Distribution Matching, Maximum Entropy, GANs, and all that

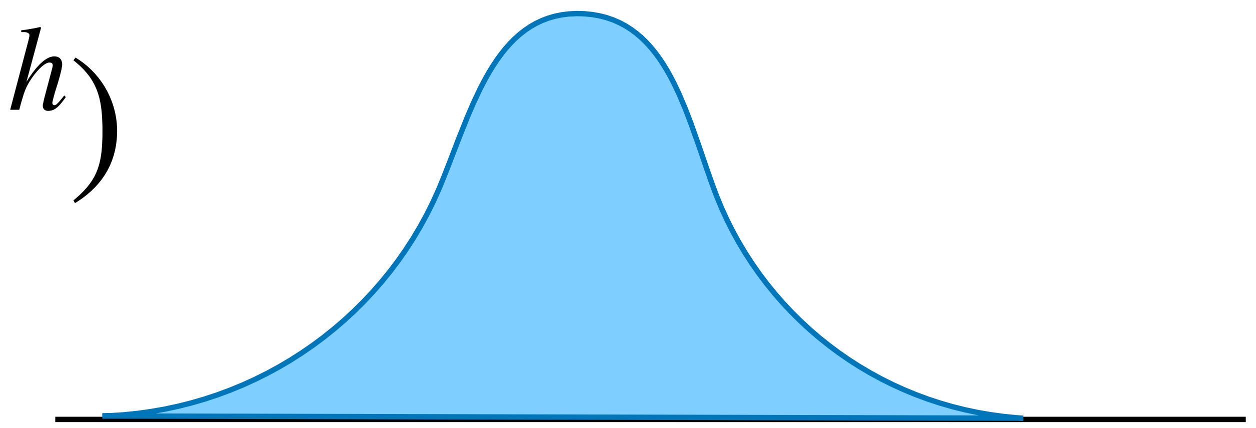
Sanjiban Choudhury

Cornell Bowers CIS
Computer Science

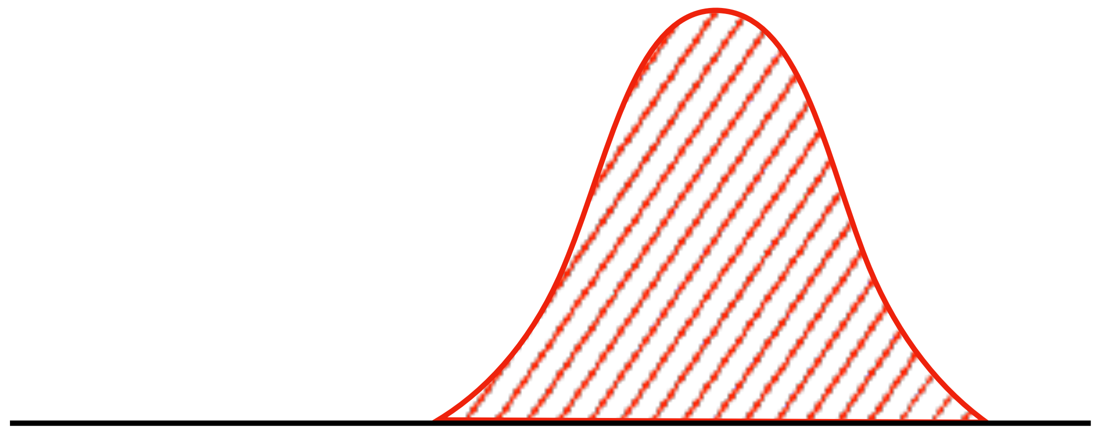
Imitation Learning is NOT blindly copying the expert's actions

The Distribution Matching Problem

$P_{expert}(\xi^h)$



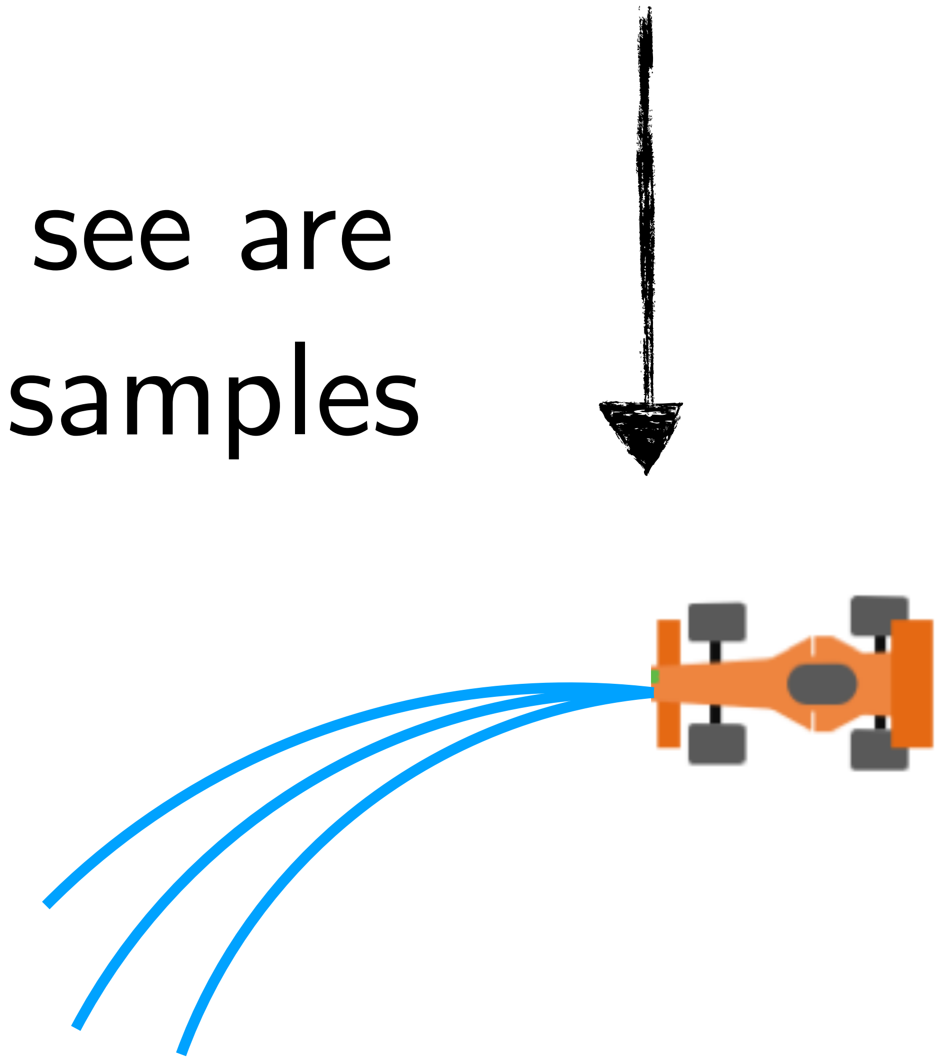
(Unknown) expert distribution



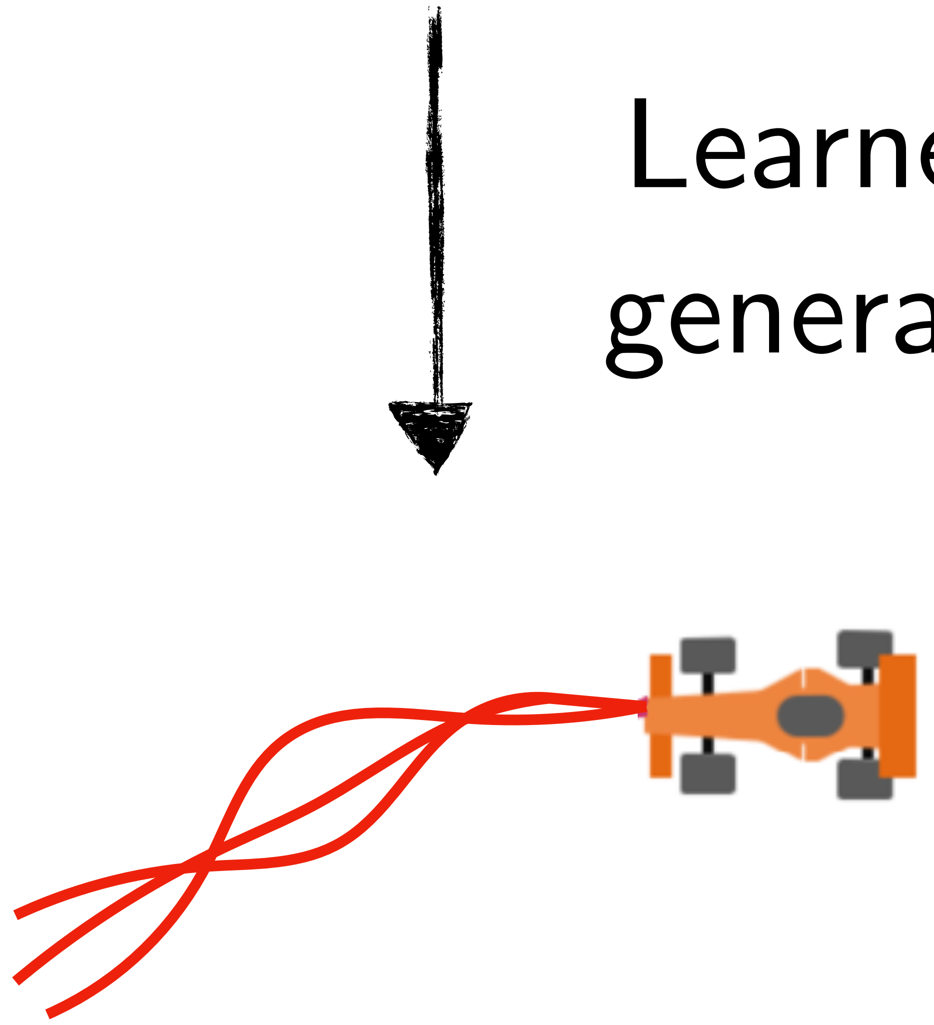
$P_\theta(\xi)$

Learn distribution over trajectories

All we see are expert samples



Learner can also generate samples



What loss should we use?

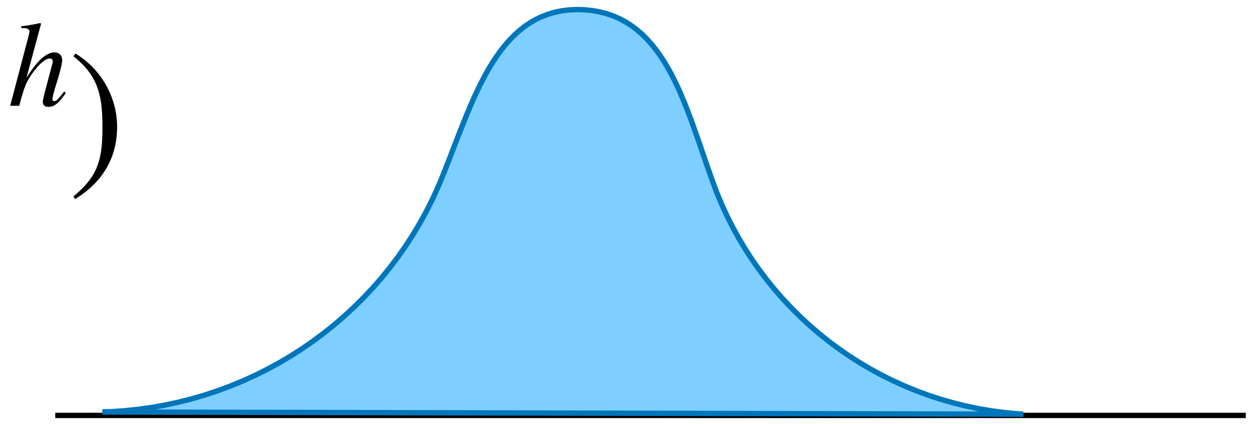
KL Divergence: A common measure!

