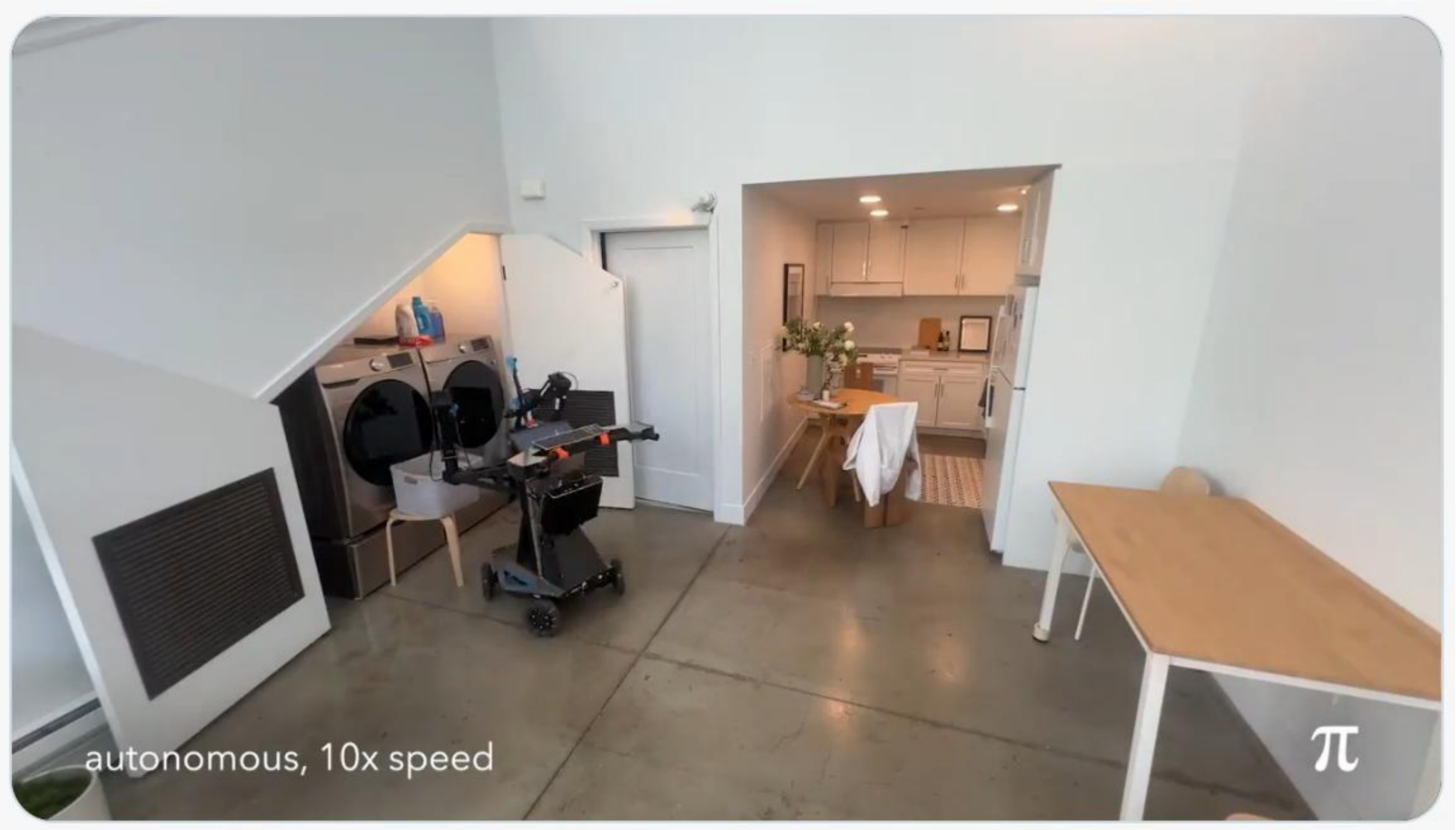
Imitation Learning

CS 6341 Robotics
Professor Yu Xiang
The University of Texas at Dallas

Some Recent Breakthroughs



Physical Intelligence https://www.physicalintelligence.company/blog/pi0

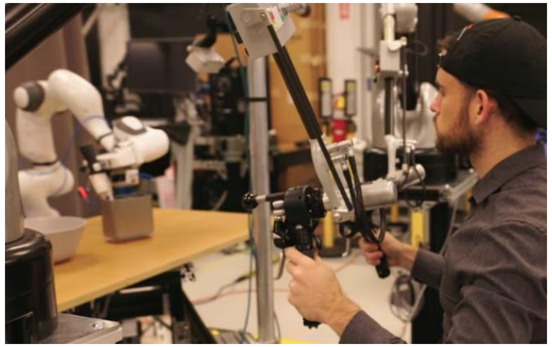
Some Recent Breakthroughs

Mobile ALOHA, Stanford, Zipeng Fu, Tony Zhao, Chelsea Finn https://mobile-aloha.github.io/

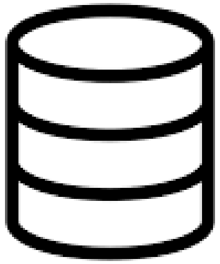
Key Ingredient: Imitation Learning

Kinesthetic Teaching

Teleoperation

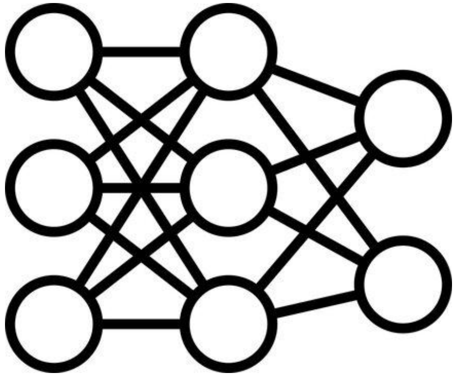


Collect Demonstrations



(state, action)

