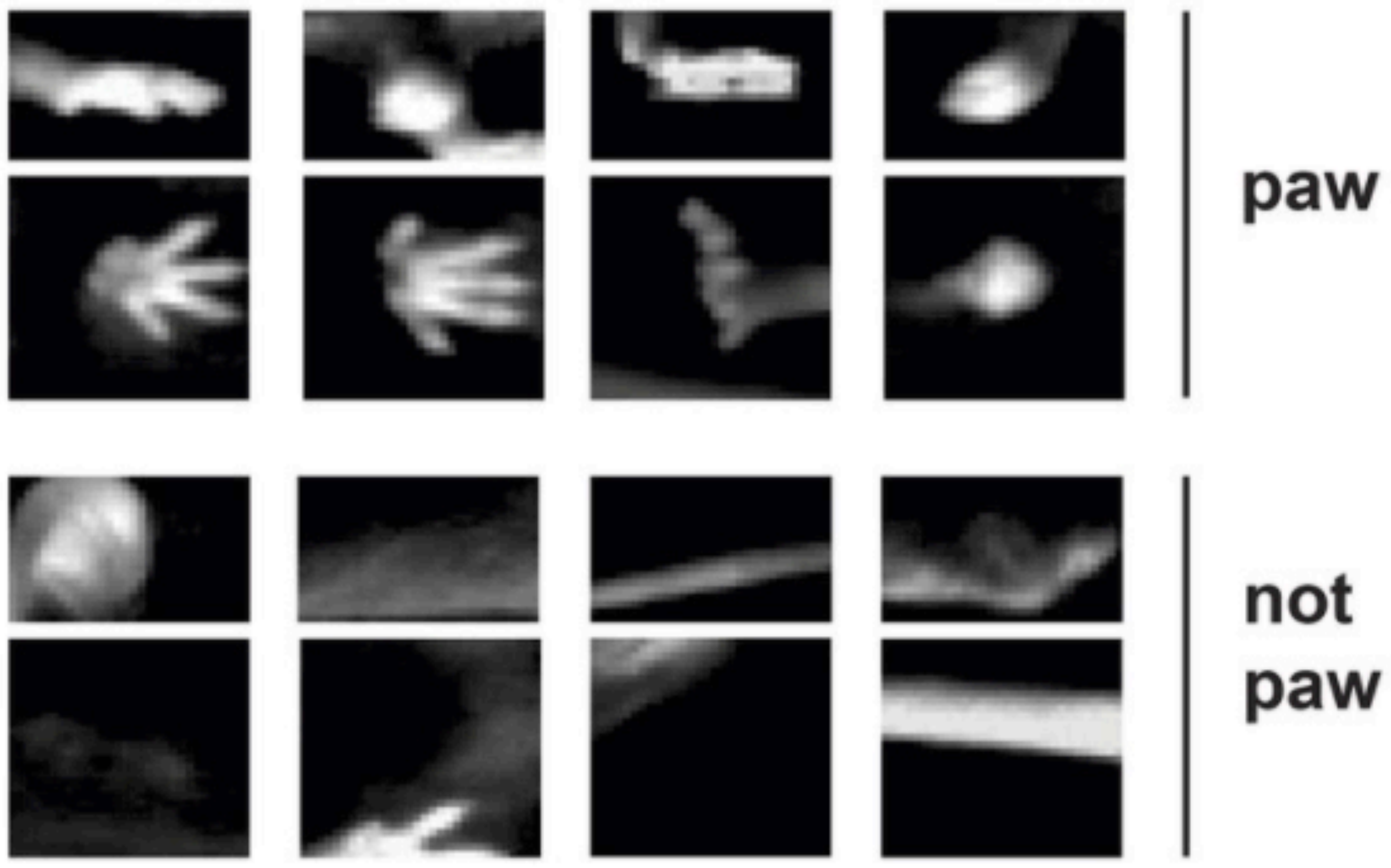
- Extract patches from the video frames and label them as positive or negative examples of a key point (e.g. paw).
- Train a binary classifier (logistic regression, SVM, neural network, etc.) to predict key point or not.
- At test time, classify each patch in the image and then pick the most likely keypoint location(s). (More on how later.)



Basic pose tracking

Mathematical formulation

- Let P_h and P_w be the height and width, respectively, of the patch (in pixels).
- N denote the number of patches
- $\mathbf{x}_n \in \mathbb{R}^{P_h \cdot P_w}$ denote the n-th patch.
- $y_n \in \{0,1\}$ denote whether or not the patch is an instance of the key point.
- $\mathbf{w} \in \mathbb{R}^{P_h \cdot P_w}$ denote the weights of our model.



Basic pose tracking

Via logistic regression

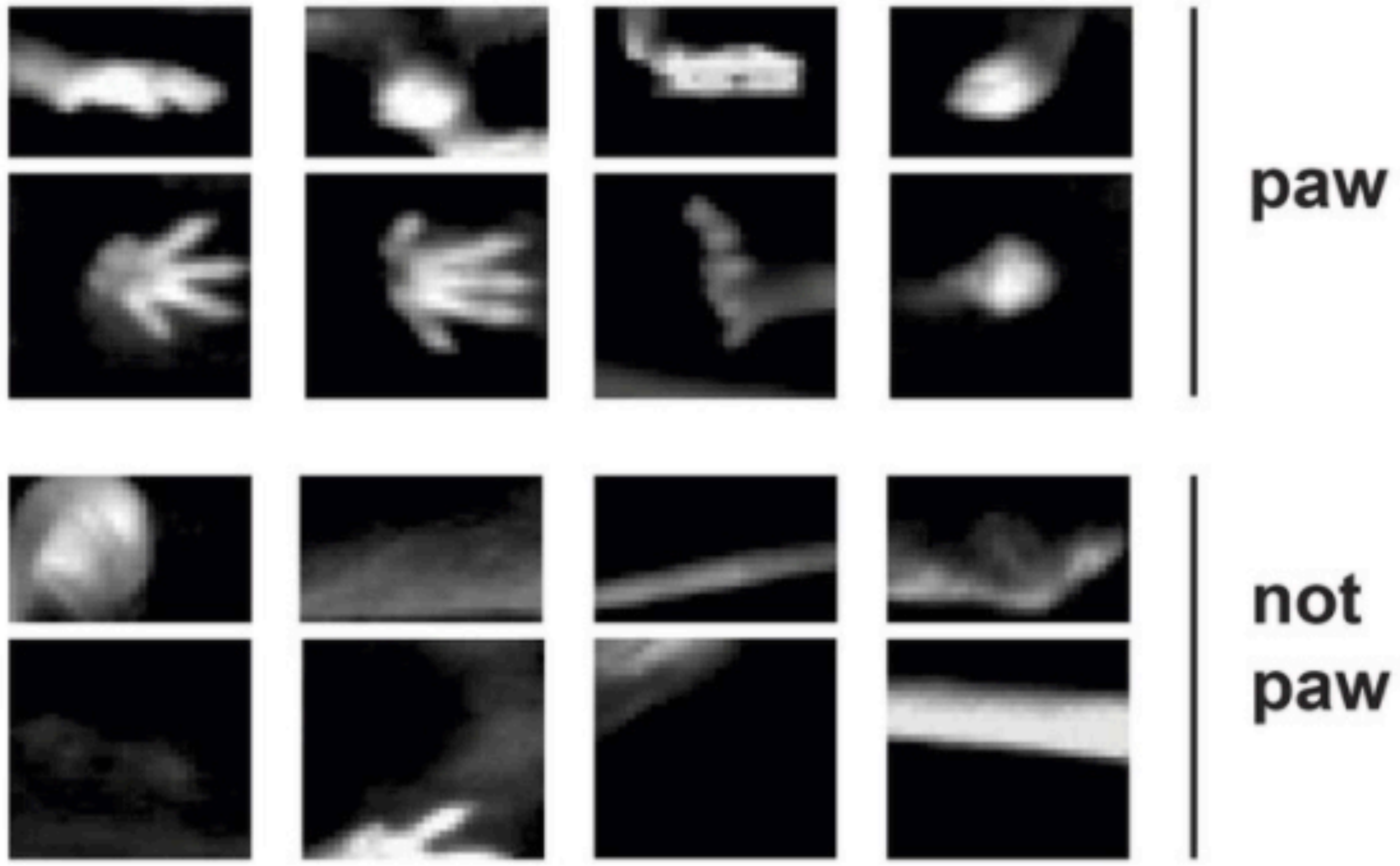
Assume

$$p(y_n \mid \mathbf{x}_n, \mathbf{w}) = \mathrm{Bern}\big(y_n \mid \sigma(\mathbf{w}^\top \mathbf{x}_n)\big)$$

where

$$\sigma(a) = \frac{e^a}{1 + e^a}$$

is the **logistic function.**



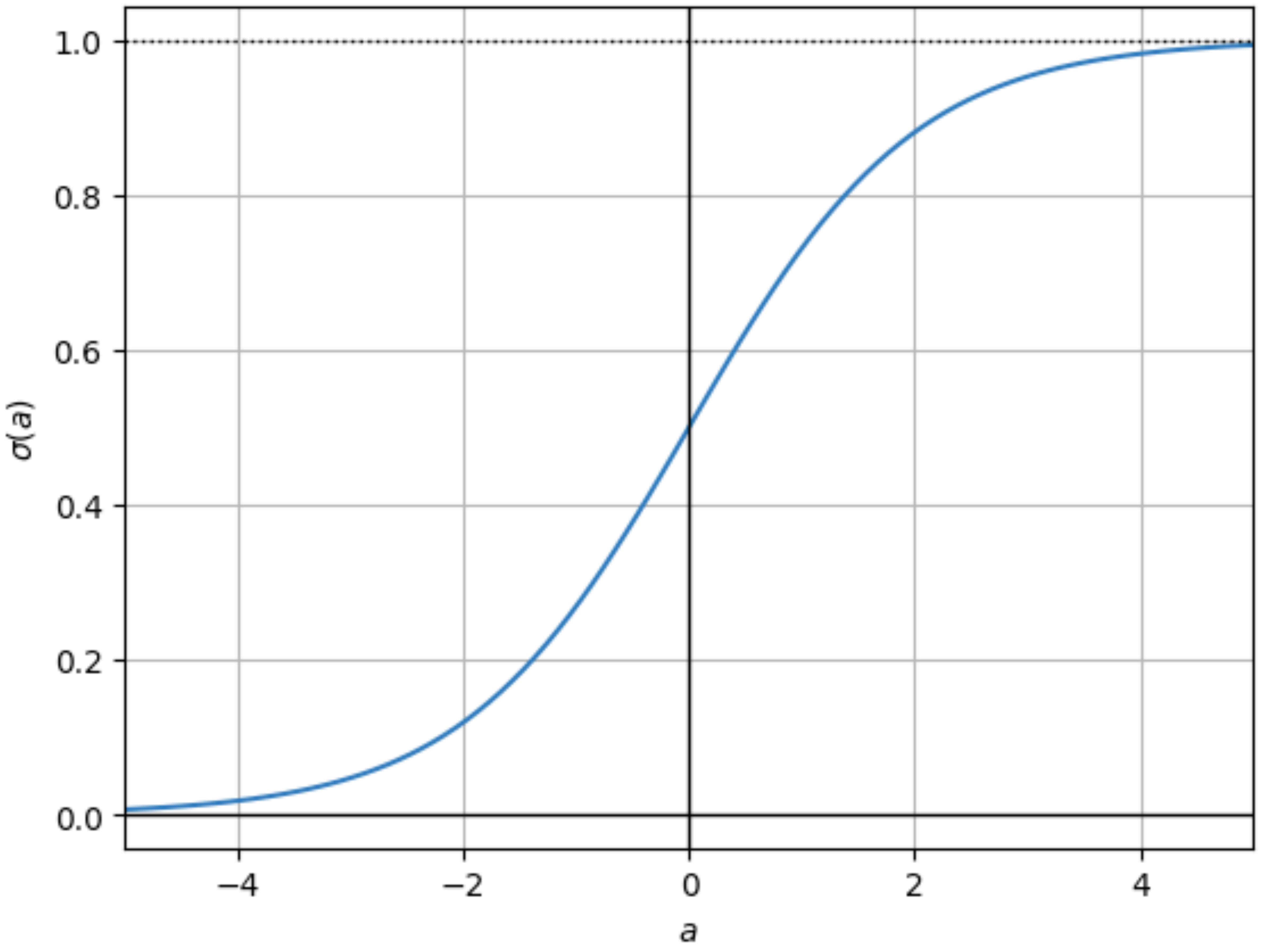
The Bernoulli distribution

The Bernoulli distribution

The **Bernoulli distribution** is a distribution over binary variables $y \in \{0, 1\}$ with probability $p \in [0, 1]$. Its pmf can be written as,

$$\mathrm{Bern}(y; p) = p^y \, (1 - p)^{(1-y)}$$

The logistic function



Basic pose tracking

Maximum likelihood estimation

$$\mathscr{L}(\mathbf{w}) = -\log p(\mathbf{y} \mid \mathbf{w}, \mathbf{X})$$

Basic pose tracking

Calculating the gradient

$$\nabla \mathscr{L}(\mathbf{w}) =$$

Basic pose tracking

The negative log likelihood is convex

- The Hessian is positive semi-definite
- $\nabla^2 \mathscr{L}(\mathbf{w}) =$

Gradient descent

Let $\mathbf{w}_0$ denote our initial setting of the weights. Gradient descent is an iterative algorithm that produces a sequence of weights $\mathbf{w}_0, \mathbf{w}_1, \ldots$ that (under certain conditions) converges to a local optimum of the objective. Since the objective is convex, all local optima are global optima. The idea is straightforward, on each iteration we update the weights by taking a step in the direction of the gradient,

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \alpha_i \nabla \mathcal{L}(\mathbf{w}_i)$$

where $\alpha_i \in \mathbb{R}_+$ is the **learning rate** (aka step size) on iteration i, and $\nabla \mathcal{L}(\mathbf{w}_i)$ is the gradient of the objective evaluated at the current weights $\mathbf{w}_i$.