Given two distributions P(x) and Q(x)

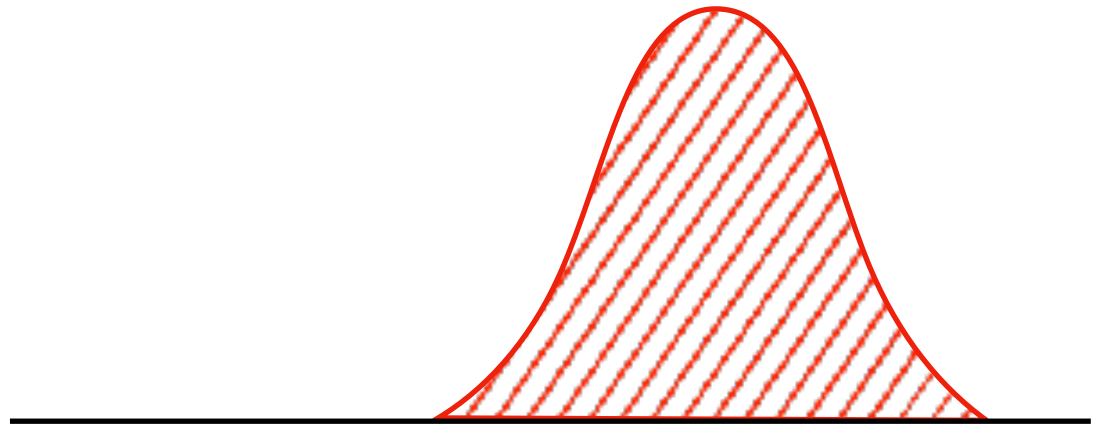
$$D_{KL}(P||Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

KL Divergence: A common measure!

$P_{expert}(\xi^h)$



(Unknown) expert distribution



$P_\theta(\xi)$

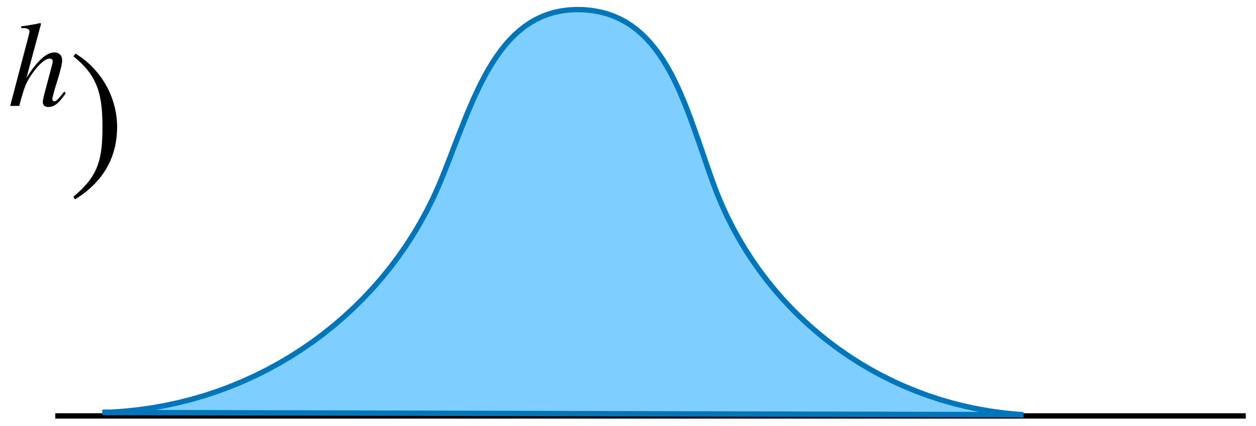
Learn distribution over trajectories

$$D_{KL}(P_{expert}||P_\theta) = \sum_{\xi} P_{expert}(\xi)\log\frac{P_{expert}(\xi)}{P_\theta(\xi)}$$

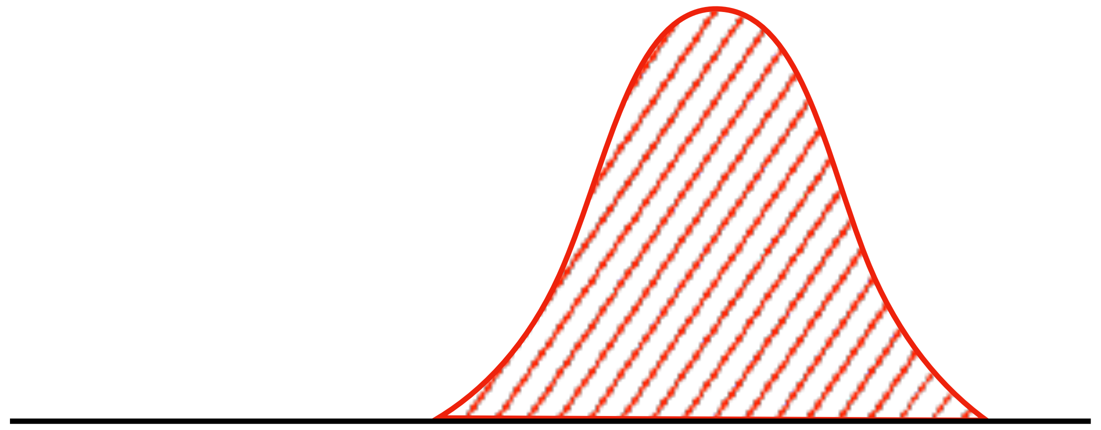
Can we $\min_\theta D_{KL}(P_{expert}||P_\theta)$ if we don't know P_{expert}?

KL Divergence: A common measure!

$P_{expert}(\xi^h)$



(Unknown) expert distribution



$P_\theta(\xi)$

Learn distribution over trajectories

Yes!

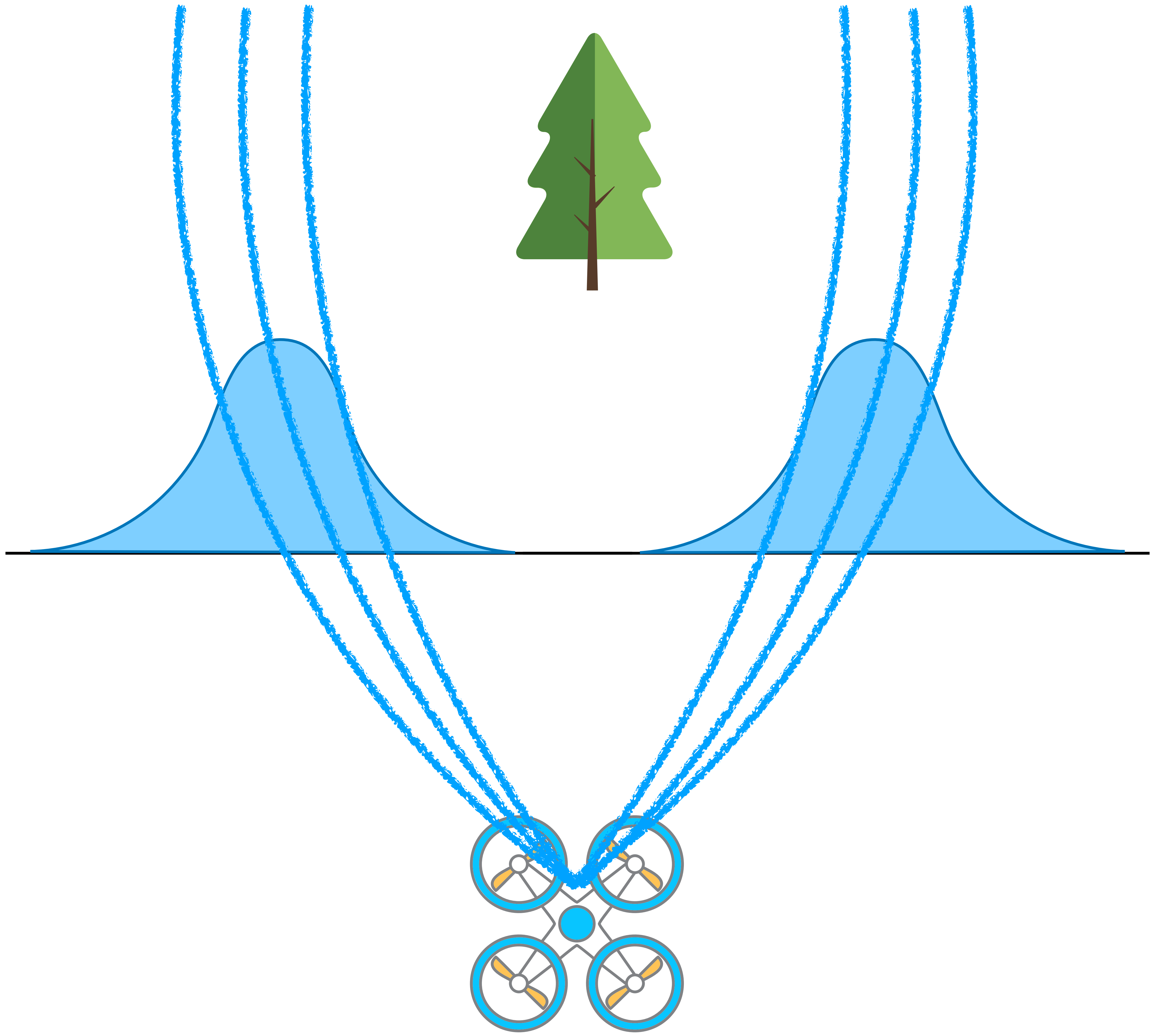
$$\min_\theta D_{KL}(P_{expert} || P_\theta) = \sum_\xi P_{expert}(\xi) \log \frac{P_{expert}(\xi)}{P_\theta(\xi)}$$

$$\min_\theta - \sum_\xi P_{expert}(\xi) \log P_\theta(\xi)$$

Only need samples from expert!

