Machine Learning Methods for Neural Data Analysis

Lecture 7: Pose Tracking to Encoding

Scott Linderman

STATS 220/320 (NBIO220, CS339N). Winter 2021.

Announcements

- Lab 3 Errata:
  - Background footprints should be normalized so that $\|u_0\|_2 = 1$, not maximum value one as it said in a code comment.
  - Lower diagonal of G should have $-e^{-1/\tau}$.
  - There are multiple ways to implement the norm constraint in CVX. The key is that they have a scalar Lagrange multiplier. Try

    ```
    lagrange_multiplier=constraints[0].dual_value
    ```

    (without the `[0]` at the end) if you have “float not subscriptable” errors.

Markerless pose tracking

- Convolutional neural networks (CNNs) are well-suited to **template matching** for key point detection.
- Pretrained CNNs for image classification offer good features for animal pose tasks. This is called **transfer learning**.
- With a relatively **small number of training examples** we can get decent accuracy on held-out data.

Markerless pose tracking

- Convolutional neural networks (CNNs) are well-suited to **template matching** for key point detection.
- Pretrained CNNs for image classification offer good features for animal pose tasks. This is called **transfer learning**.
- With a relatively **small number of training examples** we can get decent accuracy on held-out data.

Agenda

1. Pose tracking practicalities
2. [Briefly] Triangulating 3D pose from multiple 2D views

Unit 2: Encoding and Decoding Neural Spike Trains

3. Convolutional encoding models for retinal ganglion cells

Pose tracking practicalities

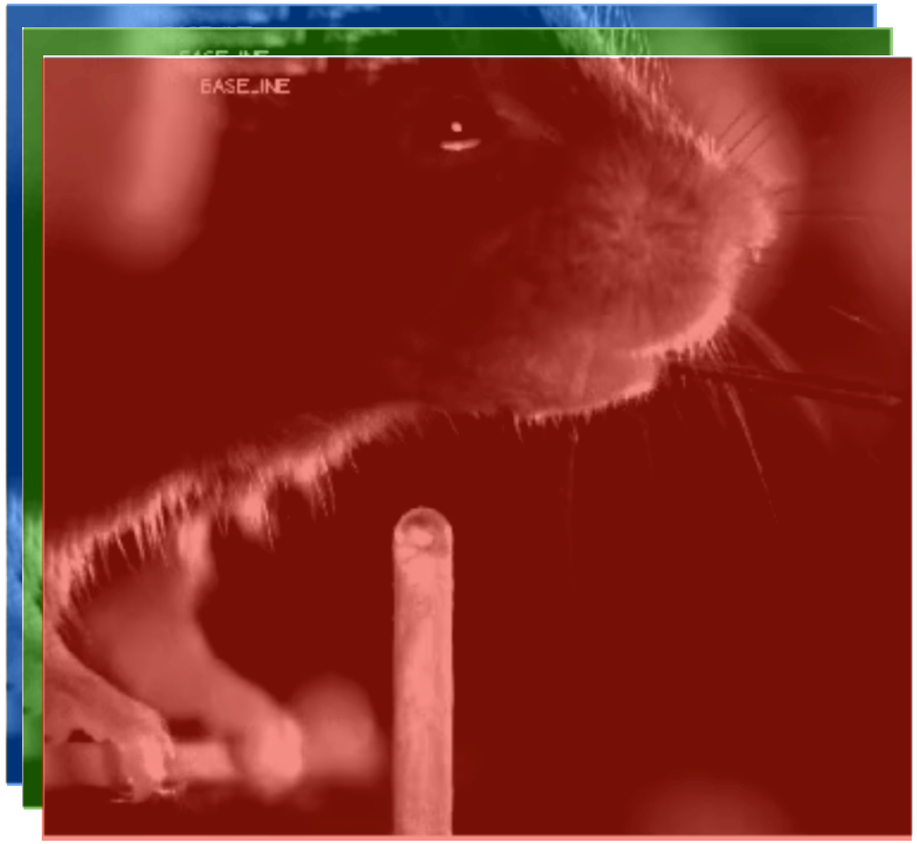
Data preprocessing

- Image is HxWx3 (RGB color channels). Permute to 3xHxW.
- In the basic model, reduce to grayscale.
- In pertained models,“normalize” to have specific mean and variance, per PyTorch convention.
- Targets come in (x,y) coordinates. Convert them to binary masks.
- Downsample the image and target by a factor of 2 for speed in the lab.
- Dilate the target to allow some room for error in the predictor.

Pose tracking practicalities

Convert to grayscale in the basic “template matching” model

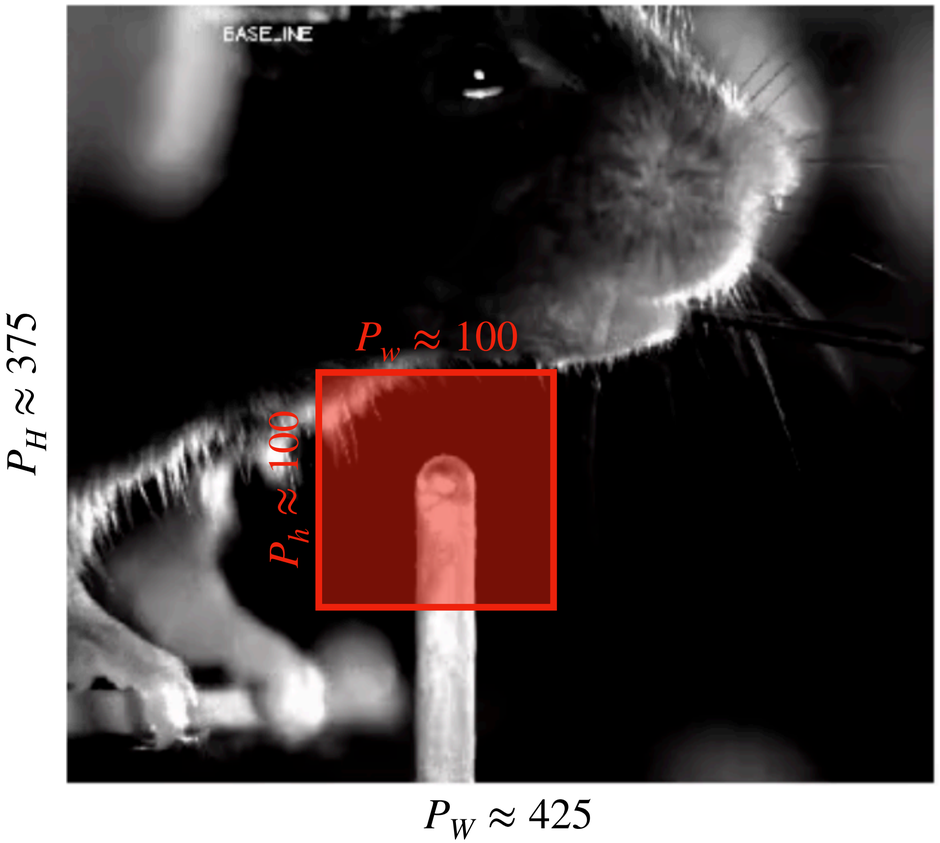
- The images actually have 3 color channels, even though they look grayscale.
- Reduce to single channel by taking the mean over the color axis.
- Again, you can think of this as weight sharing in your template.