Newton's Method

- We can obtain faster convergence rates using **second-order** methods.
- Approximate the objective with a second-order Taylor approximation around the current weights,
$$\mathscr{L}(\mathbf{w}) \approx \mathscr{L}(\mathbf{w}_i) + (\mathbf{w} - \mathbf{w}_i)^\top \nabla \mathscr{L}(\mathbf{w}_i) + \frac{1}{2}(\mathbf{w} - \mathbf{w}_i)^\top \nabla^2 \mathscr{L}(\mathbf{w}_i)(\mathbf{w} - \mathbf{w}_i)\,.$$
- **Exercise:** show that the minimum is obtained at
$\mathbf{w}_{i+1} = \mathbf{w}_i + \nabla^2 \mathscr{L}(\mathbf{w}_i)^{-1} \nabla \mathscr{L}(\mathbf{w}_i)\,.$

Computational complexity

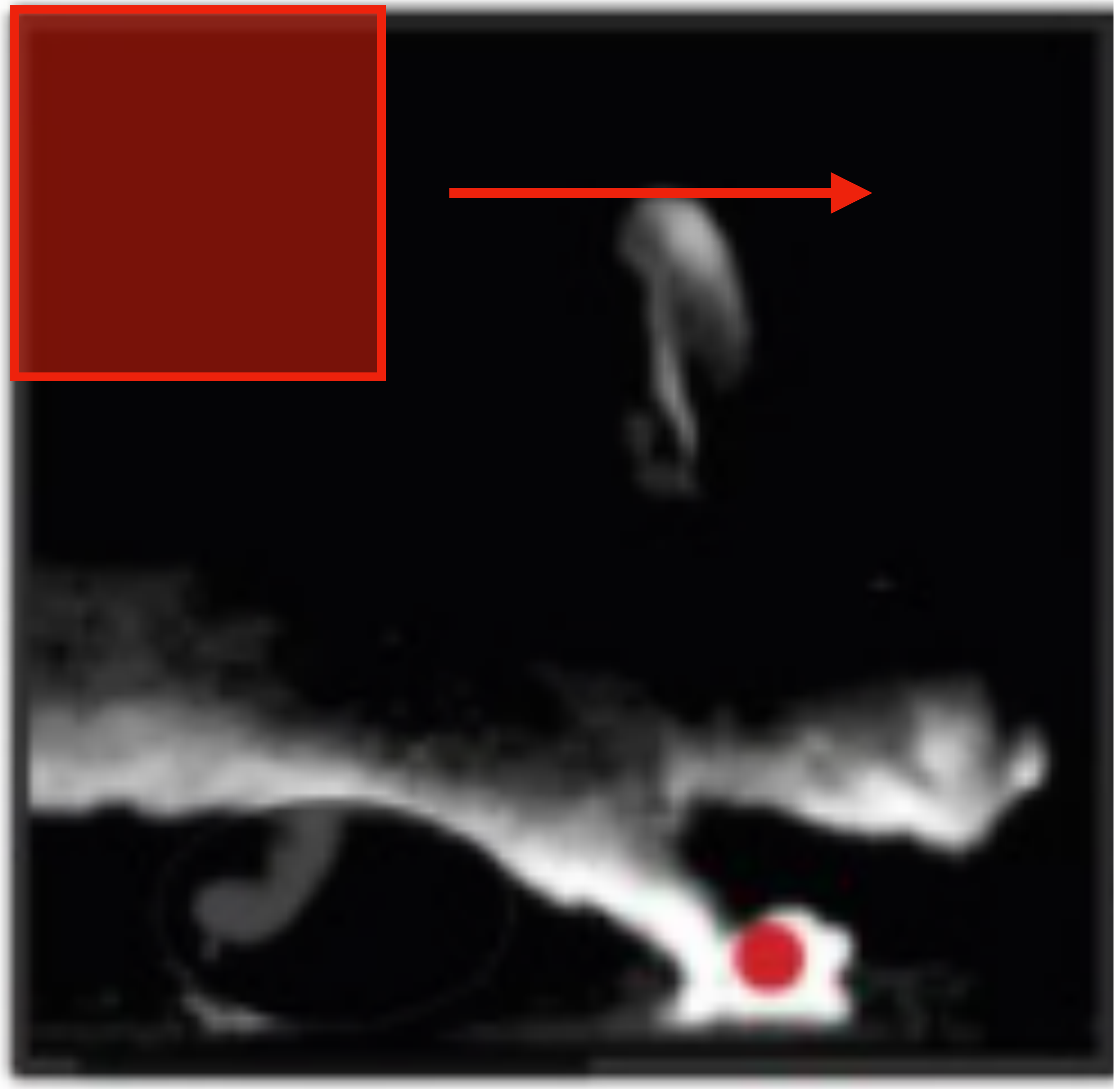
- What is the (time) complexity of gradient descent and Newton's method?
- Quasi-Newton methods like BFGS sidestep the Hessian calculation and inversion.
- SGD (with momentum) uses mini-batches of data and rolling averages of the gradient to achieve faster convergence.
- Adagrad, RMSProp, and Adam tune the learning rates as they go.

Pose tracking with convolutional neural networks

Basic pose tracking

As a one-layer convolutional neural network

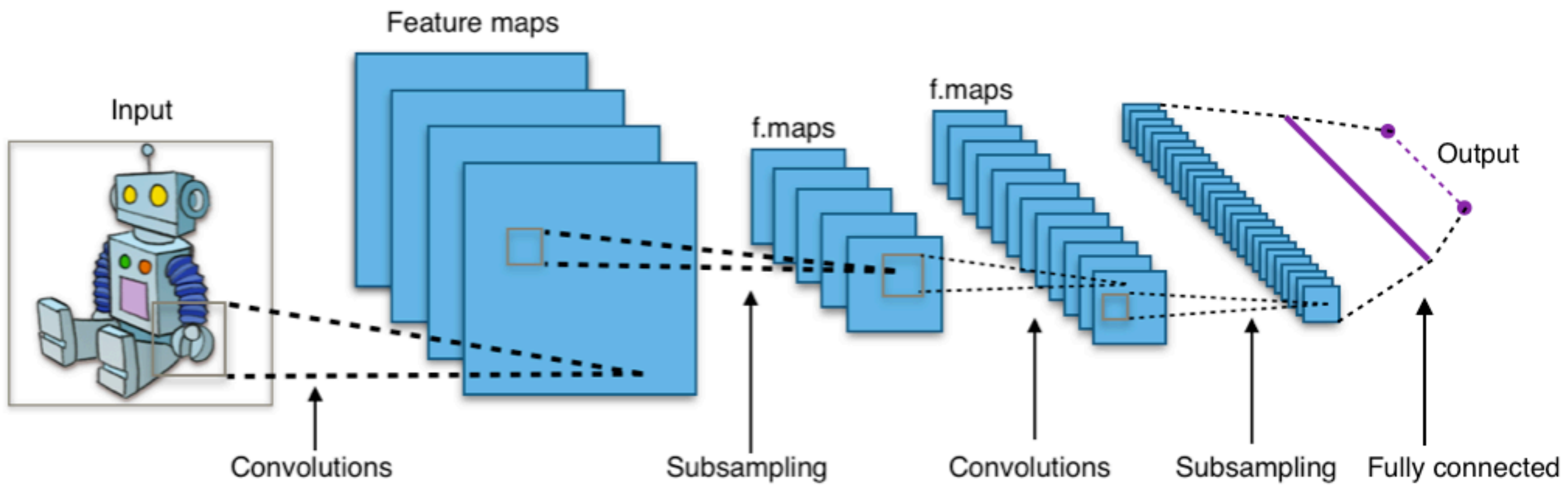
- Instead of working with patches, let's work with images directly.
- Let $\mathbf{X}_n \in \mathbb{R}^{P_H \times P_W}$ denote an image (height P_H, width P_W)
- Let $\mathbf{Y} \in \{0,1\}^{P_H \times P_W}$ indicate the location(s) of the keypoint.
- The 2D cross-correlation $\mathbf{X}_n \star \mathbf{W}$; is a sliding dot product of weights across all $P_h \times P_w$ patches in the image. It produces a $P_H \times P_W$ output.
- In PyTorch, it's implemented by the `F.conv2d` function and the `Conv2D` layer.



Basic pose tracking

Feature learning in CNNs

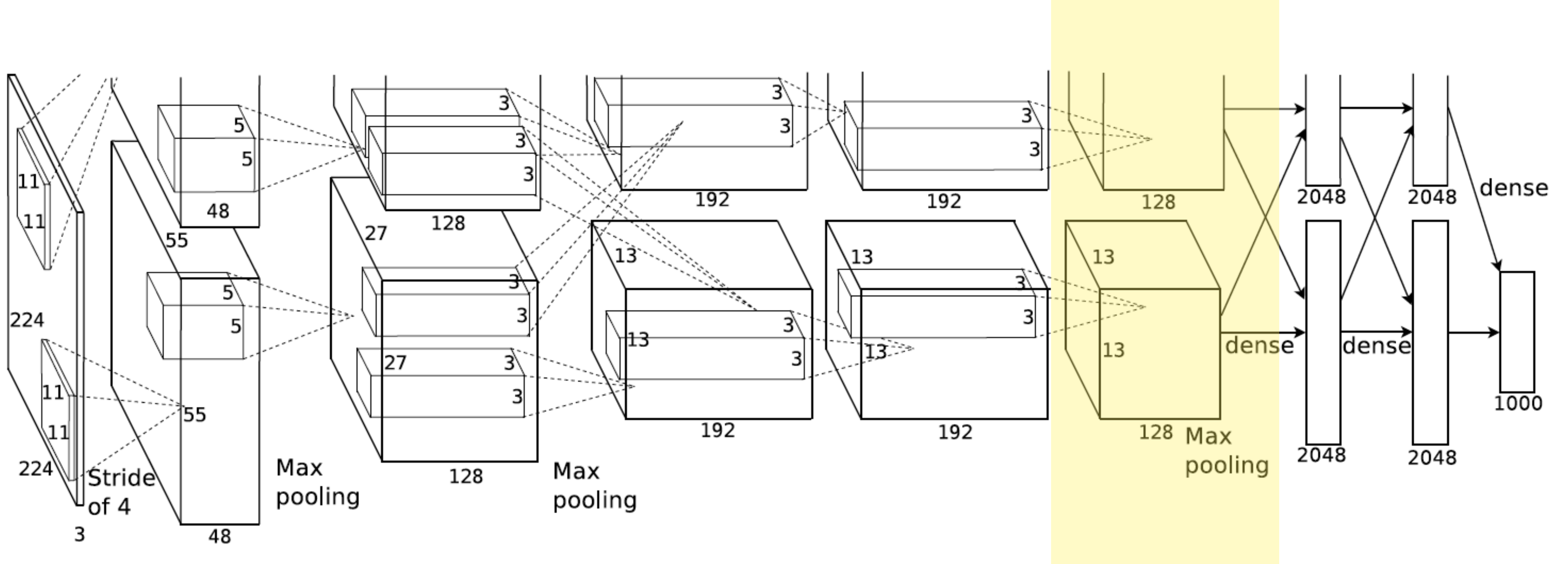
- This simple model assumes key points can be detected with a **linear classifier** using raw pixels as inputs.
- We can perform **nonlinear classification** by encoding each pixel with a vector of features.
- Rather than handcrafting these features, **learn them** from the data!



https://en.wikipedia.org/wiki/Convolutional_neural_network

Transfer Learning

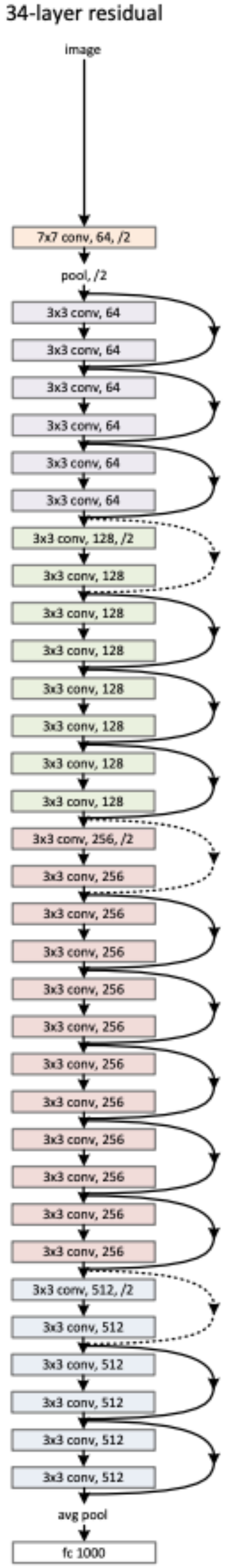
- **Idea**: rather than handcrafting features or learning them from scratch, **use a pre-trained network** for a related task.
- **Example**: use the features of a deep neural network for image classification.
- **Reroute** the output of an intermediate layer to a **new loss function.**
- Optionally, **fine tune** the weights in the early layers via stochastic gradient descent on the new loss.
- With good starting features, you **only need a few training examples** to perform animal pose estimation.



Transfer Learning
In DeepLabCut, SLEAP, etc.