$$\min_\theta - \mathbb{E}_{\xi \sim P_{expert}(\xi)} \log P_\theta(\xi)$$

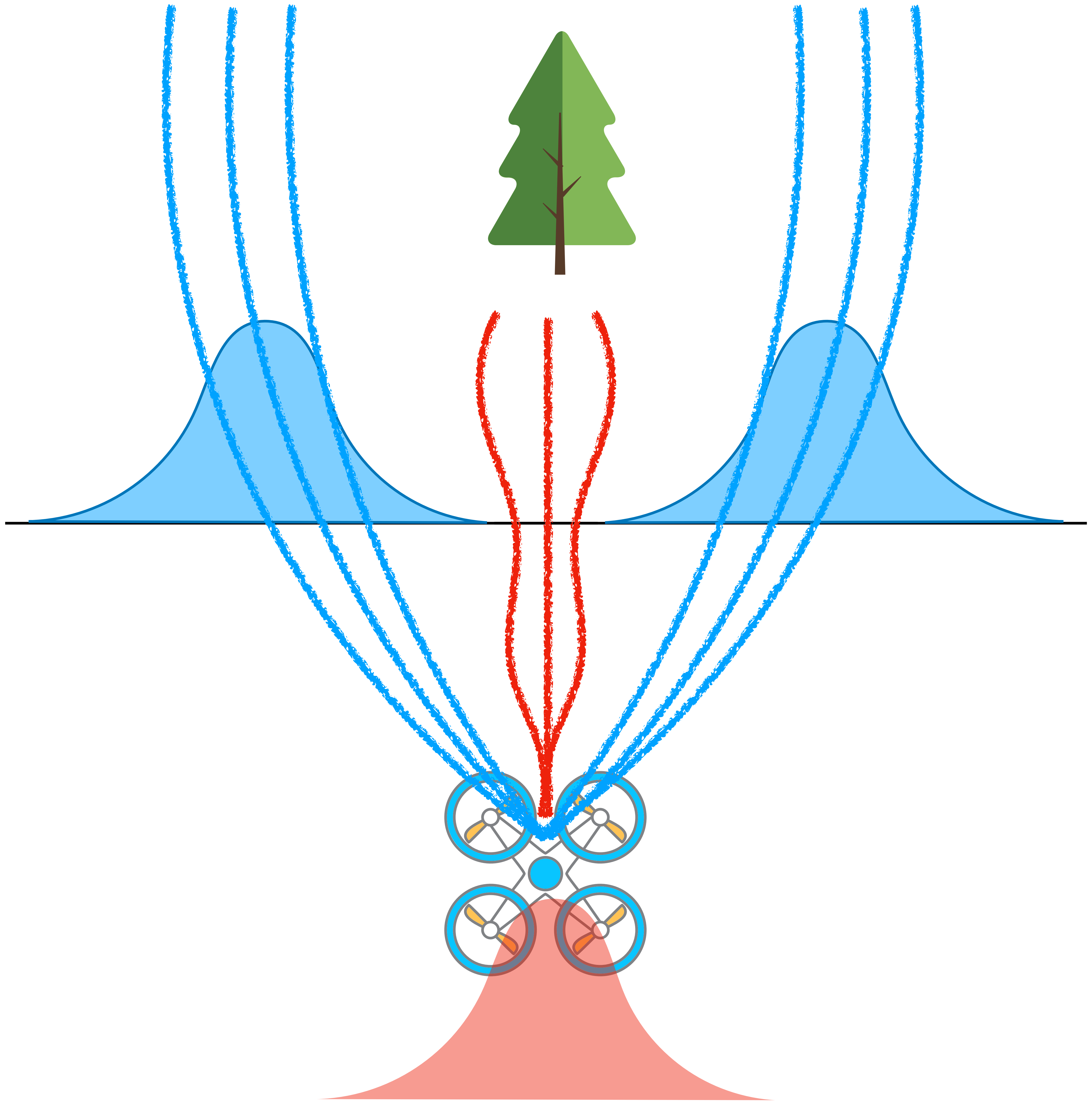
Flying through a forest



Expert flies left
and right of the tree

Given samples from expert

Flying through a forest



Expert flies left
and right of the tree

Given samples from expert

Let's say we want to learn
$P_\theta(\xi)$, a gaussian over traj

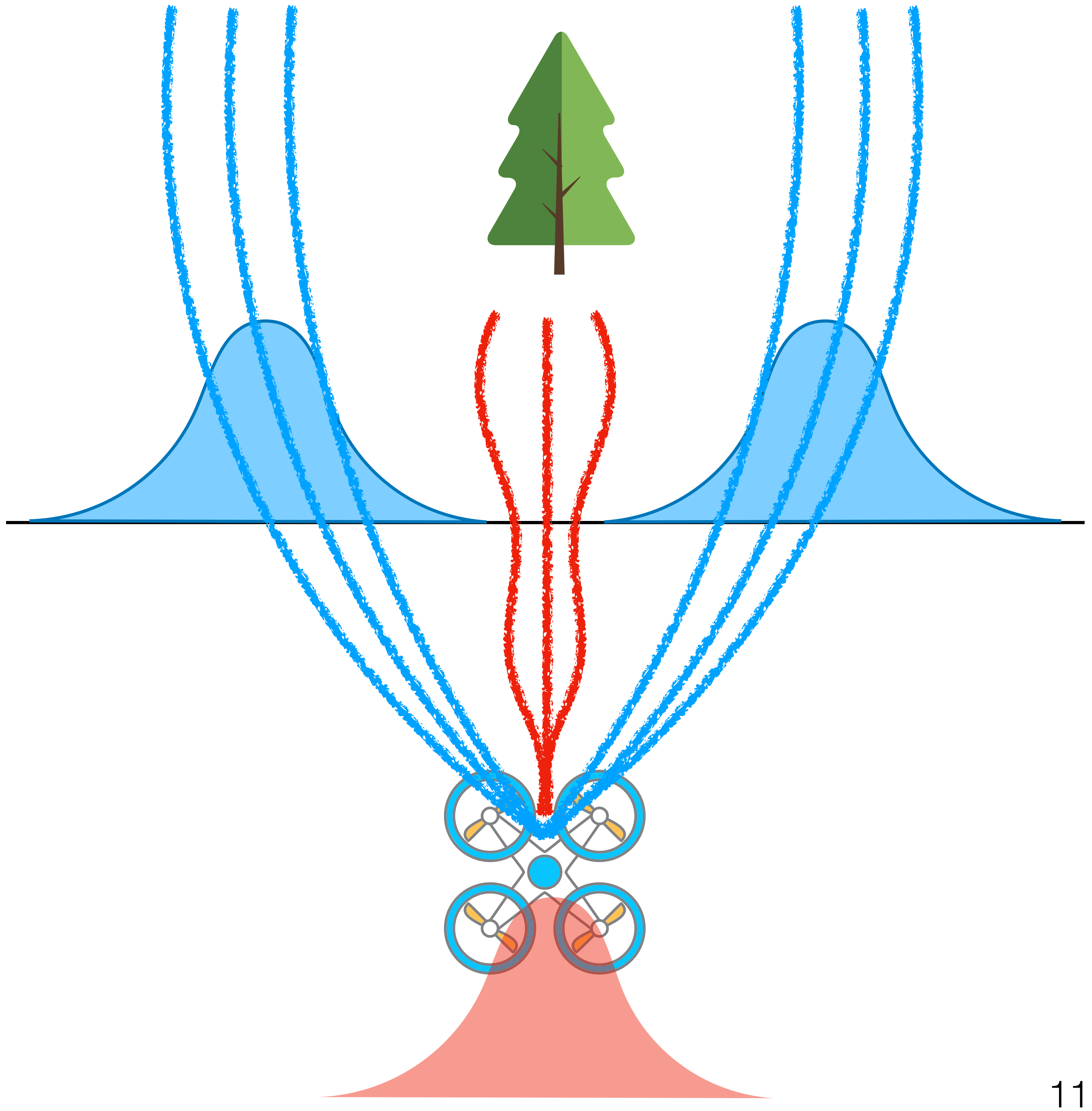
$$\min_\theta D_{KL}(P_{expert} || P_\theta)$$

What will we learn?

Activity!

Think-Pair-Share

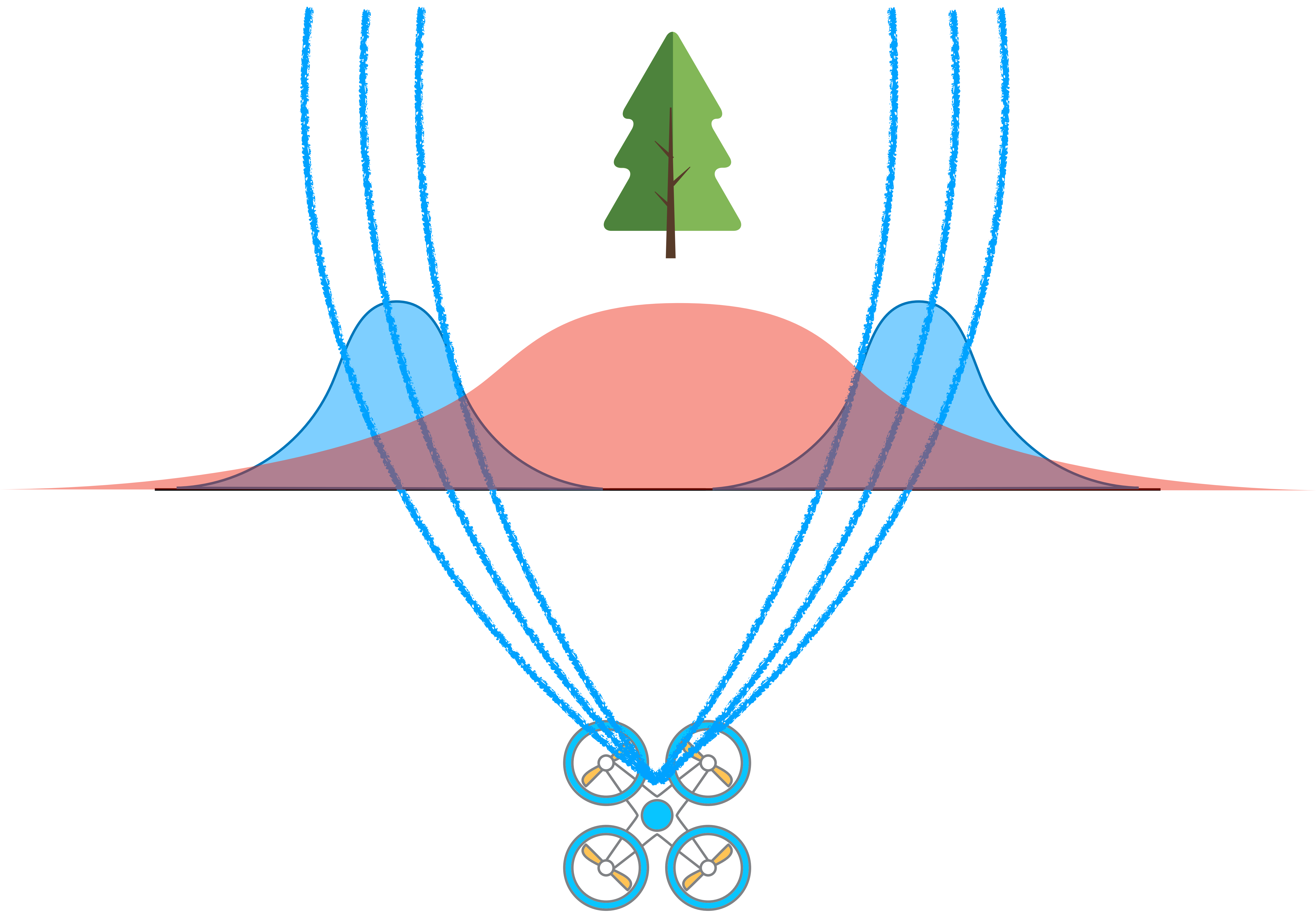
Think (30 sec): What Gaussian will we learn by minimizing KL divergence $\min_{\theta} - \mathbb{E}_{\xi \sim P_{expert}(\xi)} \log P_{\theta}(\xi)$?



Pair: Find a partner

Share (45 sec): Partners exchange ideas

Forward KL is Mode-Covering!

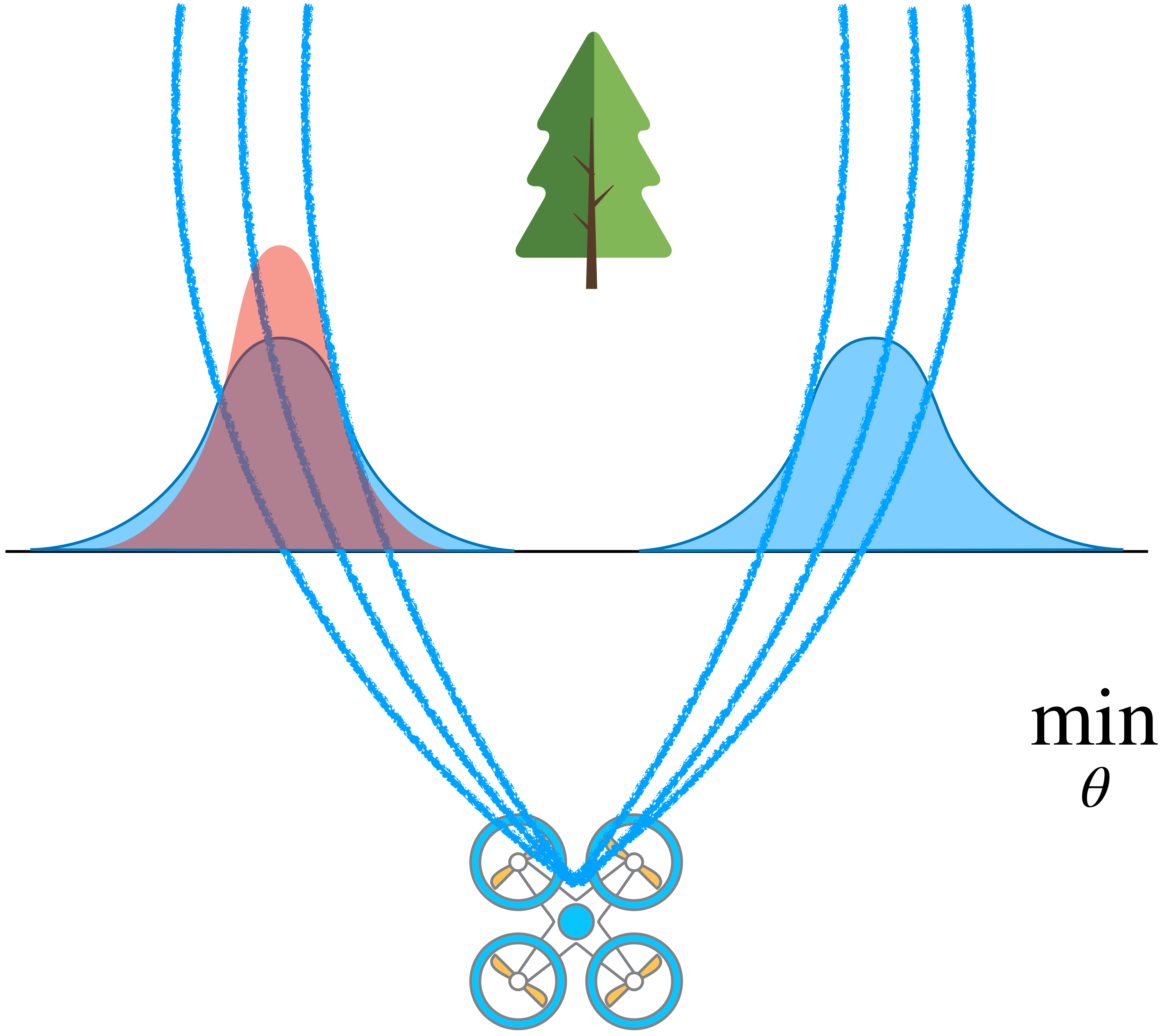


Makes sure probability is non-zero for every action the expert takes

Maximizes recall

But sacrifices precision, i.e. can leave expert support

Well what about Reverse KL?



$$\min_{\theta} D_{KL}(P_\theta || P_{expert})$$

$$\min_{\theta} \sum_{\xi} P_\theta(\xi) \log \frac{P_\theta(\xi)}{P_{expert}(\xi)}$$

$$\min_{\theta} - \sum_{\xi} P_\theta(\xi) \log P_{expert}(\xi) - H(P_\theta(\,.\,))$$

Do we know this?

