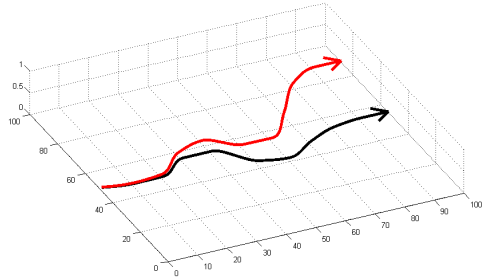
Supervised Learning of Behaviors

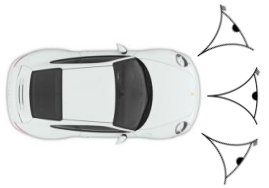
CS 185/285

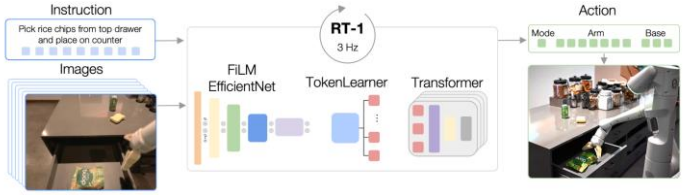
Instructor: Sergey Levine
UC Berkeley

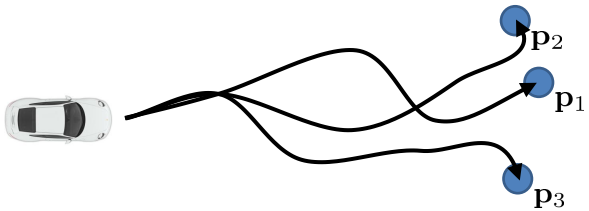
Recap

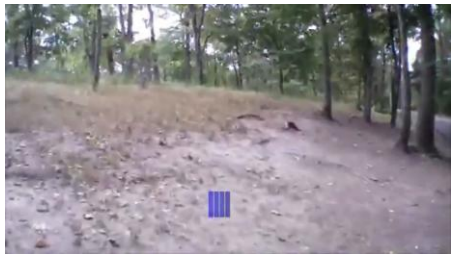
- Imitation learning via behavioral cloning is not guaranteed to work
  - This is **different** from supervised learning
  - The reason: i.i.d. assumption does not hold!
- We can formalize **why** this is and do a bit of theory
- We can address the problem in a few ways:
  - Change the algorithm (DAgger)
  - Use very powerful models that make very few mistakes
  - Be smart about how we collect (and augment) our data
  - Use multi-task learning





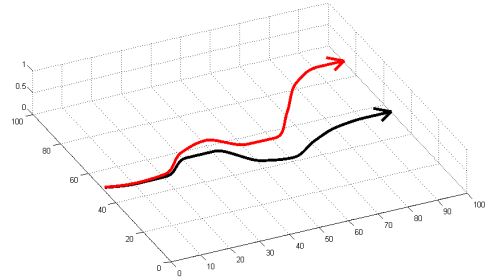


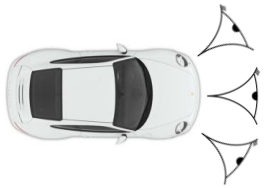


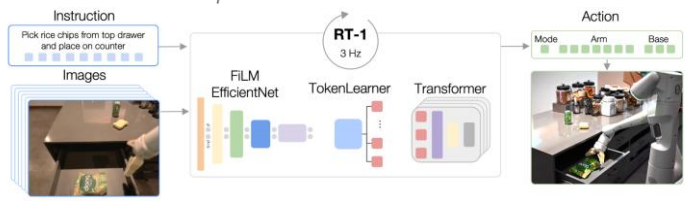


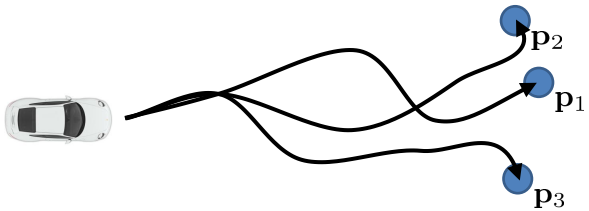
Recap

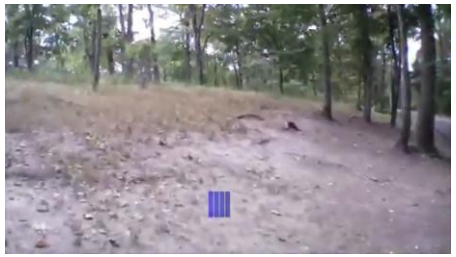
- Imitation learning via behavioral cloning is not guaranteed to work
  - This is **different** from supervised learning
  - The reason: i.i.d. assumption does not hold!
- We can formalize **why** this is and do a bit of theory
- We can address the problem in a few ways:
  - Change the algorithm (DAgger)
  - Use very powerful models that make very few mistakes
  - Be smart about how we collect (and augment) our data
  - Use multi-task learning











Part 1:
Models for imitation learning

Why might we fail to fit the expert?

1. Non-Markovian behavior
2. Multimodal behavior

$\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$

behavior depends only on current observation

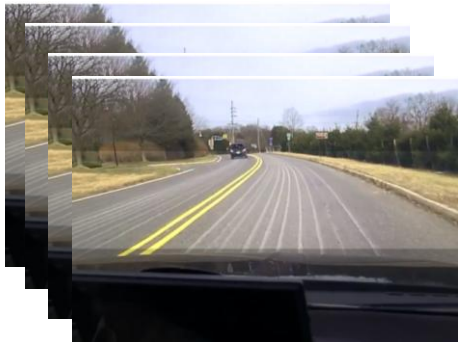
$\pi_\theta(\mathbf{a}_t|\mathbf{o}_1, ..., \mathbf{o}_t)$

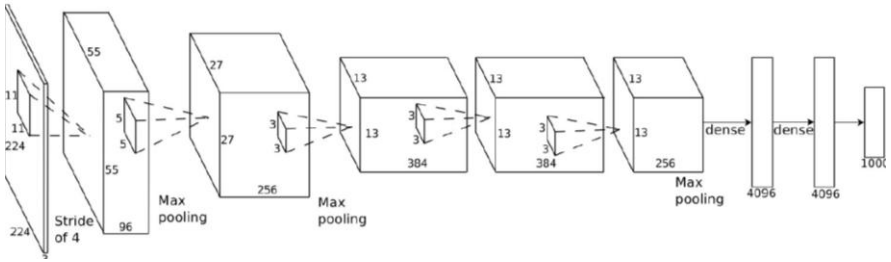
behavior depends on all past observations

If we see the same thing twice, we do the same thing twice, regardless of what happened before

Often very unnatural for human demonstrators

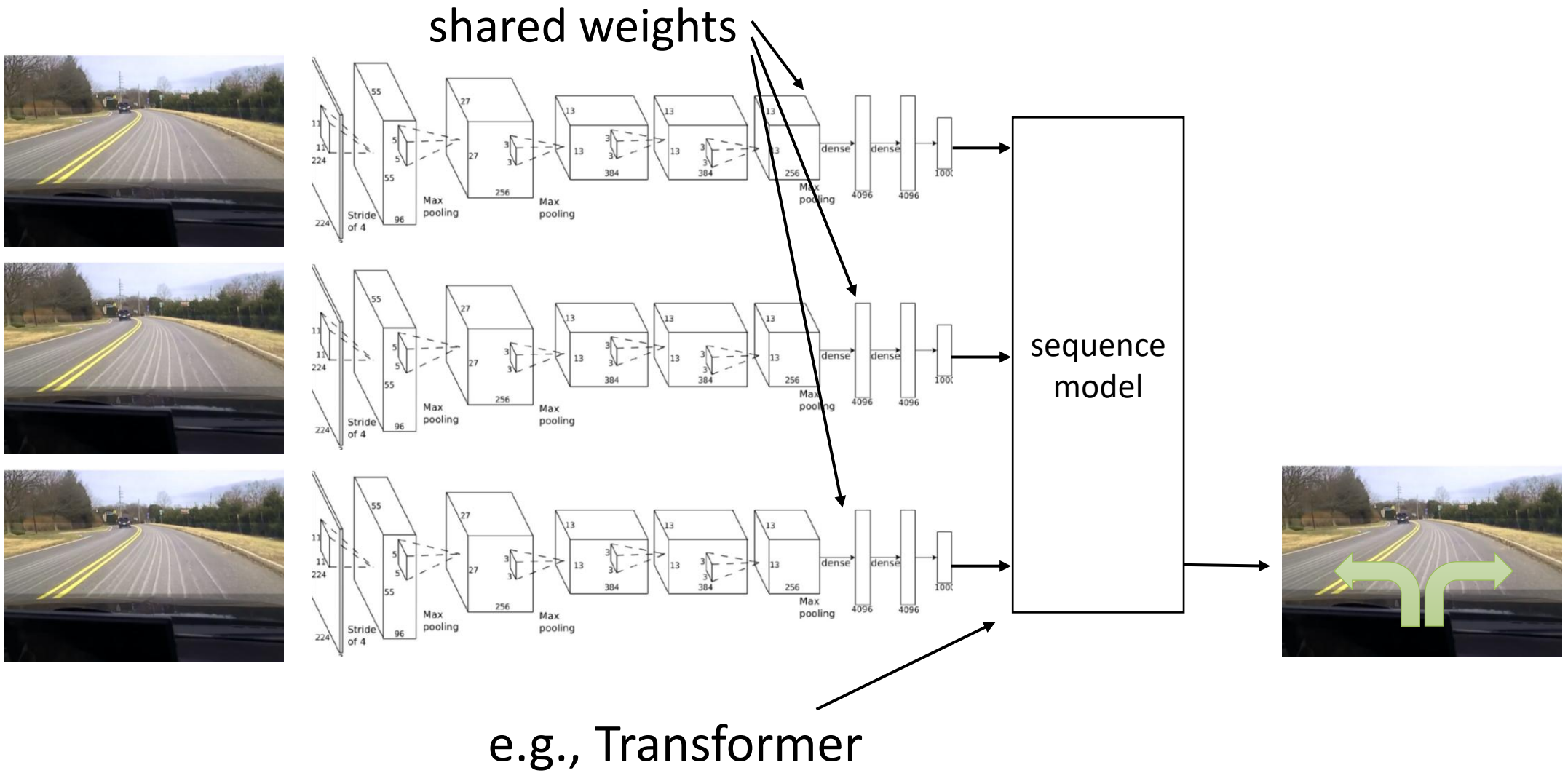
How can we use the whole history?