Pose tracking practicalities

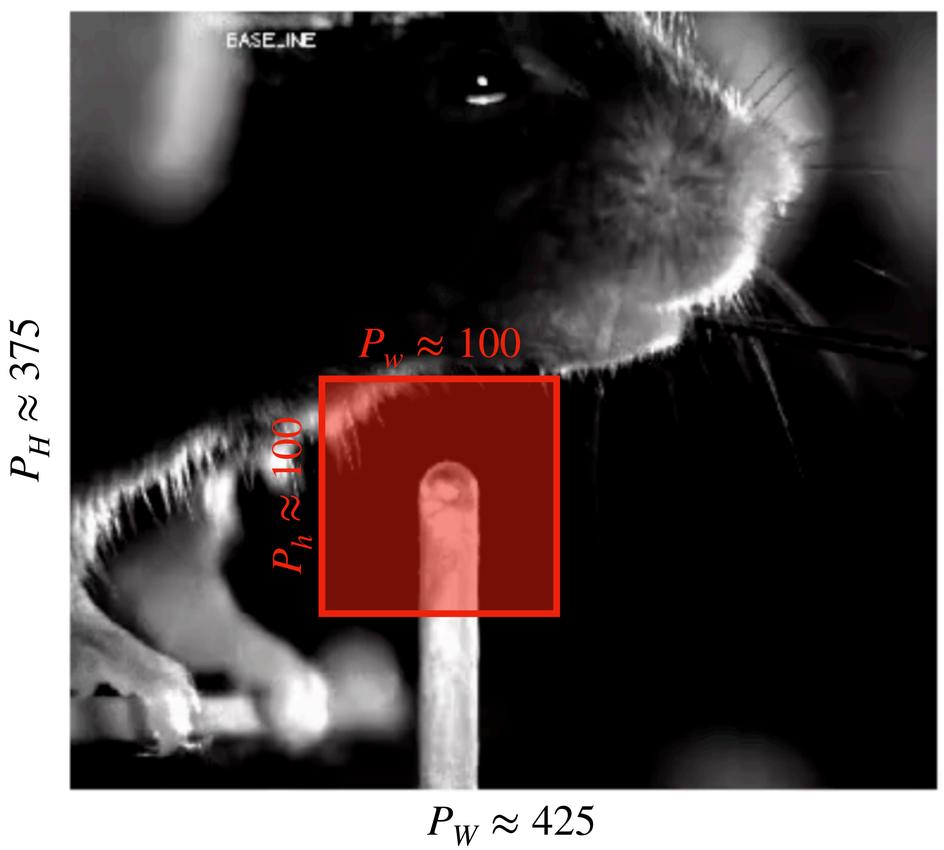
Further downsampling in the basic model

- Even in the downsampled images, we may need a pretty large patch to identify keypoints.
- Idea: just keep on downsampling!
- We'll treat the downsampling as part of the model though, because the resnet will make better use of the high-resolution (375x425) data.
- Downsampling can be implemented many ways: `avg_pool2d`, `conv_2d`, `interpolate`...

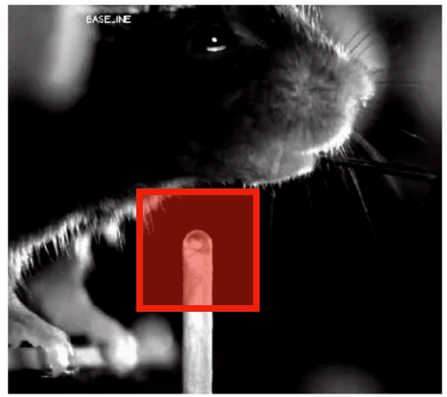


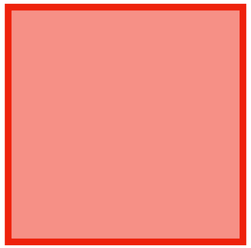
Pose tracking practicalities

Downsampling as weight sharing



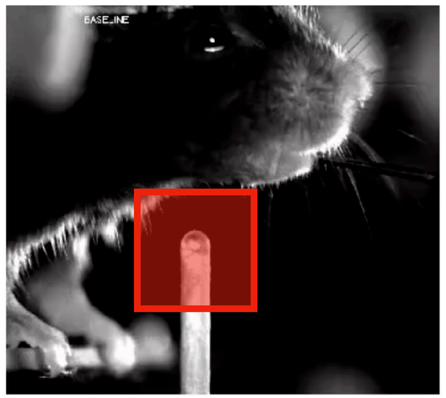
`interpolate(0.5)`
`avg_pool2d(2)`



Equivalently  = `upsample(`  `, 2)`

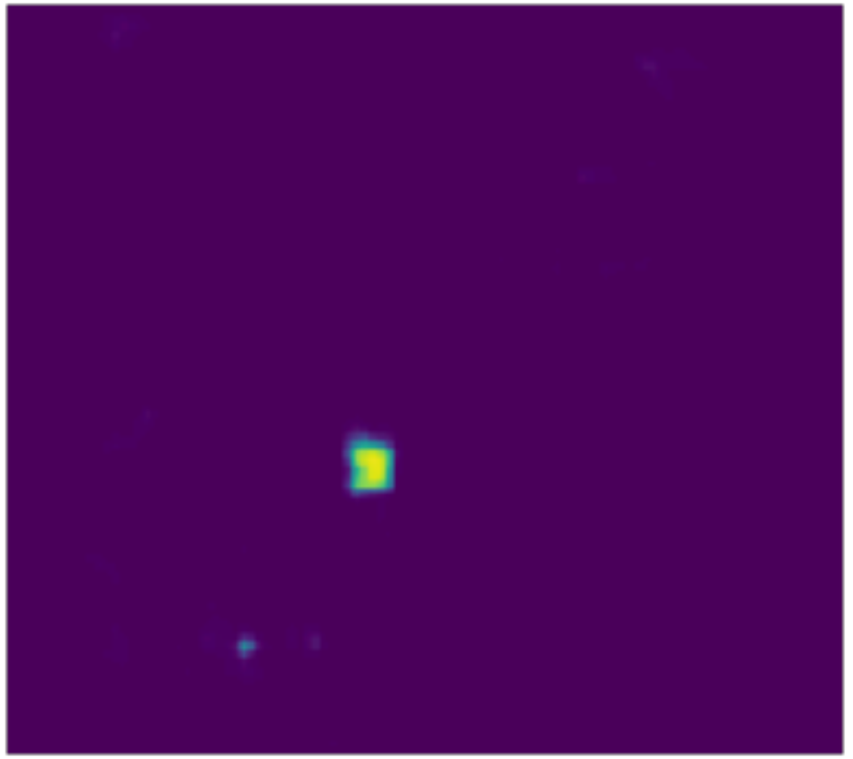
Pose tracking practicalities

Interpolating to get full-resolution logits and probabilities



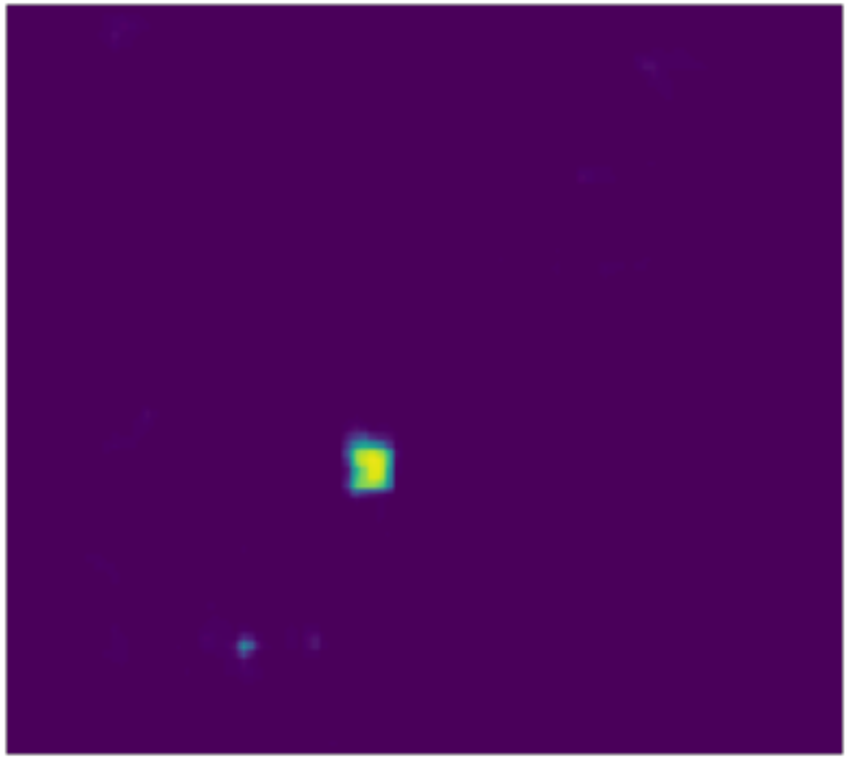
interpolate

Joystick logits U



$\sigma(\cdot)$

Joystick probabilities $\sigma(U)$



Pose tracking practicalities

Implementing a stable loss function

- Let
  - $X \in \mathbb{R}^{P_H \times P_W}$ denote the (preprocessed) input image
  - $Y_k \in \{0,1\}^{P_H \times P_W}$ denote the binary target for key point k.
  - $W_k \in \mathbb{R}^{P_h \times P_w}$ denote the weights for key point k.
- Let $U_k = X \star W_k \in \mathbb{R}^{P_H \times P_W}$ denote the "logits" for key point k; i.e. the sliding dot product of the image and weights.
- The Bernoulli log likelihood is