- DLC and SLEAP repurpose state-of-the-art deep networks for human pose detection.
- DLC starts with a residual network (resnet-50) and adds “deconvolutional” layers, as in DeeperCut for human pose estimation.
- SLEAP starts with “stacked hourglass networks” for human pose estimation.

Deep Residual Networks
(resnet-50)



Stacked Hourglass Networks

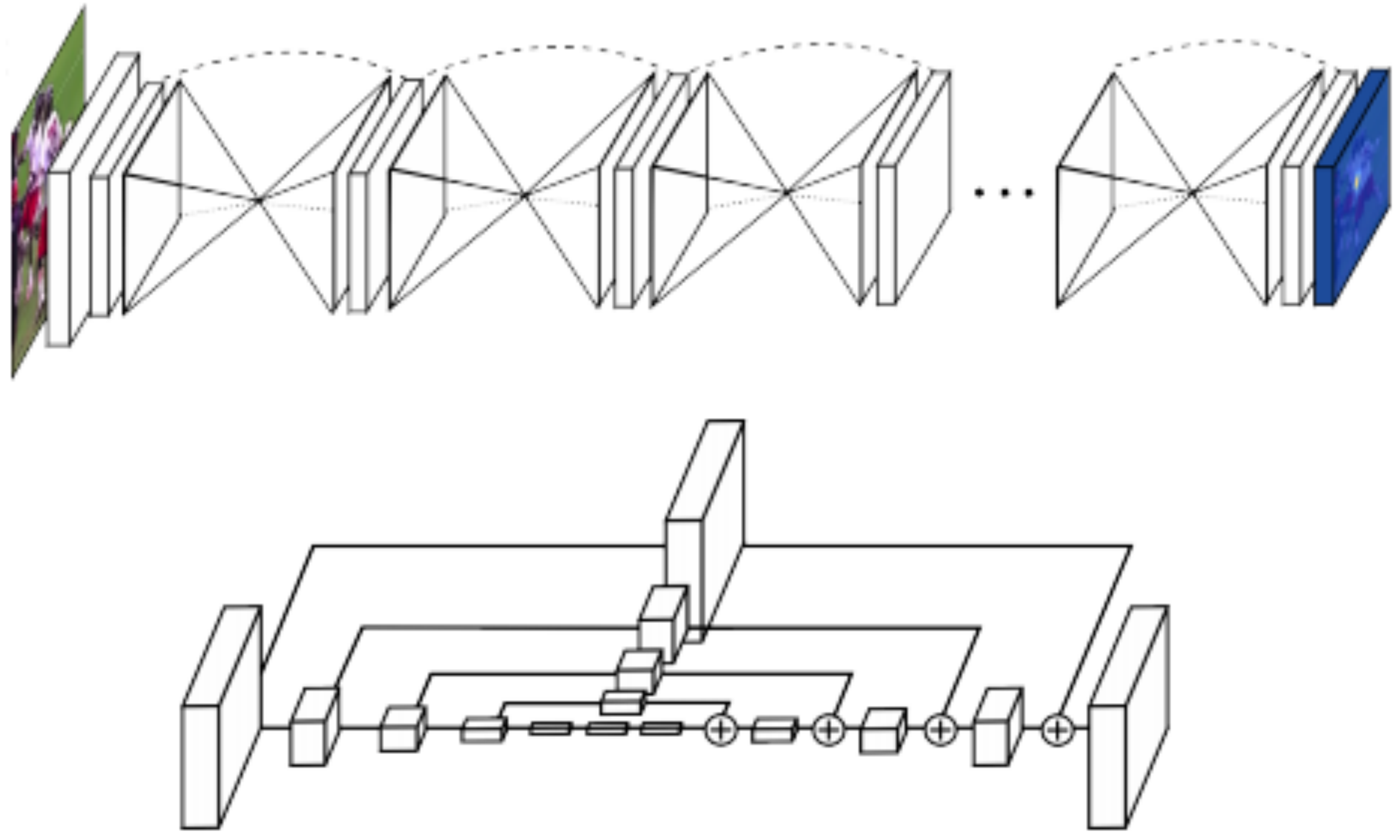
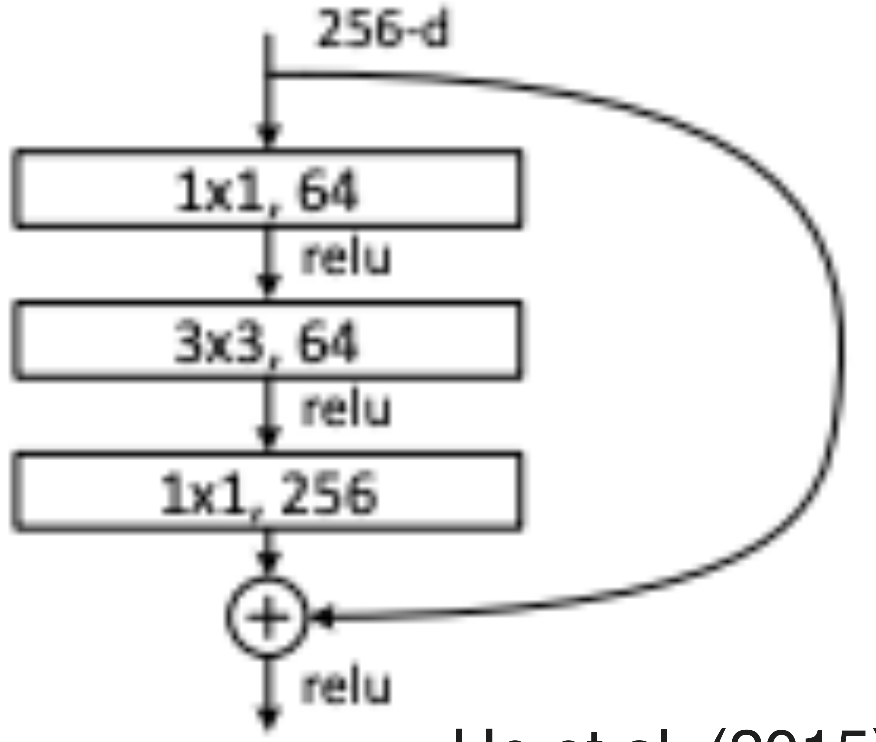


Fig. 3. An illustration of a single “hourglass” module. Each box in the figure corresponds to a residual module as seen in Figure 4. The number of features is consistent across the whole hourglass.

Newell et al (2016)

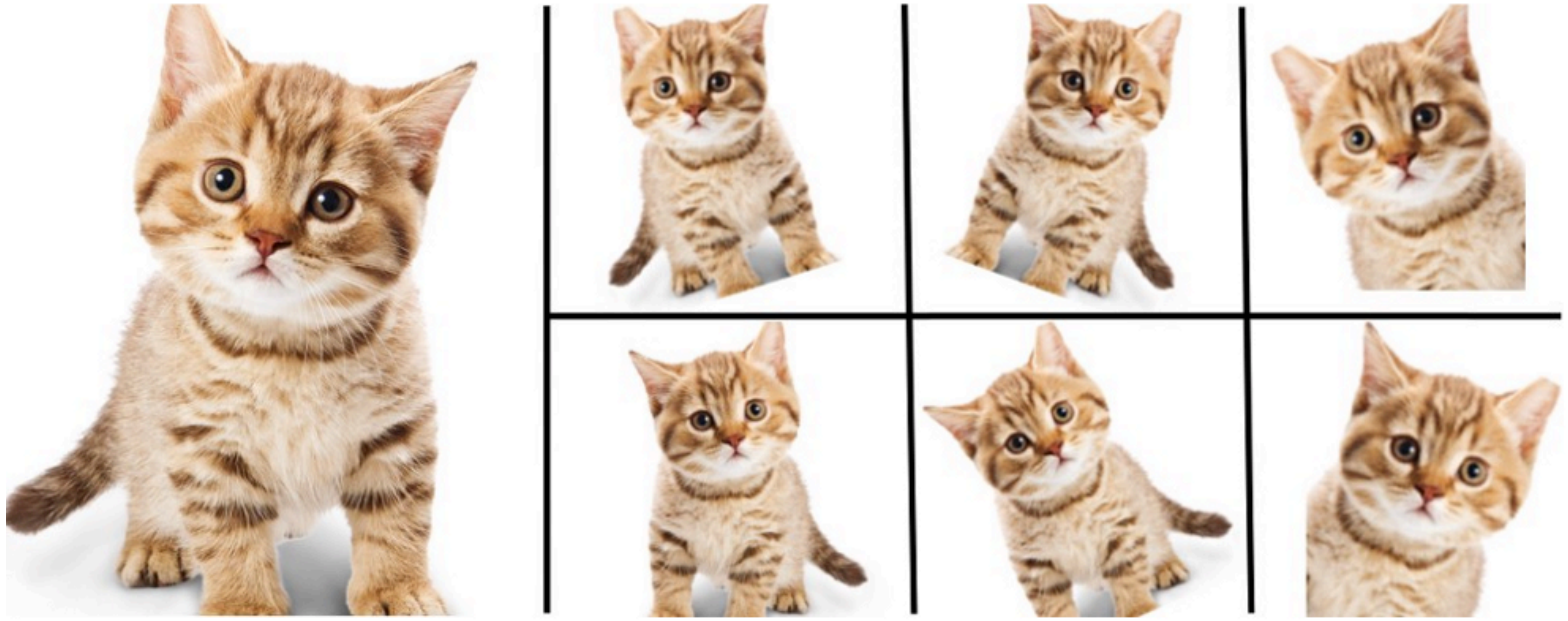


He et al (2015)

Transfer Learning

Data augmentation

- Labeling data is tedious.
- **Idea:** Make the most of each training example by making alterations your classifier should be robust to.
- Eg a cropped, rotated, and scaled paw is still a paw. A partially occluded paw is still a paw.



Structured prediction

Structured prediction

How do we aggregate key point probabilities for each pixel?

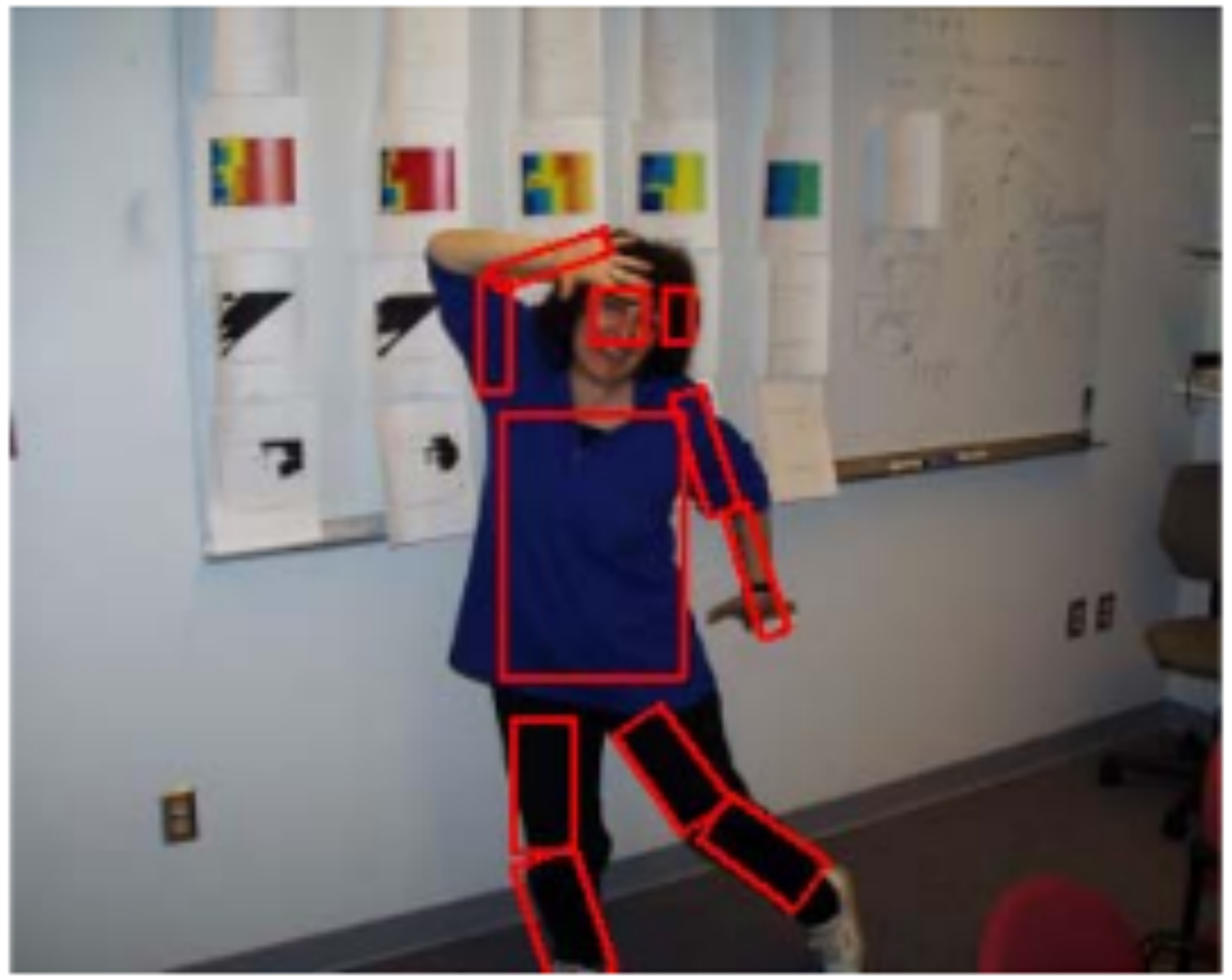
Bayesian formulation with efficient MAP inference

$$L^* = \arg\min_L \left(\sum_{(v_i, v_j) \in E} d_{ij}(l_i, l_j) - \sum_{v_i \in V} \ln g_i(I, l_i) \right)$$