A Dataset of State-Action Pairs



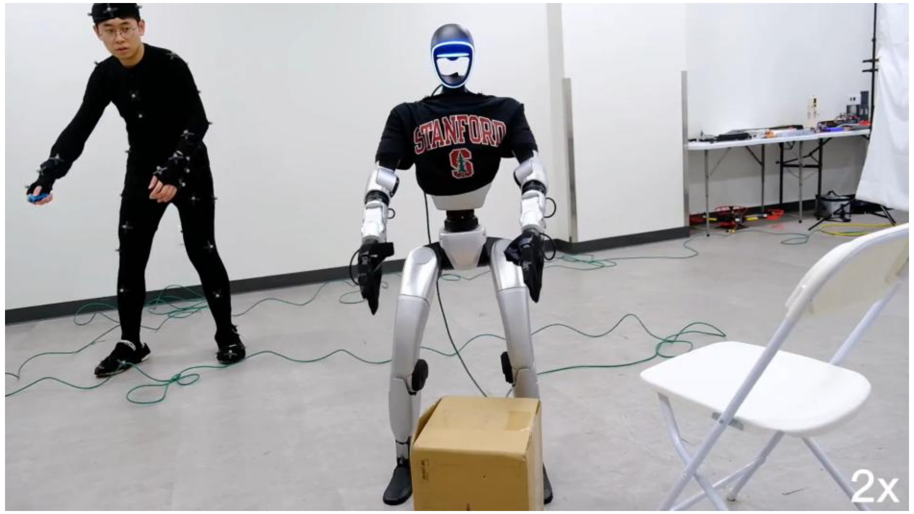
Train a Policy Network

Deploy the Policy Network

Key Ingredient: Teleoperation for Data

https://mobile-aloha.github.io/

https://mobile-tv.github.io/

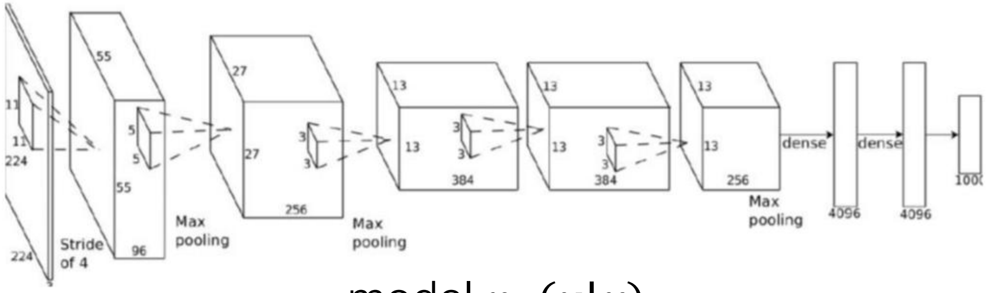


https://yanjieze.com/TWIST/

Tesla

Supervised Learning

input x



model $p_\theta(y|x)$

ragdoll
samoyed
tiger
⋮
bear

prediction y

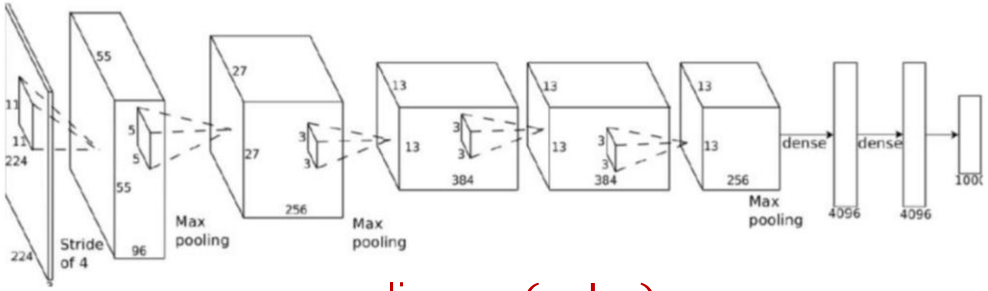
Supervised Learning: Given a training dataset of labeled data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, train the model $p_\theta(y|x)$ by minimizing a loss function (maximize likelihood in this case):

$$\theta^* = \arg\max_\theta \Sigma_D \log p_\theta(y_i|x_i)$$

Behavior Cloning

- Supervised learning from expert demonstrations

state s_t



policy $\pi_\theta(a_t|s_t)$

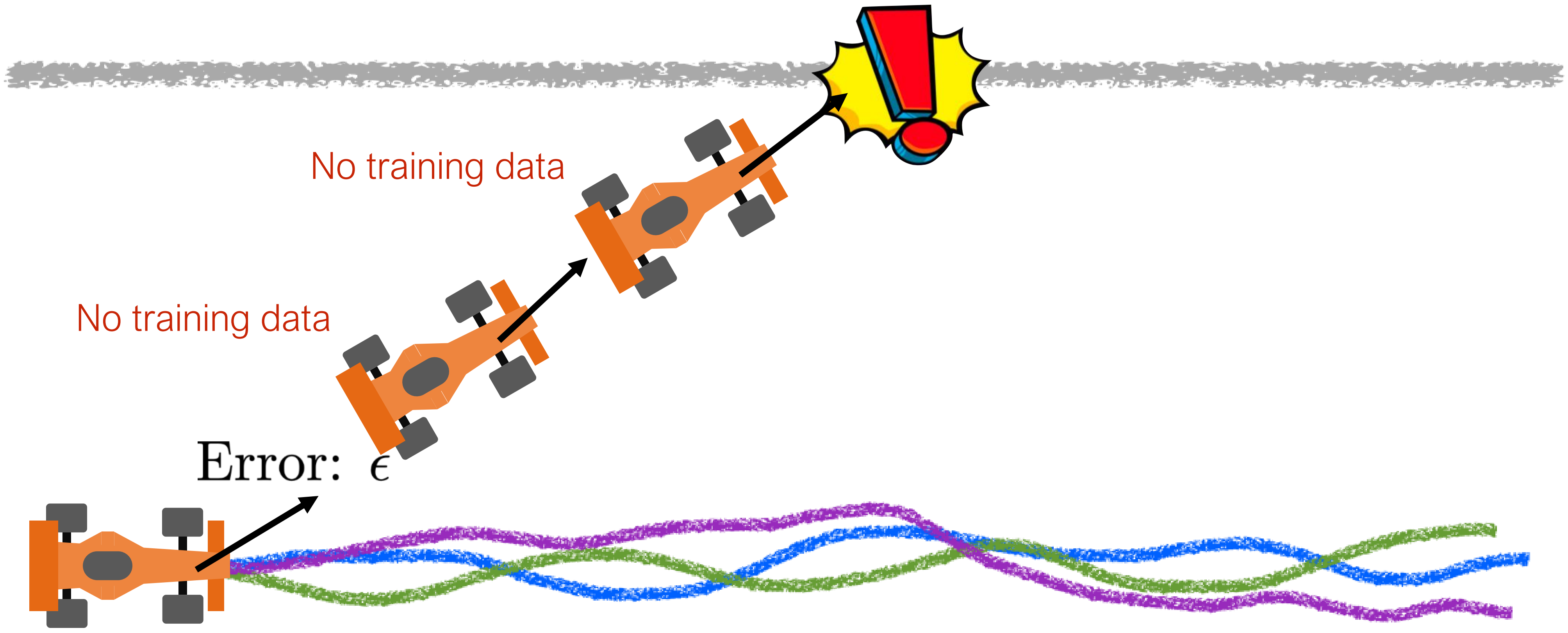
action a_t

Behavior Cloning: Given a training dataset of (expert) behaviors $\mathcal{D} = \{(s_i, a_i)\}_{i=1}^{N}$, train the policy $\pi_\theta(a_t|s_t)$ to maximize the likelihood:

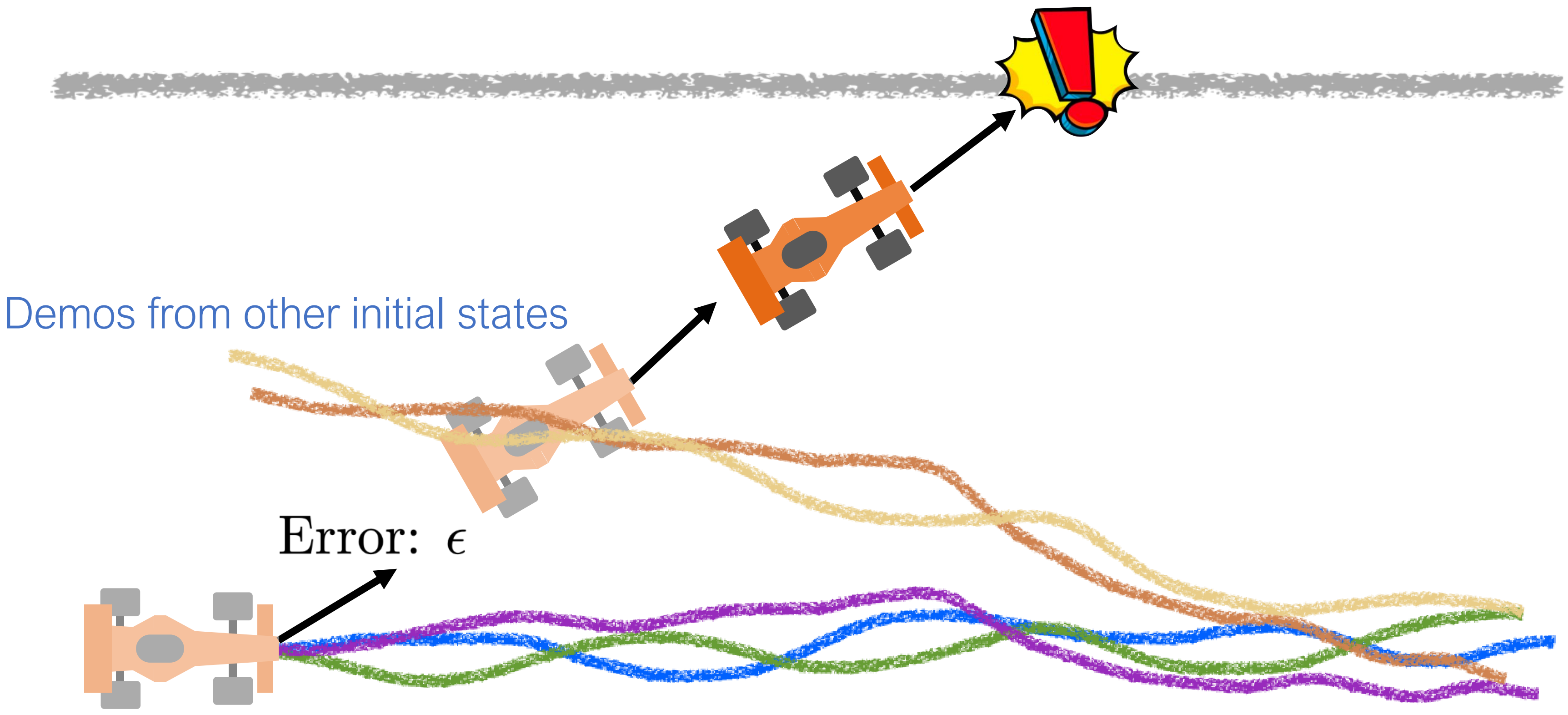
$$\theta^* = \arg\max_\theta \Sigma_D \log \pi_\theta(a_t|s_t)$$

