清華大學 Tsinghua University

Deep Reinforcement Learning

Lecture 9: Diffusion Models for RL/IL

Huazhe Xu

Tsinghua University

AI This Week

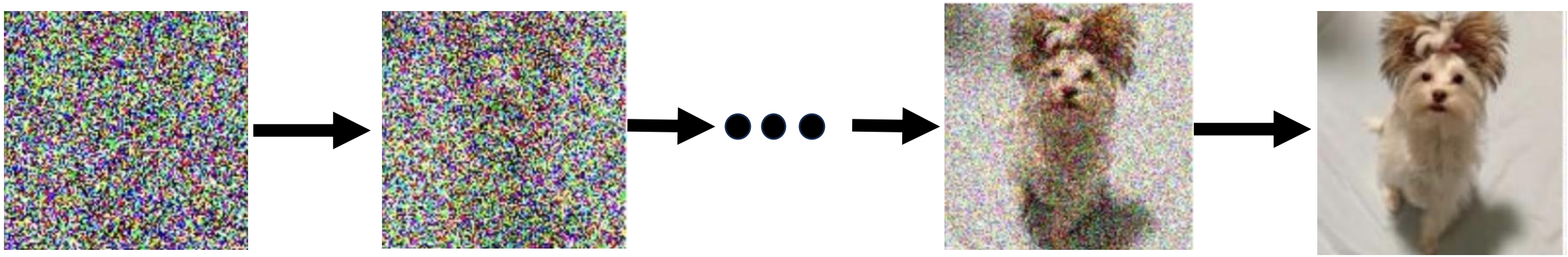
In Lec9

1. Diffusion Models
2. Diffusion in Policy Learning
3. Flow Matching

In Lec9

1. Diffusion Models
2. Diffusion Policy
3. Flow Matching

What is diffusion?

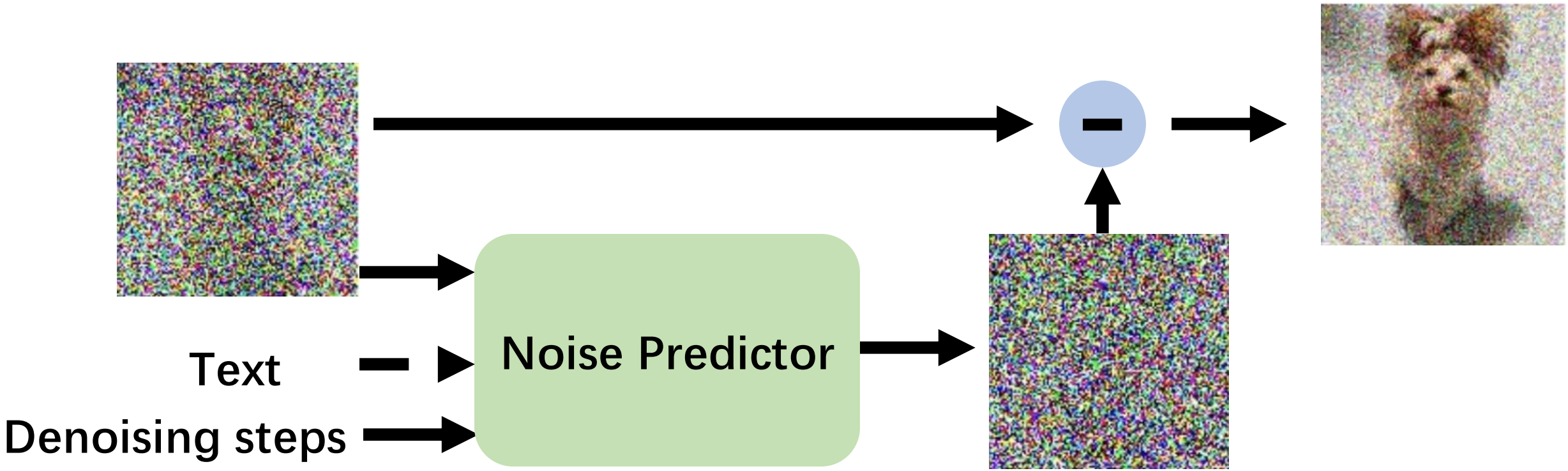


Denoising Denoising Denoising

Reverse process of adding noise to the original signals.