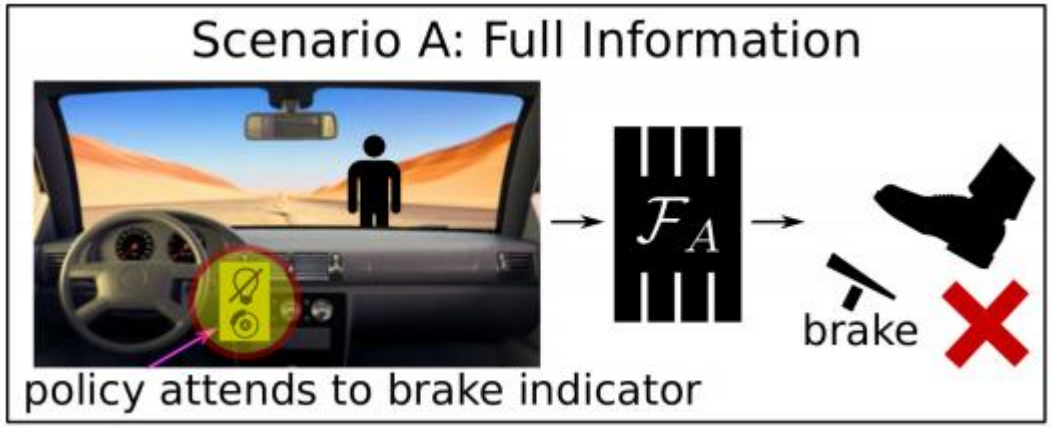


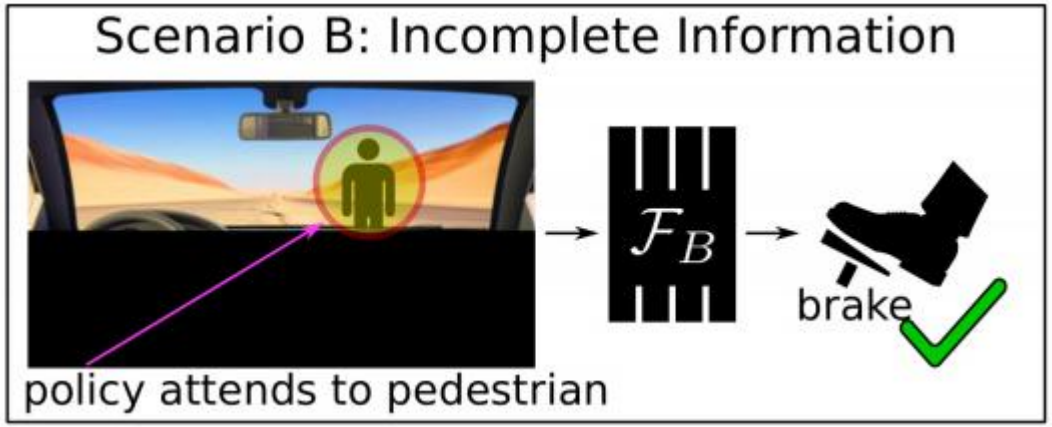
variable number of frames,
too many weights

How can we use the whole history?



Aside: why might this work **poorly**?





"causal confusion"

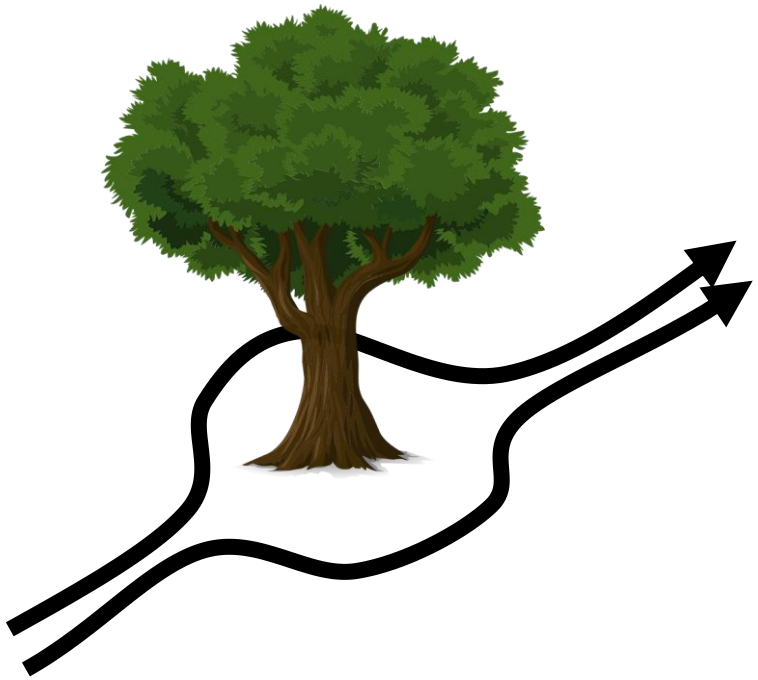
see: de Haan et al., "Causal Confusion in Imitation Learning"

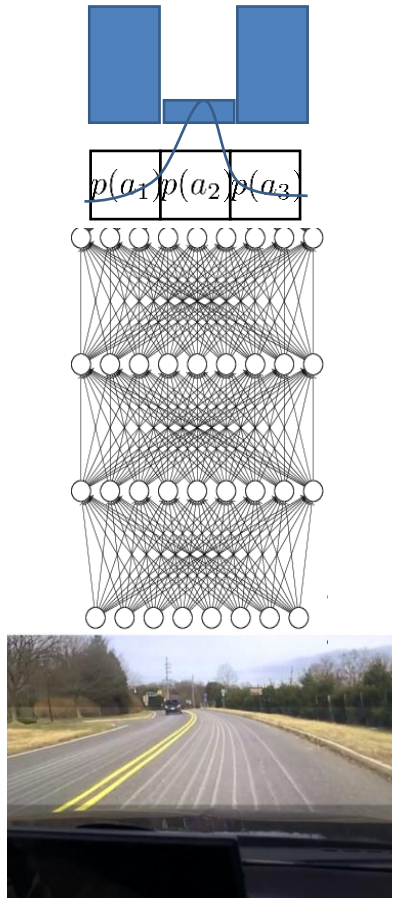
Question 1: Does including history mitigate causal confusion?

Question 2: Can DAgger mitigate causal confusion?

Why might we fail to fit the expert?

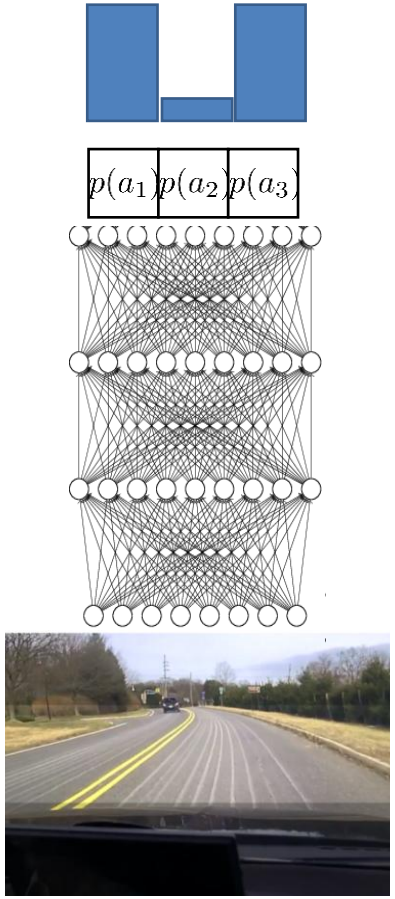
1. Non-Markovian behavior
2. Multimodal behavior





1. Discretization with high-dimensional action spaces
2. More expressive continuous distributions

Can we discretize continuous action spaces?



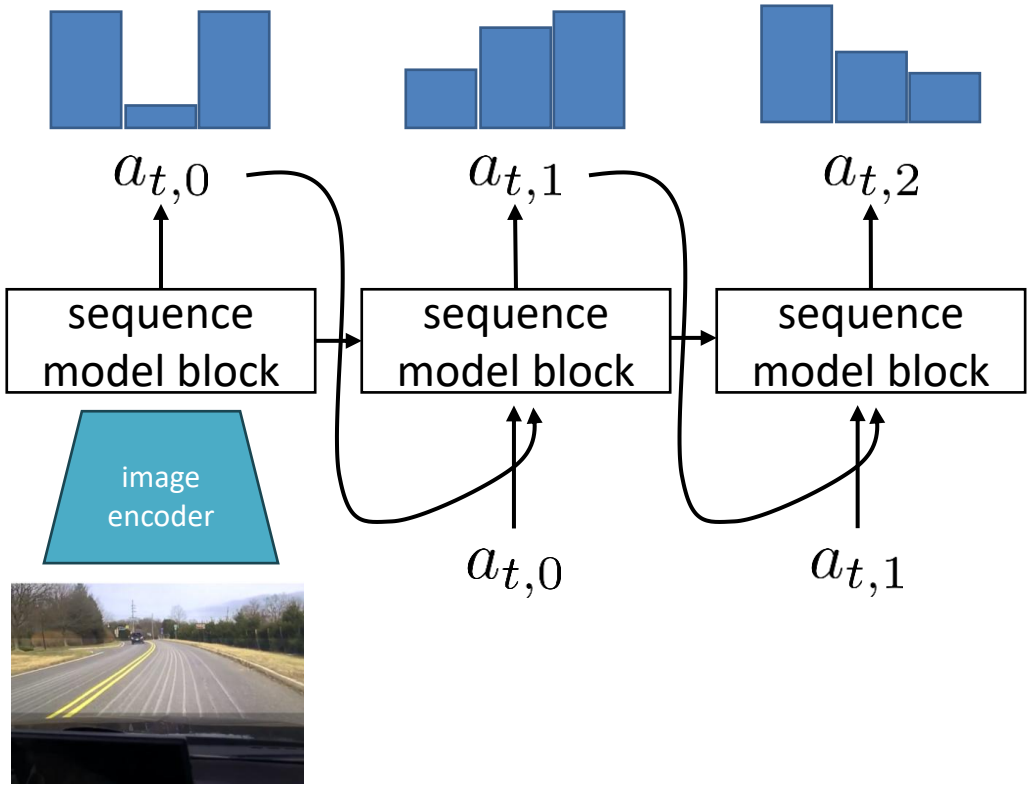
Problem: this is great for 1D actions, but in higher dimensions, discretizing the full space is impractical