Optimal configuration

Part-to-part term

Image to part term

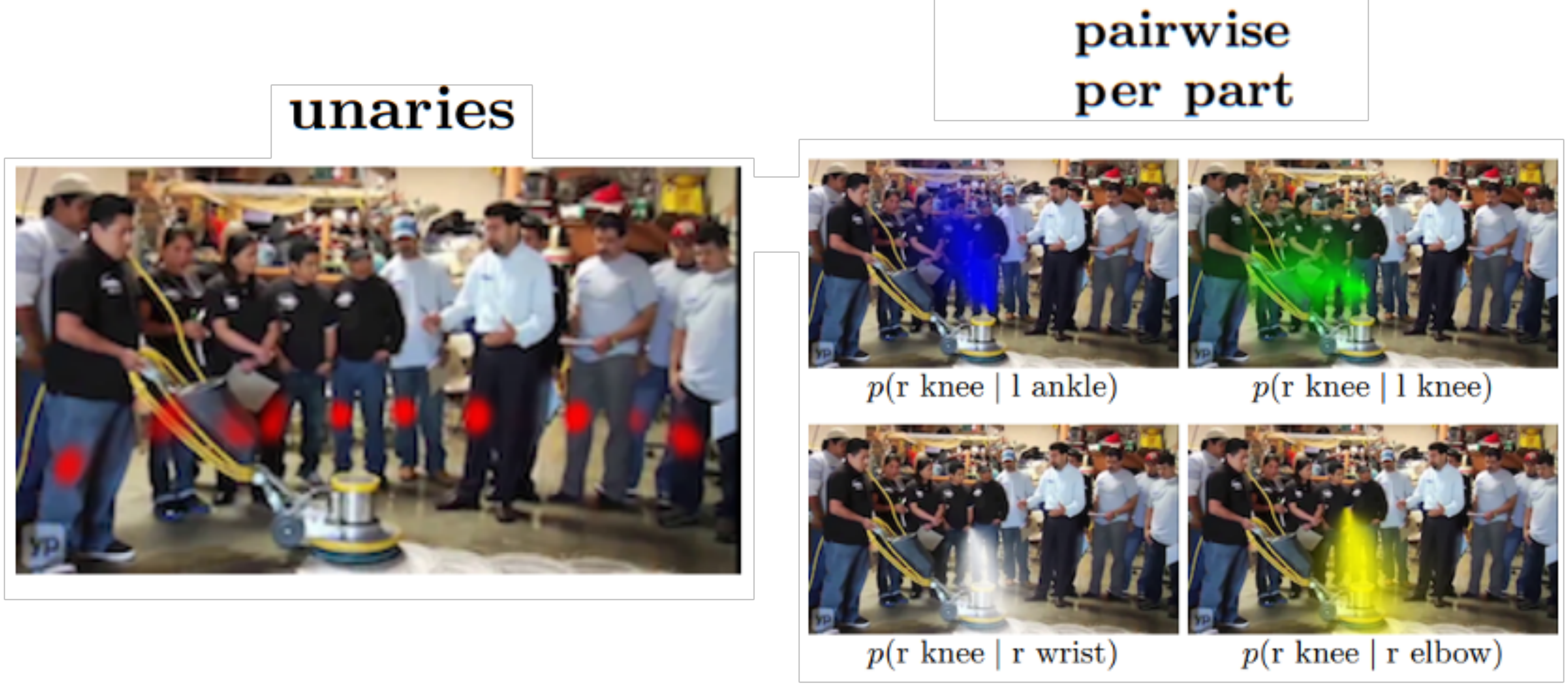


Felzenswalb & Huttenlocher. Pictorial Structures for Object Recognition (2004)
Felzenswalb & Huttenlocher. Efficient matching of pictorial structures (2005)

Slide credit: Talmo Pereira

Structured prediction

How do we aggregate key point probabilities for each pixel?



Pishchulin, Insafutdinov, Tang, Andres, Andriluka, Gehler, & Schiele. [DeepCut: Joint Subset Partition and Labeling for Multi Person Pose Estimation]() (2015)
Insafutdinov, Pishchulin, Andres, Andriluka & Schiele. [DeeperCut: A Deeper, Stronger, and Faster Multi-Person Pose Estimation Model]() (2016)

Slide credit: Talmo Pereira

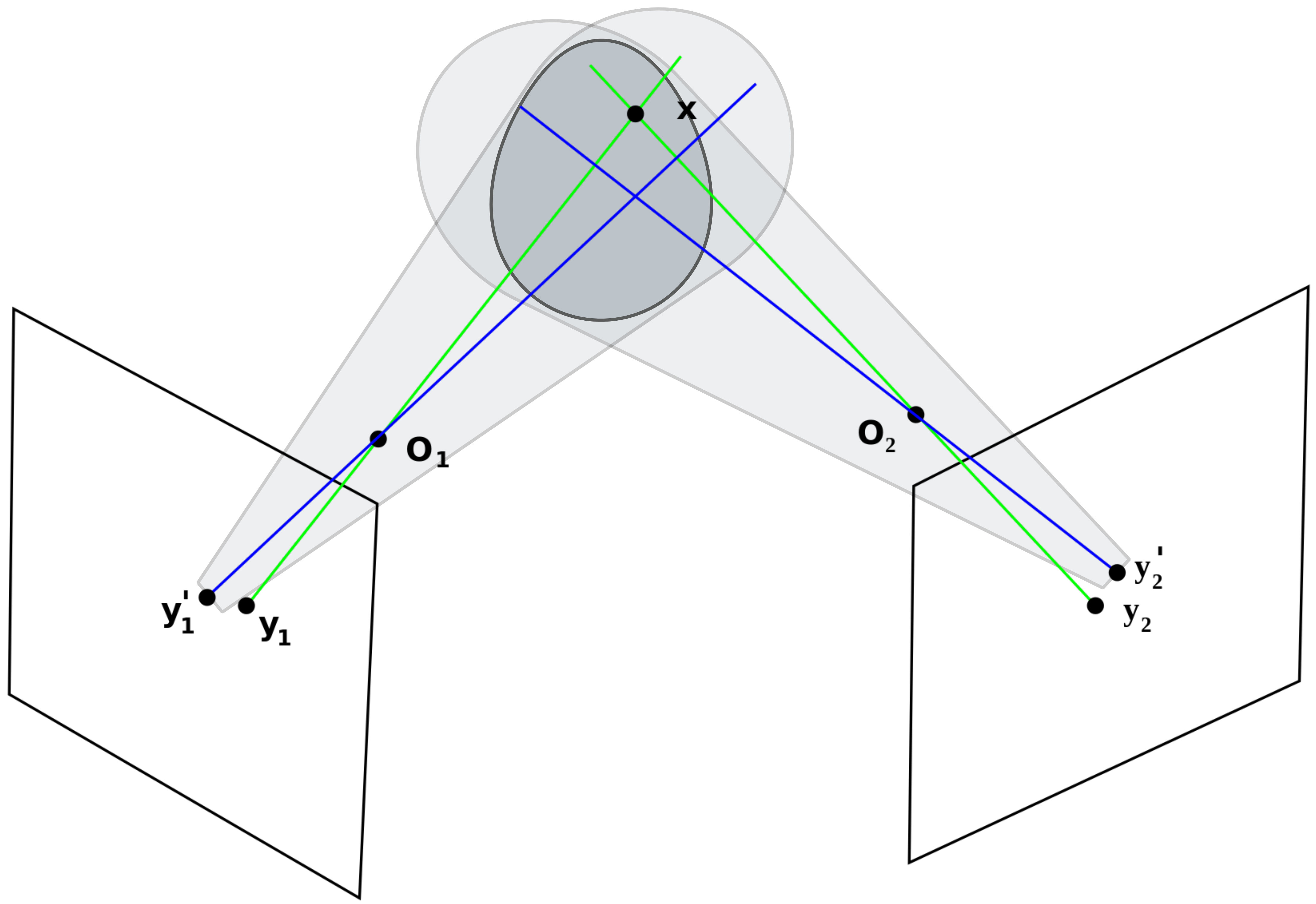
3D Pose estimation

Projective geometry

- How can we estimate 3D pose from multiple 2D camera views?
- Projective geometry makes far away objects appear smaller.

$$\vec{y}_c \approx f_c(\vec{x})$$

$$f_c(\vec{x}) = \frac{1}{w}(u, v)^\top \text{ where } (u, v, w)^\top = A_c\vec{x} + b_c,$$