credit: Hung-Yi Lee, Weinan Zhang, Yang Song

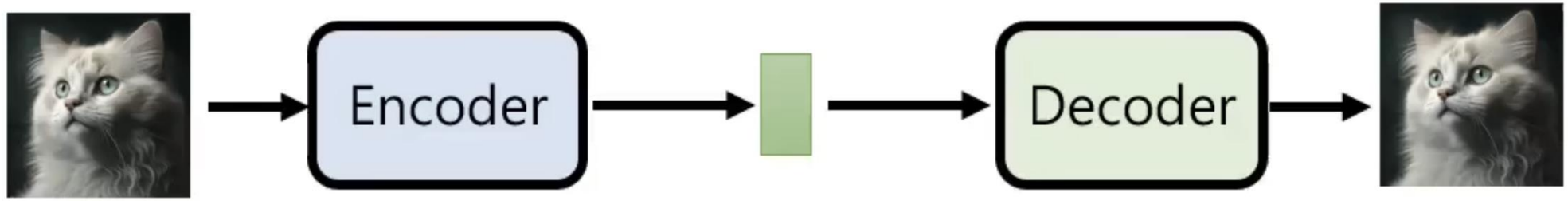
Denoising Module



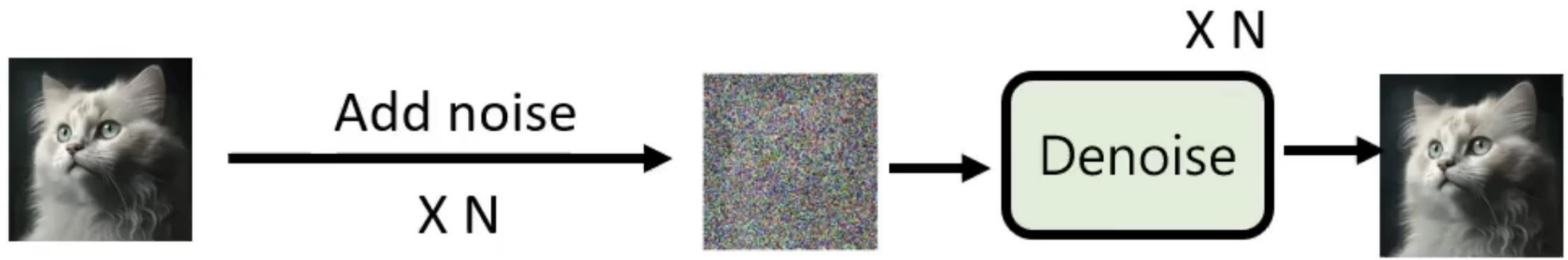
Predict the noise you added!

DDPM: https://arxiv.org/abs/2006.11239

VAE



Diffusion



Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ Sample a clean image
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ Sample a noise
5: Take gradient descent step on

$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$

6: **until** converged

Combine image with noise

Noise predictor

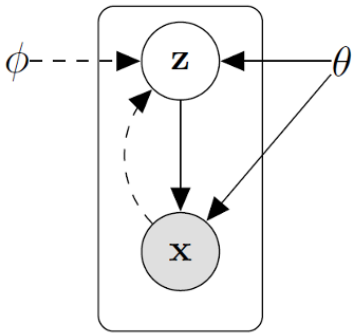
$\bar{\alpha}_1, \bar{\alpha}_2, \dots \bar{\alpha}_T$ → smaller

Algorithm 2 Sampling

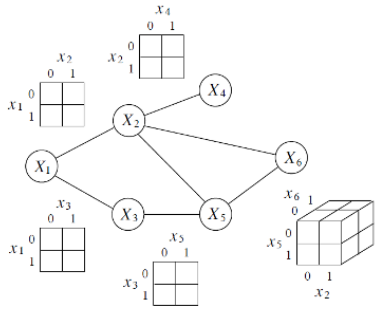
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ Sample a noise
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$ Sample another noise
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

https://lilianweng.github.io/posts/2021-07-11-diffusion-models/ for derivation

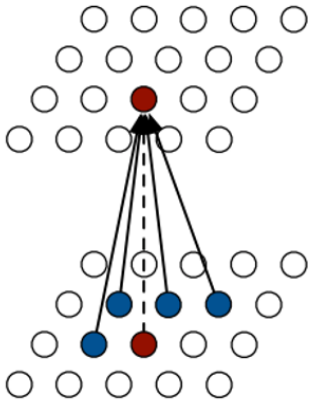
Generative Modeling by Estimating Gradients of the Data Distribution