Solution: discretize one dimension at a time

Autoregressive discretization

$$\mathbf{a}_t = \begin{pmatrix} 0.1 \\ 1.2 \\ -0.3 \end{pmatrix} \begin{matrix} a_{t,0} \\ a_{t,1} \\ a_{t,2} \end{matrix}$$

e.g., autoregressive Transformer



Why does this work?

$$\text{first step: } p(a_{t,0}|\mathbf{s}_t)$$
$$\text{second step: } p(a_{t,1}|\mathbf{s}_t, a_{t,0})$$
$$\text{third step: } p(a_{t,2}|\mathbf{s}_t, a_{t,0}, a_{t,1})$$

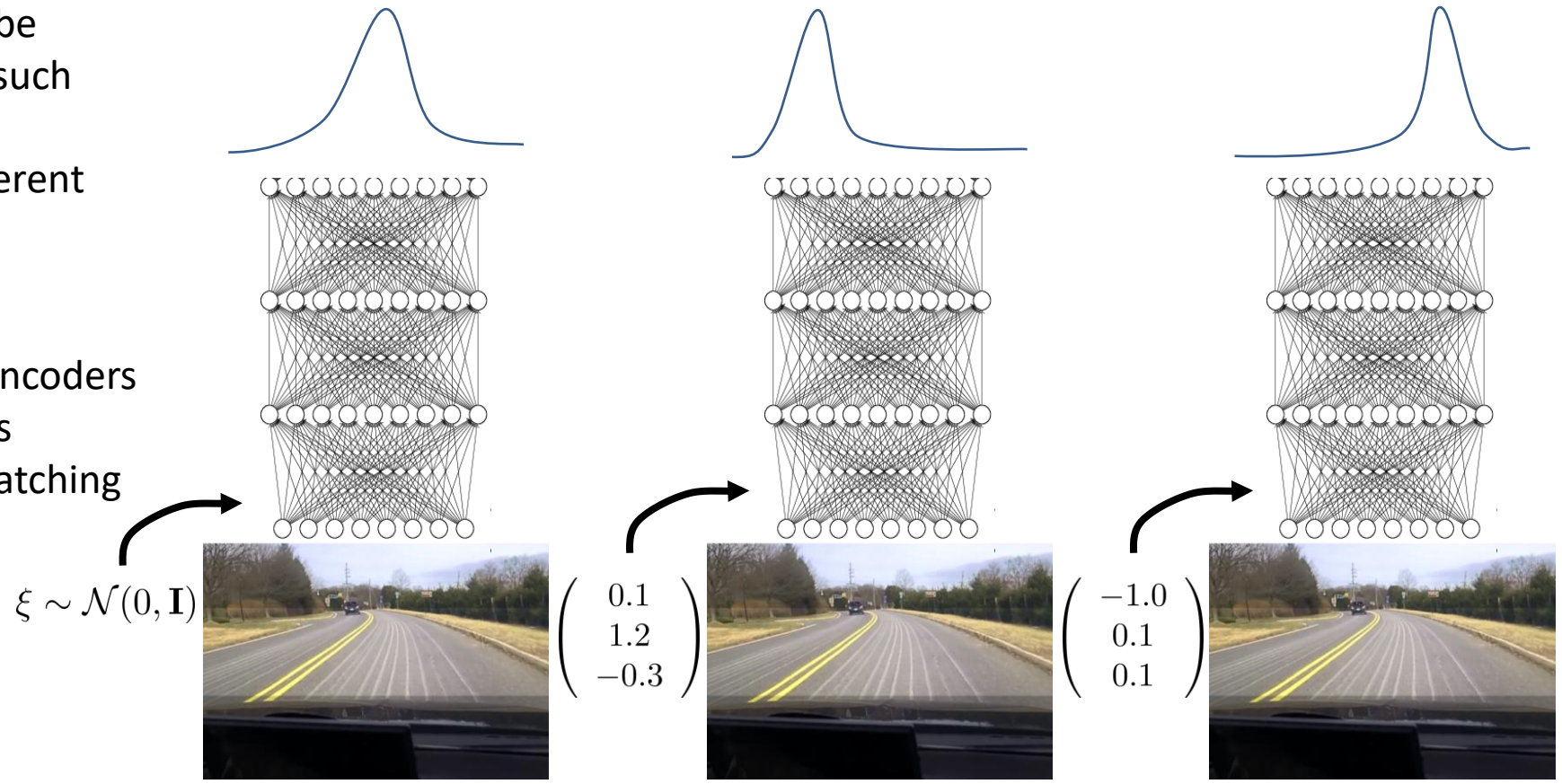
$$p(a_{t,2}|\mathbf{s}_t, a_{t,0}, a_{t,1})p(a_{t,1}|\mathbf{s}_t, a_{t,0})p(a_{t,0}|\mathbf{s}_t)$$
$$= p(a_{t,0}, a_{t,1}, a_{t,2}|\mathbf{s}_t)$$
$$= p(\mathbf{a}_t|\mathbf{s}_t)$$

Expressive continuous distributions

Key challenge: the random noise must actually be used by the model, such that different noise samples map to different action modes

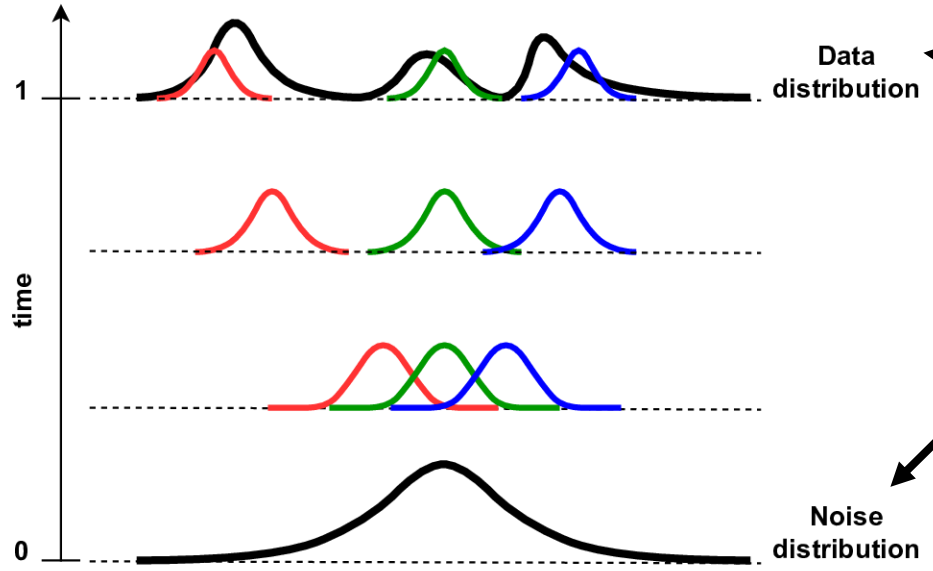
Some solutions:

- Variational autoencoders
- Normalizing flows
- Diffusion/flow matching



Flow matching & diffusion

Informal illustration



Main idea in flow matching:

to model $p(\mathbf{x})$

learn a vector field $v(\mathbf{x}_t, t)$

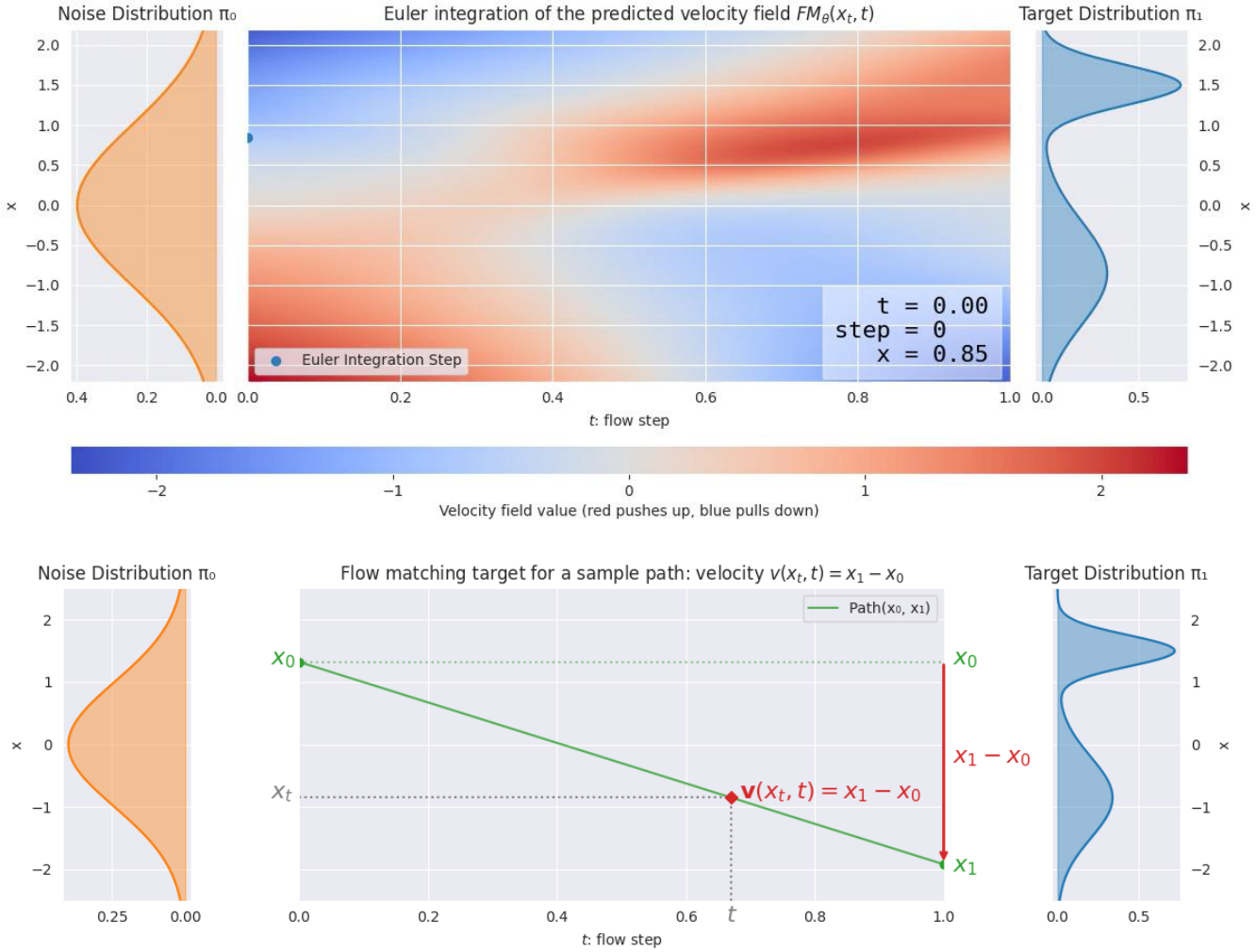
so that we can sample from $p(\mathbf{x})$

by first sampling $\mathbf{x}_0 \sim p_0(\mathbf{x}_0)$

and then integrating the vector field to get

$$\mathbf{x}_1 = \mathbf{x}_0 + \int_0^1 v(\mathbf{x}_t, t)dt$$

Flow matching overview



to sample:

1. sample $\mathbf{x}_0 \sim \mathcal{N}(0, \mathbf{I})$
2. integrate

   for $t \in \{0, \Delta t, 2\Delta t, 3\Delta t, ..., 1 - \Delta t\}$:

   $\mathbf{x}_{t+\Delta t} \leftarrow \mathbf{x}_t + v(\mathbf{x}_t, t)\Delta t$
3. return $\mathbf{x}_1$

to train:

1. sample $\mathbf{x}_0 \sim \mathcal{N}(0, \mathbf{I})$
2. sample $\mathbf{x}_1 \in \mathcal{D}$ where $\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$
3. sample $t \sim p(t)$ (e.g., $p(t) = \mathcal{U}(0, 1)$)
4. compute $\mathbf{x}_t = t\mathbf{x}_1 + (1 - t)\mathbf{x}_0$
5. update v with $\nabla \| v(\mathbf{x}_t, t) - \underbrace{(\mathbf{x}_1 - \mathbf{x}_0)}_{\text{target velocity}} \|^2$

Illustrations: Peter Roelants https://peterroelants.github.io/posts/flow_matching_intro/

Flow matching policy training implementation

1. construct minibatch

   for each element in the batch j:

   sample $(\mathbf{o}_t^{(j)}, \mathbf{a}_t^{(j)})$ from dataset

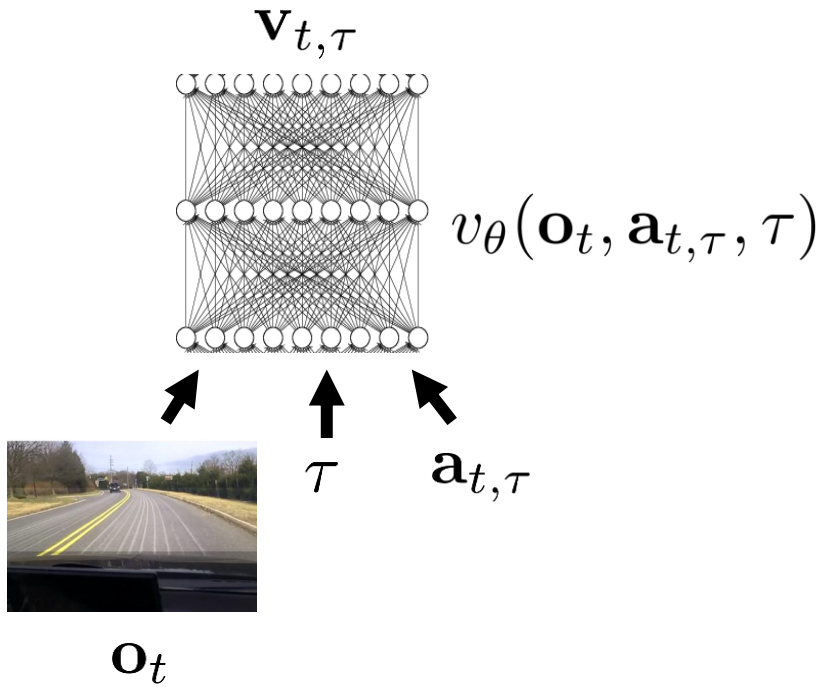
   sample $\mathbf{a}_{t,0}^{(j)} \sim \mathcal{N}(0, \mathbf{I})$

   sample $\tau^{(j)} \sim p(\tau)$ (e.g., $p(\tau) = \mathcal{U}(0,1)$)

   compute $\mathbf{a}_{t,\tau}^{(j)} = \tau^{(j)}\mathbf{a}_t^{(j)} + (1-\tau^{(j)})\mathbf{a}_{t,0}^{(j)}$

2. update $\theta \leftarrow \theta + \alpha\nabla_\theta\mathcal{L}$

   where $\mathcal{L} = \sum_{j=1}^{B} \|v_\theta(\mathbf{o}_t^{(j)}, \mathbf{a}_{t,\tau}^{(j)}, \tau^{(j)}) - (\mathbf{a}_t^{(j)} - \mathbf{a}_{t,0}^{(j)})\|^2$



Action chunking

A small detail that helps quite a lot

standard policy:

sample $\mathbf{a}_t \sim \pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$

execute $\mathbf{a}_t$ in the environment

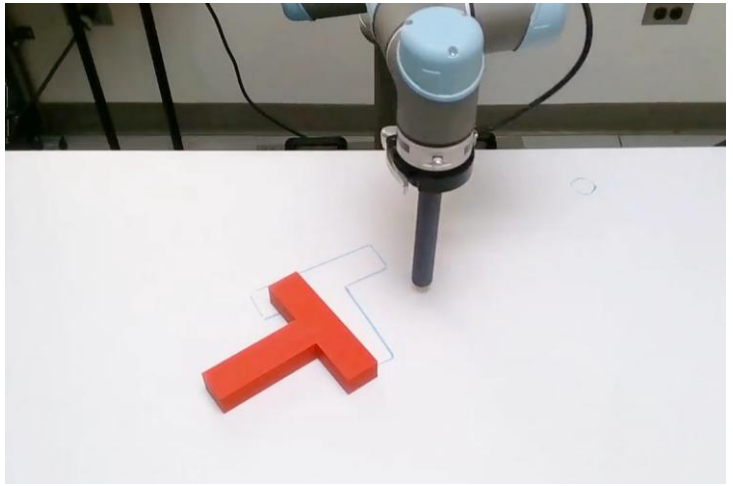
observe $\mathbf{o}_{t+1}$, repeat

action chunked policy:

sample $\mathbf{a}_{t:t+K} \sim \pi_\theta(\mathbf{a}_{t:t+K}|\mathbf{o}_t)$

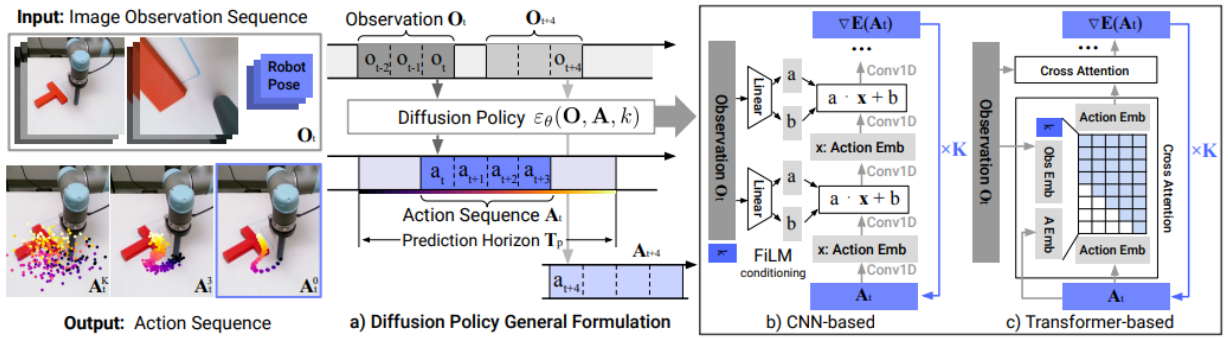
execute $\mathbf{a}_t, \mathbf{a}_{t+1}, ..., \mathbf{a}_{t+K}$ in the environment