- Classification loss
- Regression loss

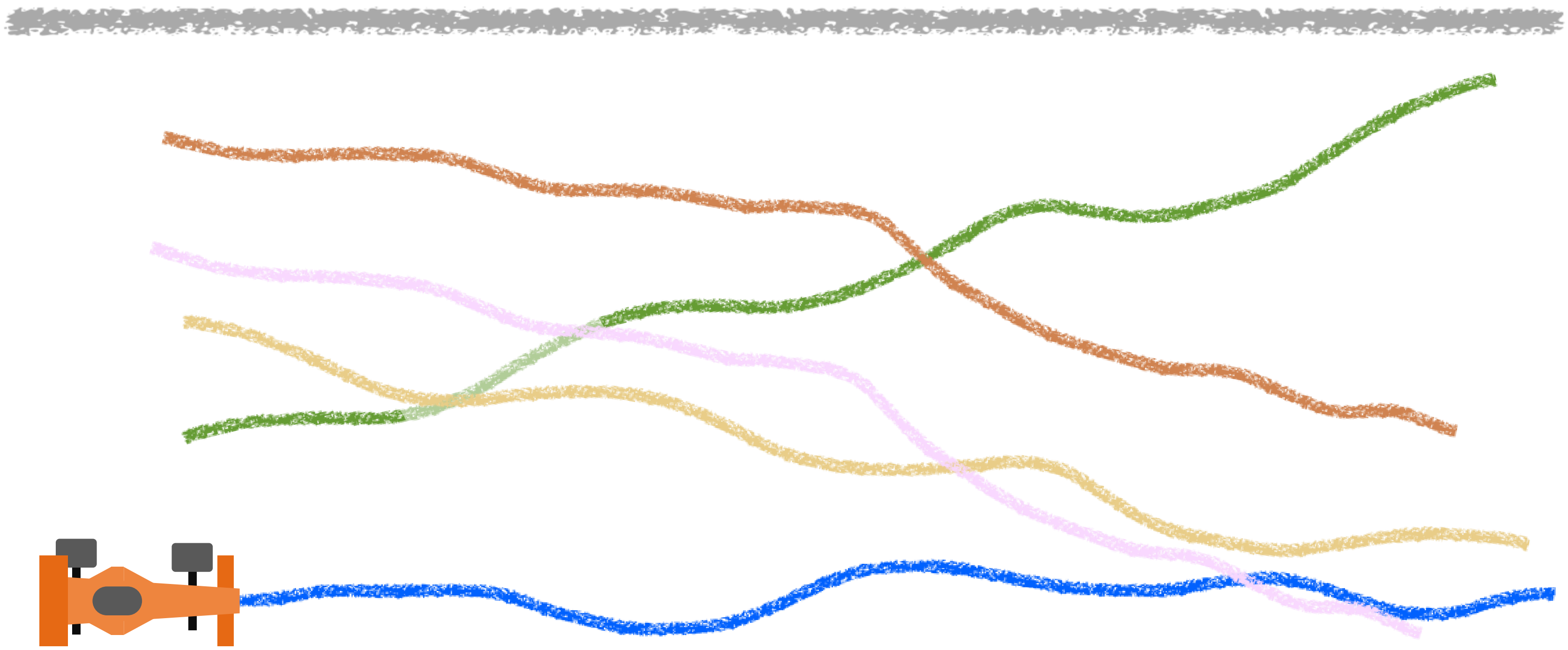
Limitation of Behavior Cloning



Collecting more demonstrations



Collecting more demonstrations

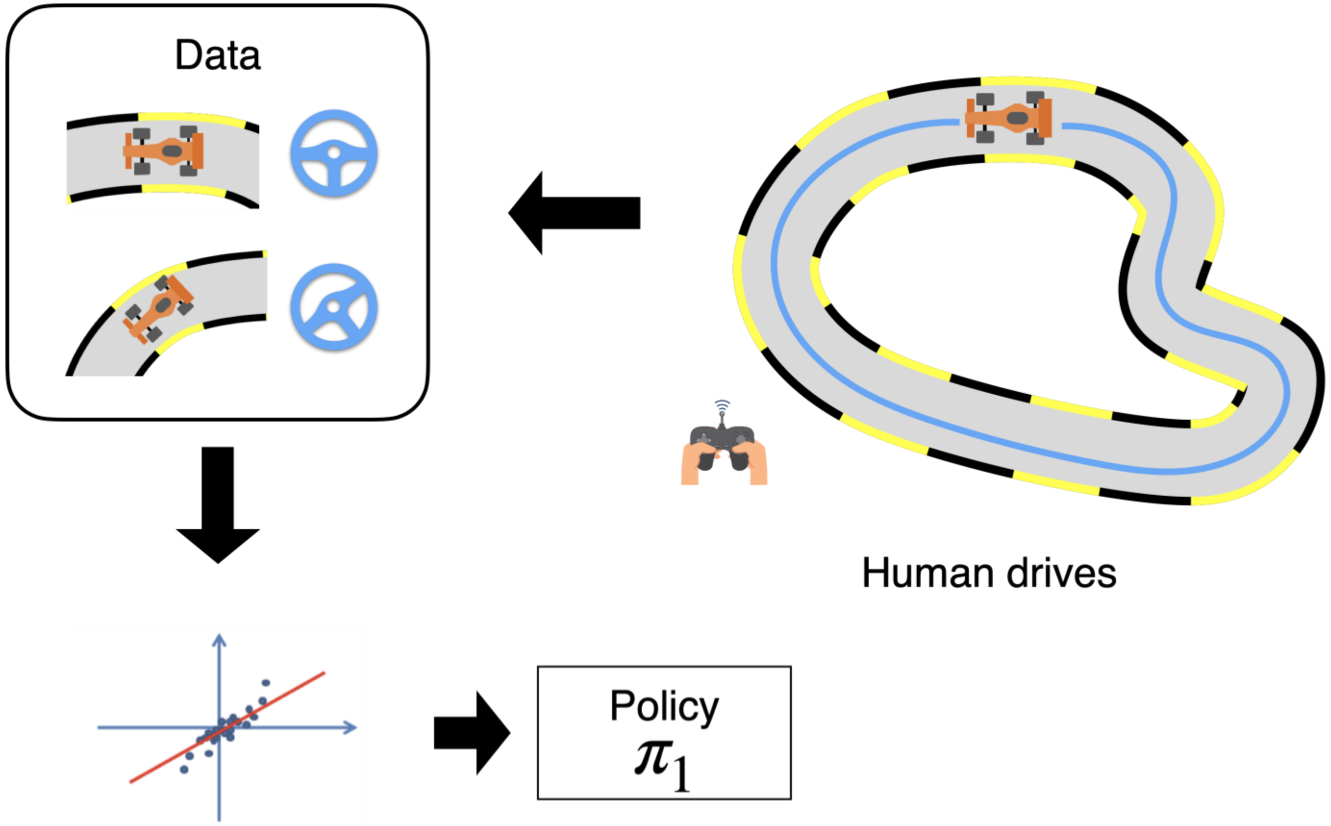


Dagger: Dataset Aggregation

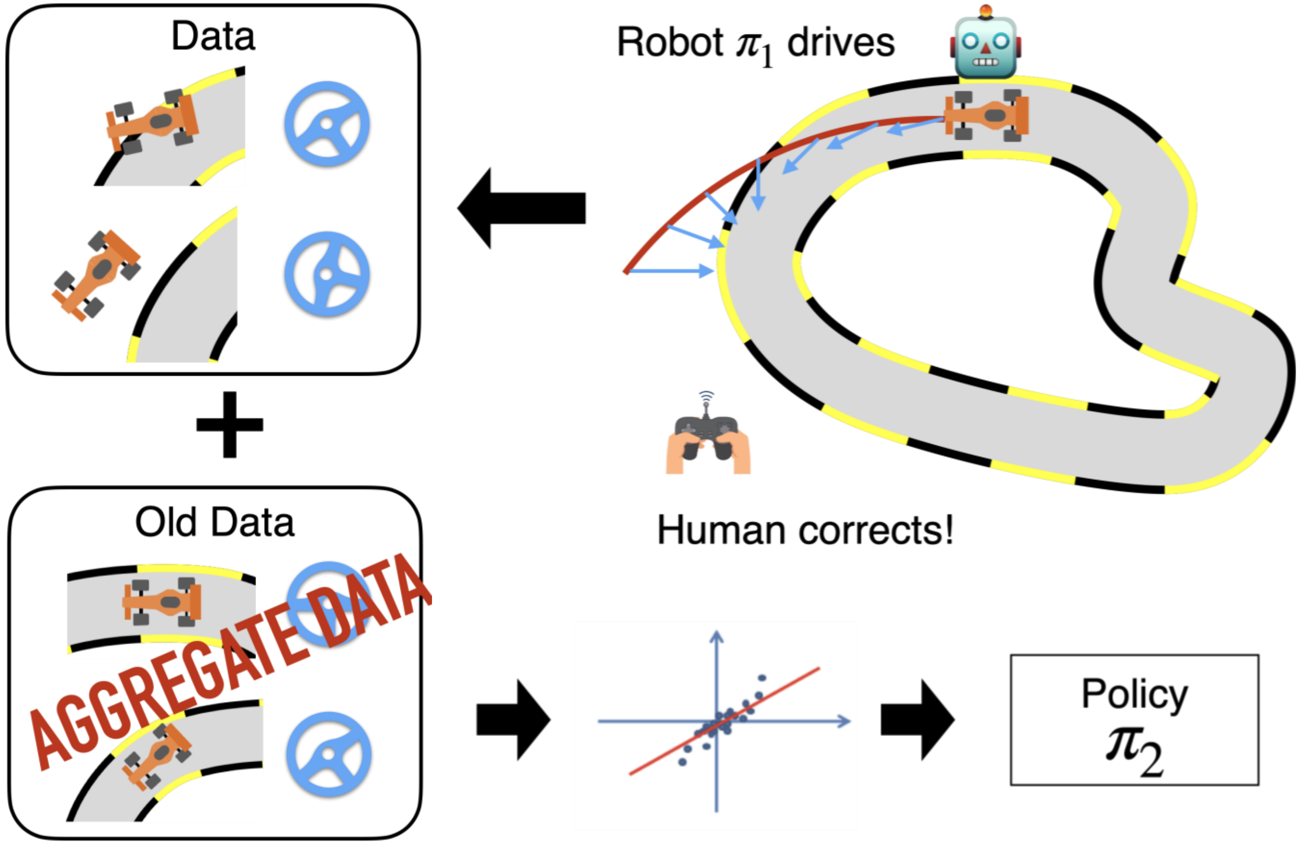
- BC trains only on expert states, but during deployment the learner visits different states → compounding errors