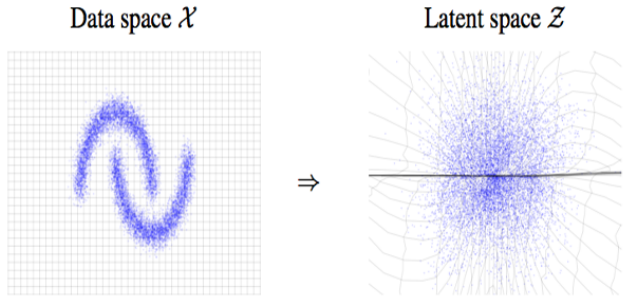
Bayesian networks (e.g., VAEs)



MRF



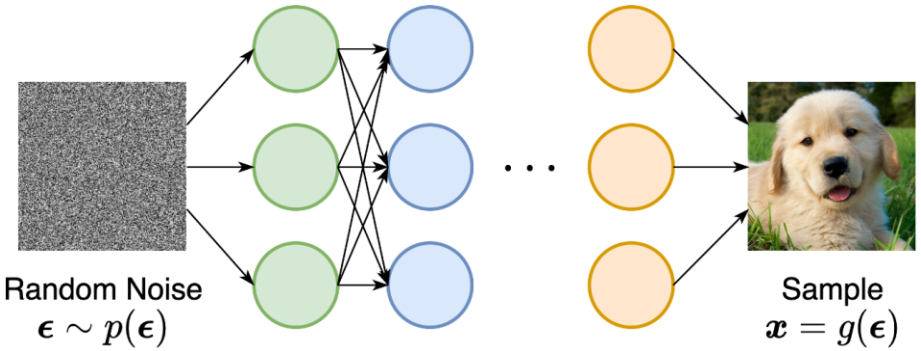
Autoregressive models



Flow models

Likelihood-based Models

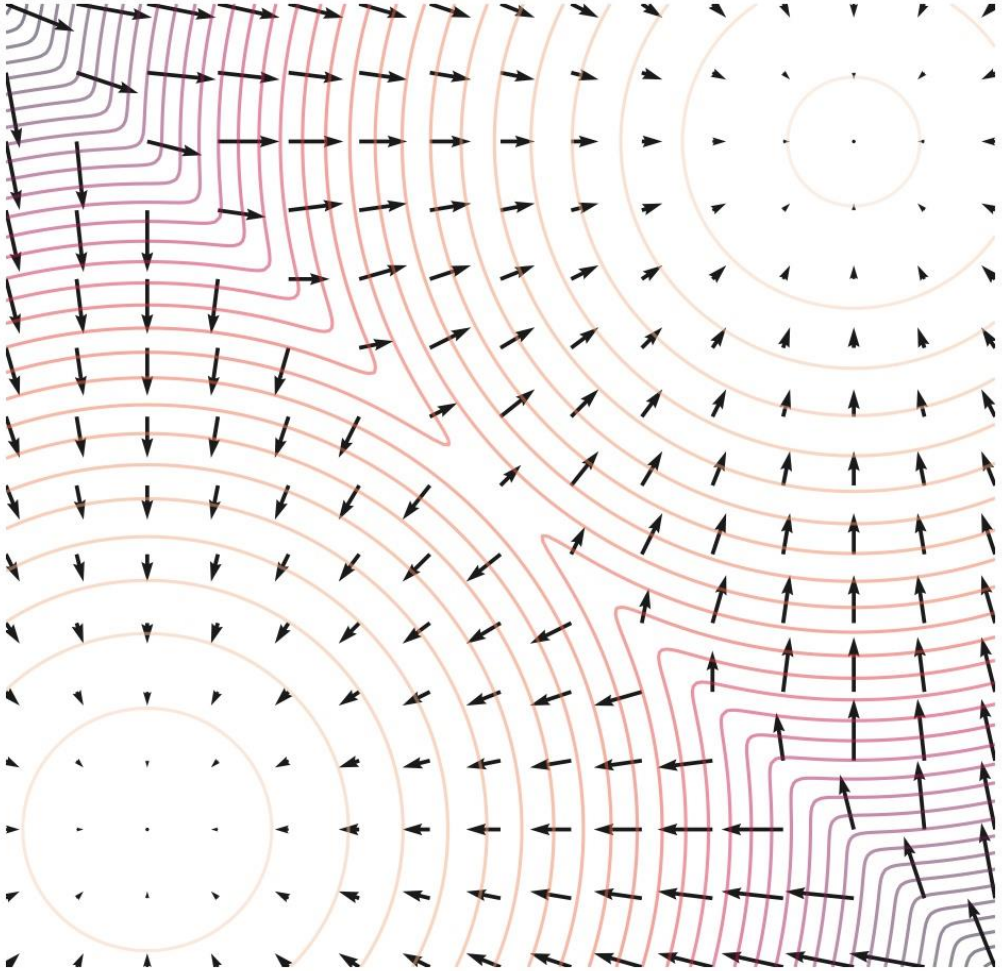
Special constraints



Implicit Generative Models

Unstable

Score-based Models



Contours: Mixture of Gaussian

Vector Field: Score function

Scores are not probabilities. No restrictions needed.

- Give dataset $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \sim p(x)$, generate a new sample
- Max Likelihood approach may have p.d.f $p_\theta(\mathbf{x}) = \frac{e^{-f_\theta(\mathbf{x})}}{Z_\theta}$ Normalizing, Usually intractable
- We can train $\max_\theta \sum_{i=1}^{N} \log p_\theta(\mathbf{x}_i)$
- The score function $\nabla_{\mathbf{x}} \log p(\mathbf{x})$
- We get rid of the Z by using