Entropy

Estimating Divergences

KL is part of a *spectrum* of divergences

f-divergence: A family of divergences

$$D_f(P||Q) = \sum_x Q(x) f\left(\frac{P(x)}{Q(x)}\right)$$

Where f() is a convex function

Ali and Silvey, 1966

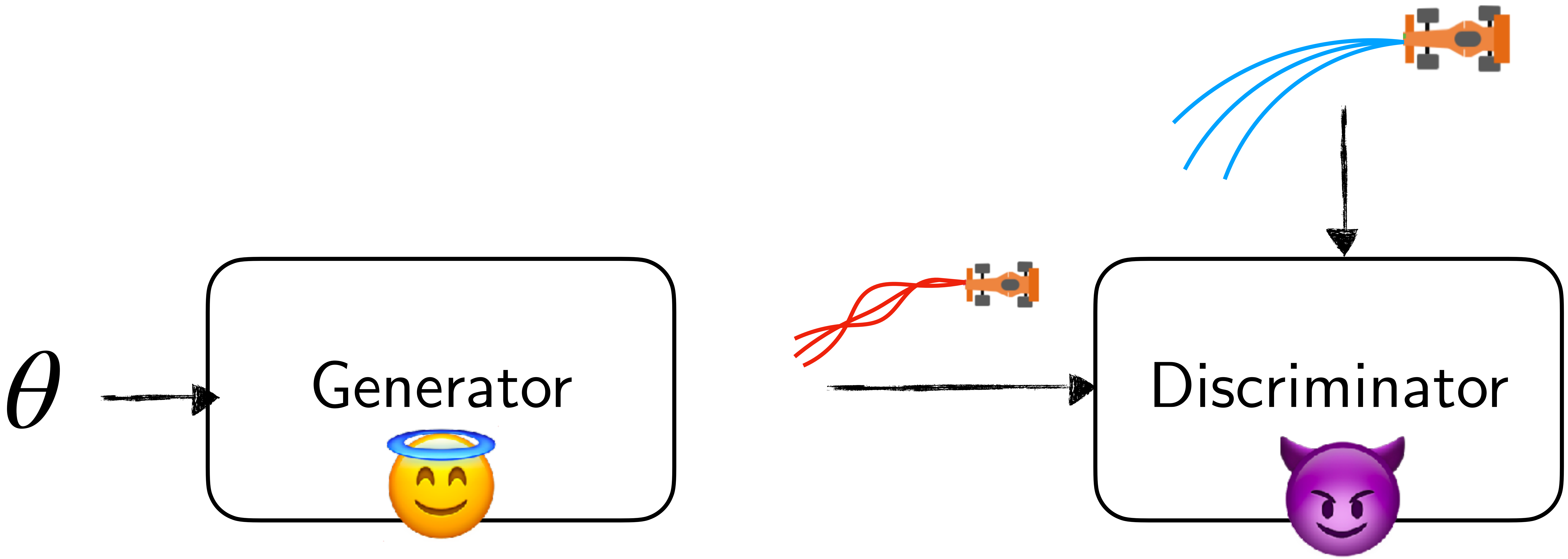
KL is part of a spectrum of divergences

Name	$D_f(P\|Q)$	Generator $f(u)$
Kullback-Leibler	$\int p(x) \log \frac{p(x)}{q(x)} \, \mathrm{d}x$	$u \log u$
Reverse KL	$\int q(x) \log \frac{q(x)}{p(x)} \, \mathrm{d}x$	$-\log u$
Pearson χ^2	$\int \frac{(q(x)-p(x))^2}{p(x)} \, \mathrm{d}x$	$(u-1)^2$
Squared Hellinger	$\int \left(\sqrt{p(x)} - \sqrt{q(x)}\right)^2 \, \mathrm{d}x$	$(\sqrt{u}-1)^2$
Jensen-Shannon	$\frac{1}{2} \int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} \, \mathrm{d}x$	$-(u+1) \log \frac{1+u}{2} + u \log u$
GAN	$\int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} \, \mathrm{d}x - \log(4)$	$u \log u - (u+1) \log(u+1)$

Nowozin et al. 2017

Okay fine ... but how do we estimate these divergences when all we have are expert samples?

Use GANs to estimate divergence!



Minimize discriminator loss!

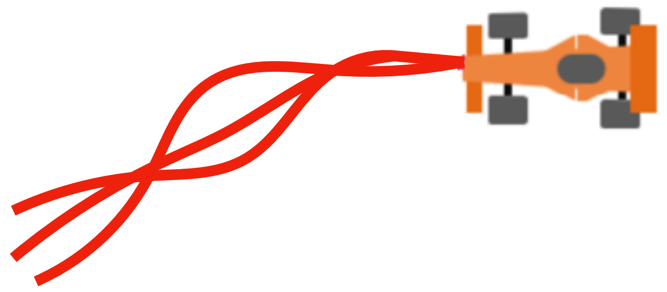
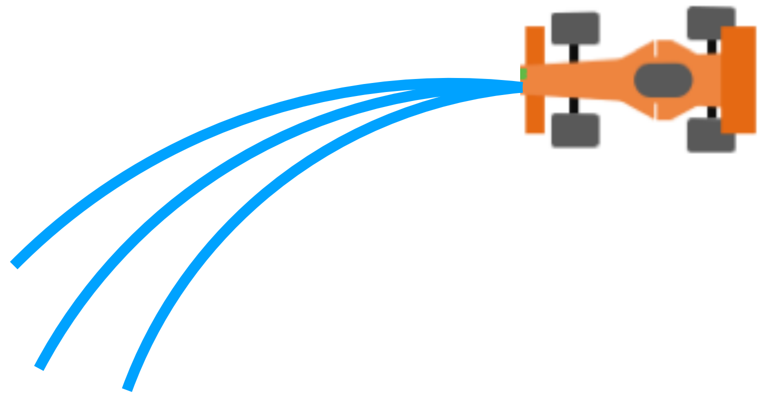
Maximally discriminate between learner and generator samples

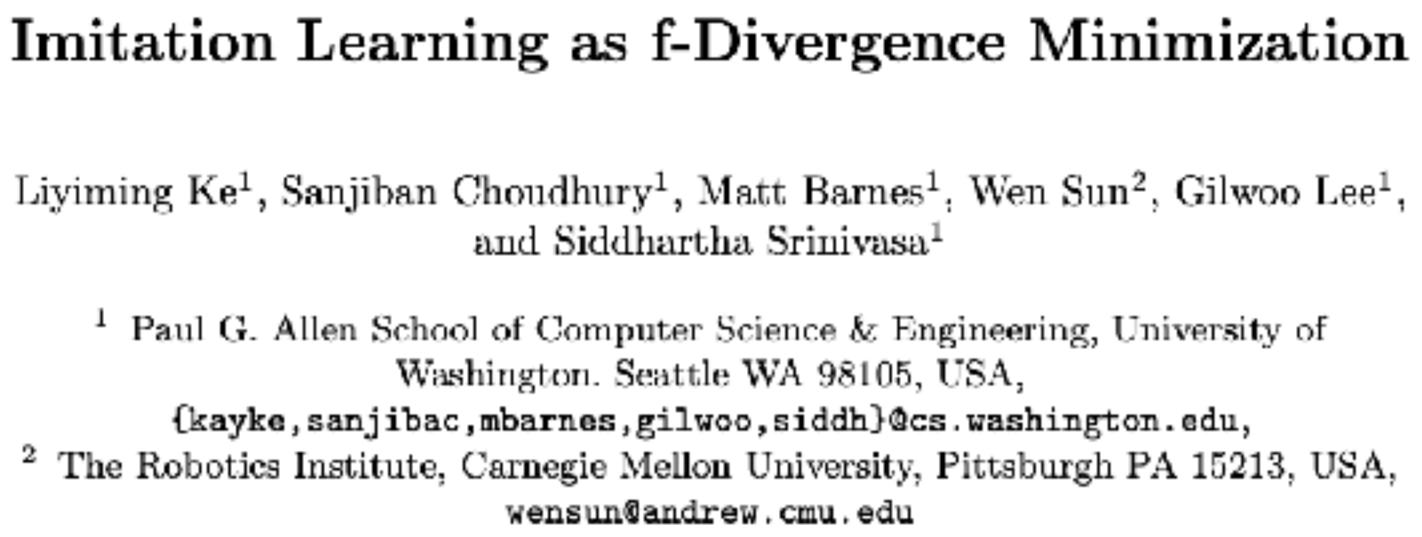
Nowozin et al. 2017

Use GANs to estimate divergence!

$$\min_{\theta} \max_{\phi} \; \mathbb{E}_{\xi \sim P_{\theta}(\xi)}[C_{\phi}(\xi)] - \mathbb{E}_{\xi \sim P_{expert}(\xi)}[f^*(C_{\phi}(\xi))]$$




Imitation Learning as f-Divergence Minimization

Liyiming Ke[1], Sanjiban Choudhury[1], Matt Barnes[1], Wen Sun[2], Gilwoo Lee[1], and Siddhartha Srinivasa[1]

[1] Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle WA 98105, USA, {kayke,sanjibac,mbarnes,gilwoo,siddh}@cs.washington.edu,
[2] The Robotics Institute, Carnegie Mellon University, Pittsburgh PA 15213, USA, wensun@andrew.cmu.edu

😇 The Rise of Adversarial Imitation Learning 😈

JS-Divergence

Generative Adversarial Imitation Learning

Jonathan Ho
Stanford University
hoj@cs.stanford.edu

Stefano Ermon
Stanford University
ermon@cs.stanford.edu

Reverse-KL Divergence

LEARNING ROBUST REWARDS WITH ADVERSARIAL INVERSE REINFORCEMENT LEARNING

Justin Fu, Katie Luo, Sergey Levine
Department of Electrical Engineering and Computer Science
University of California, Berkeley
Berkeley, CA 94720, USA
justinjfu@eecs.berkeley.edu, katieluo@berkeley.edu,
svlevine@eecs.berkeley.edu

Jeffrey Divergence

R2P2: A ReparameteRized Pushforward Policy for Diverse, Precise Generative Path Forecasting

Nicholas Rhinehart[1,2][0000–0003–4242–1236], Kris M. Kitani[1][0000–0002–9389–4060], and Paul Vernaza[2][0000–0002–2745–1894]

[1] Carnegie Mellon University, Pittsburgh PA 15213, USA
[2] NEC Labs America, Cupertino, CA 95014, USA

State-Marginal f-divergence

f-IRL: Inverse Reinforcement Learning via State Marginal Matching

Tianwei Ni*, Harshit Sikchi*, Yufei Wang*, Tejus Gupta*, Lisa Lee†, Benjamin Eysenbach†
Carnegie Mellon University
{tianwein, hsikchi, yufeiw2, tejusg, lslee, beysenba}@cs.cmu.edu

Which divergence do we care about?

What divergence do we care about?

f-divergence are great and all, but which one
do we actually care about?

What divergence do we care about?