- DAgger solves this by *actively collecting data on the learner's own state distribution* and getting expert labels on those states

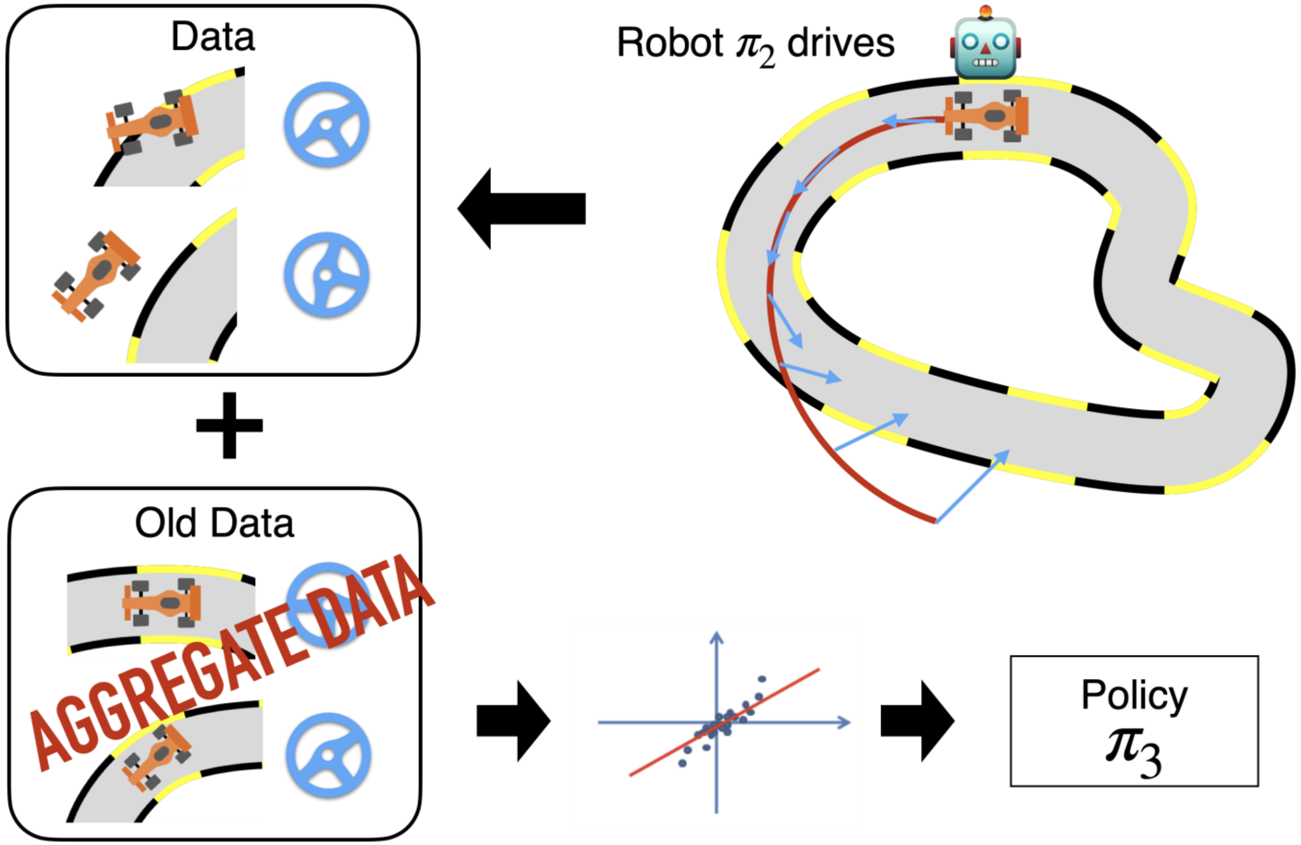
Dagger: Dataset Aggregation



Dagger: Dataset Aggregation

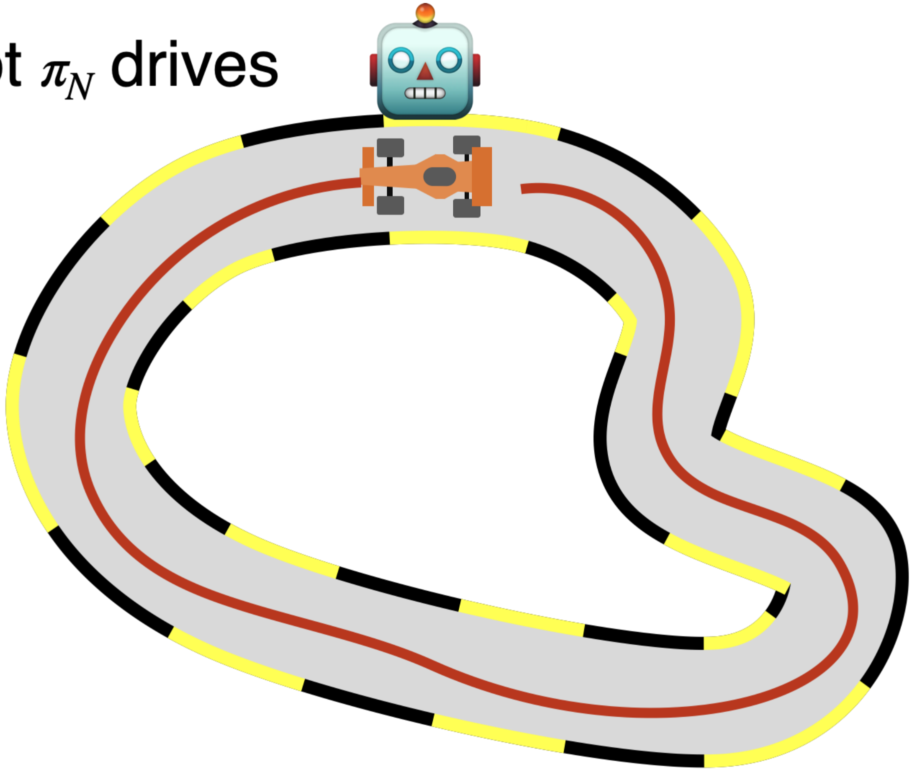


Dagger: Dataset Aggregation



Dagger: Dataset Aggregation

Robot π_N drives



After many iterations ….
we are able to drive like a human!