$$\mathbf{s}_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}) = -\nabla_{\mathbf{x}} f_\theta(\mathbf{x}) - \underbrace{\nabla_{\mathbf{x}} \log Z_\theta}_{=0} = -\nabla_{\mathbf{x}} f_\theta(\mathbf{x}).$$

- Loss

Unknown

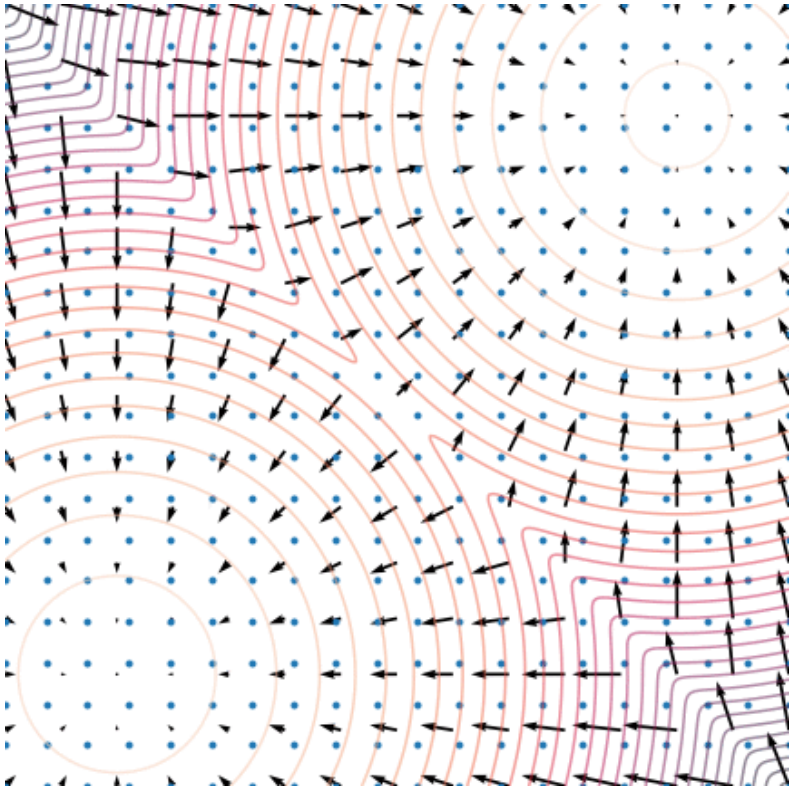
$$\mathbb{E}_{p(\mathbf{x})}\left[\|\nabla_{\mathbf{x}} \log p(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x})\|_2^2\right]$$

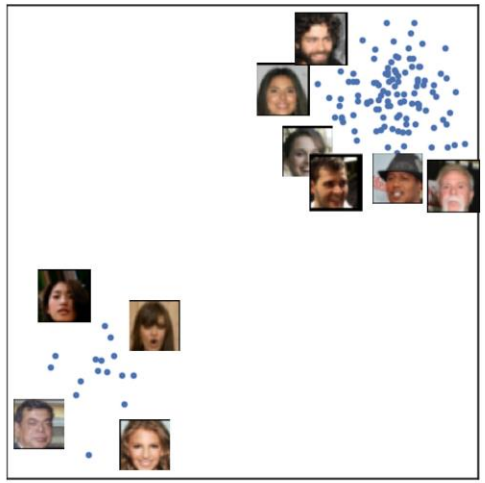
$$J(\boldsymbol{\theta}) \triangleq \mathbb{E}_{p_d}\left[\operatorname{tr}(\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x};\boldsymbol{\theta})) + \frac{1}{2}\|\mathbf{s}_m(\mathbf{x};\boldsymbol{\theta})\|_2^2\right.$$

Estimation of non-normalized statistical models by score matching
A. Hyvarinen.
Journal of Machine Learning Research, Vol 6(Apr), pp. 695--709. 2005.