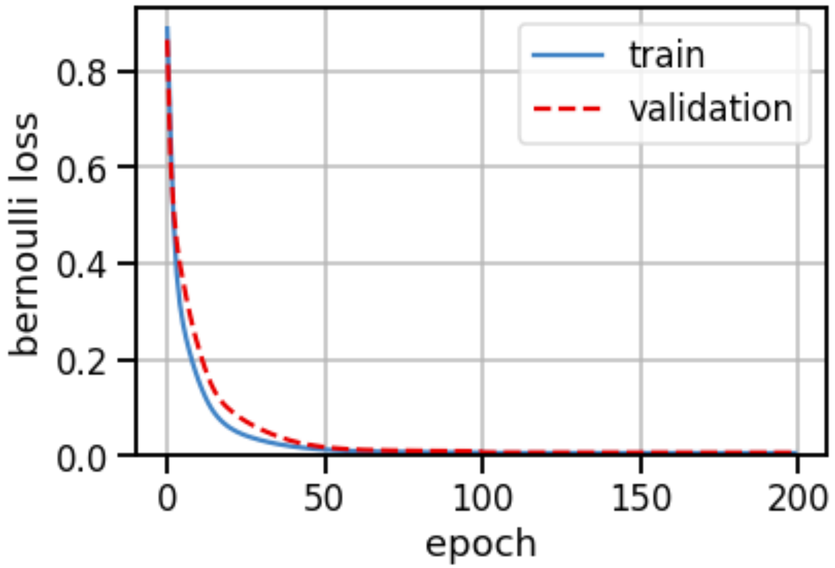
$$\log p(Y \mid X, W) = \sum_{i=1}^{H} \sum_{j=1}^{W} u_{kij} y_{kij} - \log(1 + e^{u_{kij}})$$

- We need to be careful evaluating it because $e^{u_{kij}}$ can blow up. Solution: `logsumexp` trick.

Pose tracking practicalities

Stochastic gradient descent

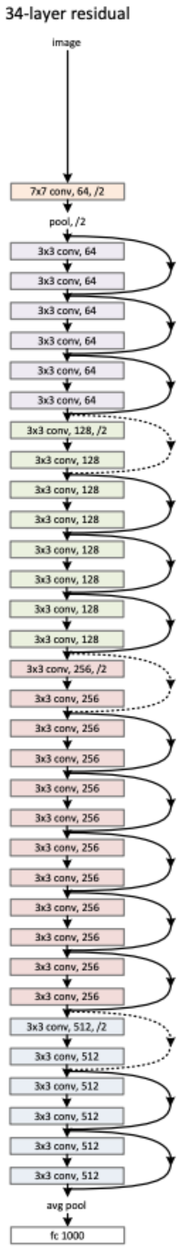
- For each training image, **evaluate the loss** function.
- Compute **gradient** with respect to the weights using reverse-mode automatic differentiation.
- Take a step in the direction of the **gradient** with current learning rate α.
- An **epoch** is a single sweep through the training data.
- After each epoch, compute the loss on held-out **validation data.**
- Optionally, decay the learning rate according to a **schedule**; e.g. every 100 epochs scale α by factor of 0.1.



Pose tracking practicalities

Residual network (resnet50) architecture

- ResNet50 (He et al, 2015) has a bunch of “bottleneck” layers.
- Each bottleneck **downsamples** by a factor of 2, leading to 4x fewer “pixels.”
- But, each bottleneck **increases the number of channels** by a factor of 2 as well.
- Result: 2x compression with fewer pixels but more channels.
- Contrast this with the naive downsampling we proposed for the basic model.



3D Triangulation

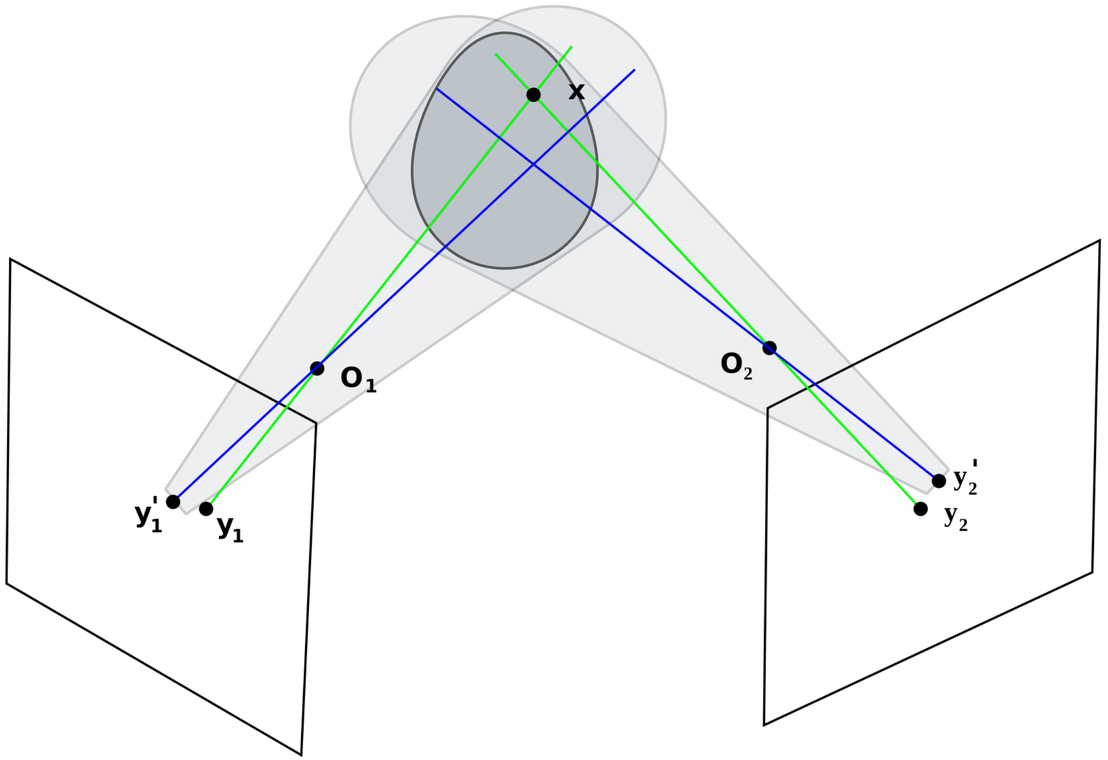
Basic triangulation

Projective geometry

- Projective geometry makes far away objects appear smaller.