What we actually care about is matching Performance Difference

$$J(\pi) = J(\pi^*)$$

$$\mathbb{E}_{\xi \sim P_\theta(\xi)} c(\xi) = \mathbb{E}_{\xi \sim P_{expert}(\xi)} c(\xi)$$

But we don't know the costs c(.)

What divergence do we care about?

What we actually care about is matching Performance Difference

$$J(\pi) = J(\pi^*)$$

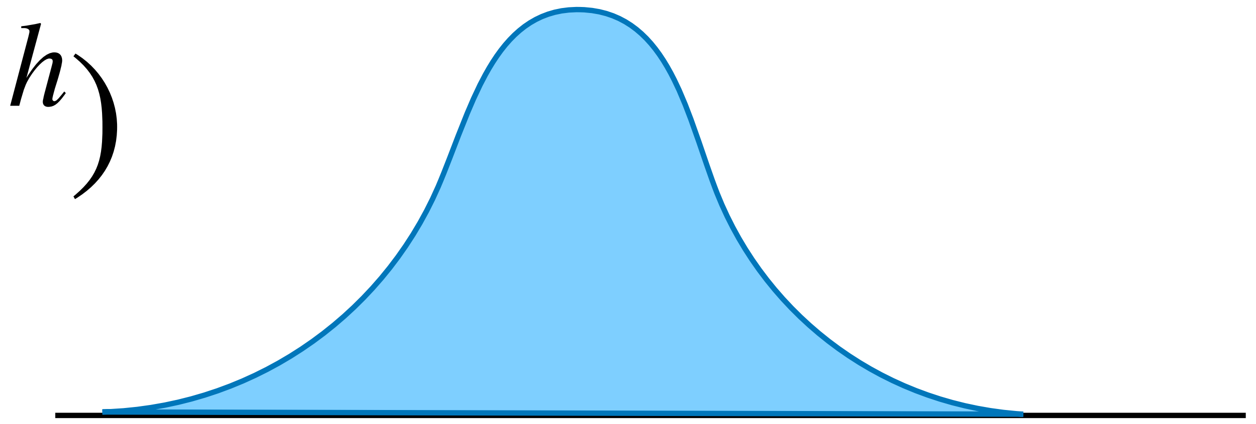
$$\mathbb{E}_{\xi \sim P_\theta(\xi)} c(\xi) = \mathbb{E}_{\xi \sim P_{expert}(\xi)} c(\xi)$$

But we don't know the costs c(.)

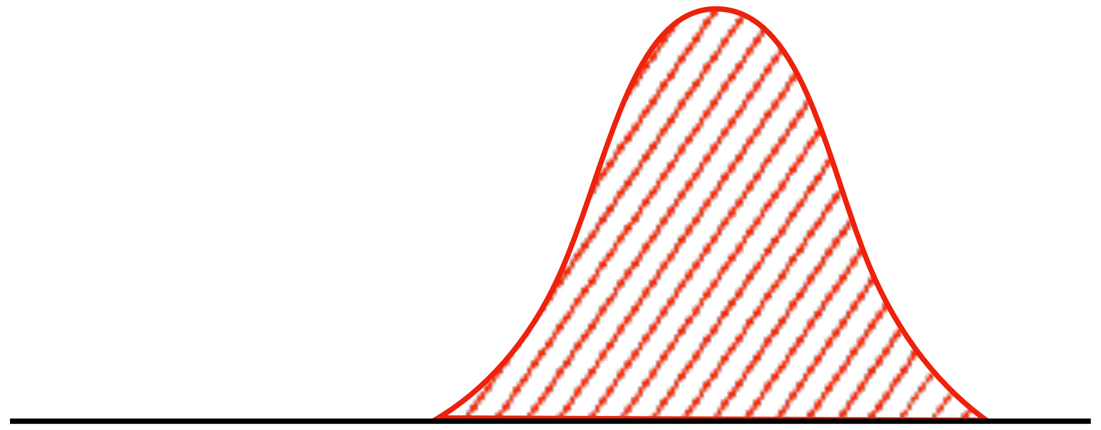
Costs are just weighted combination of features. What if we just matched all the expected features?

Proposal: Match cost features!

$P_{expert}(\xi^h)$



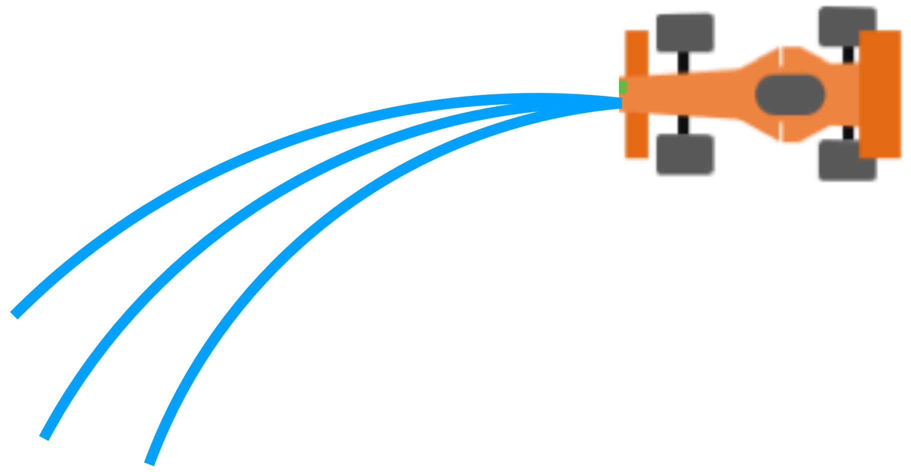
(Unknown) expert distribution



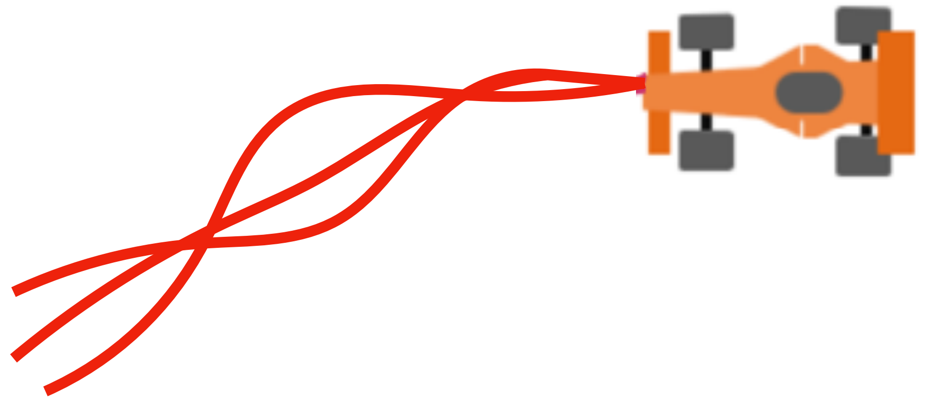
$P_\theta(\xi)$

Learn distribution over trajectories

All we see are expert samples

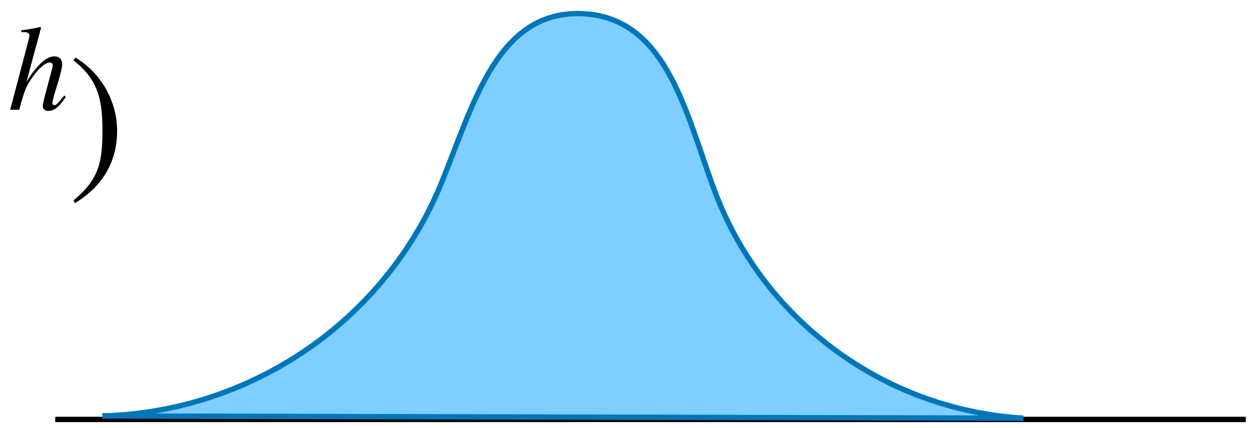


Learner can also generate samples

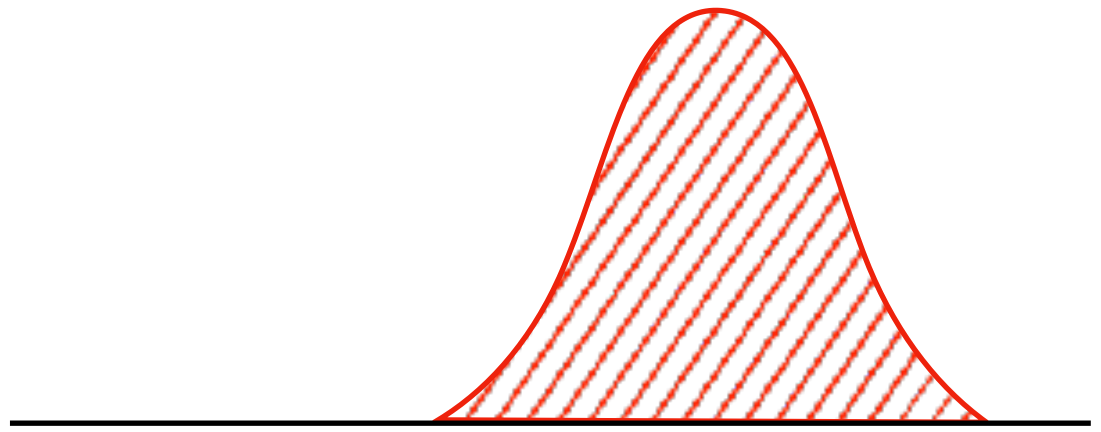


Proposal: Match cost features!

$P_{expert}(\xi^h)$



(Unknown) expert distribution

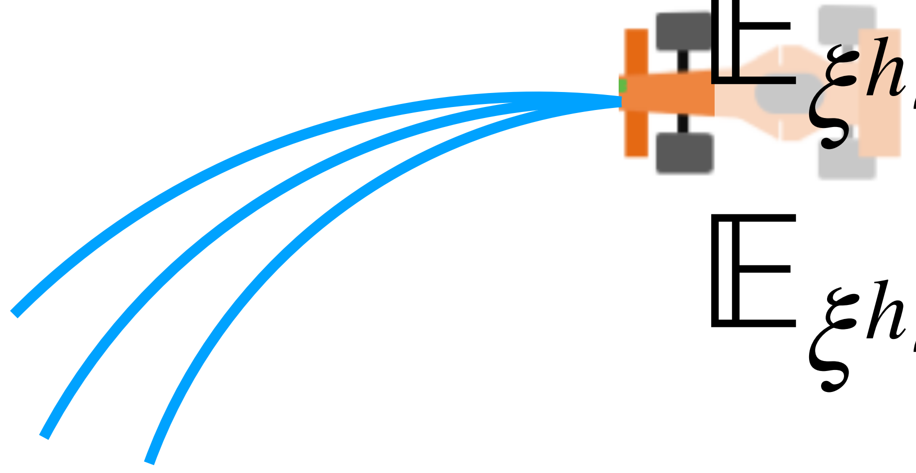


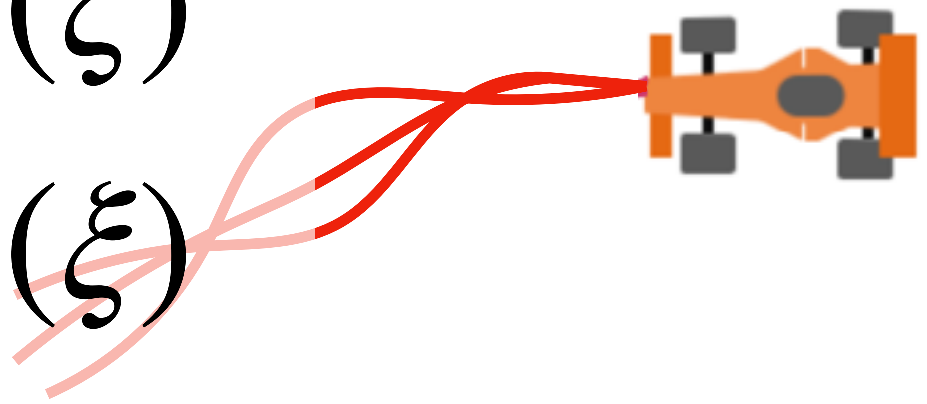
$P_\theta(\xi)$

Learn distribution over trajectories

All we see are expert samples

Learner can also generate samples





$$\mathbb{E}_{\xi^h \sim P_{expert}(.)}\, f_1(\xi^h) = \mathbb{E}_{\xi \sim P_\theta(.)}\, f_1(\xi)$$

$$\mathbb{E}_{\xi^h \sim P_{expert}(.)}\, f_2(\xi^h) = \mathbb{E}_{\xi \sim P_\theta(.)}\, f_2(\xi)$$

$$\vdots$$

$$\mathbb{E}_{\xi^h \sim P_{expert}(.)}\, f_k(\xi^h) = \mathbb{E}_{\xi \sim P_\theta(.)}\, f_k(\xi)$$

Let's formalize!