- Sampling by Langevin Dynamics

$$\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \epsilon \nabla_{\mathbf{x}} \log p(\mathbf{x}) + \sqrt{2\epsilon}\mathbf{z}_i, \quad i = 0, 1, \cdots, K,$$

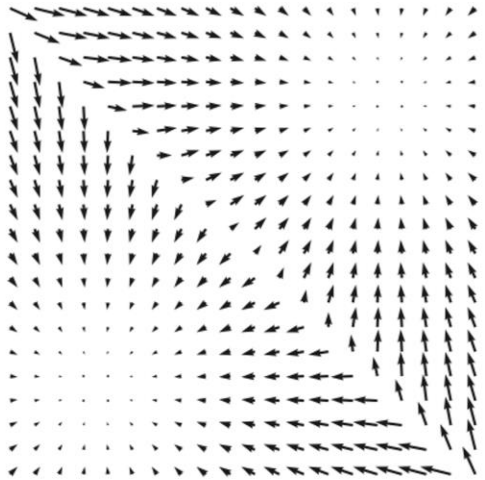




Data samples

$$\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p(\mathbf{x})$$

score matching



Scores

$$\mathbf{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p(\mathbf{x})$$

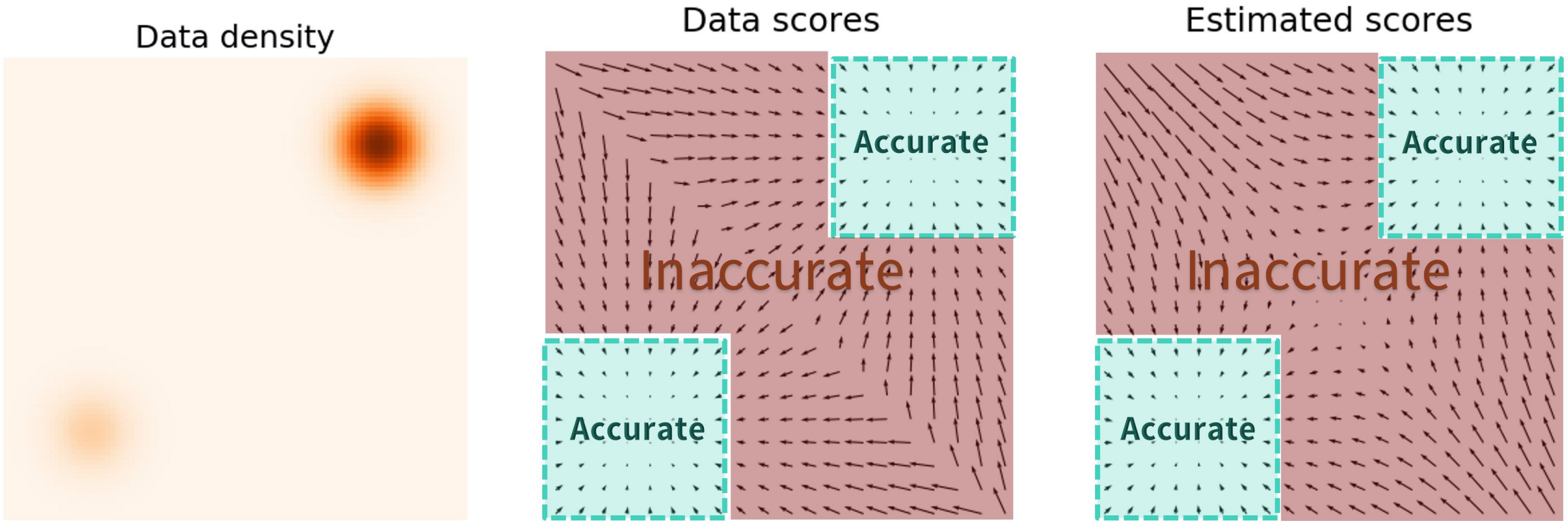