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

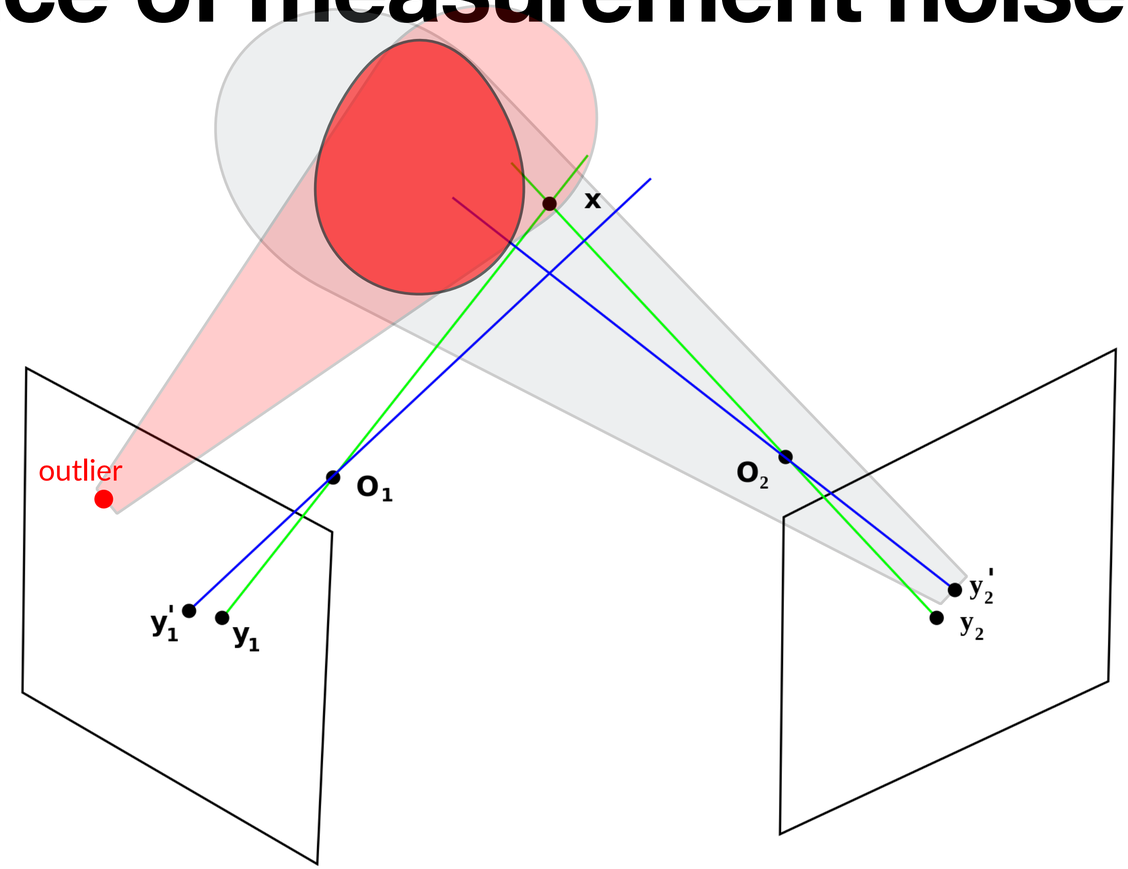
$$f_c(\vec{x}) = \frac{1}{w}(u, v)^\top \text{ where } (u, v, w)^\top = A_c\vec{x} + b_c,$$



Modified from wikipedia.org

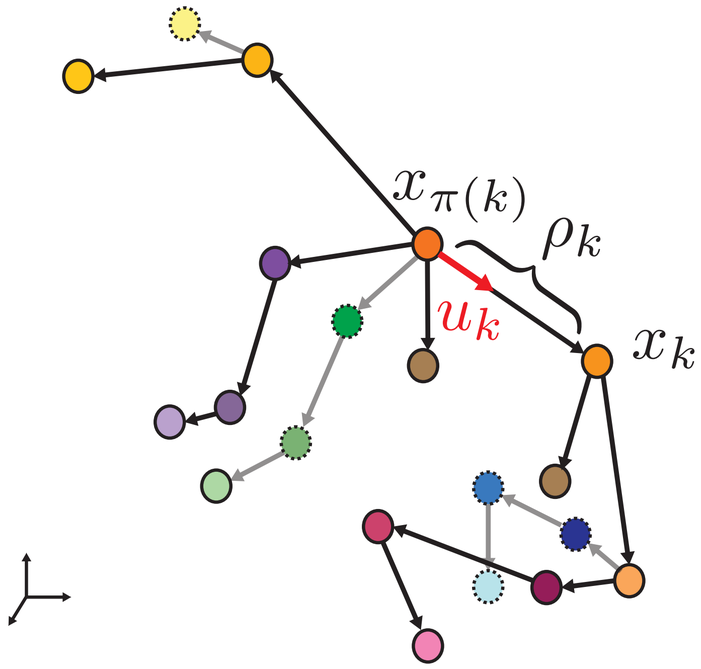
Triangulation in the presence of measurement noise

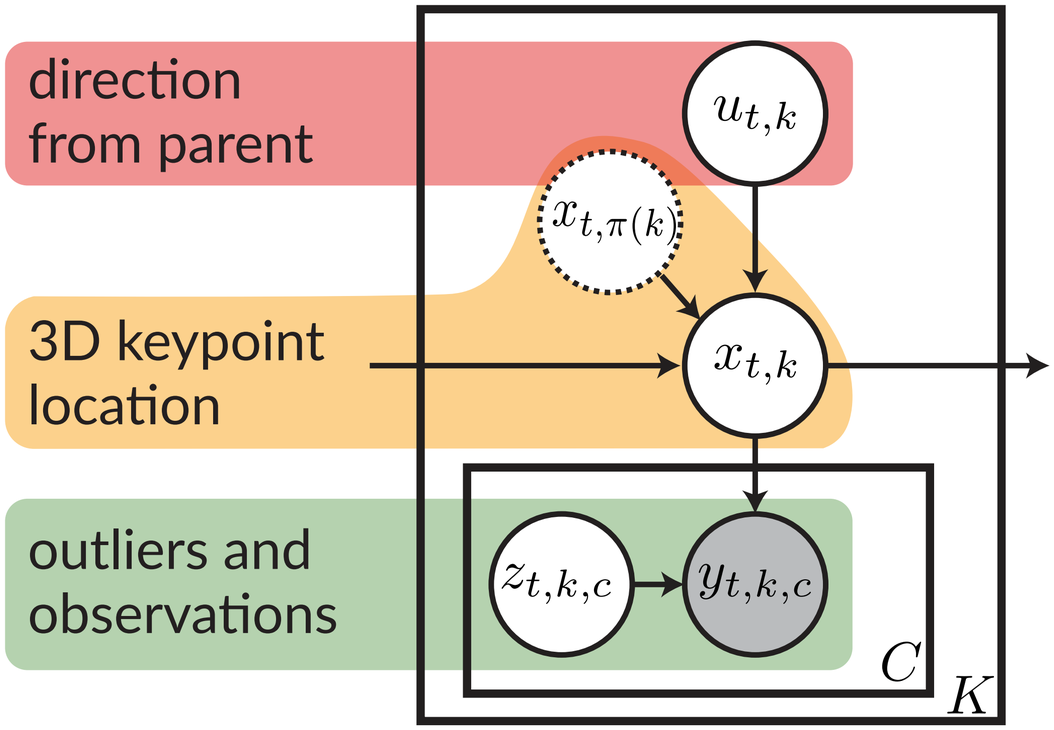
- Projective geometry makes far away objects appear smaller.
- Outliers in 2D estimates can severely affect 3D triangulation.
- Typical approaches:
  - More data
  - Temporal constraints
  - Median filtering (DLC-3D) / RANSAC
  - Robust noise models