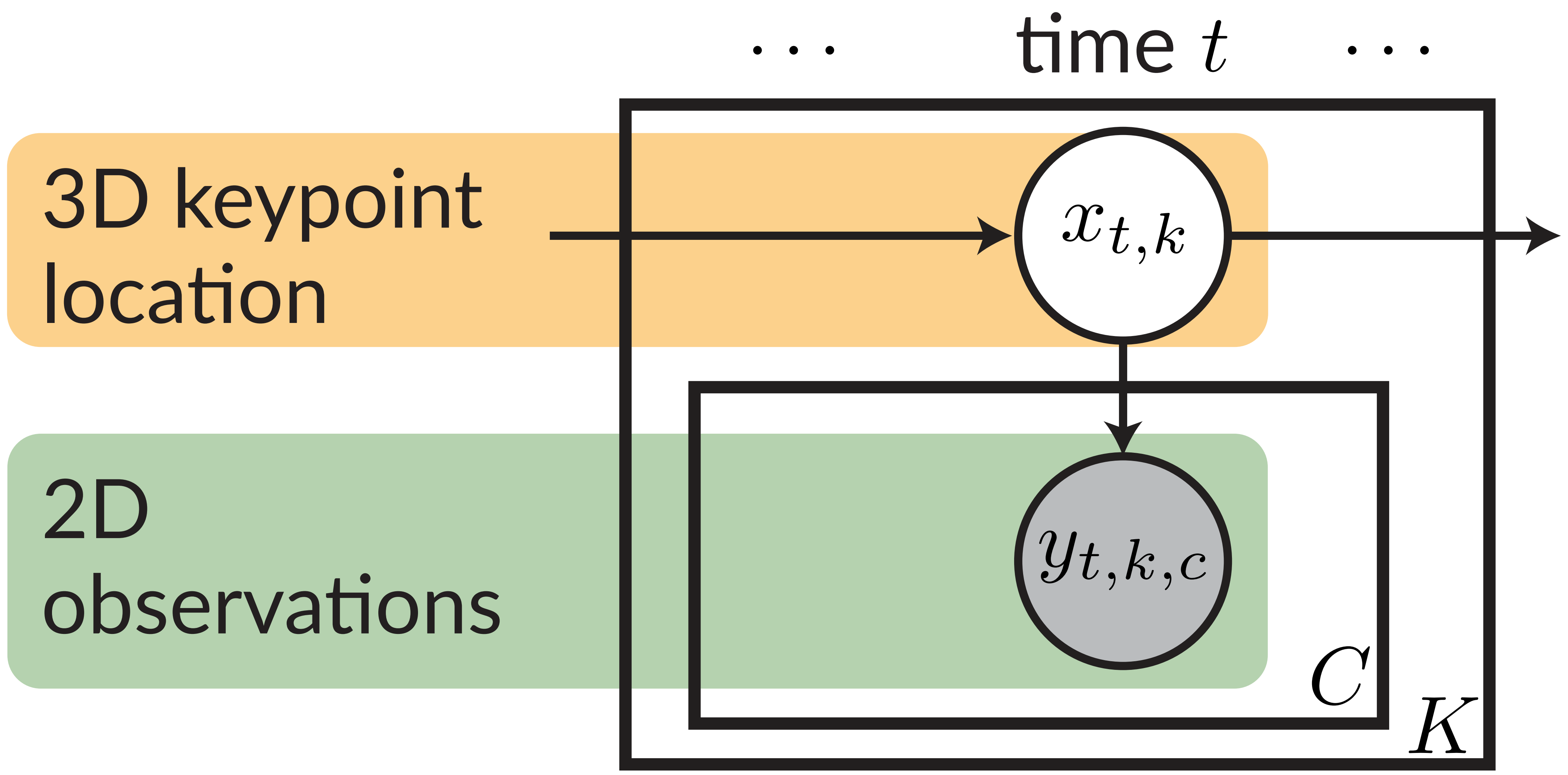


Modified from wikipedia.org

3D Pose estimation

Model 0: Bayesian triangulation of 3D pose from 2D observations

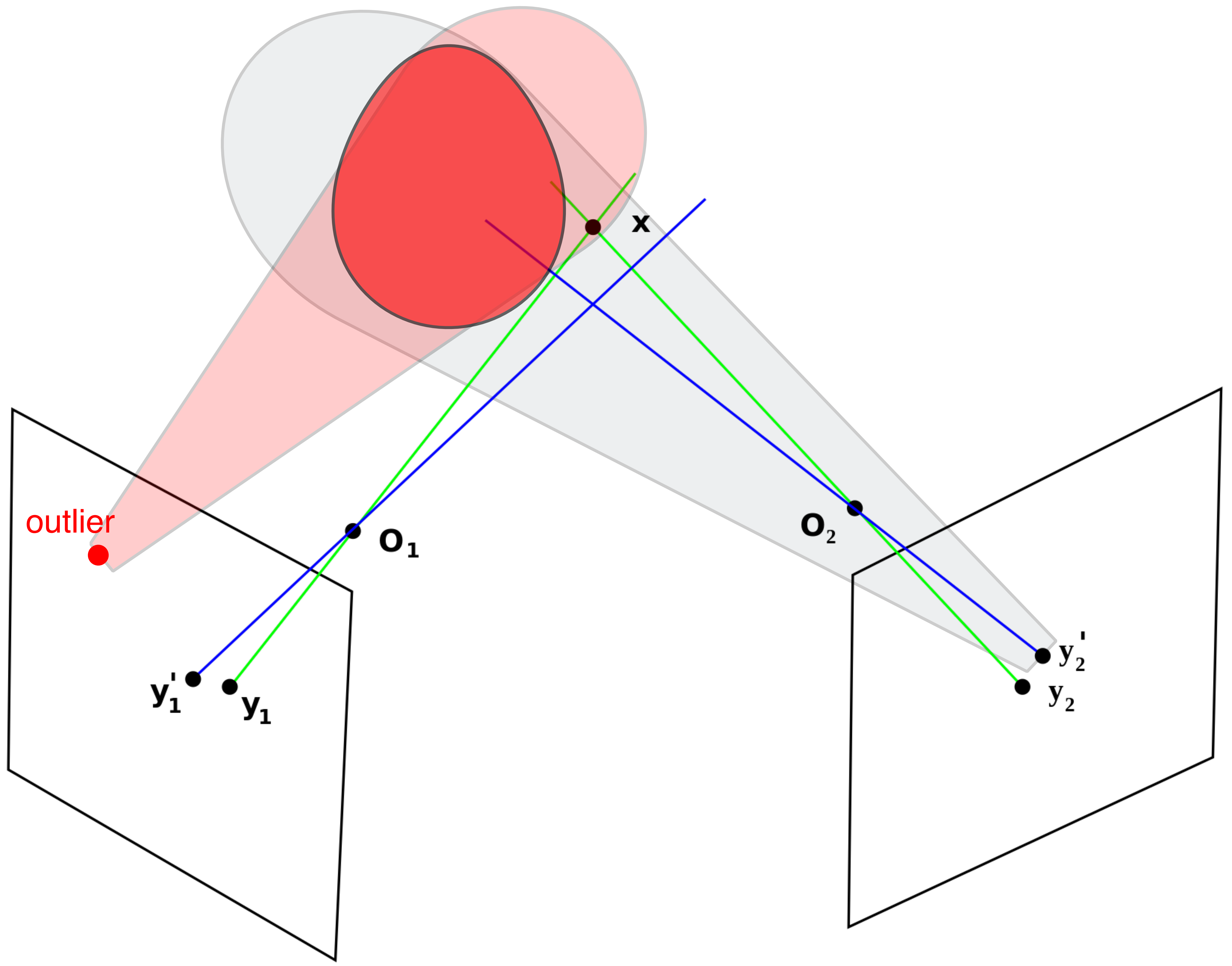
$$x_{t,k} \sim \mathcal{N}(x_{t-1,k},\, \eta^2 I)$$
$$y_{t,k,c} \sim \mathcal{N}\left(f_c(x_{t,k}),\, \omega^2 I\right)$$



T time steps
K keypoints
C cameras

3D Pose estimation
Triangulation in the presence of measurement noise

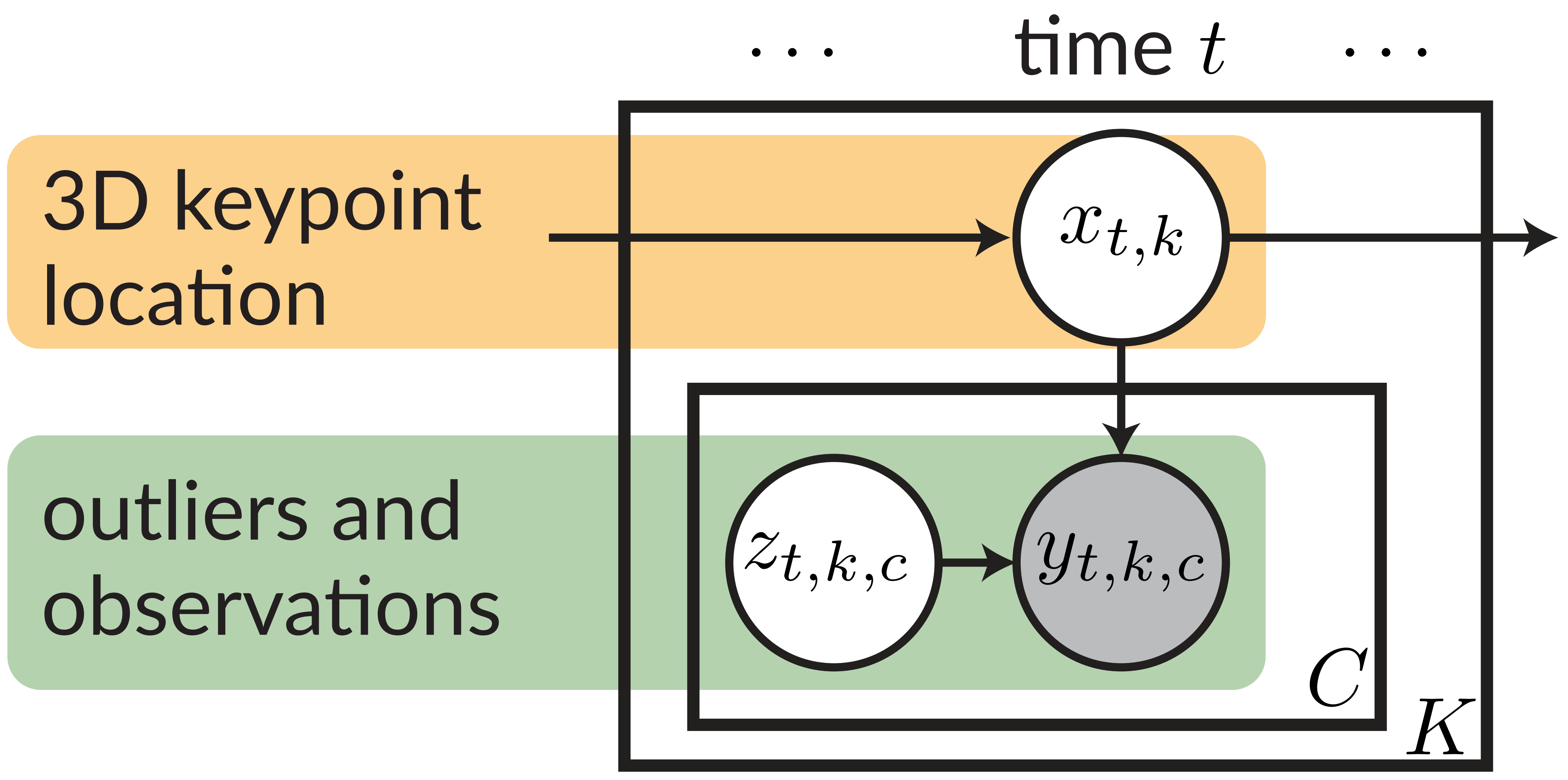
- Projective geometry makes far away objects appear smaller.
- Outliers in 2D estimates can severely affect 3D triangulation.
- Typical approaches:
  - More data
  - Temporal constraints
  - Median filtering (DLC-3D) / RANSAC
  - Robust noise models



Modified from wikipedia.org

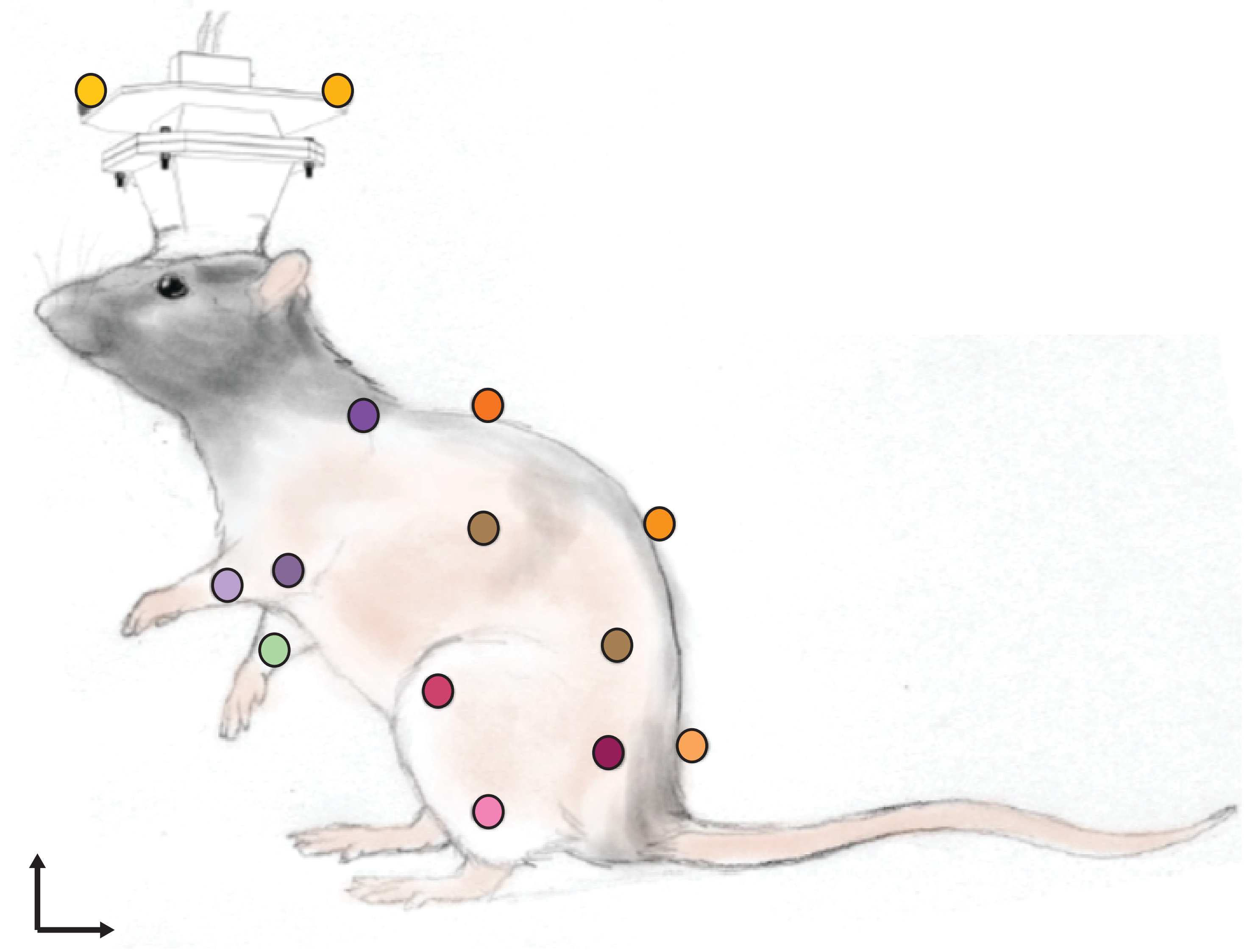
Model 1: Robust Bayesian triangulation of 3D pose from 2D observations

- Projective geometry makes far away objects appear smaller.
- Outliers in 2D estimates can severely affect 3D triangulation.
- Typical approaches:
  - More data
  - Temporal constraints
  - Median filtering (DLC-3D) / RANSAC
  - **Robust noise models**



Triangulation in the presence of measurement noise

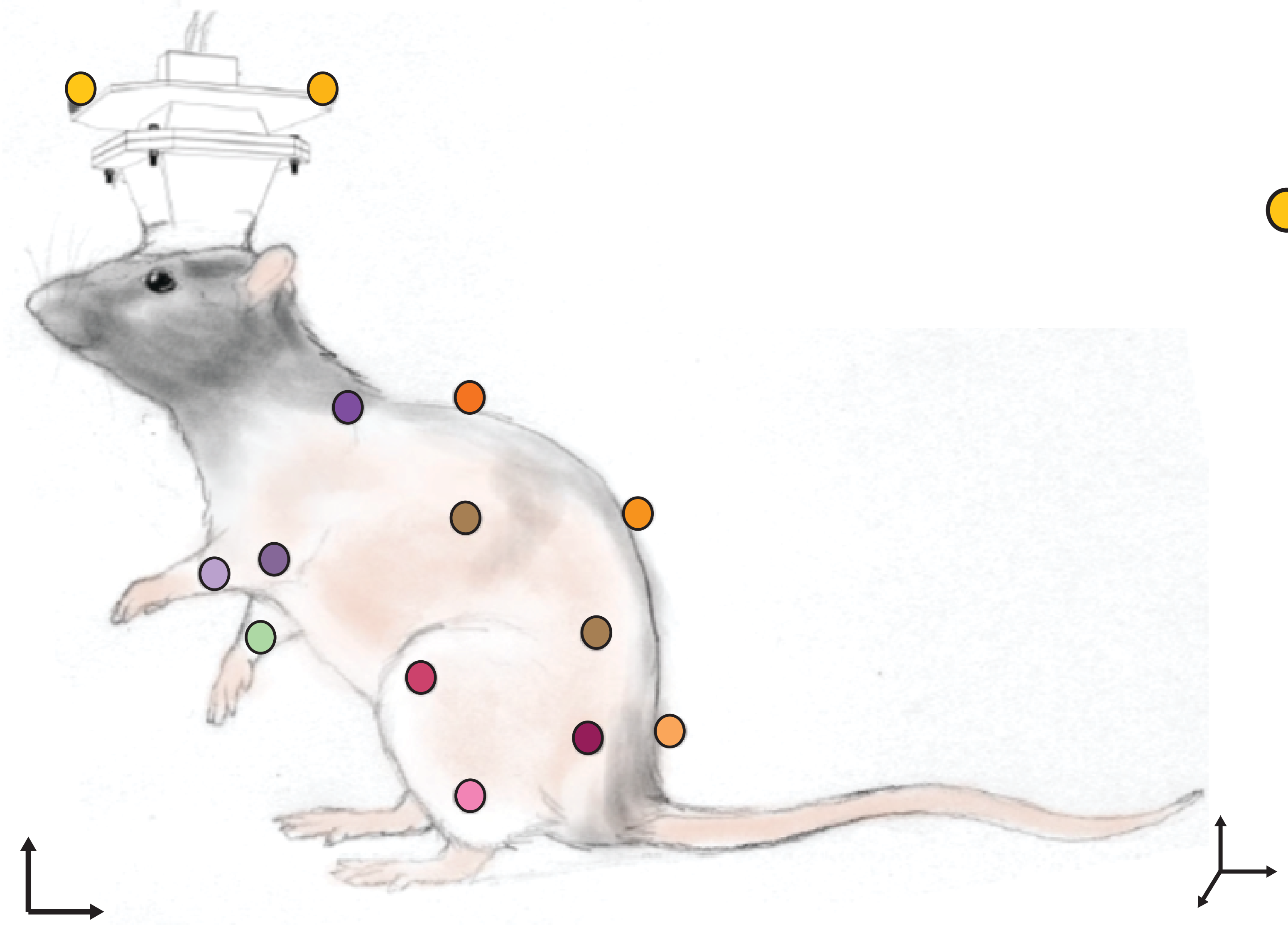
- Projective geometry makes far away objects appear smaller.
- Outliers in 2D estimates can severely affect 3D triangulation.
- Typical approaches:
  - More data
  - Temporal constraints
  - Median filtering (DLC-3D) / RANSAC
  - Robust noise models
  - **Spatial constraints**