Maximum Entropy Inverse Optimal Control

Maximum Entropy Inverse Reinforcement Learning

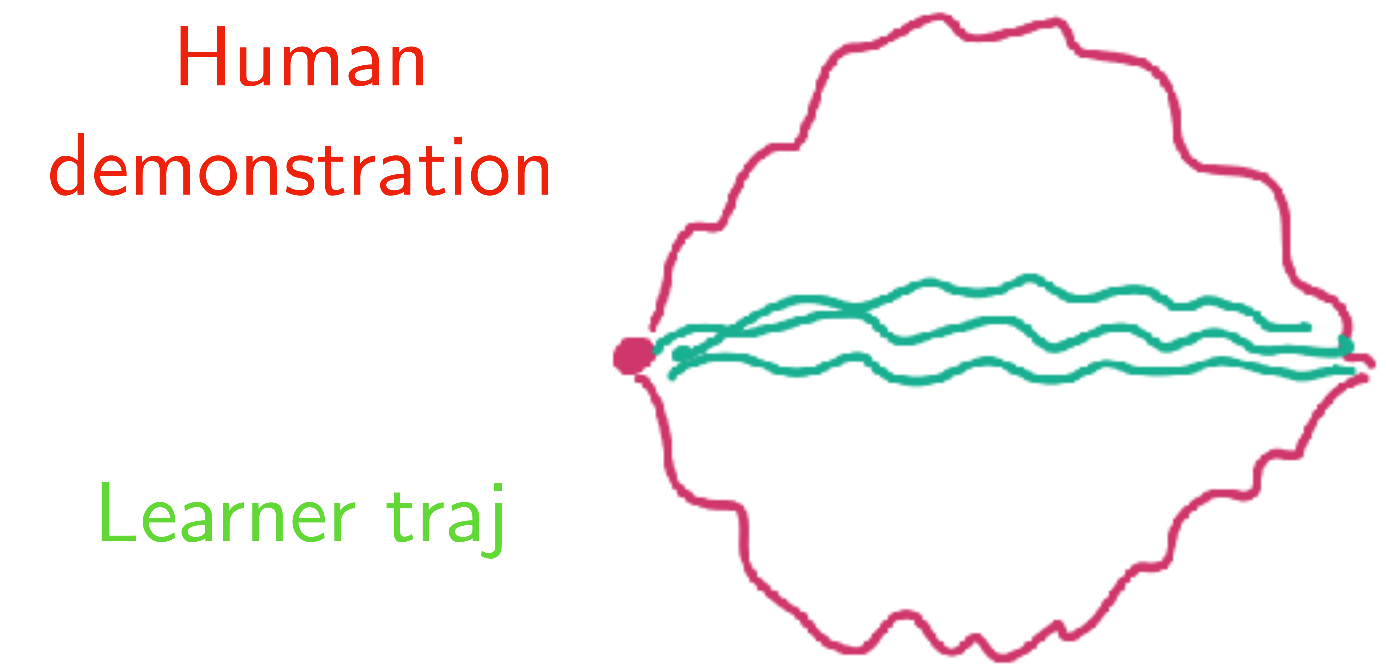
Brian D. Ziebart, Andrew Maas, J.Andrew Bagnell, and **Anind K. Dey**
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
bziebart@cs.cmu.edu, amaas@andrew.cmu.edu, dbagnell@ri.cmu.edu, anind@cs.cmu.edu

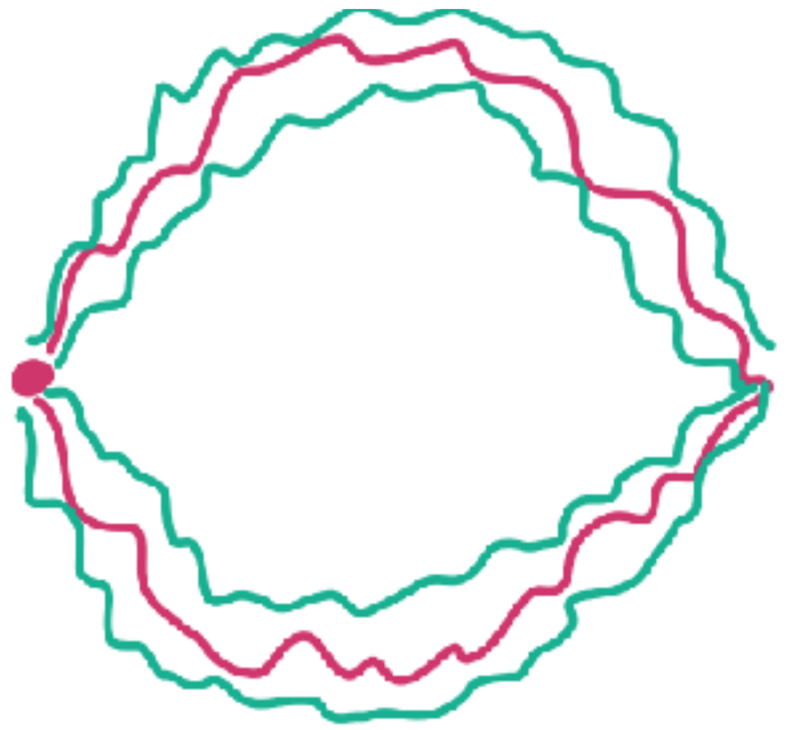
Maximum Entropy Inverse Optimal Control

LEO: Learning Energy-based Models in Factor Graph Optimization

Paloma Sodhi[1,2], Eric Dexheimer[1], Mustafa Mukadam[2], Stuart Anderson[2], Michael Kaess[1]
[1]Carnegie Mellon University, [2] Facebook AI Research

Maximum Entropy Inverse Optimal Control





Given dataset: $\{\xi_i^h, \phi_i\}_{i=1}^N$

(Human demo) *(Map)*

Solve for cost $C_\theta(\xi)$

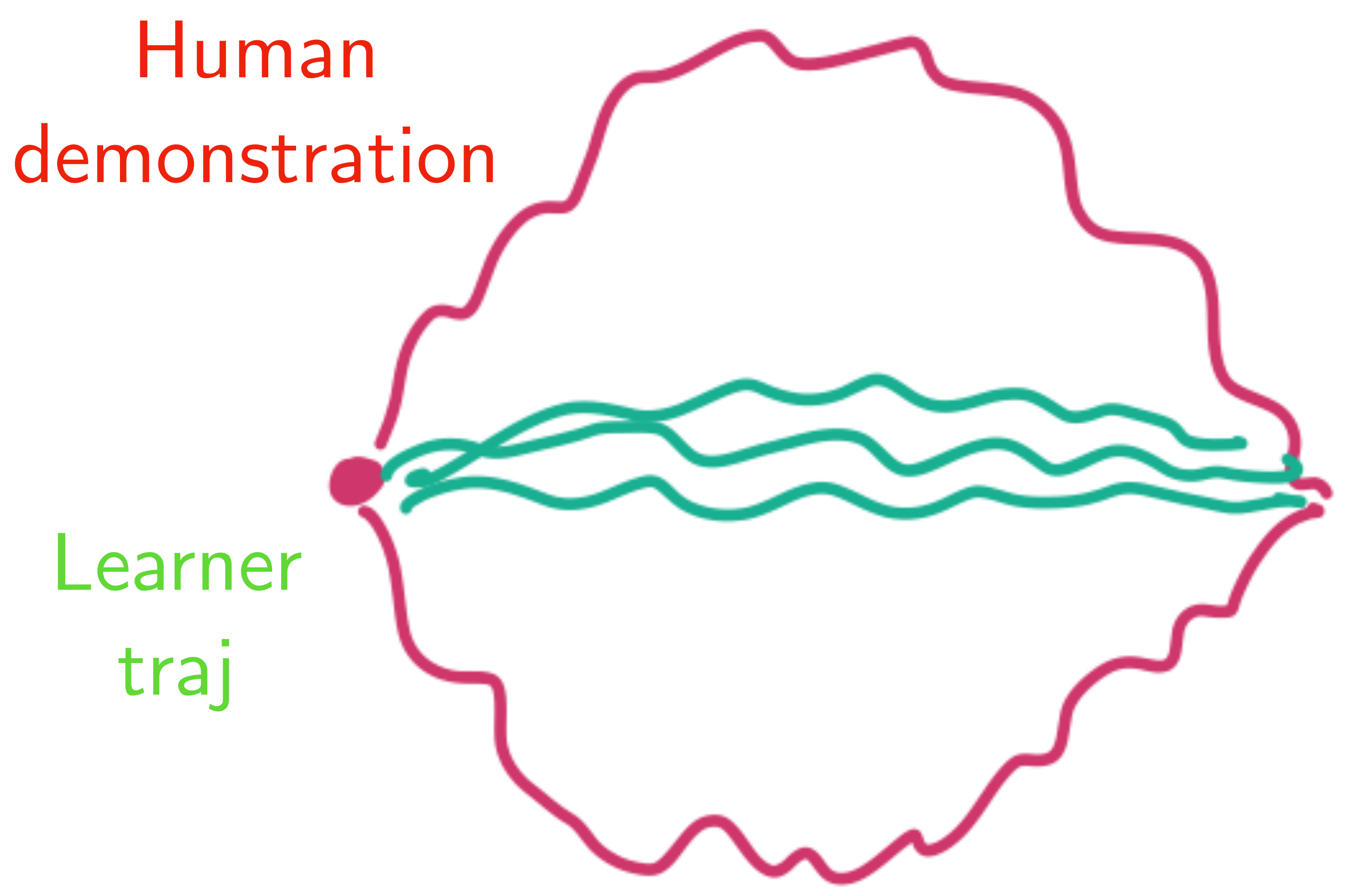
$$\min_\theta \frac{1}{N} \sum_{i=1}^{N} -\log P_\theta(\xi_i^h \,|\, \phi_i)$$

Max lik. of human traj

$$P_\theta(\xi \,|\, \phi) = \frac{1}{Z(\theta, \phi)} \exp(-C_\theta(\xi, \phi))$$