Modified from wikipedia.org

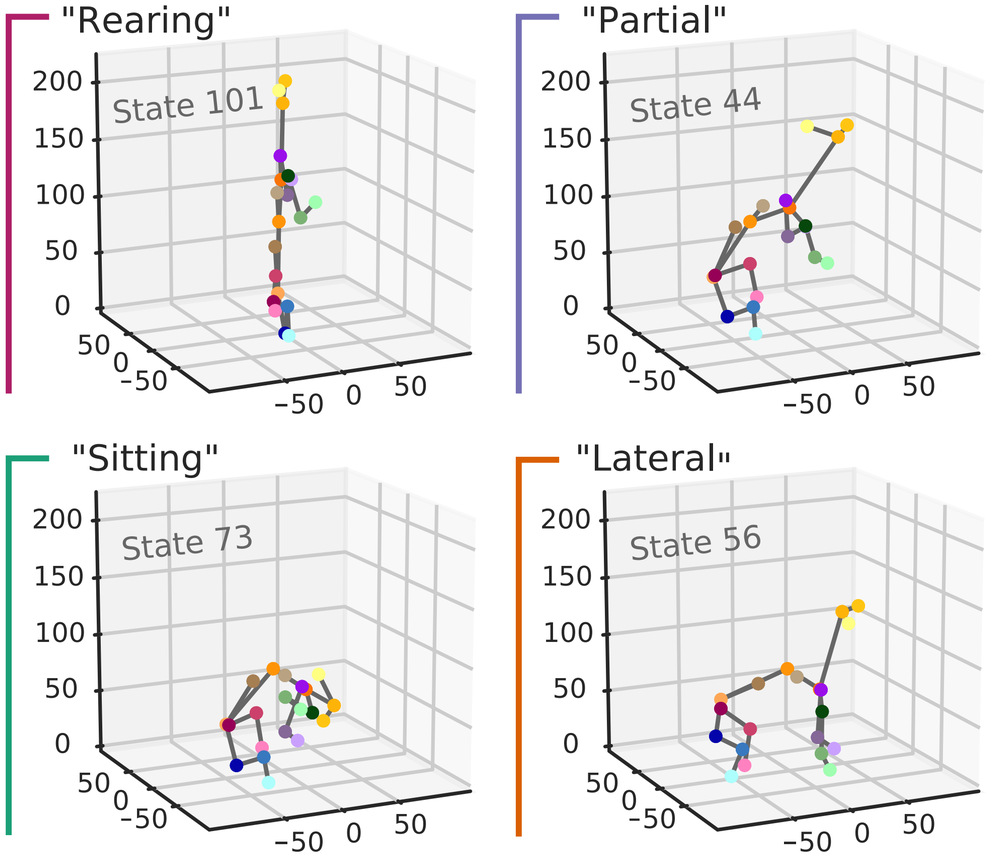
Bayesian triangulation

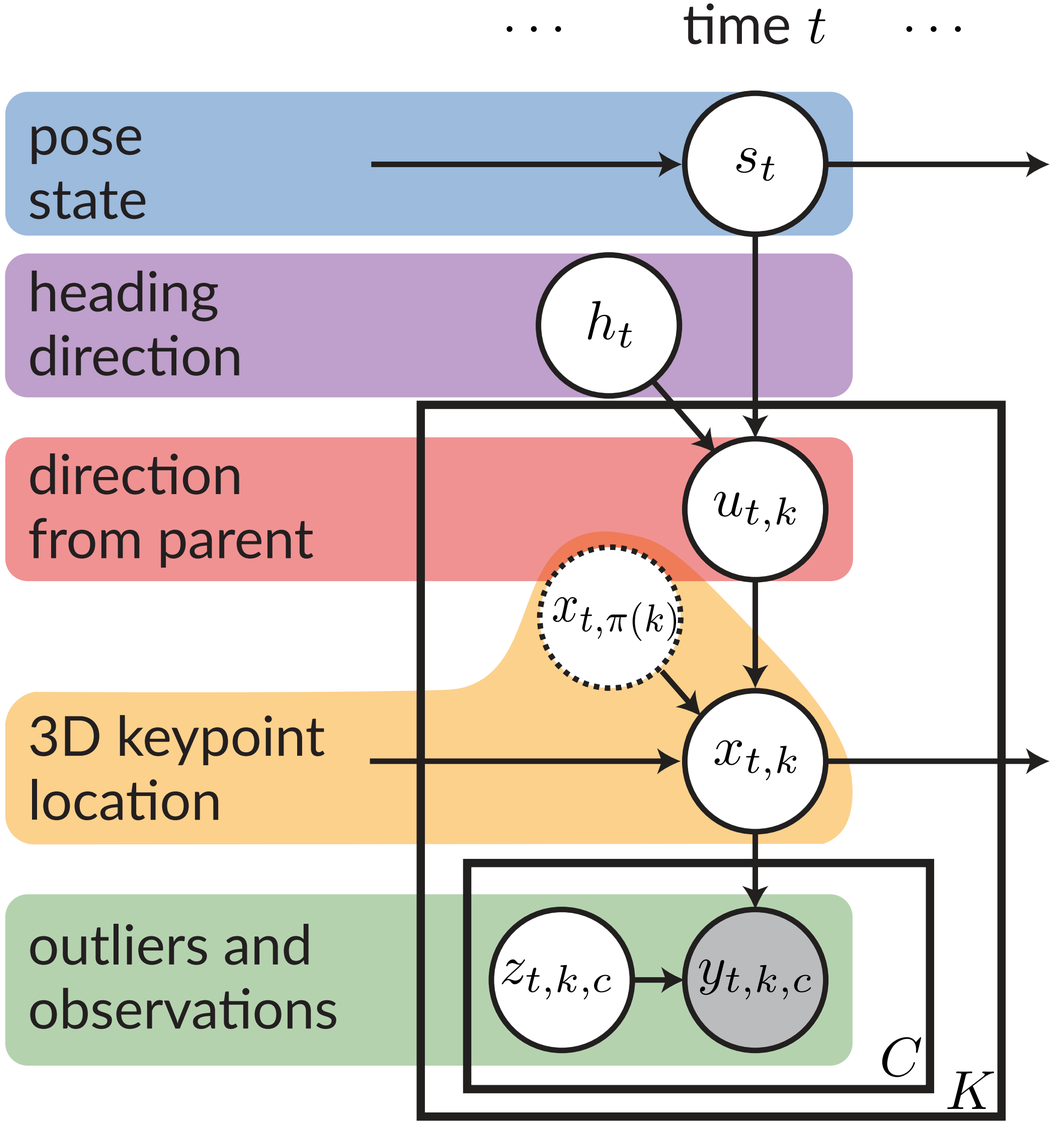




Zhang et al (*AISTATS*, 2021)

GIMBAL: Capturing correlations in direction vectors with pose states



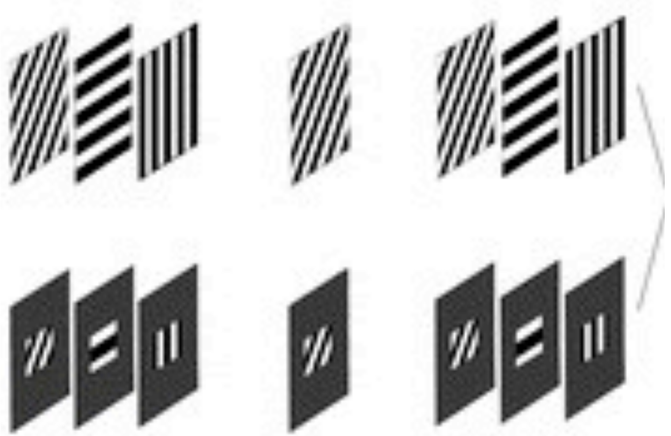


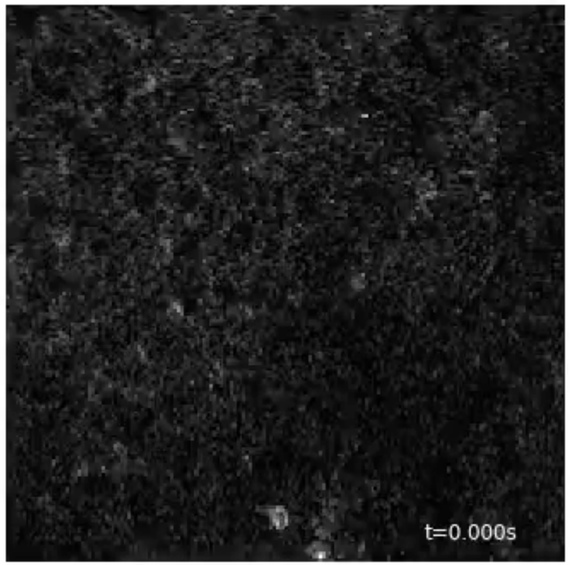
Zhang et al (*AISTATS*, 2021)