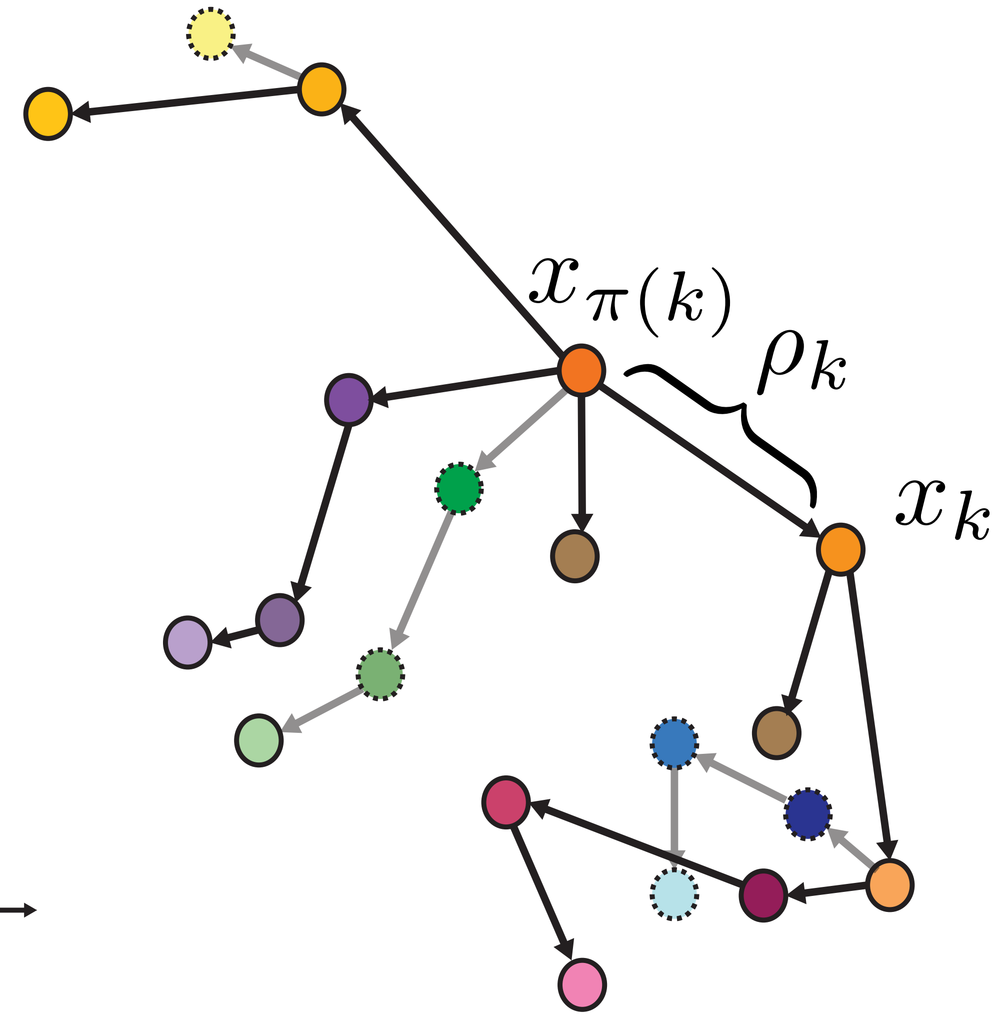


A probabilistic view of spatial constraints

- Common approach:

$$p(x) \propto \prod_k \mathcal{N}(\|x_k - x_{\pi(k)}\|; \rho_k, \sigma^2 I)$$





A probabilistic view of spatial constraints

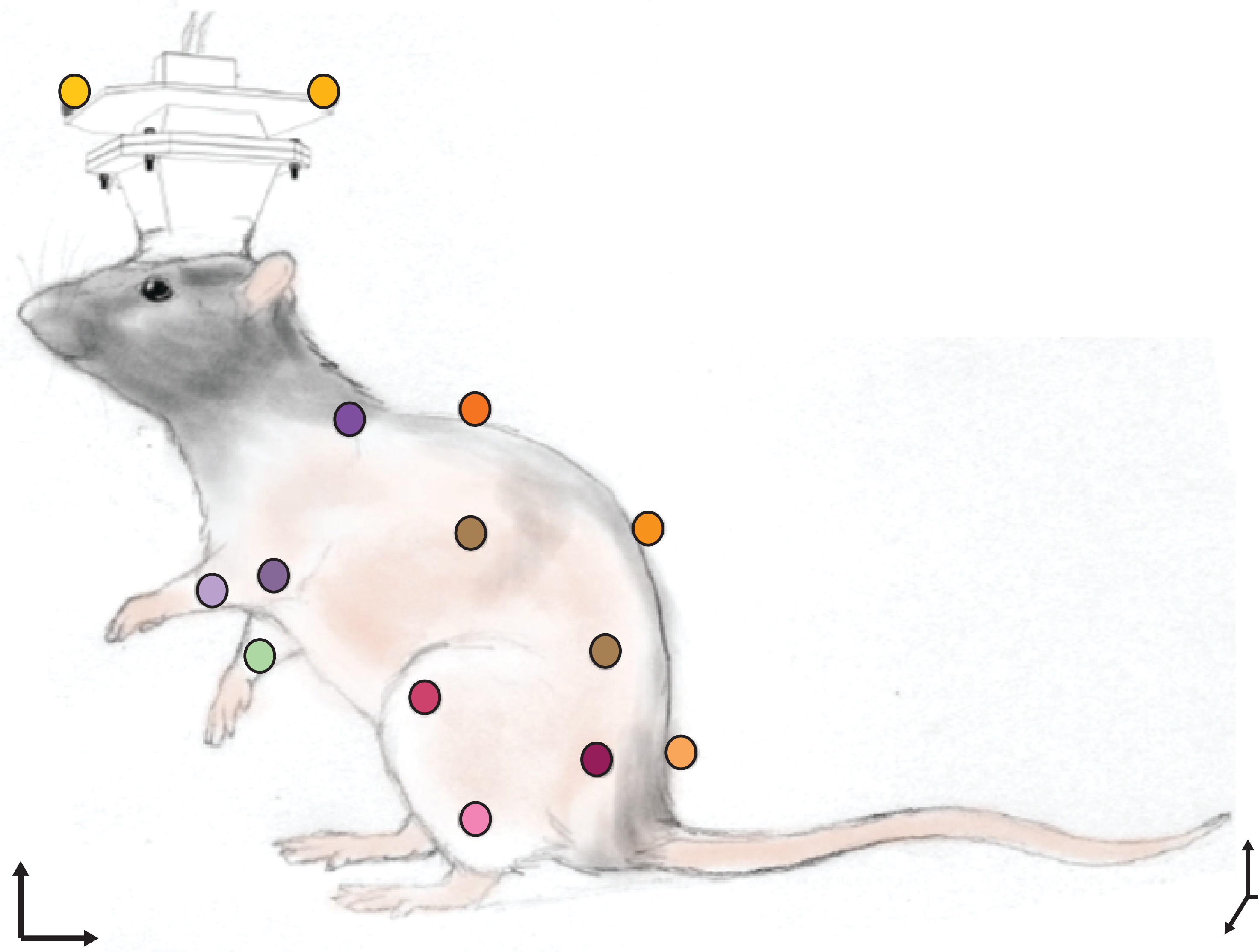
- Common approach:

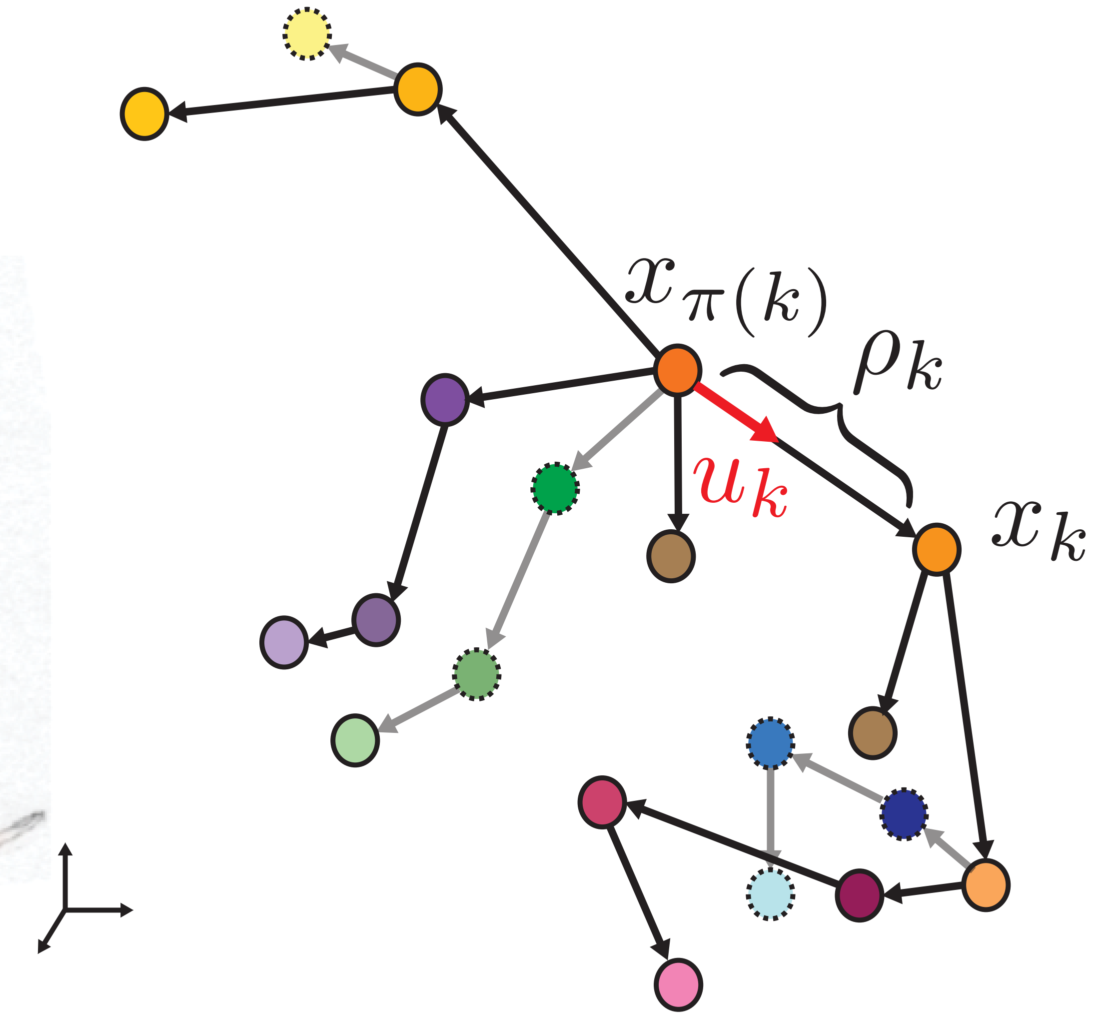
$$p(x) \propto \prod_k \mathcal{N}(\|x_k - x_{\pi(k)}\|; \rho_k, \sigma^2 I)$$