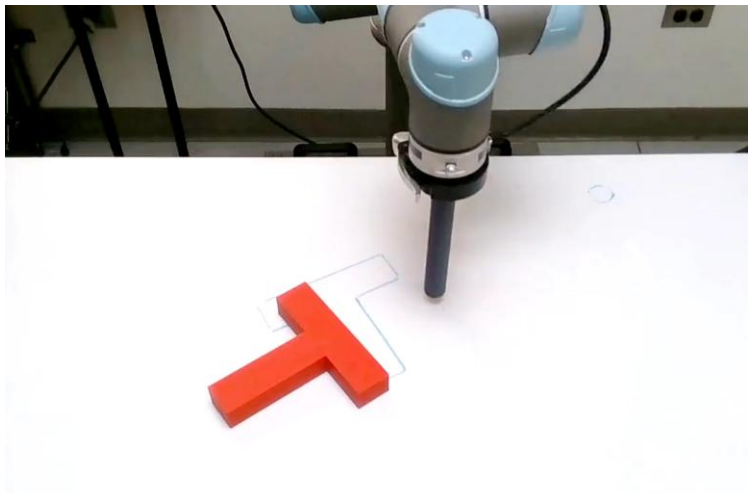
observe $\mathbf{o}_{t+K+1}$, repeat

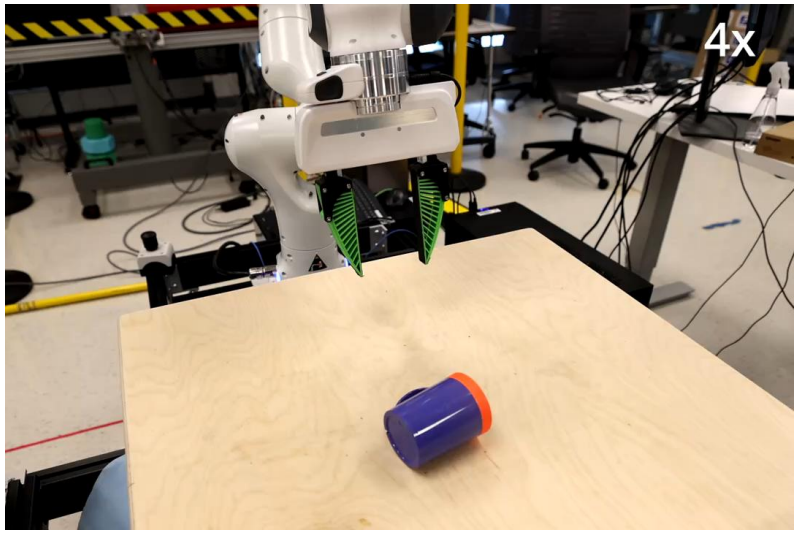


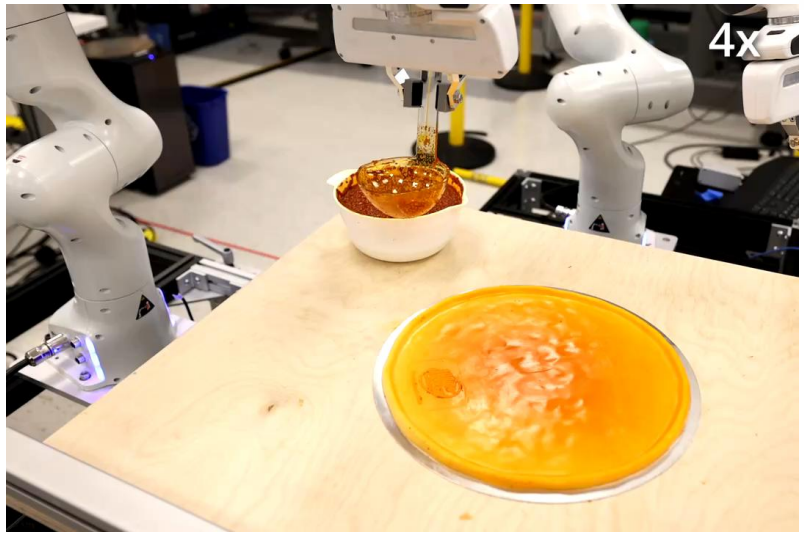
Chi et al. Diffusion policy, 2024

Case study: imitation with diffusion models





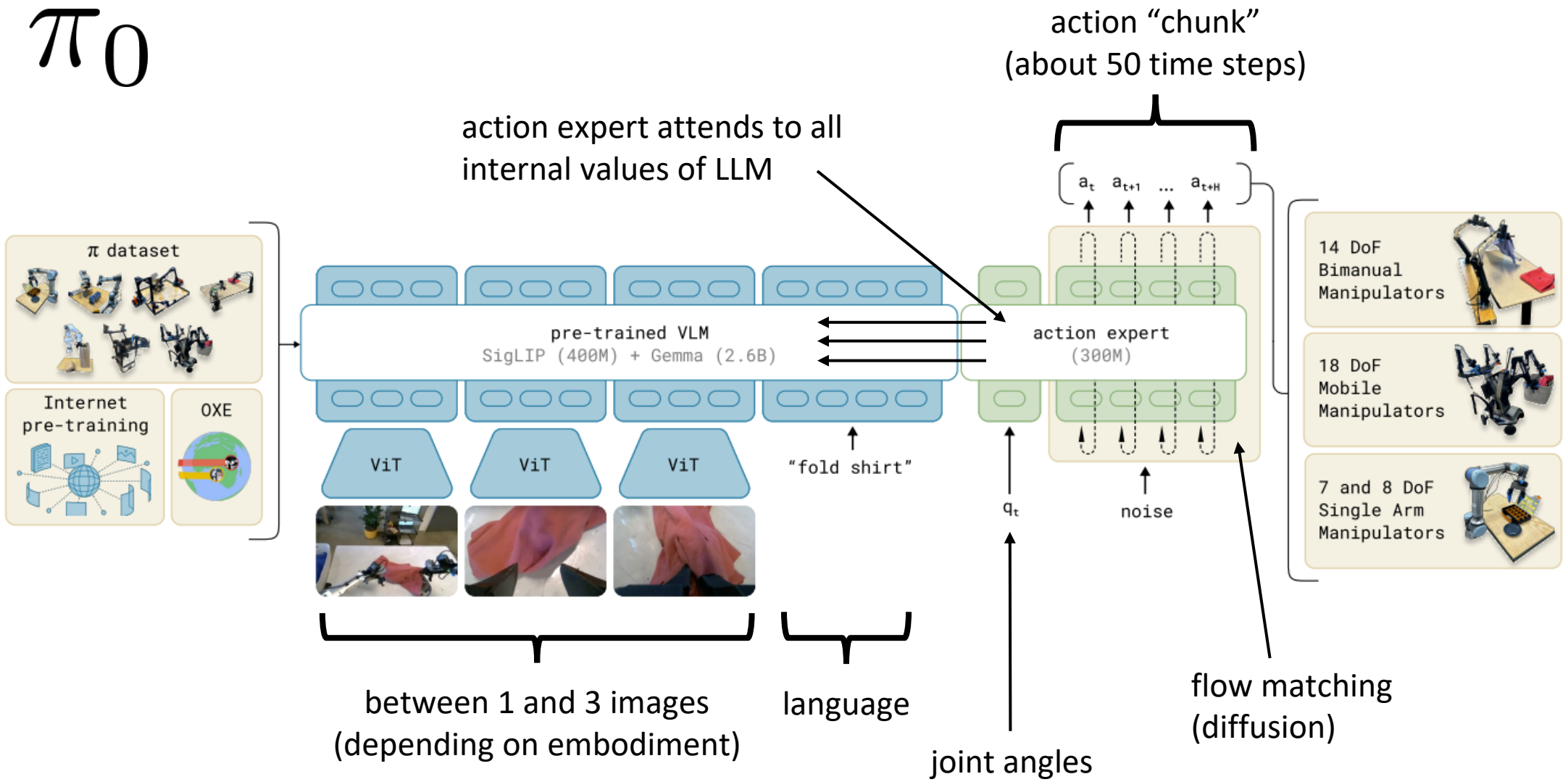




Chi et al. **Diffusion Policy: Visuomotor Policy Learning via Action Diffusion.** 2023

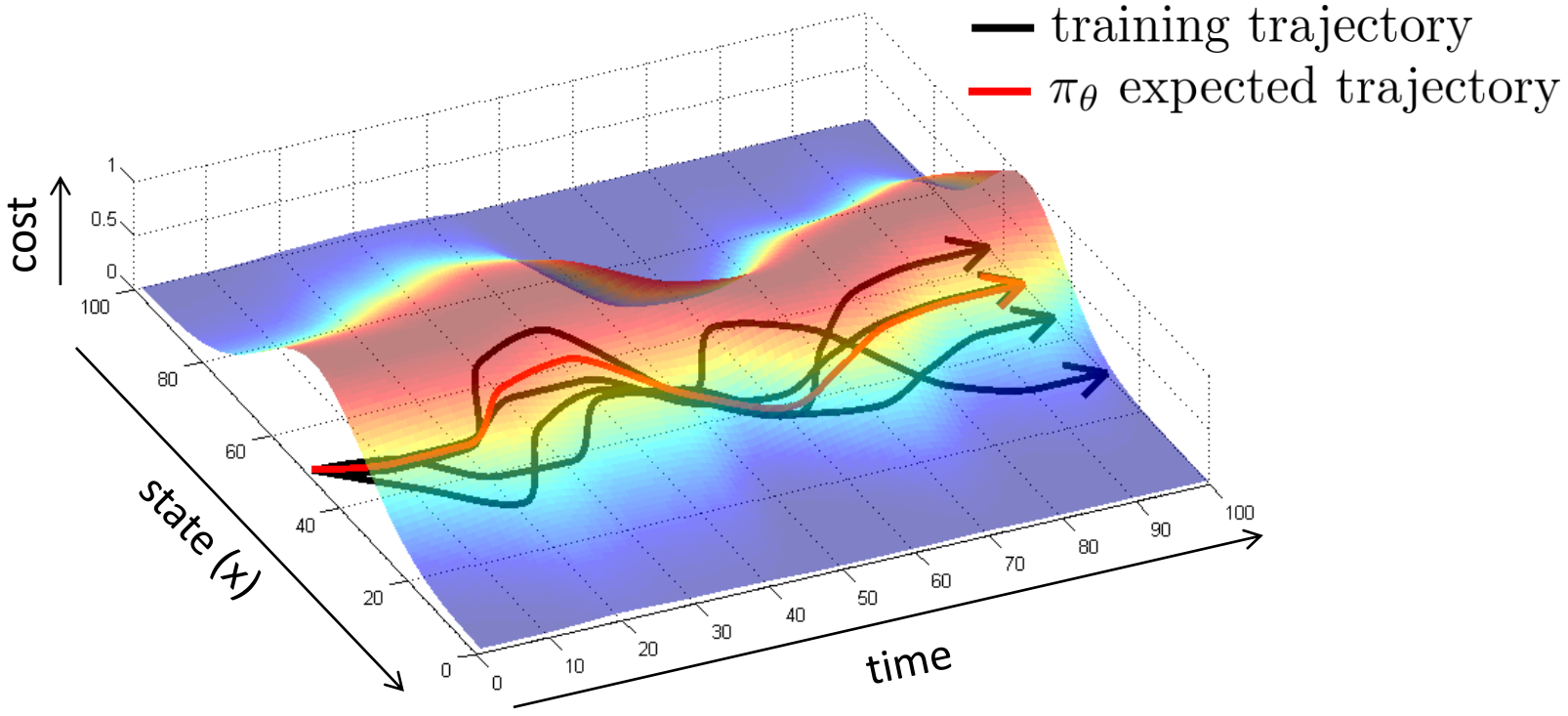
Case study: diffusion + chunking + pre-training

π_0



Part 2:
Narrow vs broad data

Some common tricks



- Intentionally add **mistakes** and **corrections**
  - The mistakes hurt, but the corrections help, often more than the mistakes hurt
- Use **data augmentation**
  - Add some “fake” data that illustrates corrections (e.g., side-facing cameras)

Imitation learning with pre-training

Problem: we want the model to see lots of (bad) situations, so that it knows how to handle them

but we don't want to teach it to enter those (bad) situations!

"bad" data

+ sees lots of situations ("broad")

- has suboptimal actions

"good" data

+ has great actions

- doesn't see many situations ("narrow")

Can we get the best of both worlds?