More costly traj, less likely

Maximum Entropy Inverse Optimal Control



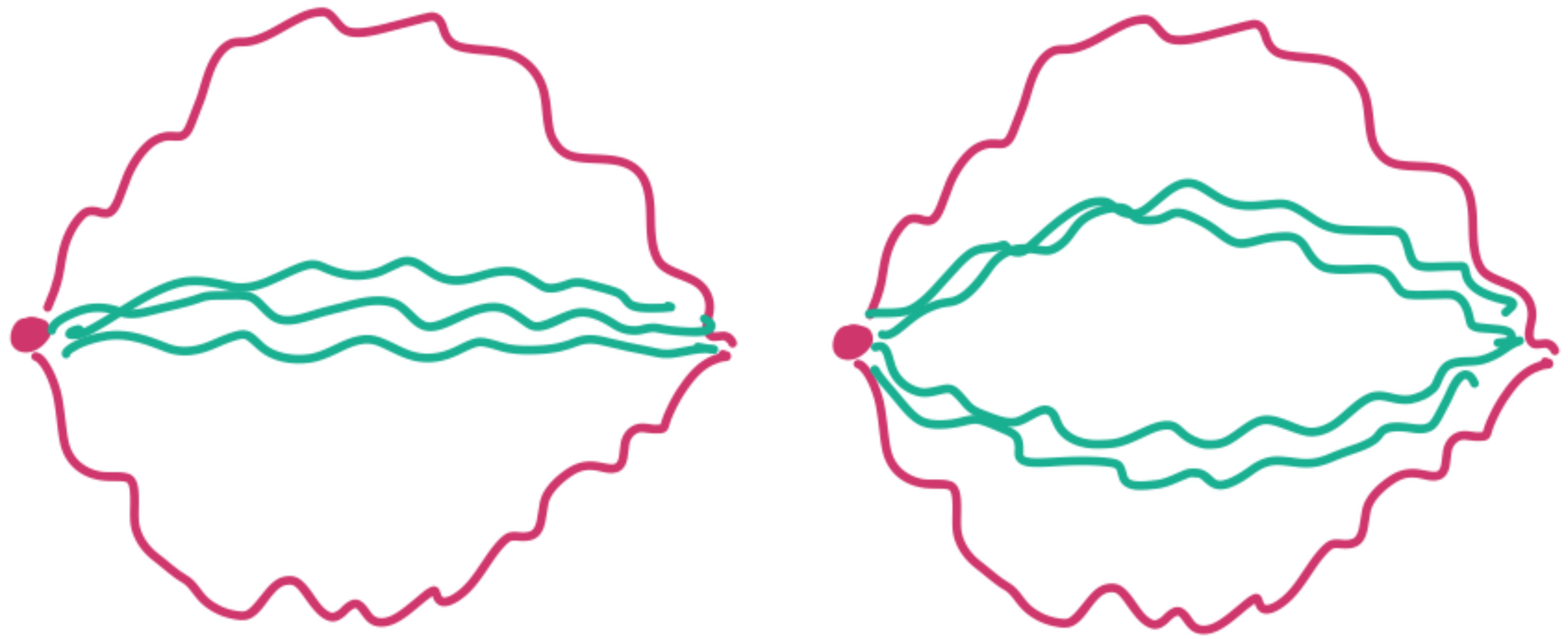
for $i = 1, \ldots, N$ # Loop over datapoints

$$\xi_i \sim \frac{1}{Z} \exp\left(-C_\theta(\xi, \phi_i)\right)$$ # Call planner!

$$\theta^+ = \theta - \eta[\nabla_\theta C_\theta(\xi_i^h, \phi_i) - \nabla_\theta C_\theta(\xi_i, \phi_i)]$$ # Update cost

(Push down human cost) *(Push up planner cost)*

Maximum Entropy Inverse Optimal Control



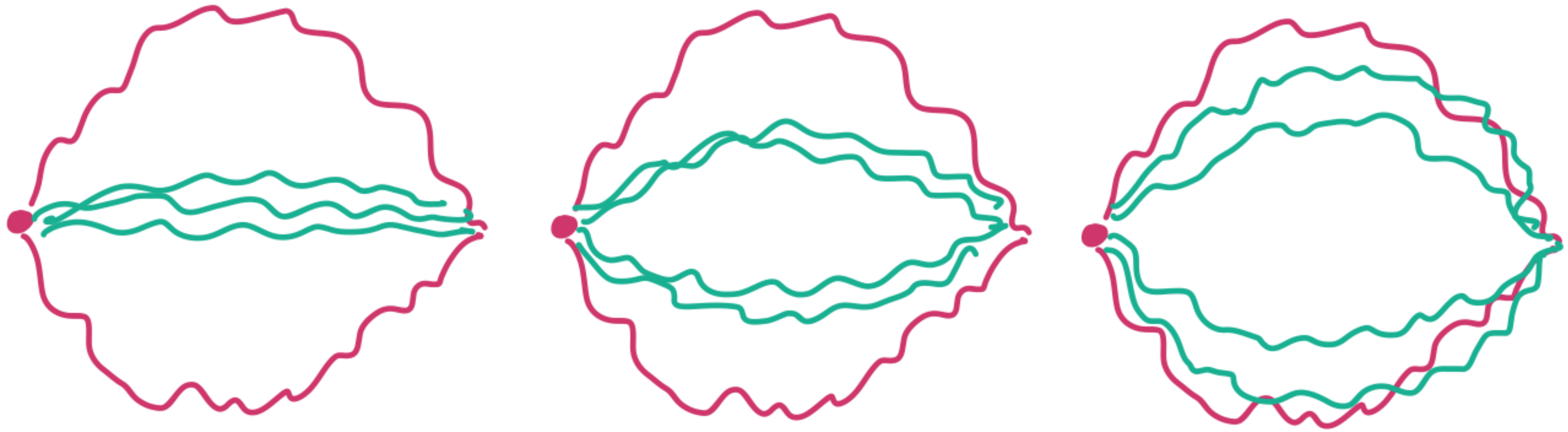
for $i = 1, \ldots, N$ # Loop over datapoints

$$\xi_i \sim \frac{1}{Z} \exp\left(-C_\theta(\xi, \phi_i)\right)$$ # Call planner!

$$\theta^+ = \theta - \eta[\nabla_\theta C_\theta(\xi_i^h, \phi_i) - \nabla_\theta C_\theta(\xi_i, \phi_i)]$$ # Update cost

(Push down human cost) *(Push up planner cost)*

Maximum Entropy Inverse Optimal Control



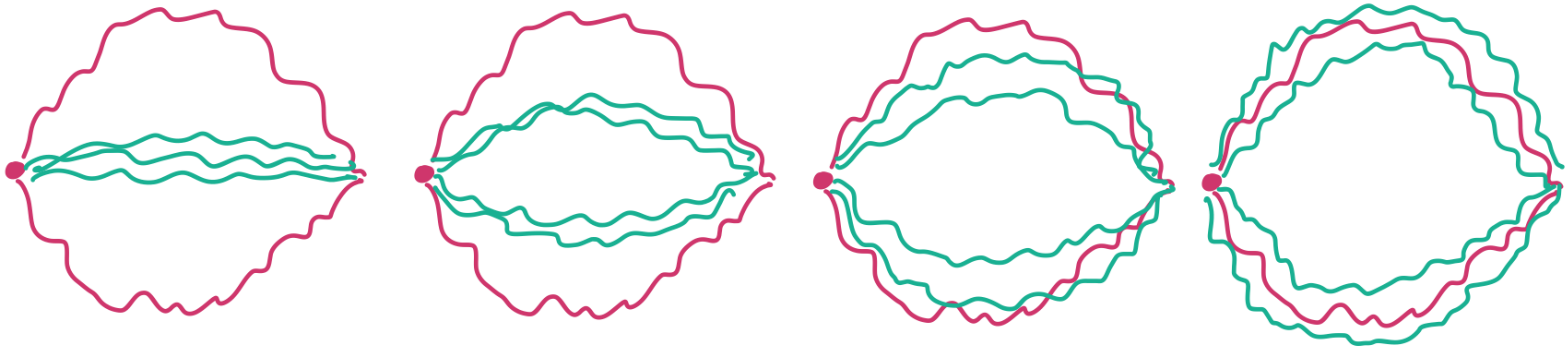
for $i = 1, \ldots, N$ # Loop over datapoints

$$\xi_i \sim \frac{1}{Z} \exp\left(-C_\theta(\xi, \phi_i)\right)$$ # Call planner!

$$\theta^+ = \theta - \eta[\nabla_\theta C_\theta(\xi_i^h, \phi_i) - \nabla_\theta C_\theta(\xi_i, \phi_i)]$$ # Update cost

(Push down human cost) *(Push up planner cost)*

Maximum Entropy Inverse Optimal Control



for $i = 1, \ldots, N$ # Loop over datapoints

$$\xi_i \sim \frac{1}{Z} \exp\left(-C_\theta(\xi, \phi_i)\right)$$ # Call planner!

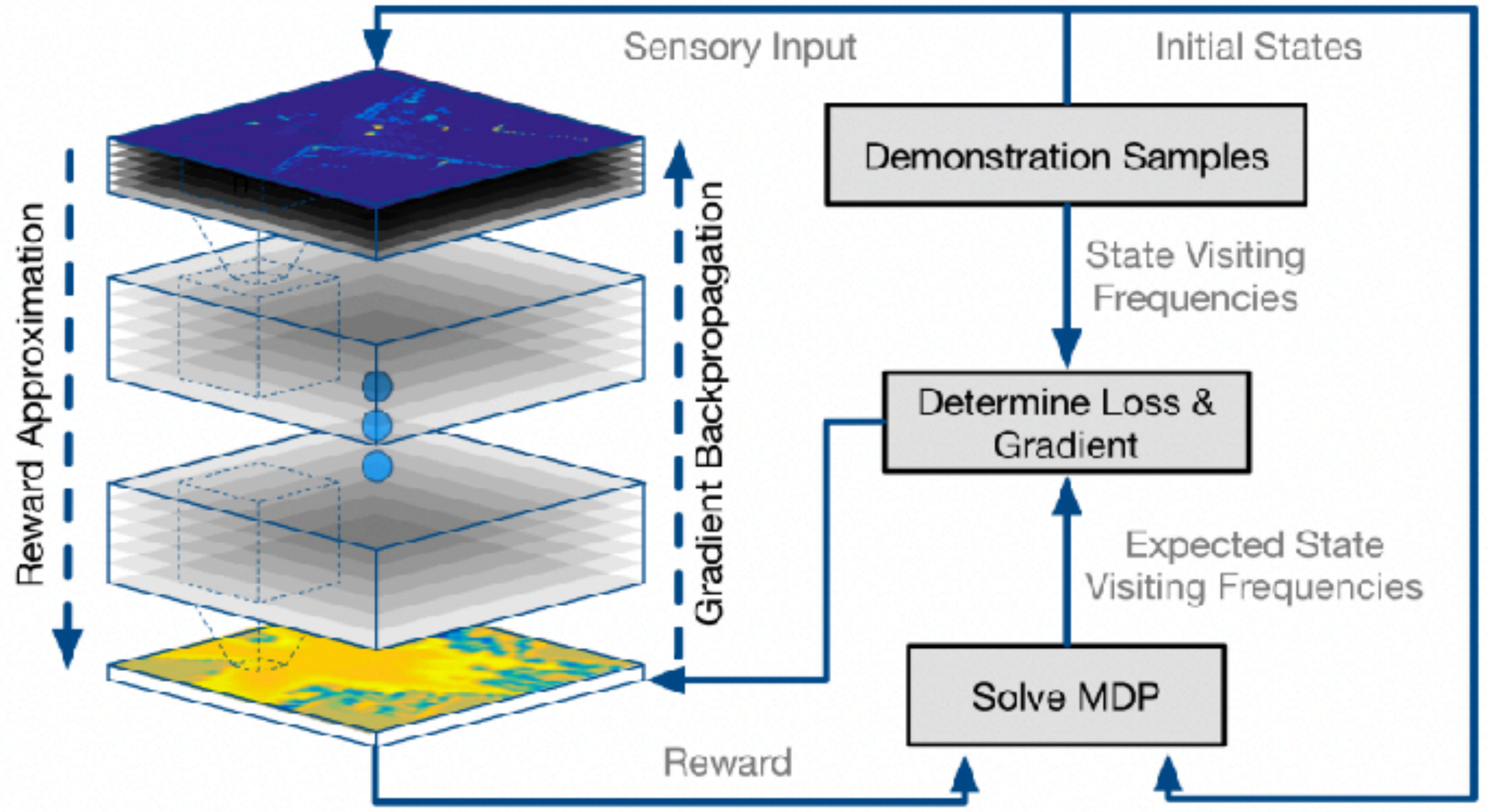
$$\theta^+ = \theta - \eta[\nabla_\theta C_\theta(\xi_i^h, \phi_i) - \nabla_\theta C_\theta(\xi_i, \phi_i)]$$ # Update cost