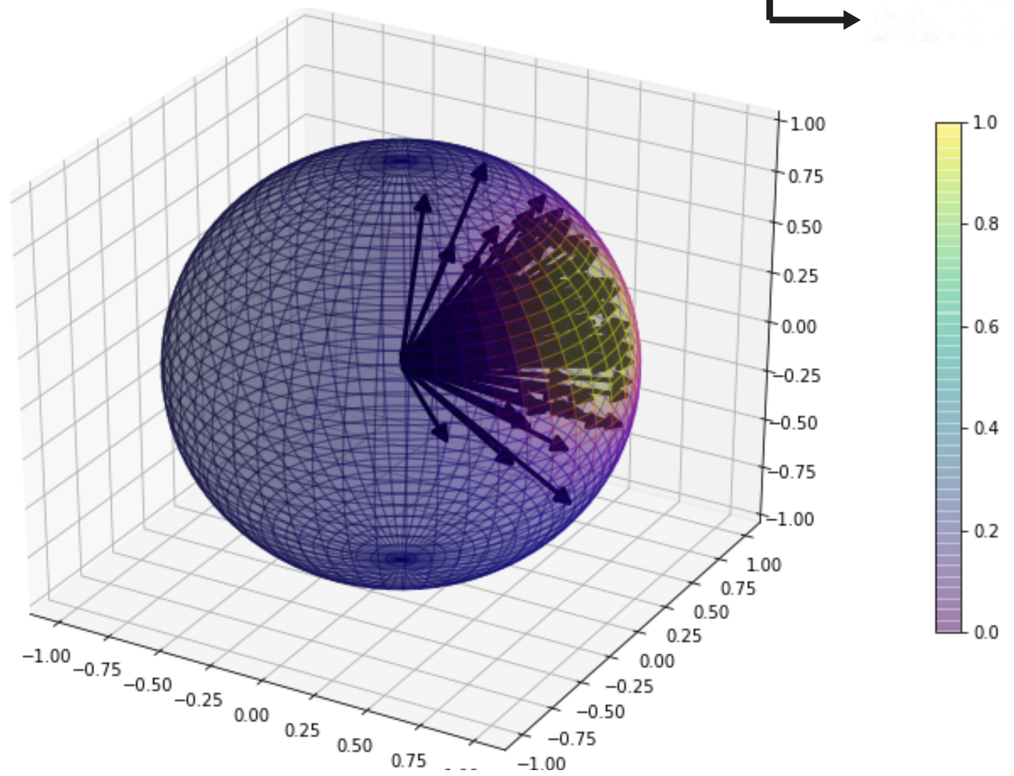
- Alternative:

$$u_k \sim \mathrm{Unif}(\mathbb{S}_2)$$
$$x_k \mid u_k \sim \mathcal{N}(x_{\pi(k)} + \rho_k u_k,\, \sigma^2 I)$$

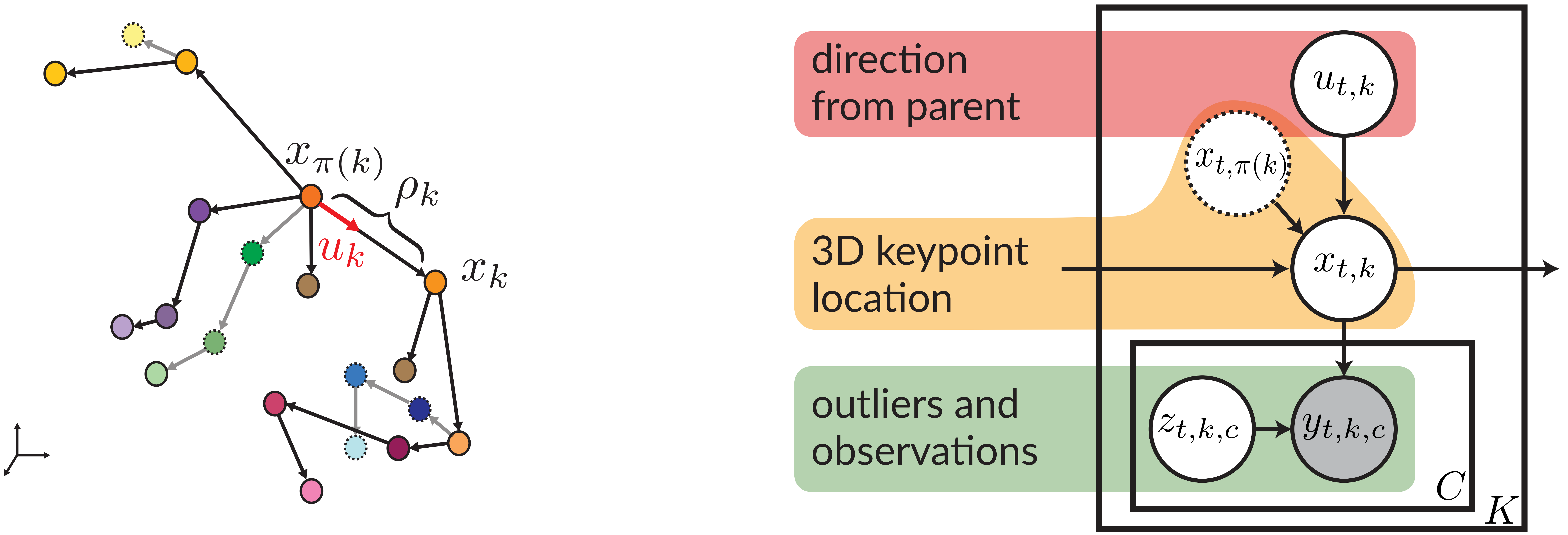
- Are these equivalent?



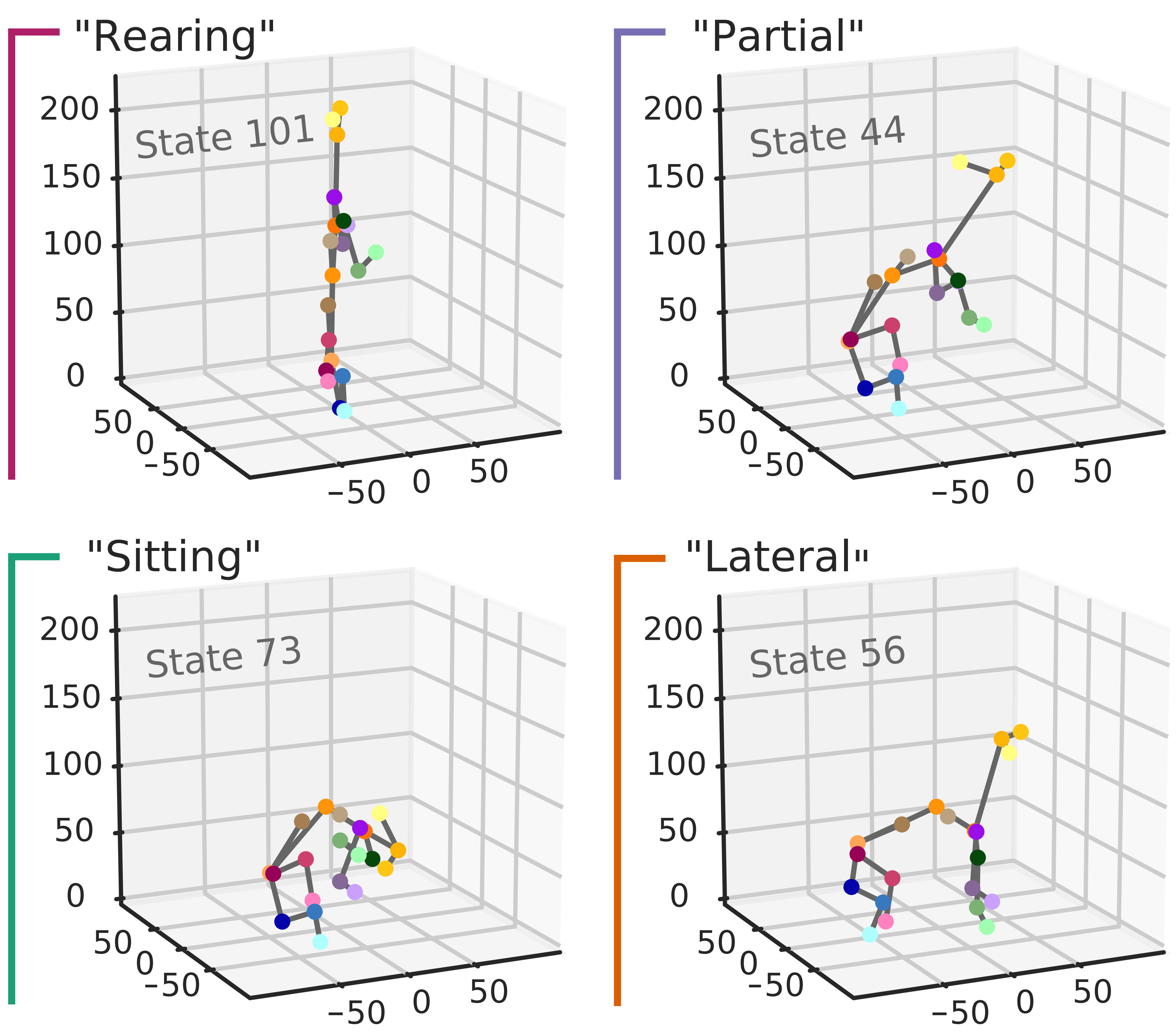




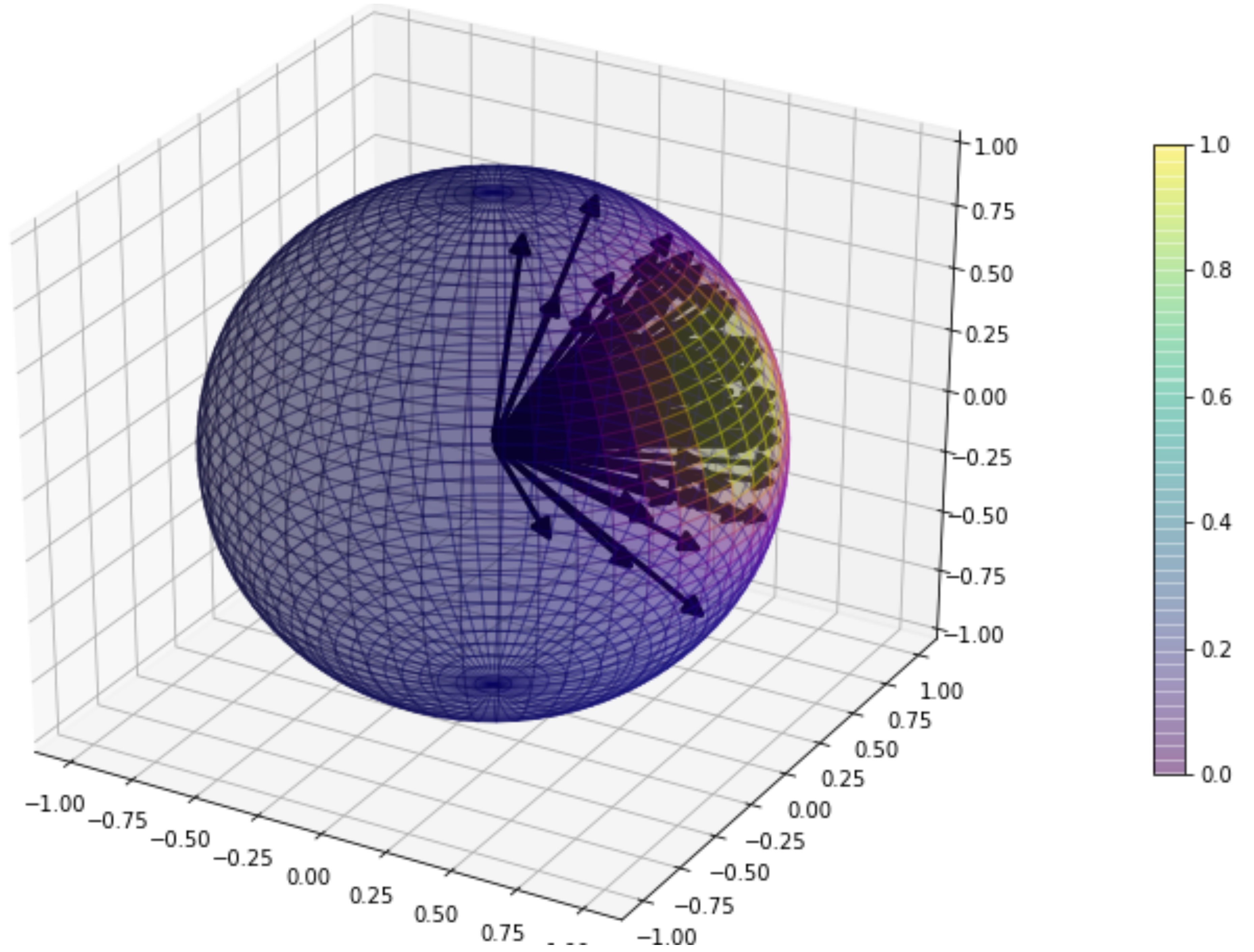
Model 2: Incorporating distance priors on keypoint configurations



Why stop at distances? Poses involve correlated directions!