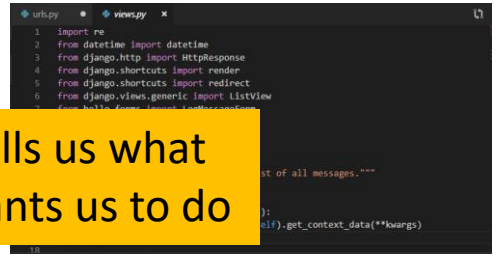
Imitation learning with pre-training

pre-training

post-training

This data gives us the diverse knowledge to do it

This data tells us what the user wants us to do

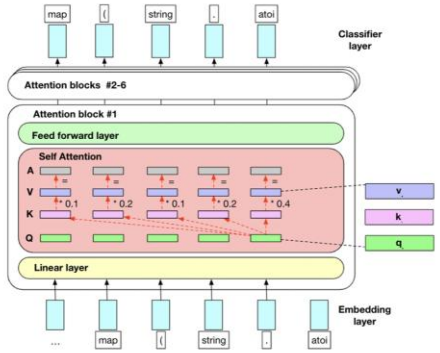


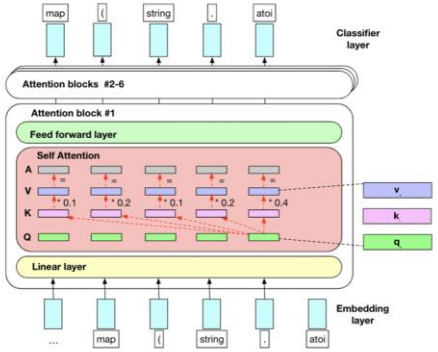
high-quality SFT datasets

“bad” but broad data

“good” but narrow data

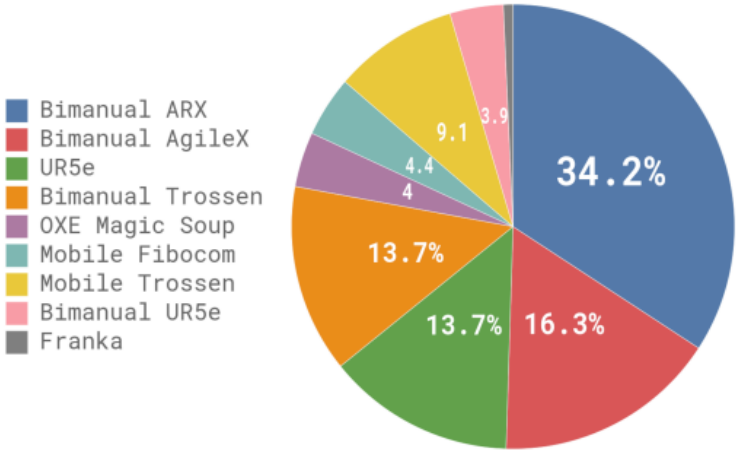
“raw” base model not very useful, but knows lots of things





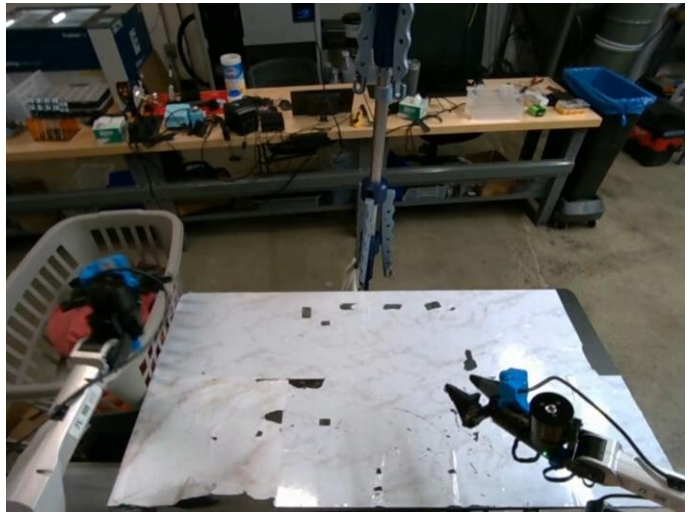
π_0 pre-training and post-training data

pre-training data



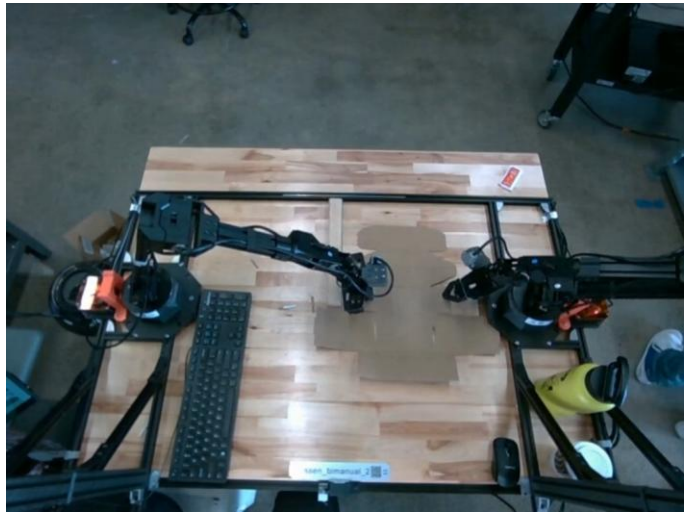
about 10,000 hours of data

post-training data
(laundry folding)



about 20 hours of data

post-training data
(build box)

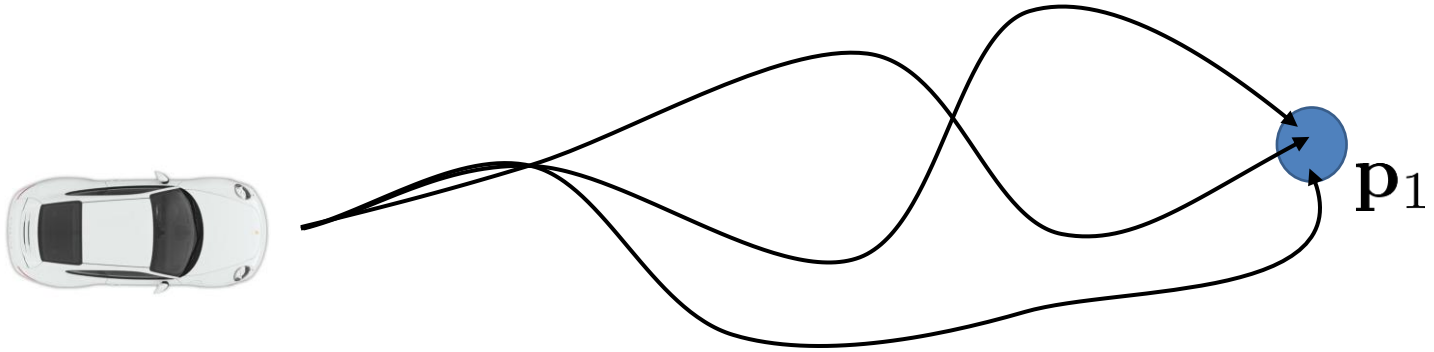


about 20 hours of data

- High-quality but **narrow** data
- Illustrates consistent strategies to perform a task **well**
- By itself doesn't work – robot gets confused if it makes a mistake
- Works great when combined with **pre-training**

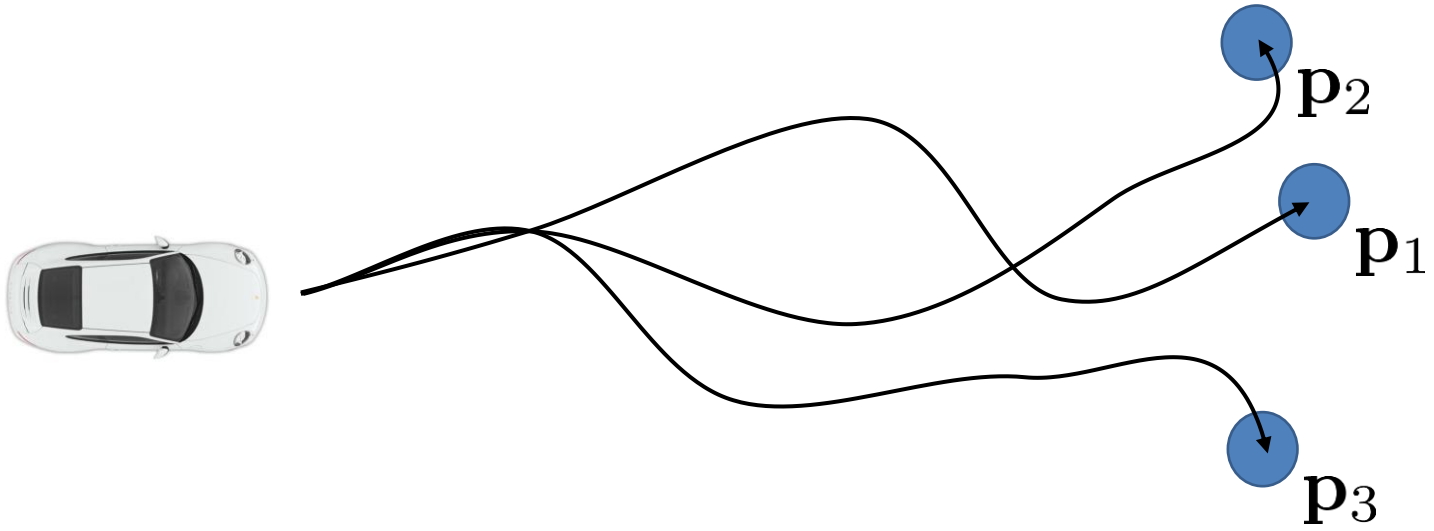
Part 3:
Multi-task learning to the rescue

Does learning **many** tasks become easier?