(Push down human cost) *(Push up planner cost)*

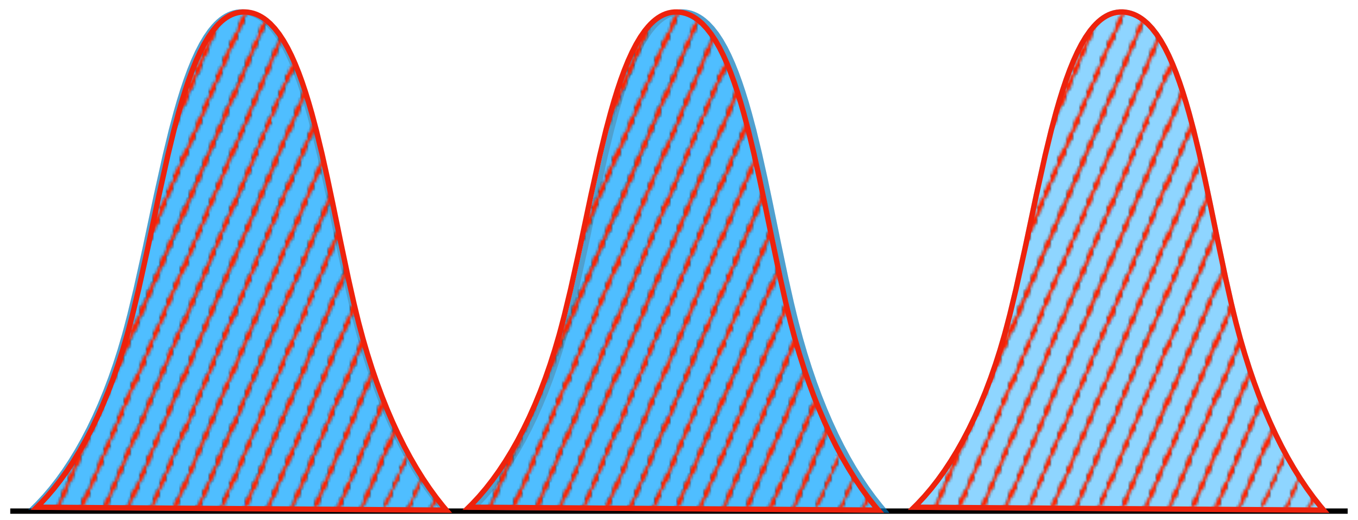
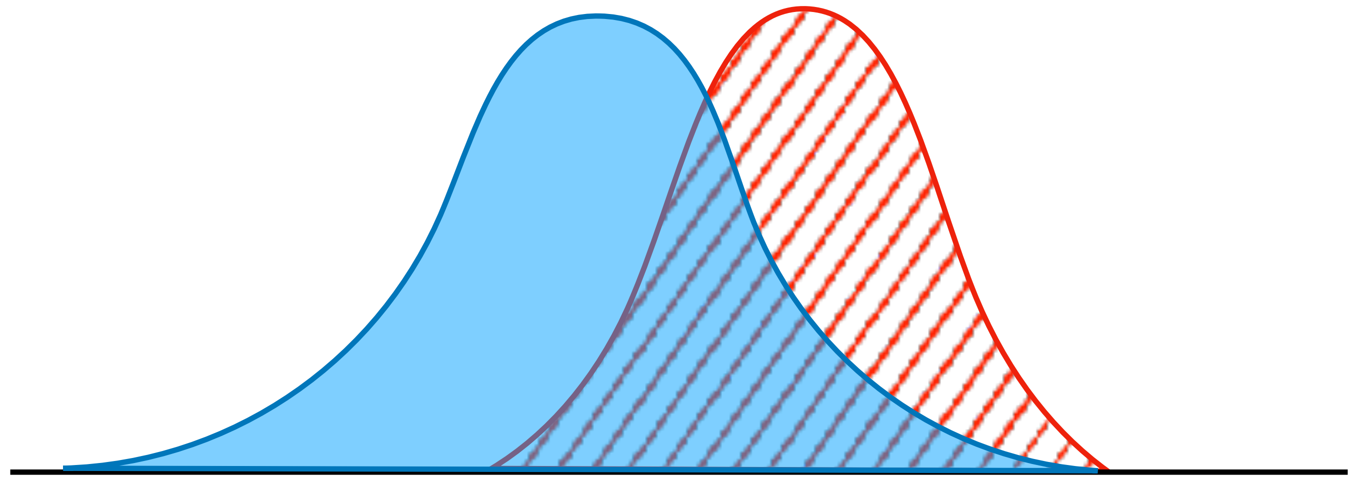
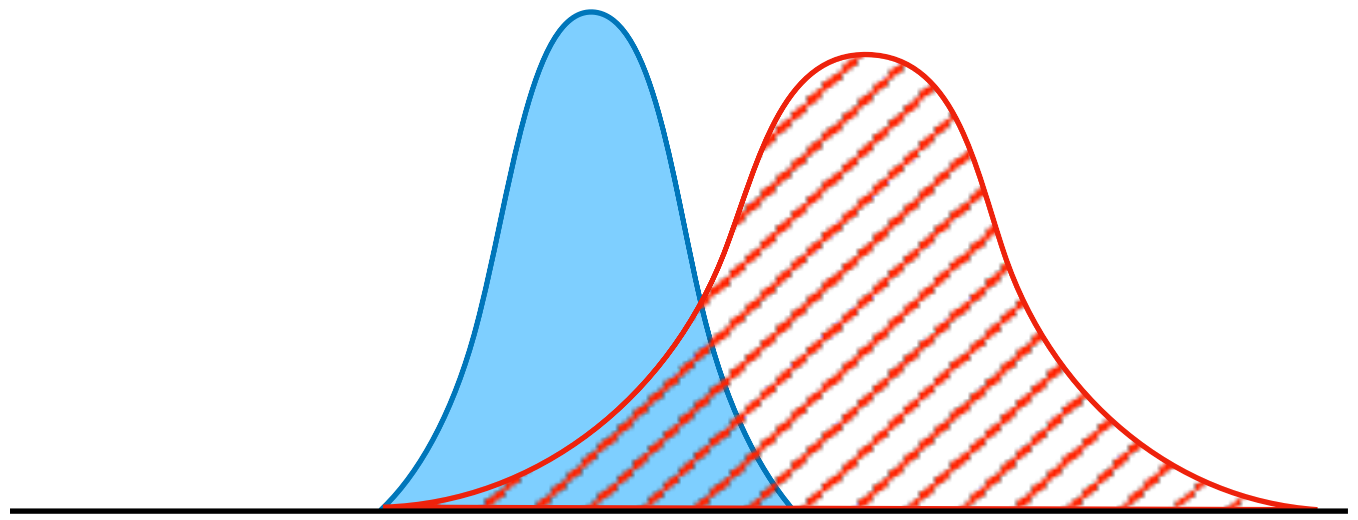
Deep Max Ent

Watch This: Scalable Cost-Function Learning for Path Planning in Urban Environments

Markus Wulfmeier[1], Dominic Zeng Wang[1] and Ingmar Posner[1]

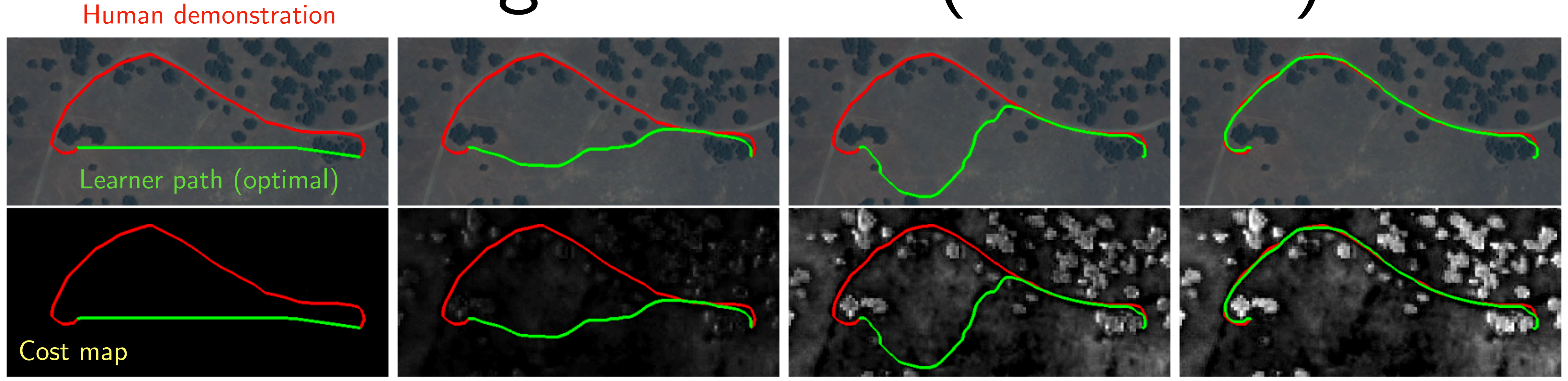



autonomous execution
1x real-time
PR2
goal
our method
IOC samples from $q(u_t|x_t)$

	Easy	**Medium**	**Hard**
Setting	Expert is realizable $\pi^E \in \Pi$	Non-realizable expert but full expert support	Non-realizable expert + limited expert support
	As $N \rightarrow \infty$, drive down $\epsilon = 0$ (or Bayes error)	Even as $N \rightarrow \infty$, behavior cloning $O(\epsilon C T)$ where C is conc. coeff	Even as $N \rightarrow \infty$, behavior cloning $O(\epsilon T^2)$
			
Solution	Nothing special. Collect lots of data and do Behavior Cloning	Requires interactive simulator (MaxEntIRL) to match distribution $\Rightarrow O(\epsilon T)$	Requires interactive expert (DAGGER / EIL) to provide labels $\Rightarrow O(\epsilon T)$

tl;dr

Learning to Search (LEARCH)



for $i = 1,\ldots,N$ # Loop over datapoints

$$\xi_i^* = \min_{\xi}[C_\theta(\xi,\phi_i) - \gamma(\xi,\xi^h)]$$ # Call planner!

$$\theta^+ = \theta - \eta[\nabla_\theta C_\theta(\xi_i^h,\phi_i) - \nabla_\theta C_\theta(\xi_i^*,\phi_i) + \nabla_\theta R(\theta)]$$

(Push down human cost) *(Push up planner cost)* # Update cost

x

When the expert is

Suboptimal

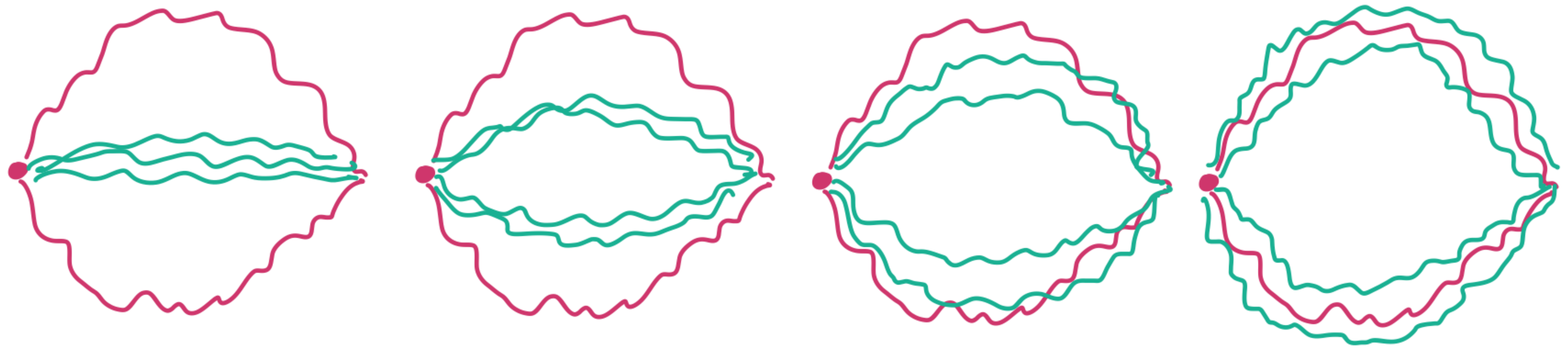
Noisy

Privileged Information

LEARCH does NOT converge!!

x

Maximum Entropy Inverse Optimal Control



for $i = 1,\ldots,N$ # Loop over datapoints

$$\xi_i \sim \frac{1}{Z}\exp\left(-C_\theta(\xi,\phi_i)\right)$$ # Call planner!

$$\theta^+ = \theta - \eta[\nabla_\theta C_\theta(\xi_i^h,\phi_i) - \nabla_\theta C_\theta(\xi_i,\phi_i)]$$ # Update cost

(Push down human cost) *(Push up planner cost)*

x