Dagger: Dataset Aggregation

Initialize with a random policy π_1 # Can be BC

Initialize empty data buffer $\mathcal{D} \leftarrow \{\}$

For $i = 1, \dots, N$

Execute policy π_i in the real world and collect data

$$\mathcal{D}_i = \{s_0, a_0, s_1, a_1, \dots\}$$ # Also called a rollout

Query the expert for the optimal action on learner states

$$\mathcal{D}_i = \{s_0, \pi^\star(s_0), s_1, \pi^\star(s_1), \dots\}$$

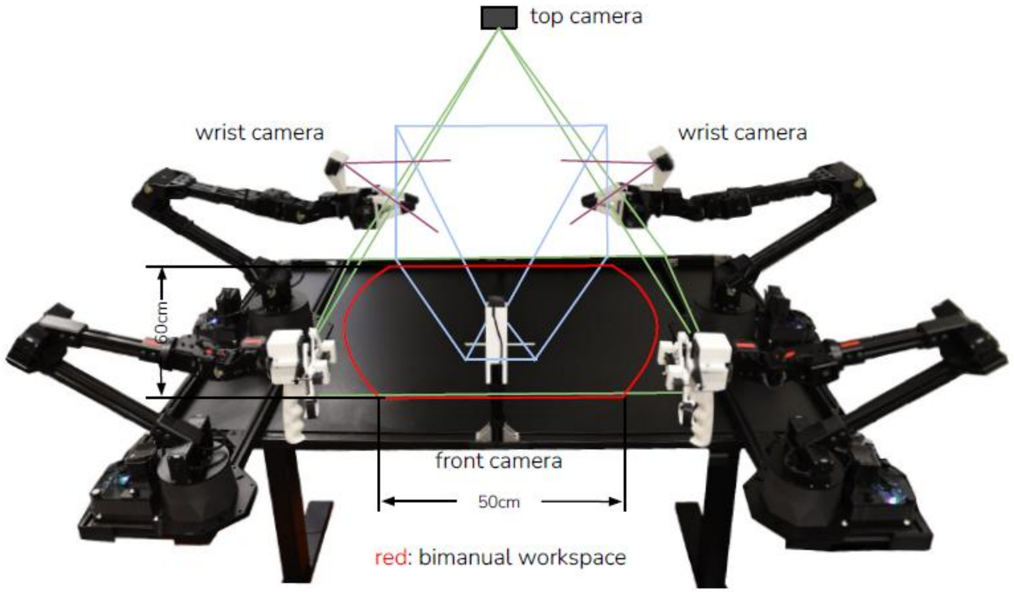
Aggregate data $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_i$

Train a new learner on this dataset $\pi_{i+1} \leftarrow \text{Train}(\mathcal{D})$

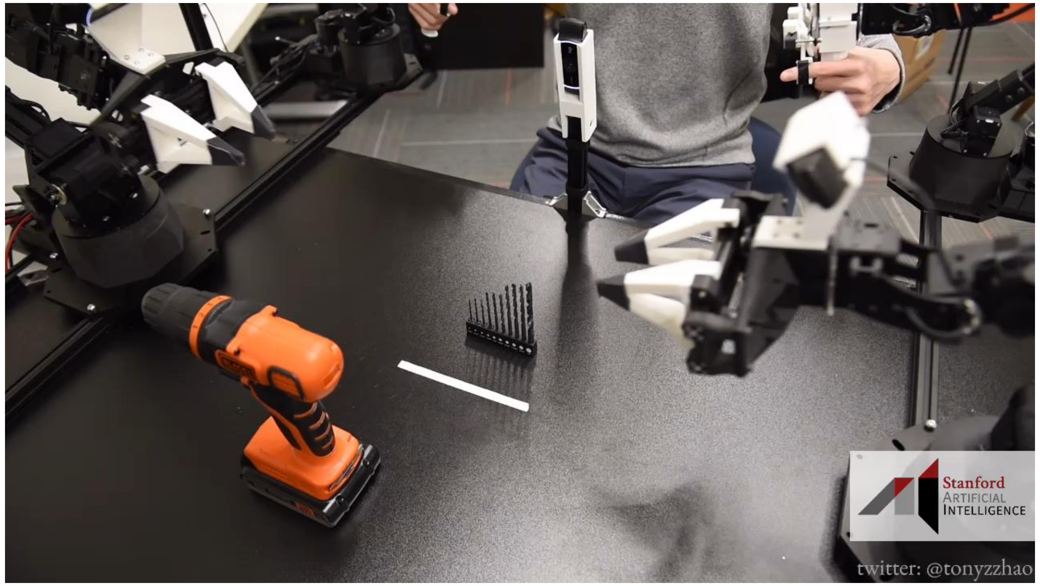
Select the best policy in $\pi_{1:N+1}$

ACT: Action Chunking with Transformers

- ACT to predict the sequence of future actions given the current observations $\pi_\theta(a_{t:t+k}|s_t)$
  - mitigate compounding error, deal with non-Markovian or noisy demonstrations