$\pi_\theta(\mathbf{a}|\mathbf{s})$

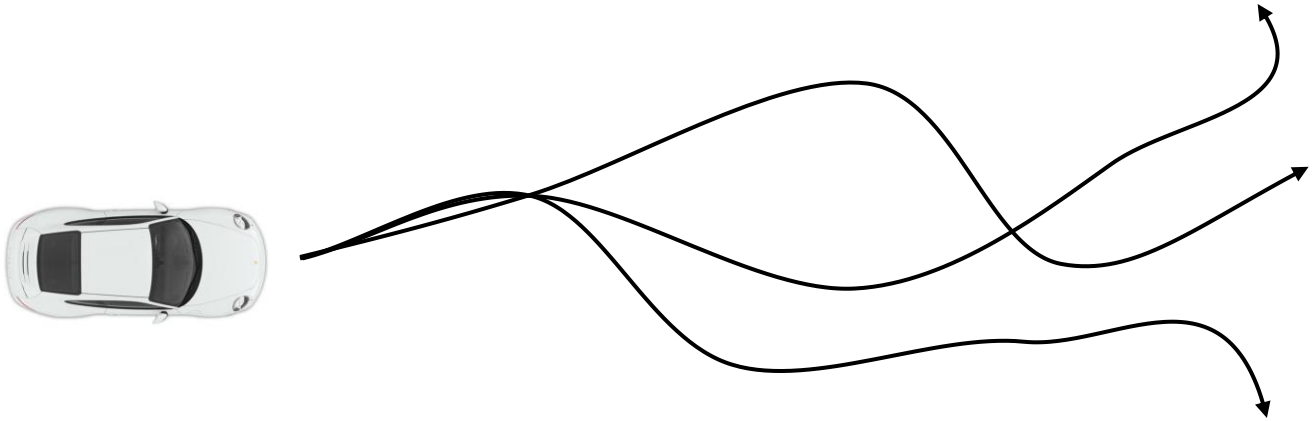
policy for reaching $\mathbf{p}_1$



$\pi_\theta(\mathbf{a}|\mathbf{s}, \mathbf{p})$

policy for reaching *any* $\mathbf{p}$

Goal-conditioned behavioral cloning



training time:

demo 1: $\{\mathbf{s}_1, \mathbf{a}_t, \ldots, \mathbf{s}_{T-1}, \mathbf{a}_{T-1}, \mathbf{s}_T\}$ ⟵ successful demo for reaching $\mathbf{s}_T$

demo 2: $\{\mathbf{s}_1, \mathbf{a}_t, \ldots, \mathbf{s}_{T-1}, \mathbf{a}_{T-1}, \mathbf{s}_T\}$

demo 3: $\{\mathbf{s}_1, \mathbf{a}_t, \ldots, \mathbf{s}_{T-1}, \mathbf{a}_{T-1}, \mathbf{s}_T\}$

learn $\pi_\theta(\mathbf{a}|\mathbf{s}, \mathbf{g})$ ⟵ goal state

for each demo $\{\mathbf{s}_1^i, \mathbf{a}_1^i, \ldots, \mathbf{s}_{T-1}^i, \mathbf{a}_{T-1}^i, \mathbf{s}_T^i\}$

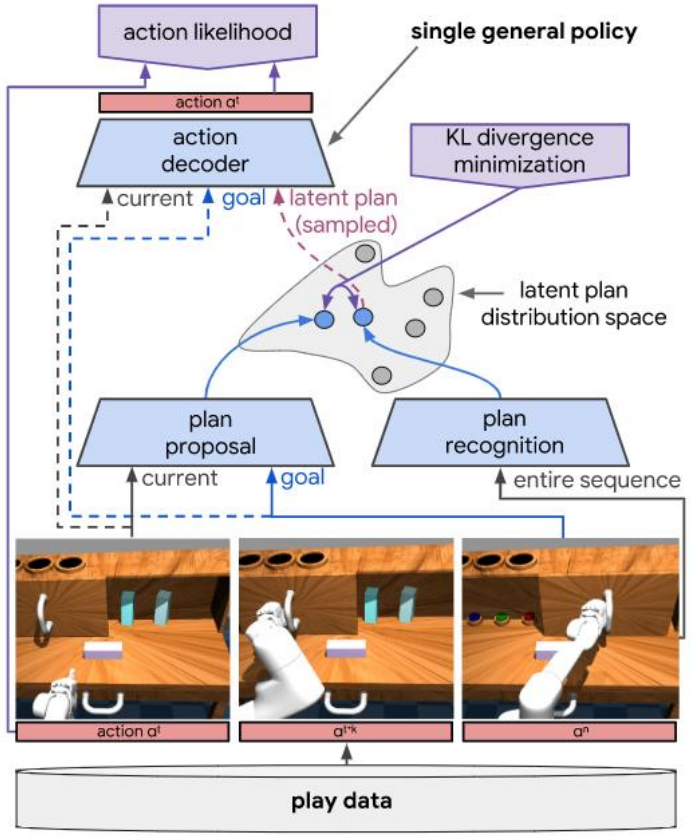
maximize $\log \pi_\theta(\mathbf{a}_t^i|\mathbf{s}_t^i, \mathbf{g} = \mathbf{s}_T^i)$

We see distributional shift in **two** places here!

Can you figure out what the second place is?

Learning Latent Plans from Play

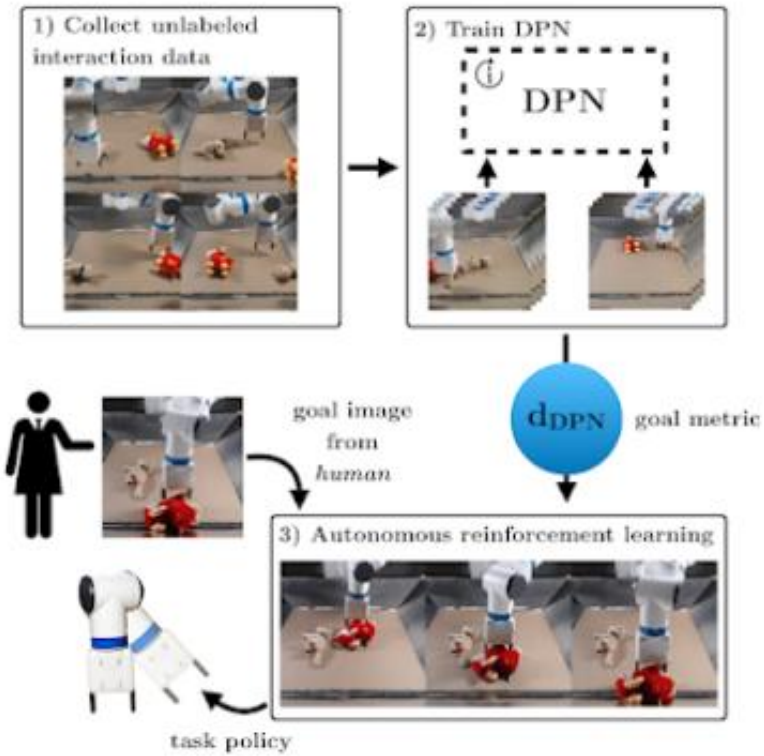
COREY LYNCH Google Brain
MOHI KHANSARI Google X
TED XIAO Google Brain
VIKASH KUMAR Google Brain
JONATHAN TOMPSON Google Brain
SERGEY LEVINE Google Brain
PIERRE SERMANET Google Brain



Unsupervised Visuomotor Control through Distributional Planning Networks

Tianhe Yu, Gleb Shevchuk, Dorsa Sadigh, Chelsea Finn

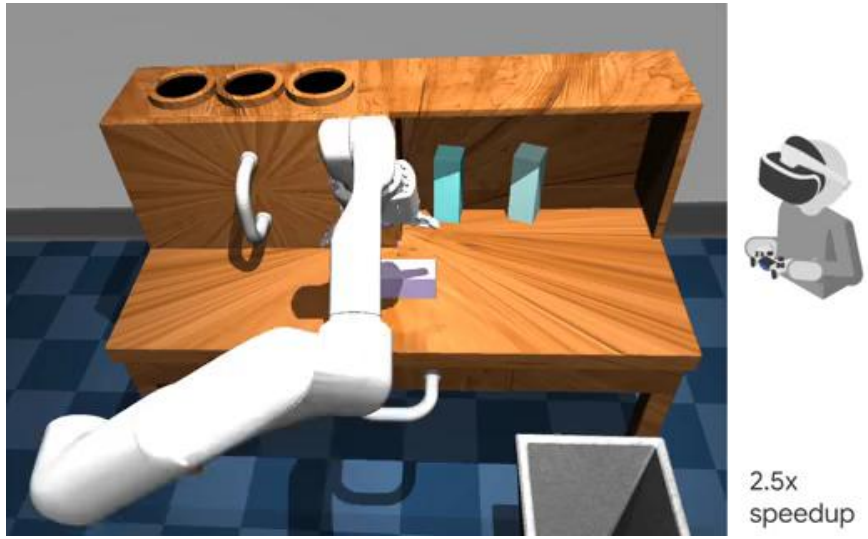
Stanford University



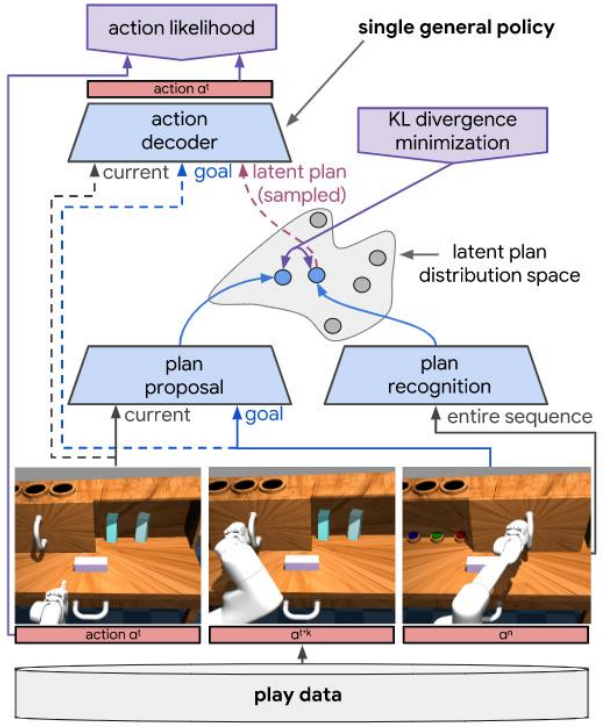
Learning Latent Plans from Play

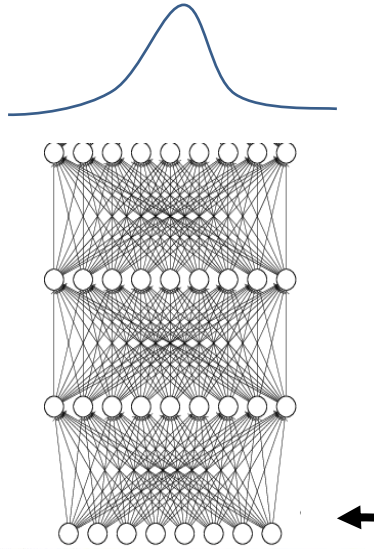
COREY LYNCH Google Brain
MOHI KHANSARI Google X
TED XIAO Google Brain
VIKASH KUMAR Google Brain
JONATHAN TOMPSON Google Brain
SERGEY LEVINE Google Brain
PIERRE SERMANET Google Brain