Unit 2: Encoding and decoding

Big picture

Stimulus —*encoding*→ Neural Recording —*decoding*→ Behavioral Video

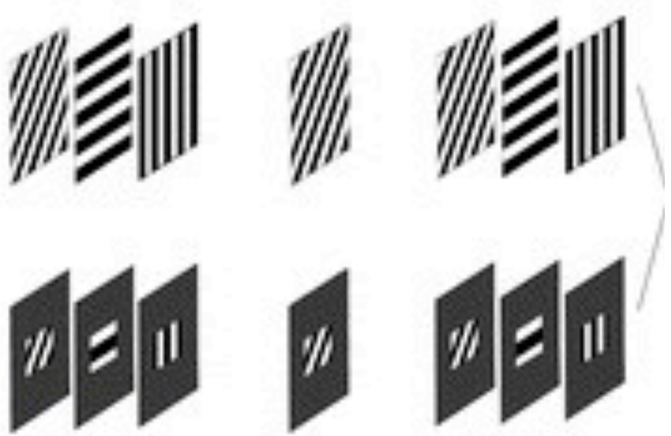


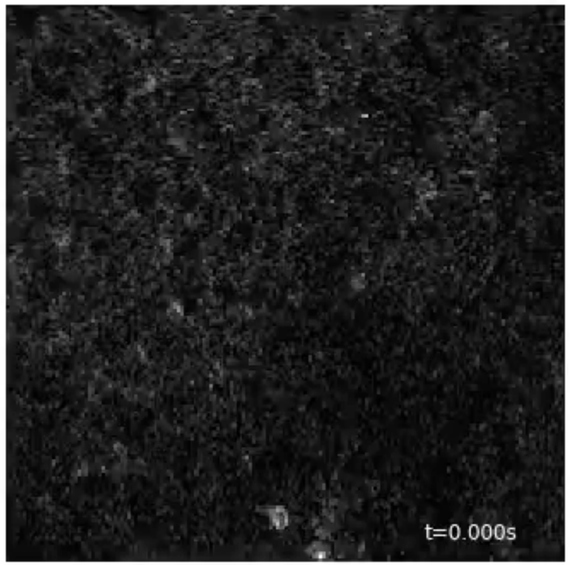


In statistics lingo, it's all regression.

Big picture

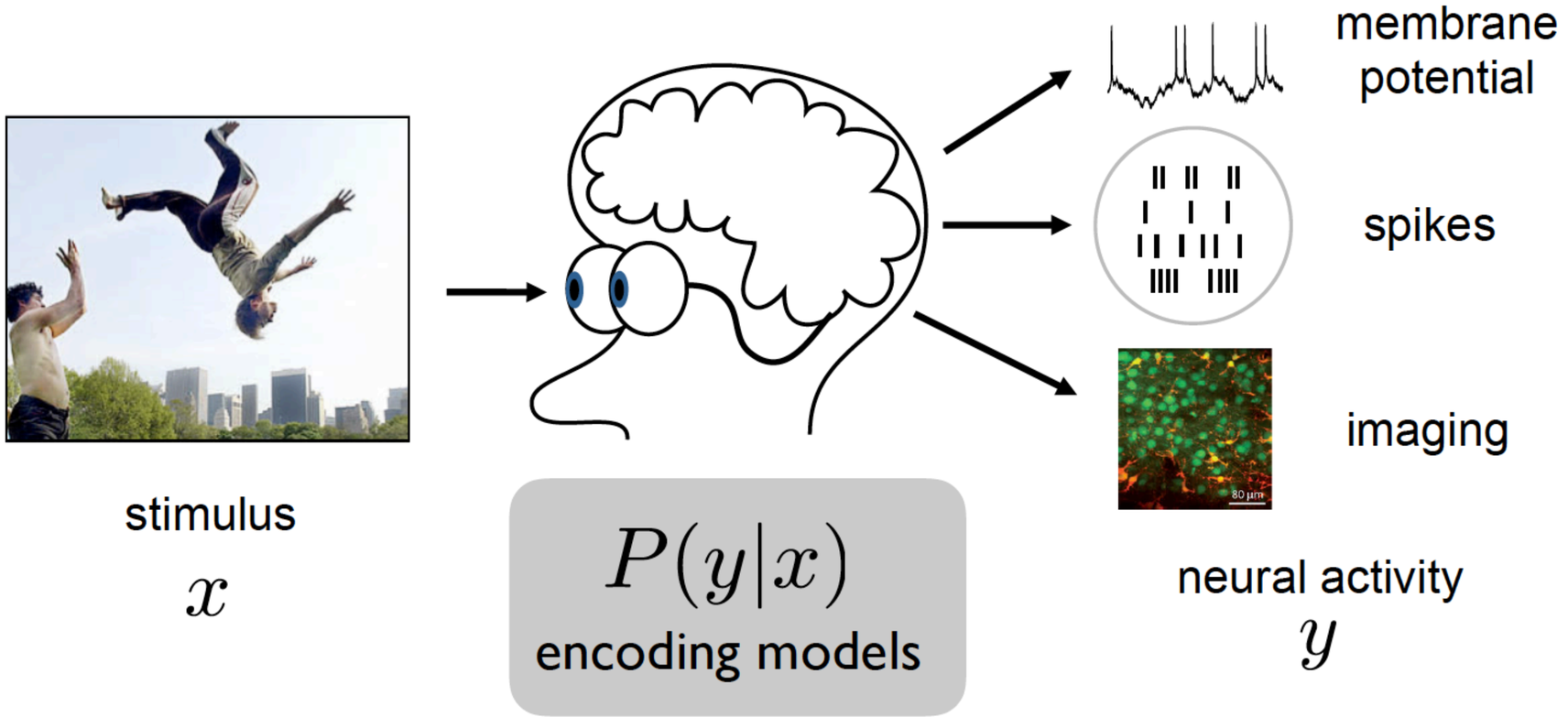
Stimulus —*encoding*→ Neural Recording —*decoding*→ Behavioral Video





In statistics lingo, it's all regression.

Encoding models

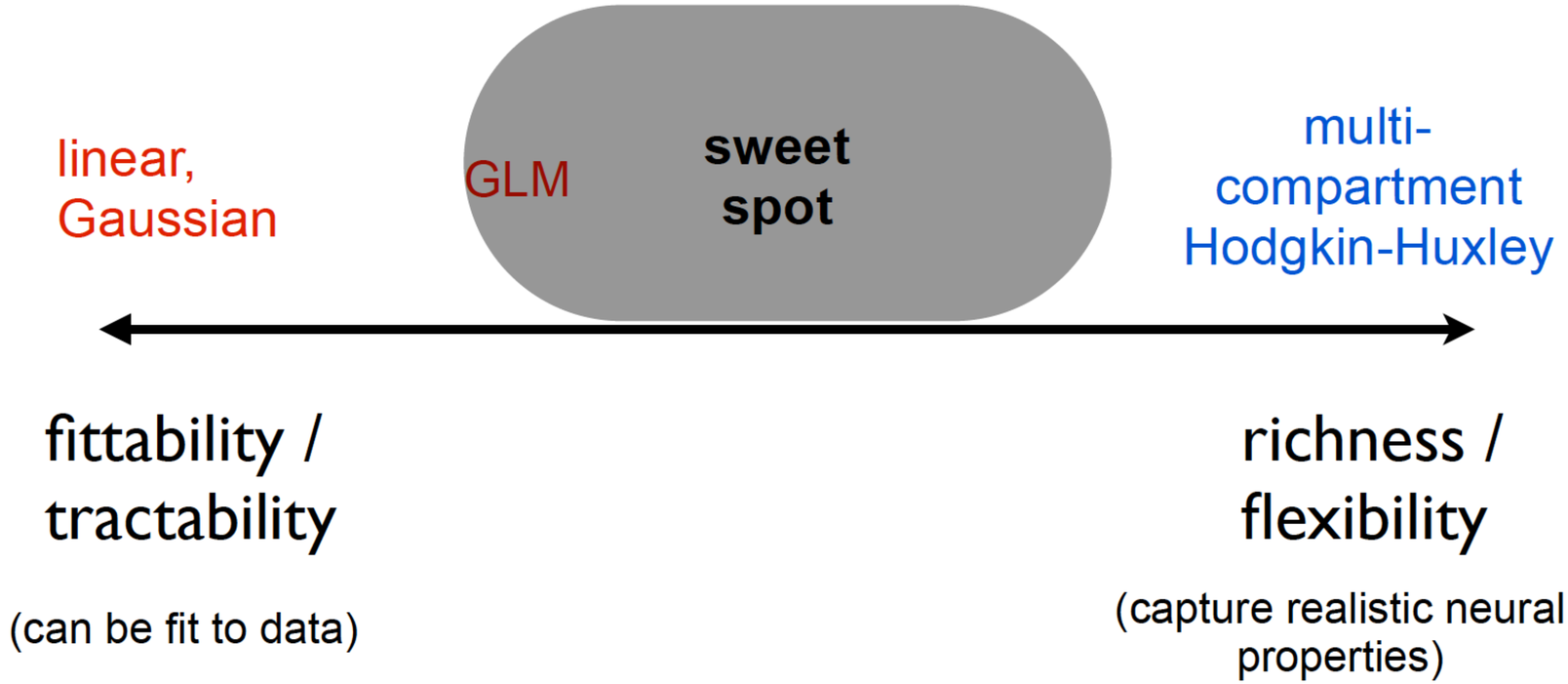


Approach:
- develop flexible statistical models of P(y|x)
- quantify information carried in neural responses