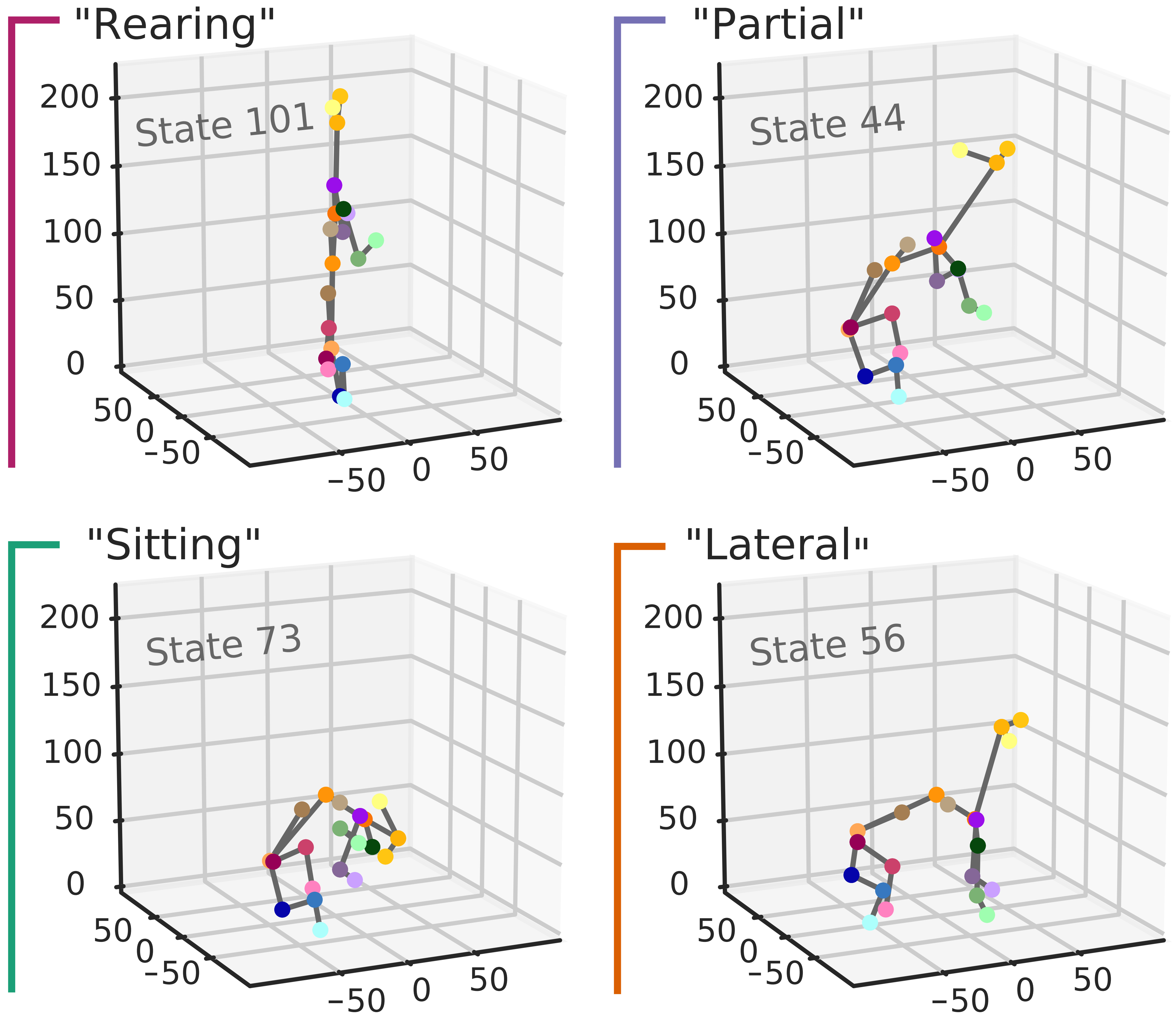


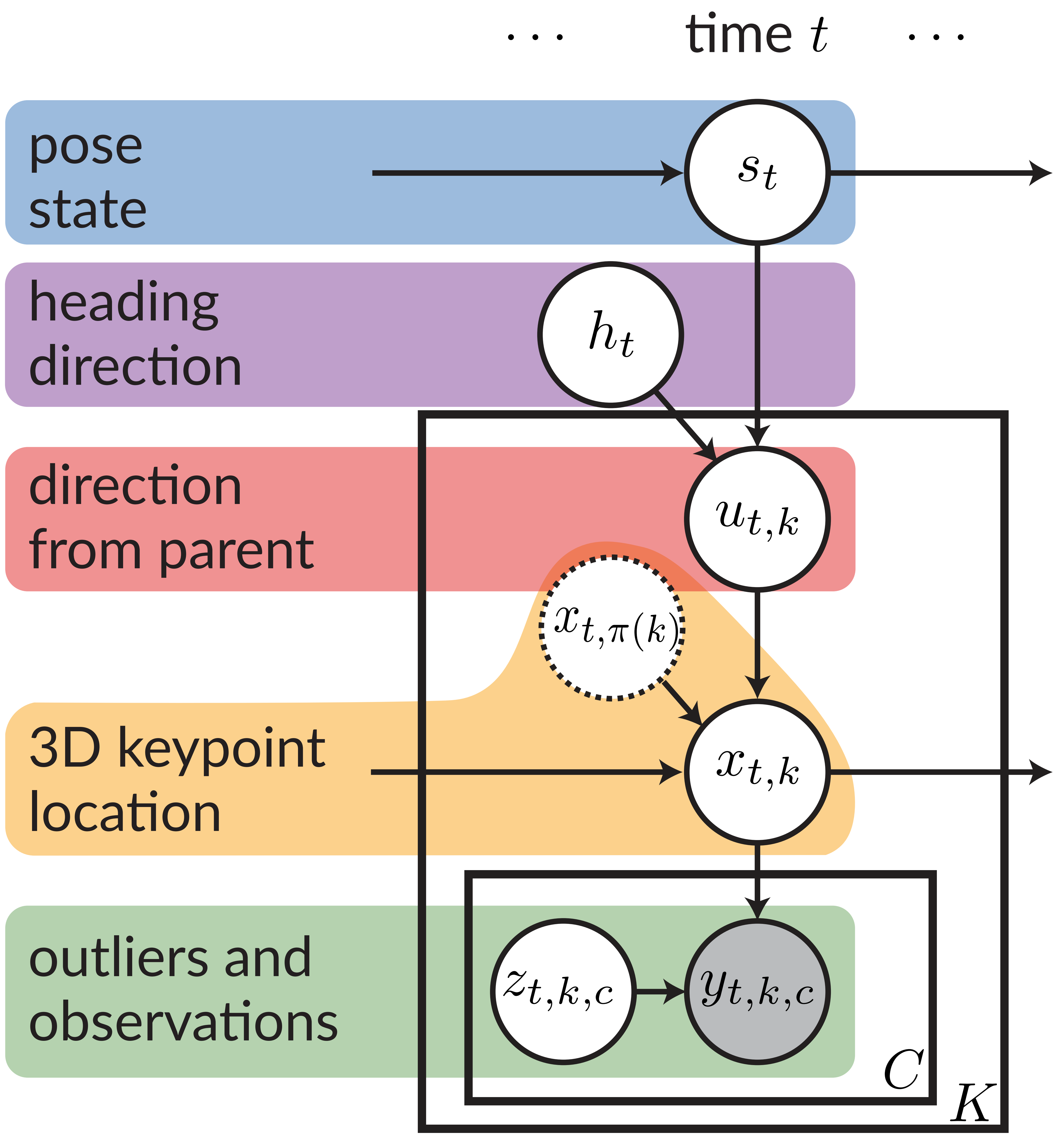
von Mises-Fisher distribution
on the sphere



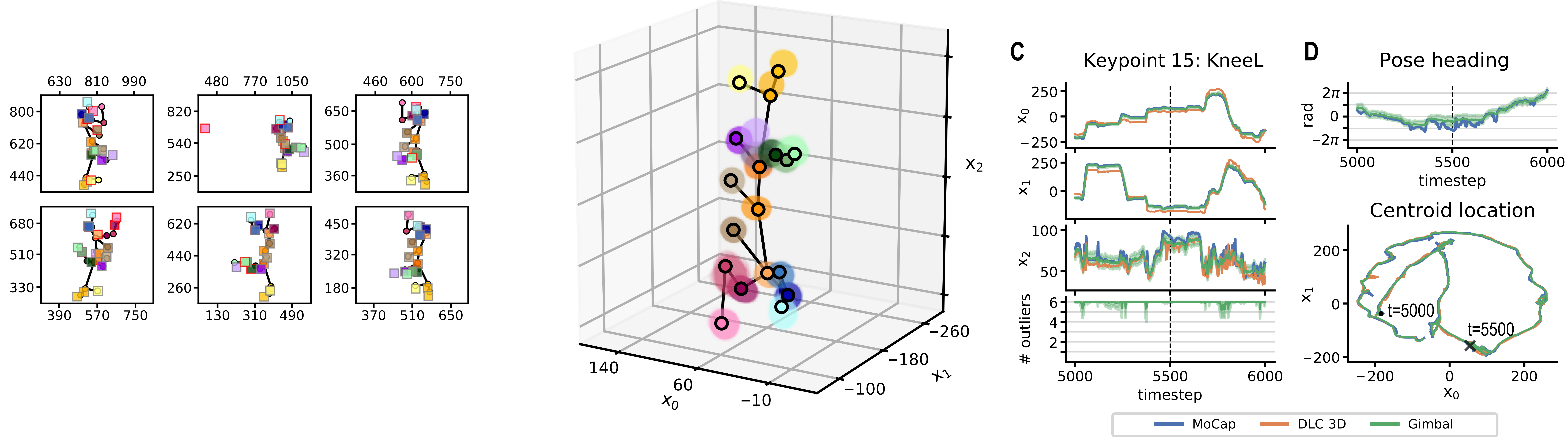
$$u_k \sim \mathrm{vMF}(\mu_{s_t,k}, \kappa_{s_t,k})$$

GIMBAL: Capturing correlations in direction vectors with pose states





GIMBAL yields posterior distributions on 3D pose given 2D estimates



Structured priors improve 3D pose estimates

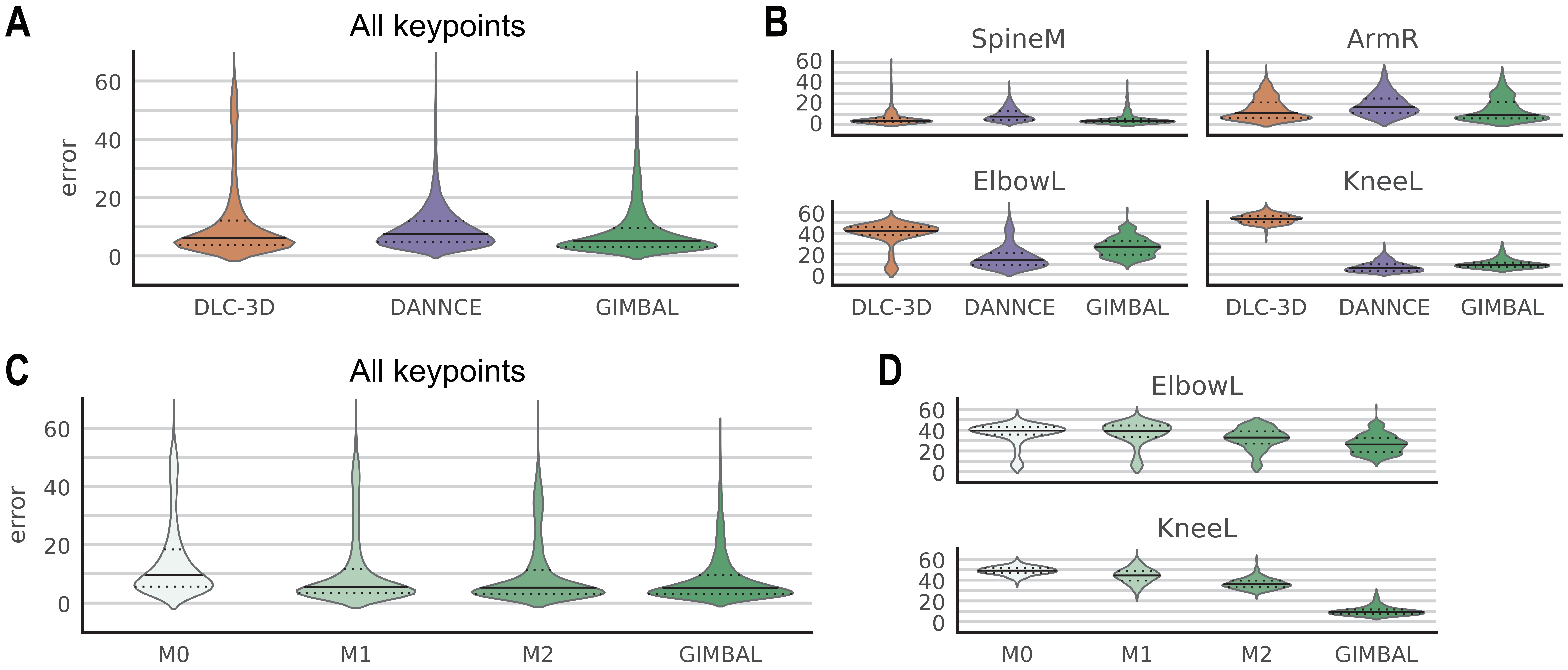
Table 1: Mean position error (MPE) averaged over all keypoints, for different pose estimation models. Calculated with unmodified predictions (raw) and after applying rigid Procrustes analysis (RPA). Units: mm.

	DLC-3D	DANNCE	GIMBAL
Raw	11.41	9.25	**8.01**
RPA	11.17	7.38	**6.88**

Table 2: Same as Table 1, with results for special submodels of GIMBAL.

	M0	M1	M2	GIMBAL
Raw	14.96	10.71	9.65	**8.01**
RPA	14.07	10.43	8.97	**6.88**

Structured priors improve 3D pose estimates



Conclusion

- **Precise behavior quantifications** are critical for understanding how neural activity relates to behavioral output.
- **Markerless pose tracking** methods have made it much easier to obtain such quantifications.
- **Convolutional neural networks** are naturally suited to this task.
- With **transfer learning,** we can leverage state-of-the-art deep networks for image classification to warm-start pose tracking.
- We can t**riangulate 3D pose** from 2D images using projecting geometry and spatiotemporal priors.

Further reading

- Datta, Sandeep Robert, et al. "Computational neuroethology: a call to action." *Neuron* 104.1 (2019): 11-24.
- Mathis, Alexander, et al. "DeepLabCut: markerless pose estimation of user-defined body parts with deep learning." *Nature neuroscience* 21.9 (2018): 1281-1289.
- Pereira, Talmo D., et al. "Fast animal pose estimation using deep neural networks." Nature methods 16.1 (2019): 117-125.
- He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.