1. Collect **data**

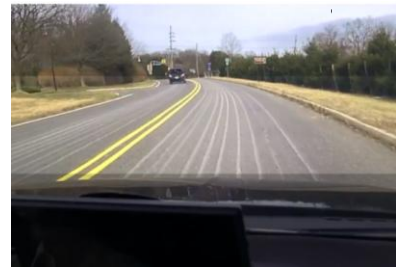


2. Train **goal conditioned** policy





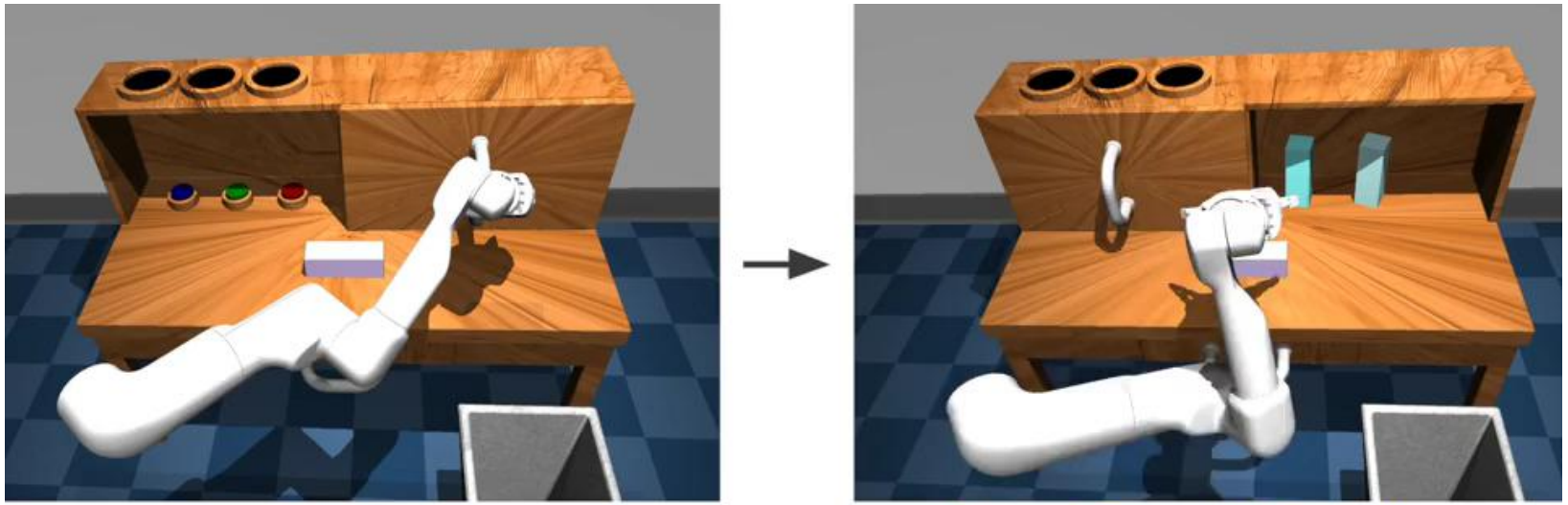
$$\xi \sim \mathcal{N}(0, \mathbf{I})$$



Learning Latent Plans from Play

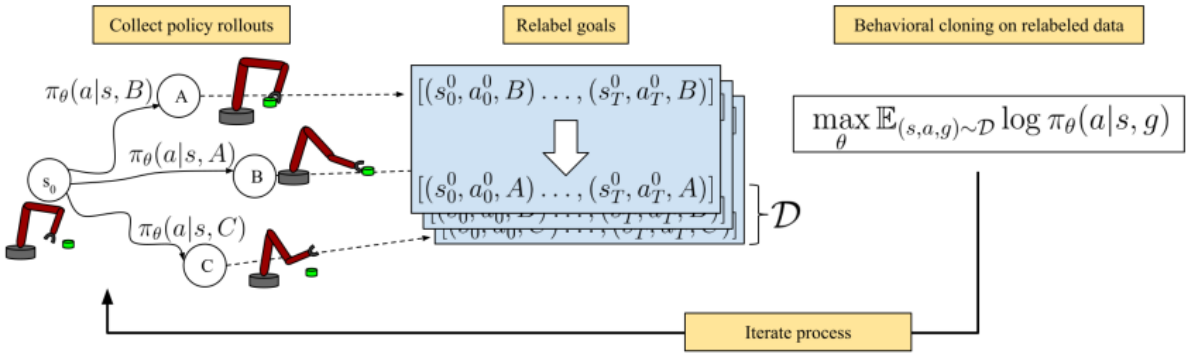
COREY LYNCH Google Brain
MOHI KHANSARI Google X
TED XIAO Google Brain
VIKASH KUMAR Google Brain
JONATHAN TOMPSON Google Brain
SERGEY LEVINE Google Brain
PIERRE SERMANET Google Brain

3. Reach goals



Going beyond just imitation?

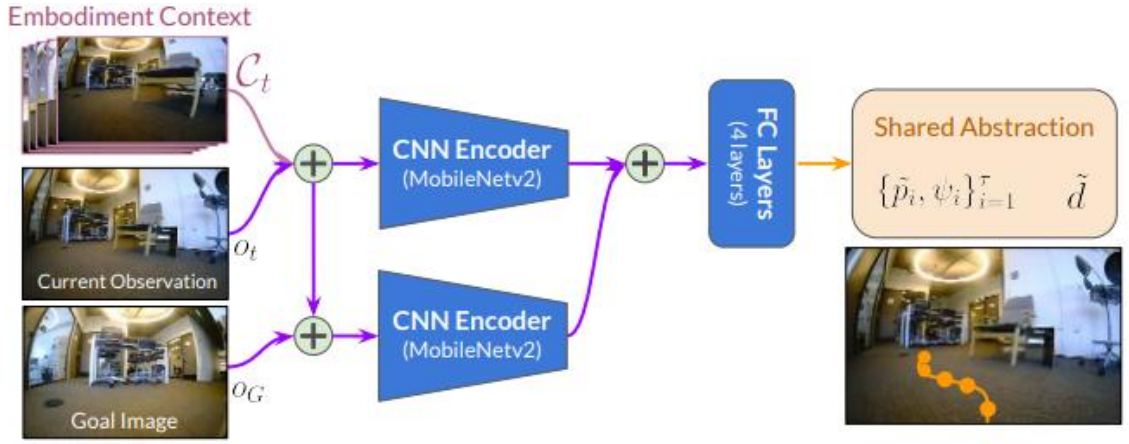
Learning to Reach Goals via Iterated Supervised Learning

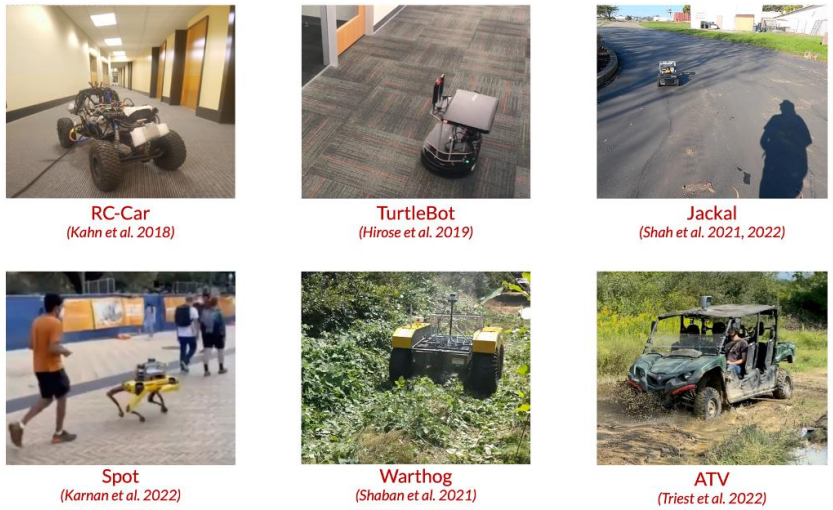


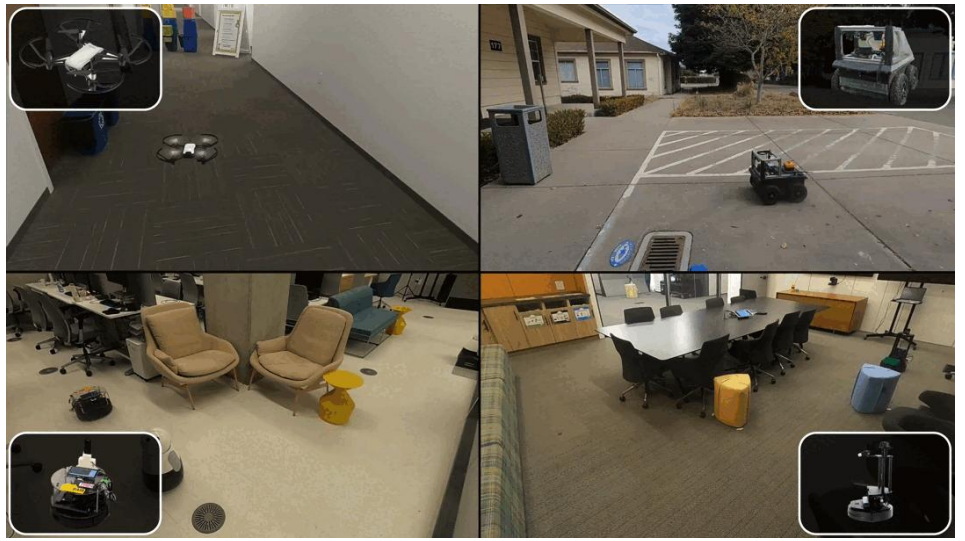
- Start with a **random** policy
- Collect data with **random** goals
- Treat this data as “demonstrations” for the goals that were reached
- Use this to improve the policy
- Repeat

Goal-conditioned BC at a huge scale

	Dataset	Platform	Speed	Amt.	Environment
1	GoStanford [26]	TurtleBot2	0.5m/s	14h	office
2	RECON [32]	Jackal	1m/s	25h	off-road
3	CoryHall [35]	RC Car	1.2m/s	2h	hallways
4	Berkeley [33]	Jackal	2m/s	4h	suburban
5	SCAND-S [36]	Spot	1.5m/s	8h	sidewalks
6	SCAND-J [36]	Jackal	2m/s	1h	sidewalks
7	Seattle [37]	Warthog	5m/s	1h	off-road
8	TartanDrive [38]	ATV	10m/s	5h	off-road
	Ours			60h	







Shah*, Sridhar*, Bhorkar, Hirose, Levine. **GNM: A General Navigation Model to Drive Any Robot**. 2022.

Also related (for later…)

- Similar principle but with reinforcement learning
- This will make more sense later once we cover off-policy value-based RL algorithms
- Worth mentioning because this idea has been used widely outside of imitation