Slides from Jonathan Pillow

Encoding models

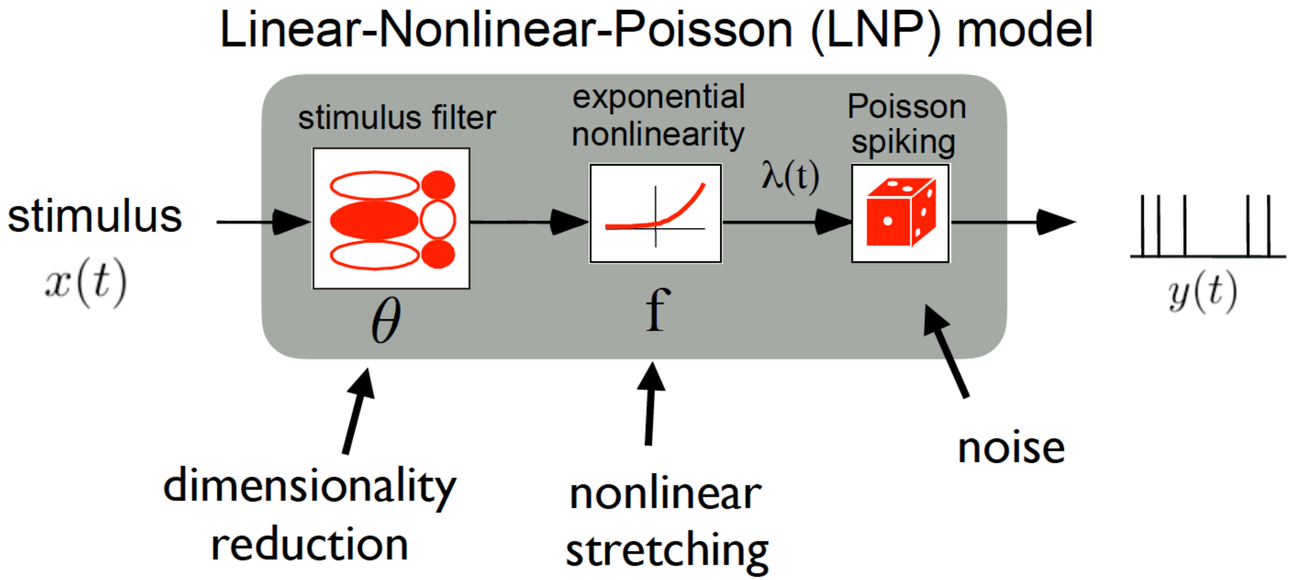
“Fittability” vs Flexibility



Slides from Jonathan Pillow

Encoding models

Basic linear-nonlinear-Poisson (LNP) model



spike rate $\lambda = f(\vec{k} \cdot \vec{x})$

spike count $y \sim \mathrm{Poiss}(\lambda)$

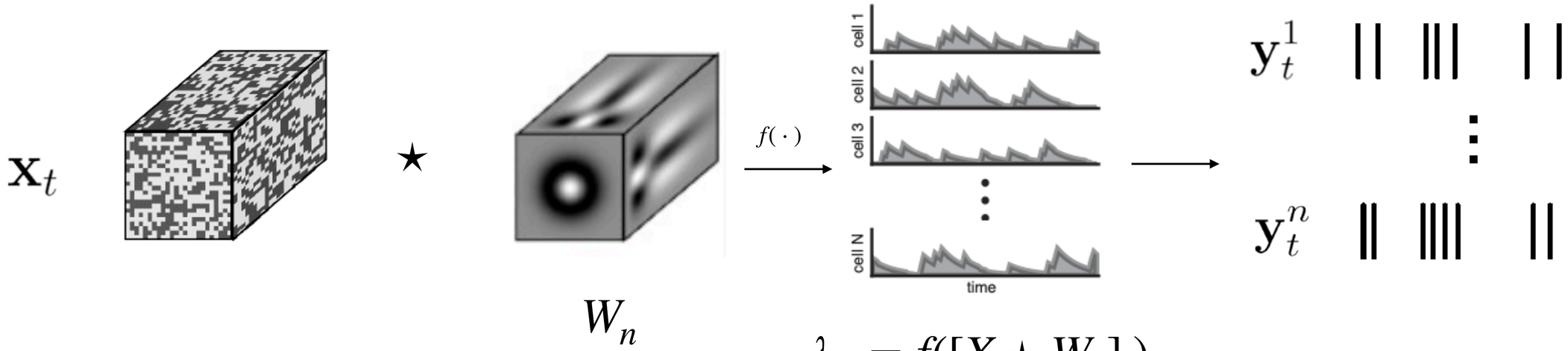
In statistics lingo, a generalized linear model (GLM).