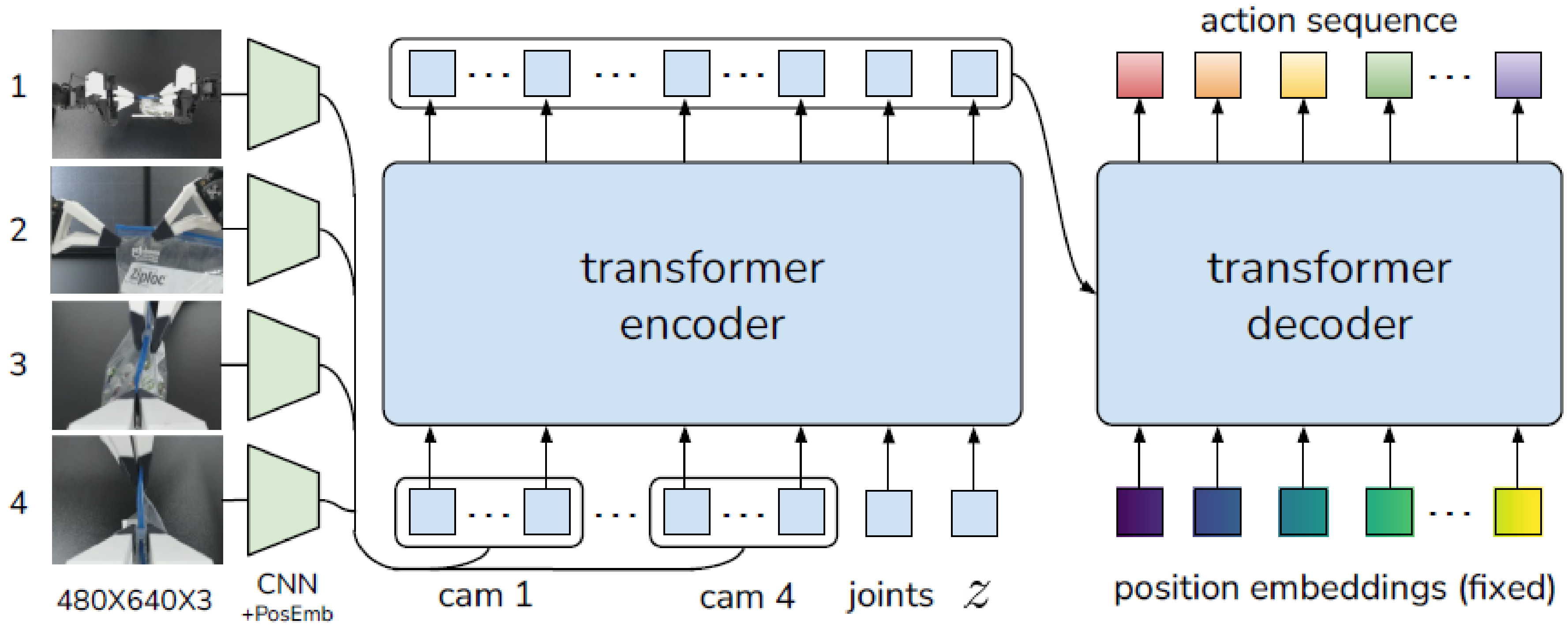
Aloha System



https://tonyzhaozh.github.io/aloha/

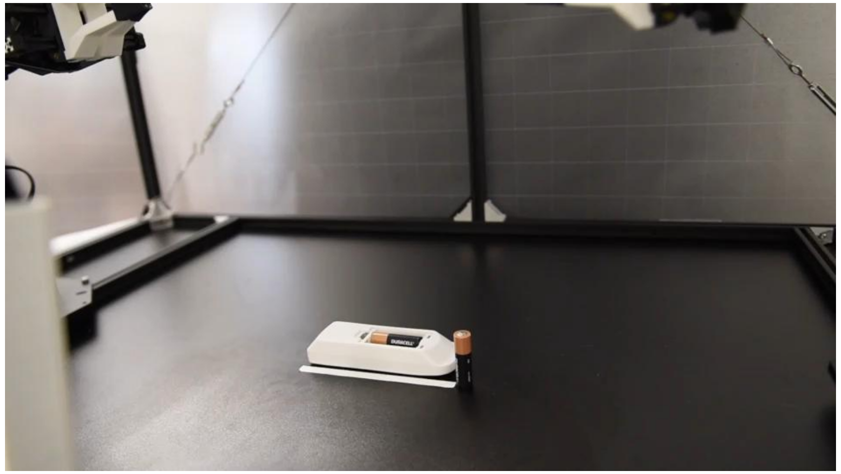
ACT: Action Chunking with Transformers

- ACT to predict the sequence of future actions given the current observations $\pi_\theta(a_{t:t+k}|s_t)$

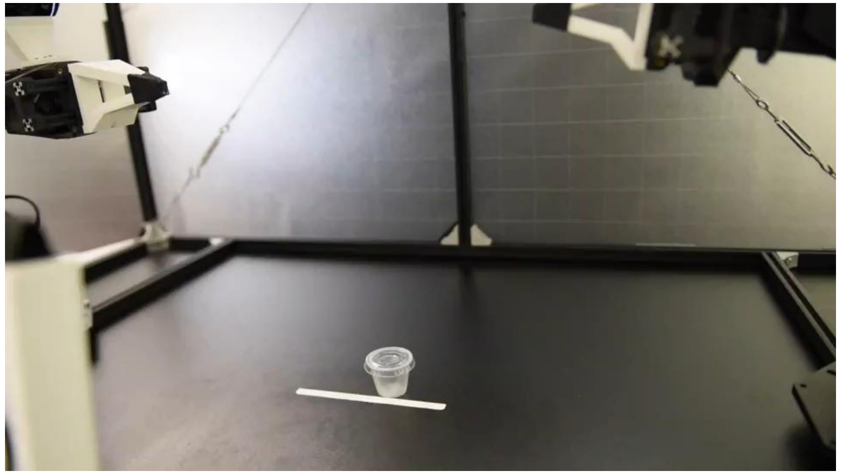


ACT: Action Chunking with Transformers

- 50 demonstration per task, chunk size 90

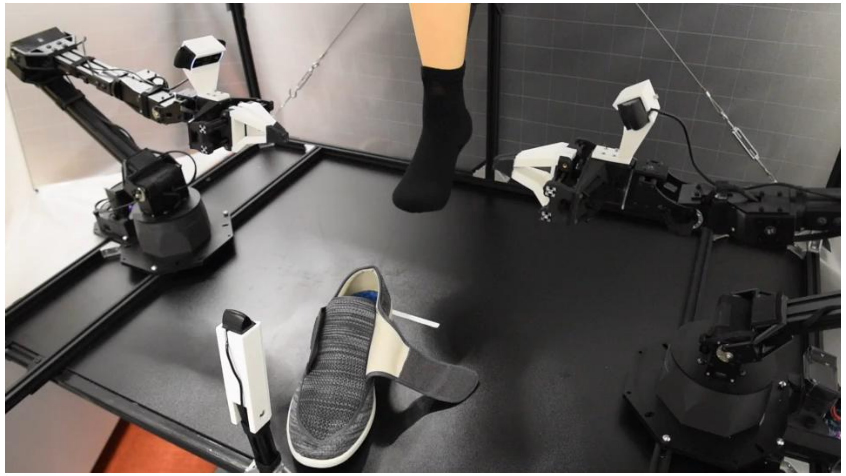


96%



84%

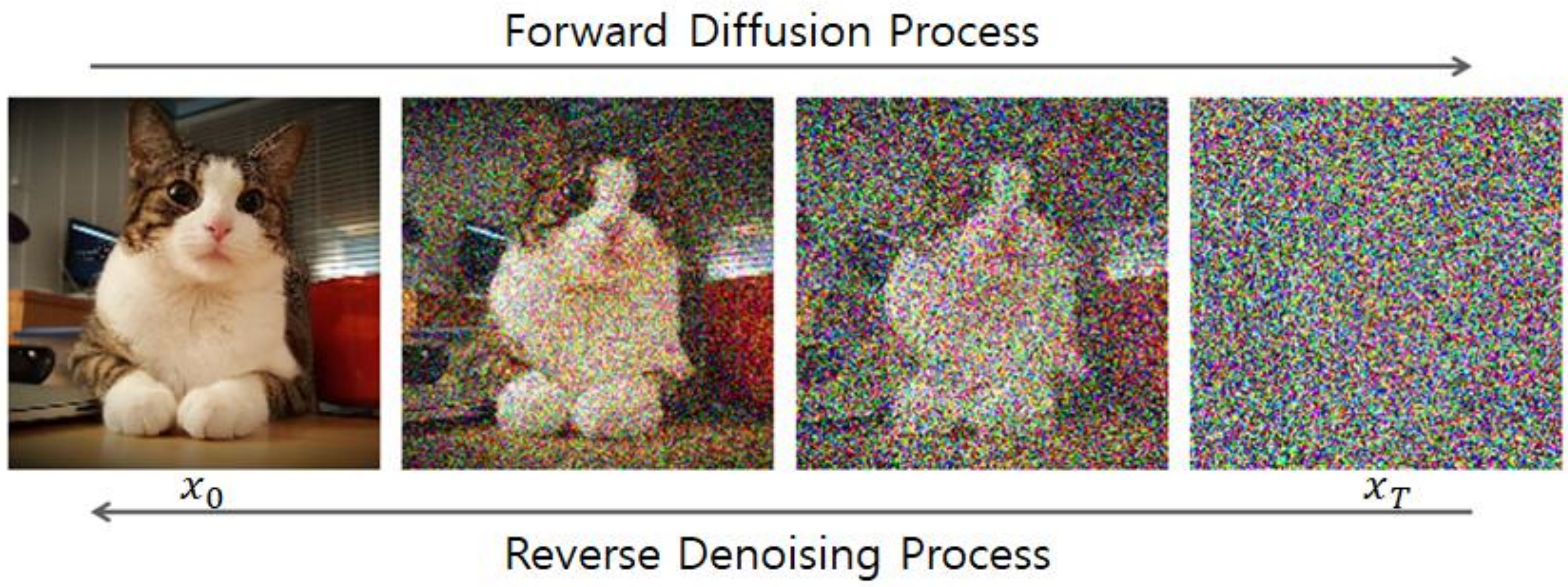
64%



92%

Diffusion Policy

- Uses Denoising Diffusion Probabilistic Models (DDPMs) to generate actions



Denoising Diffusion Probabilistic Models (DDPMs)

- **Forward process** (left → right): adds Gaussian noise step by step

$$x_0 \sim q(x_0) \quad \text{(real data)}$$

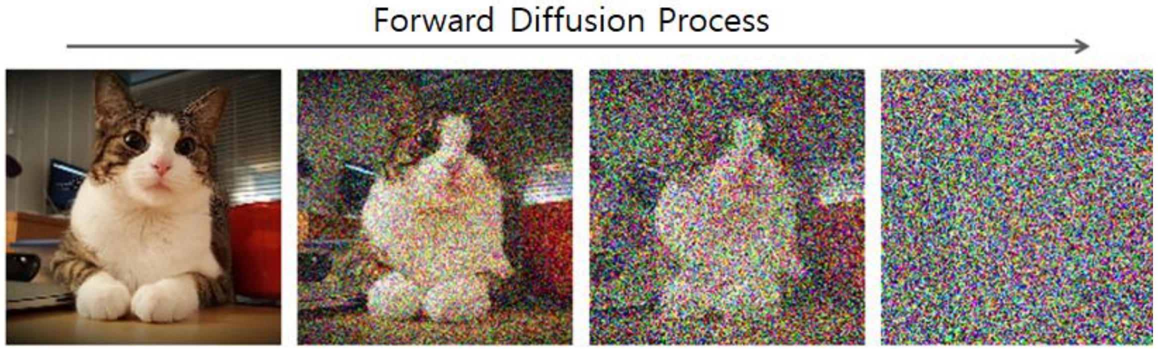
$$x_0 \to x_1 \to \cdots \to x_T \sim \mathcal{N}(0, I)$$

$$q(x_t \mid x_{t-1}) = \mathcal{N}\left(\sqrt{1-\beta_t}\, x_{t-1},\ \beta_t I\right)$$

$$x_t = \sqrt{\alpha_t} x_0 + \sqrt{1-\alpha_t}\, \epsilon, \quad \alpha_t = \prod_{i=1}^{t} (1-\beta_i)$$