Slides from Jonathan Pillow

Encoding models

Generalized linear models

Stimulus **Filter Weights** **Firing Rates** **Neural Recording**



$$\lambda_{nt} = f([X \star W_n]_t)$$

Slides from Jonathan Pillow

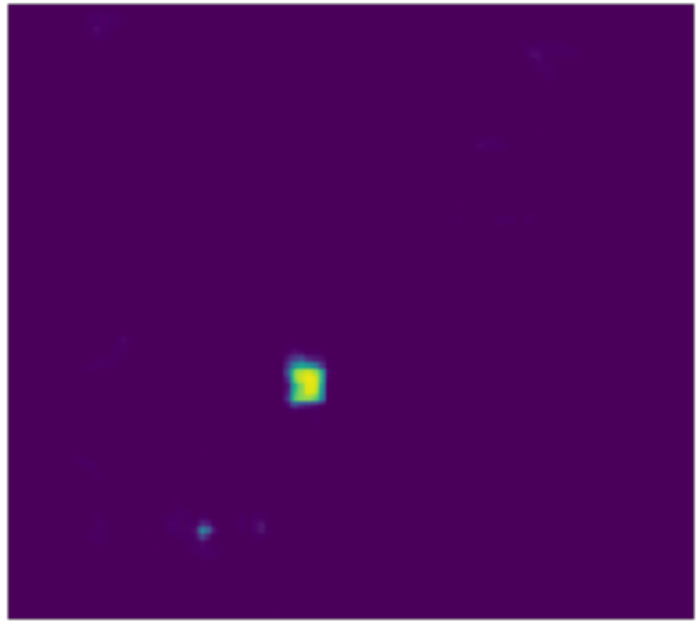
Encoding models

This should look familiar…

input

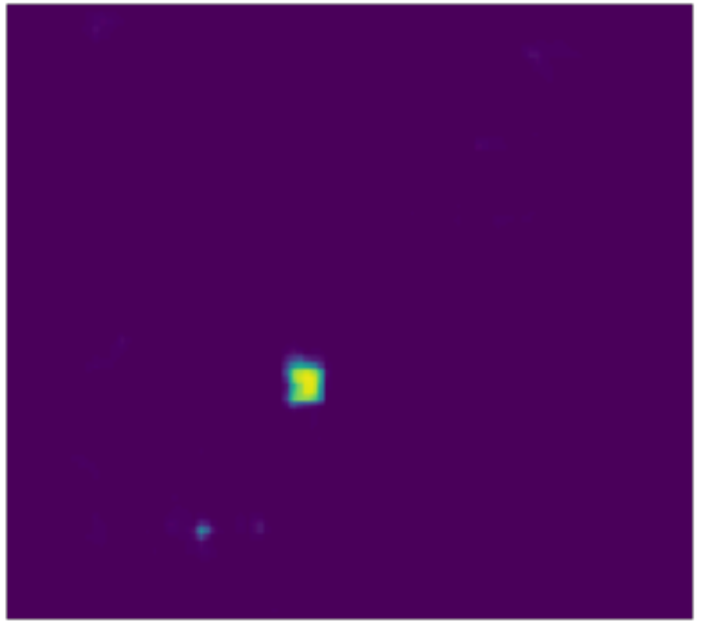
$\star$

logits



$\sigma(\cdot)$

probabilities



Encoding models

Convolutional models of retinal ganglion cell responses

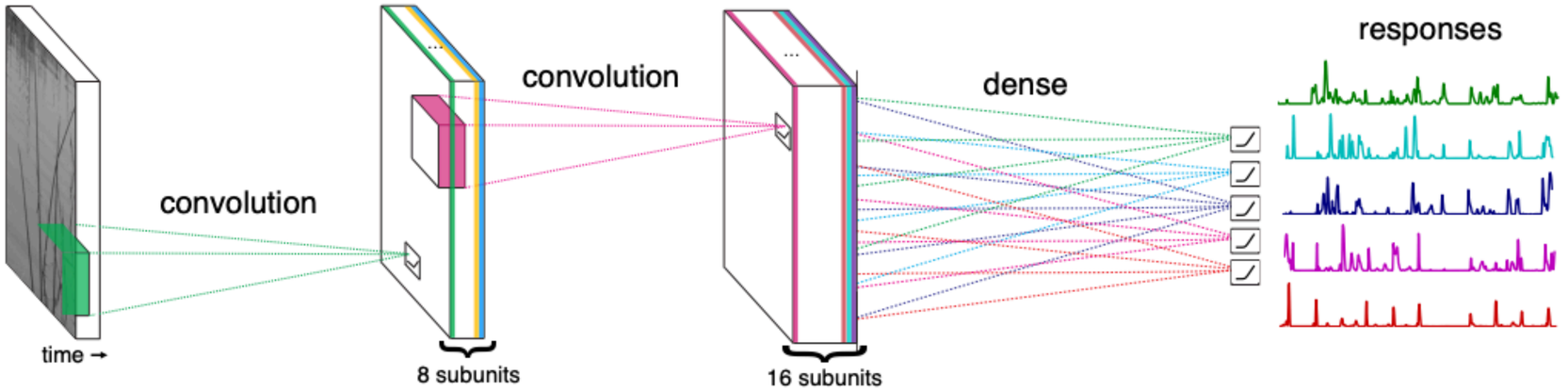
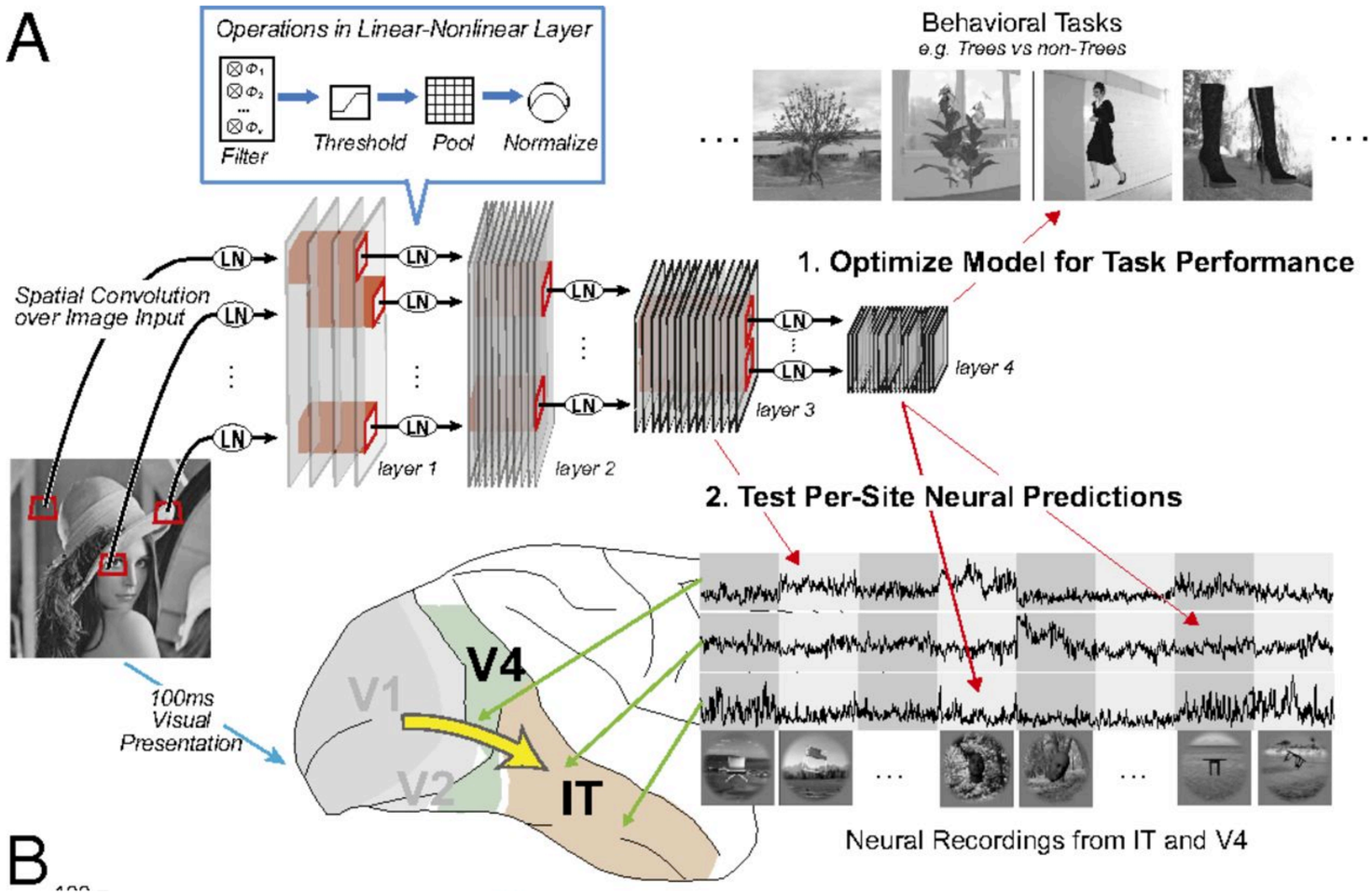


Figure 1: A schematic of the model architecture. The stimulus was convolved with 8 learned spatiotemporal filters whose activations were rectified. The second convolutional layer then projected the activity of these subunits through spatial filters onto 16 subunit types, whose activity was linearly combined and passed through a final soft rectifying nonlinearity to yield the predicted response.

McIntosh et al (*NeurIPS*, 2016)

Encoding models

Convolutional models of retinal ganglion cell responses



Yamins et al (*PNAS*, 2014)

Conclusion

- **Convolutional neural networks** are naturally suited to pose tracking tasks.
- With **transfer learning,** we can leverage state-of-the-art deep networks for image classification to warm-start pose tracking.
- We can **triangulate 3D pose** from 2D images using projecting geometry and spatiotemporal priors.
- CNNs will be a recurring theme: not only are they useful for signal extraction, they’re also our **state-of-the-art models for neural encoding.**

Further reading

- Mathis, Alexander, et al. "DeepLabCut: markerless pose estimation of user-defined body parts with deep learning." *Nature neuroscience* 21.9 (2018): 1281-1289.

- He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

- Pillow, Jonathan. *Cosyne Tutorial,* 2018. http://pillowlab.princeton.edu/pubs/pillow_TutorialSlides_Cosyne2018.pdf