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

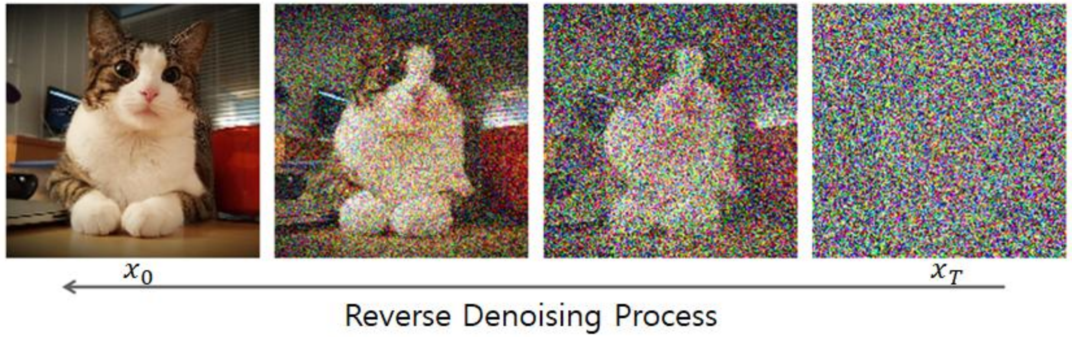
$\beta_1, \beta_2, \ldots, \beta_T$ Known and fixed



Denoising Diffusion Probabilistic Models (DDPMs)

- **Reverse process** (right → left): learned denoising network predicts noise or x_{t-1}

$$x_T \to x_{T-1} \to \cdots \to x_0$$



Training loss $\mathcal{L} = \mathbb{E}_{x_0,t,\epsilon}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|^2\right]$

- ϵ is known
- x_t is computed from x_0 and ϵ
- The model learns to recover the added noise

$$x_t = \sqrt{\alpha_t}x_0 + \sqrt{1-\alpha_t}\,\epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

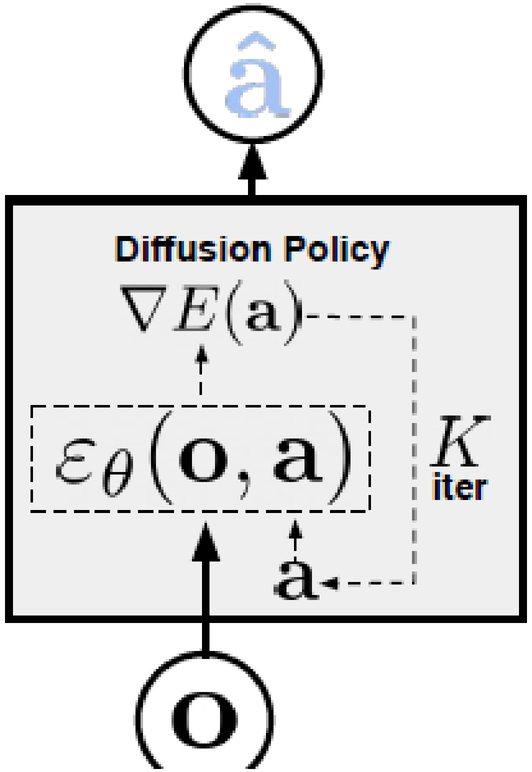
The reverse process uses a neural network **to predict that same noise**: $\epsilon_\theta(x_t, t) \approx \epsilon$

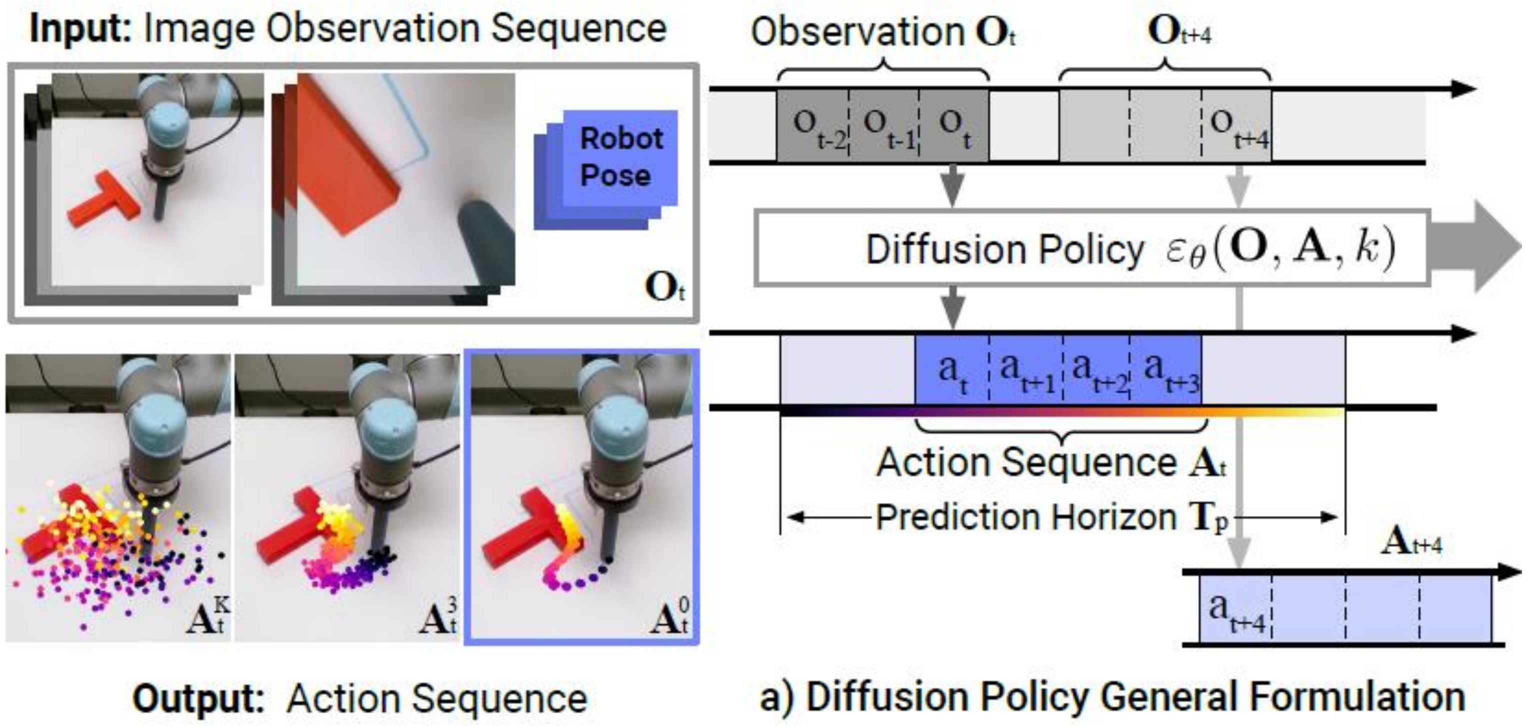
$$\hat{x}_0 = \frac{x_t - \sqrt{1-\alpha_t}\,\epsilon_\theta(x_t, t)}{\sqrt{\alpha_t}}$$

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}(\mu_\theta(x_t, t),\ \Sigma_t)$$

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{1-\beta_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\epsilon_\theta(x_t, t)\right)$$

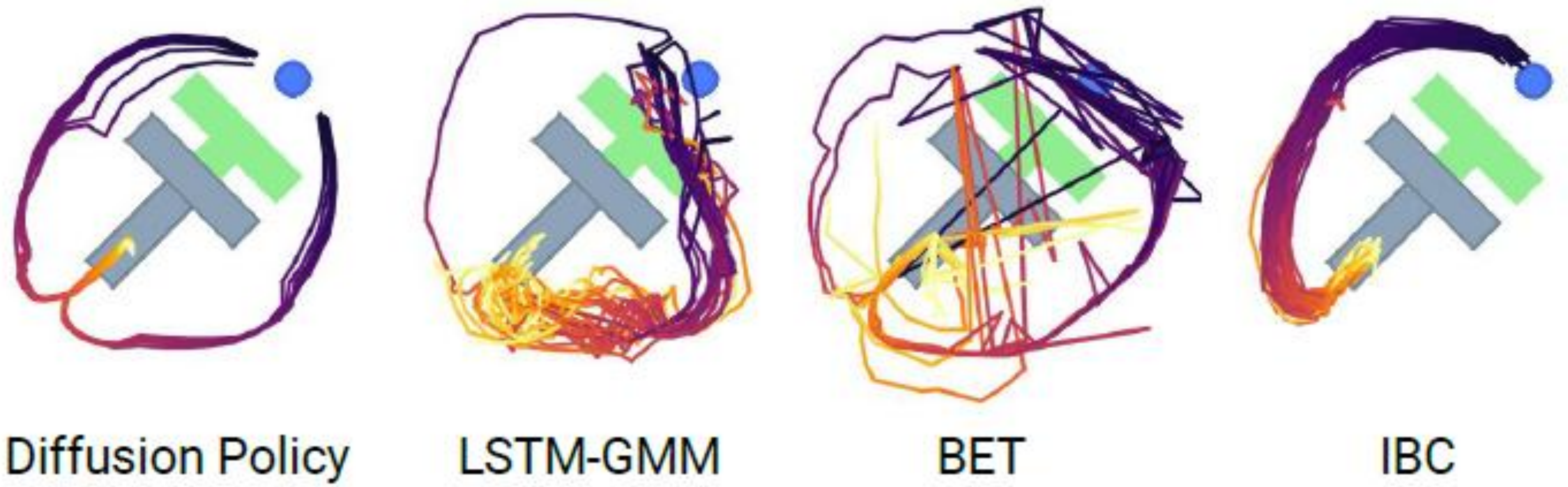
Diffusion Policy



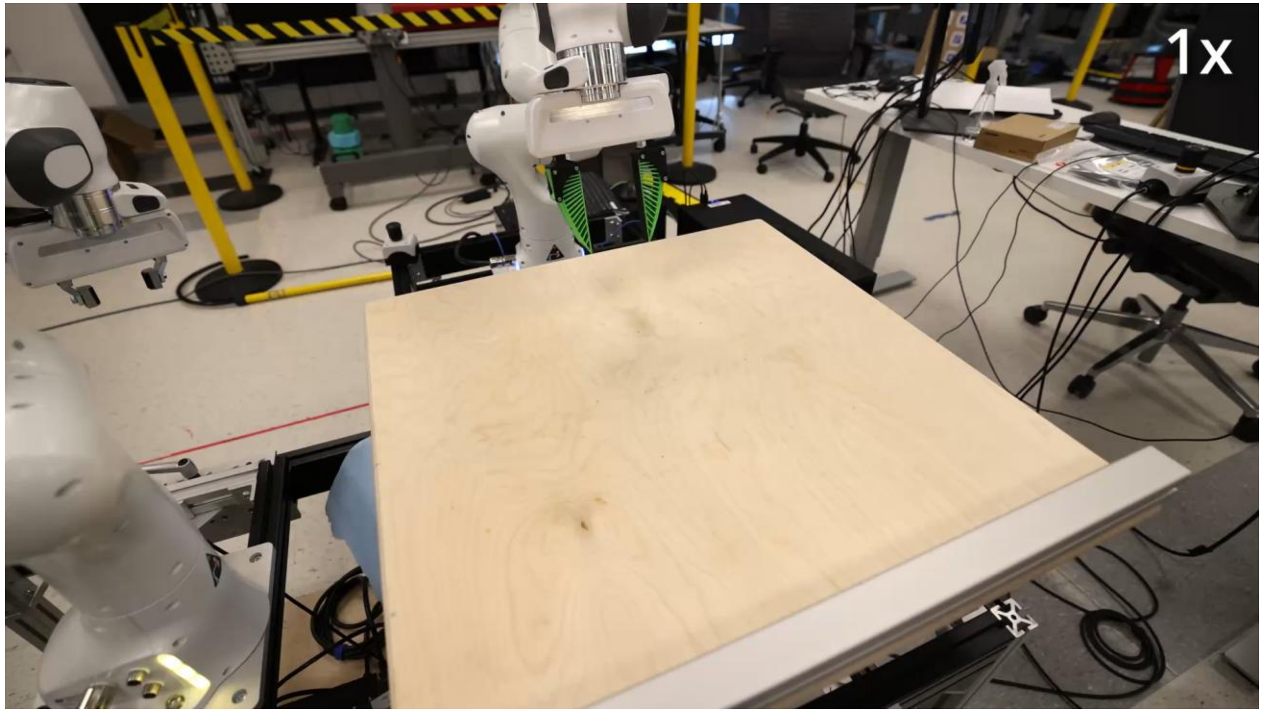


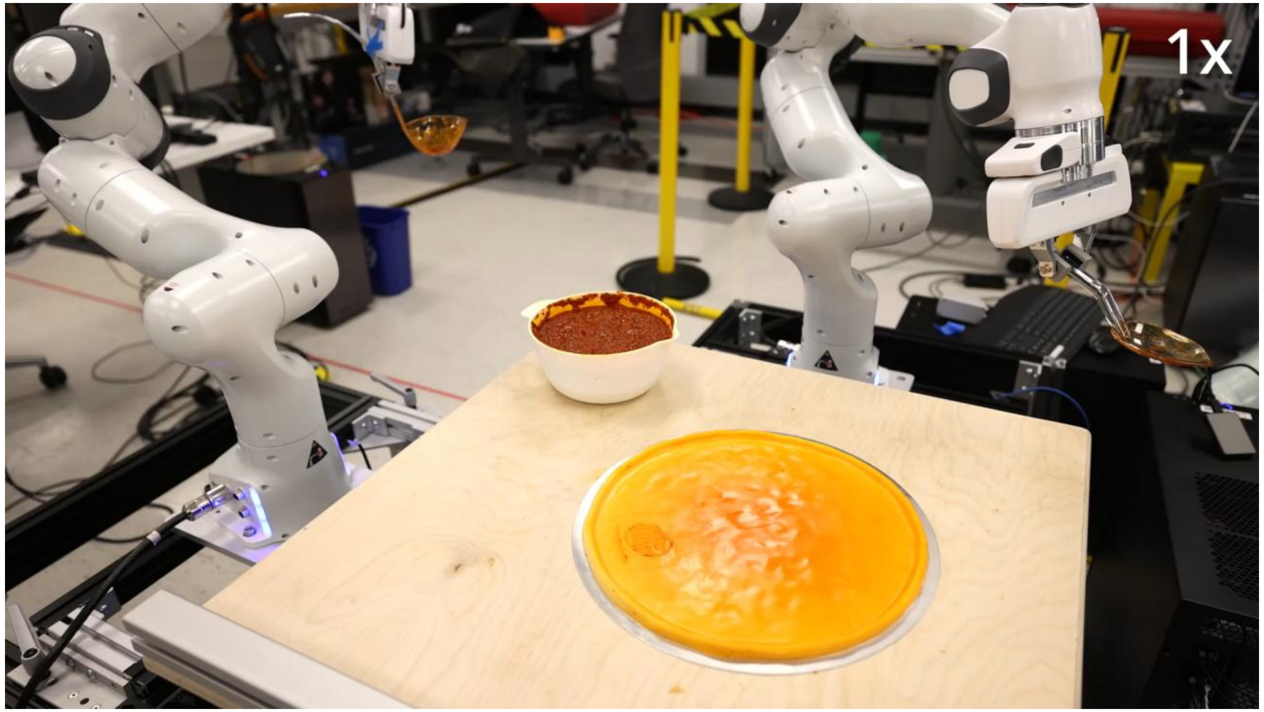
Diffusion Policy

- Better modeling of multi-modal demonstrations
- Scalability to high-dimensional action spaces



Diffusion Policy





https://diffusion-policy.cs.columbia.edu/

Summary

- Imitation learning
  - Behavior cloning
  - Dagger
  - ACT
  - Diffusion policy

Further Reading

- Dagger: A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning https://arxiv.org/abs/1011.0686

- ACT: Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware https://arxiv.org/abs/2304.13705

- Diffusion Policy: Visuomotor Policy Learning via Action Diffusion https://arxiv.org/abs/2303.04137v4