Langevin dynamics

New samples

- Inaccurate Estimation

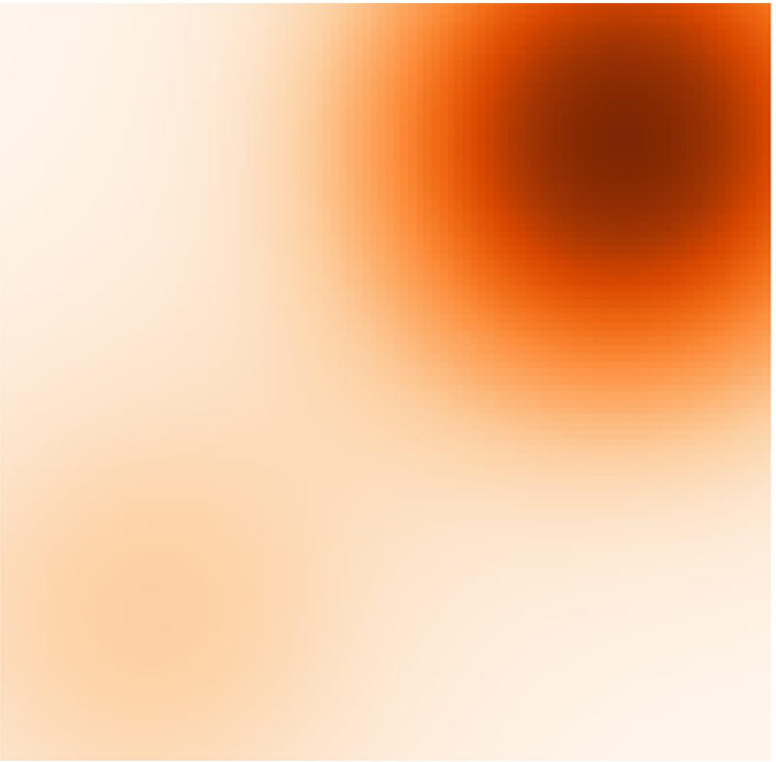
$$\mathbb{E}_{p(\mathbf{x})}\left[\|\nabla_{\mathbf{x}}\log p(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x})\|_2^2\right] = \int p(\mathbf{x})\|\nabla_{\mathbf{x}}\log p(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x})\|_2^2\,\mathrm{d}\mathbf{x}$$

Weight



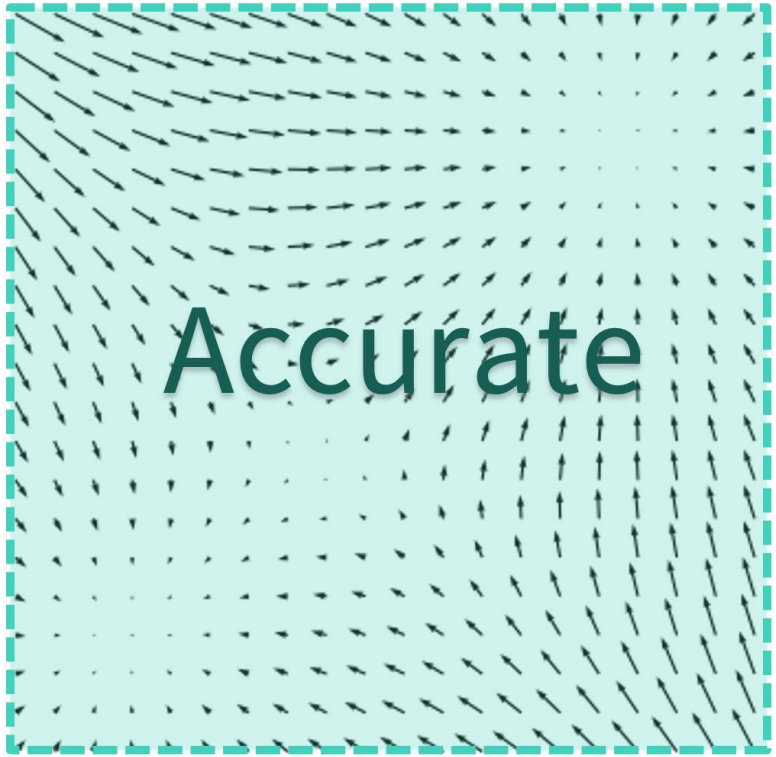
- Perturb the samples

Perturbed density



Perturbed scores

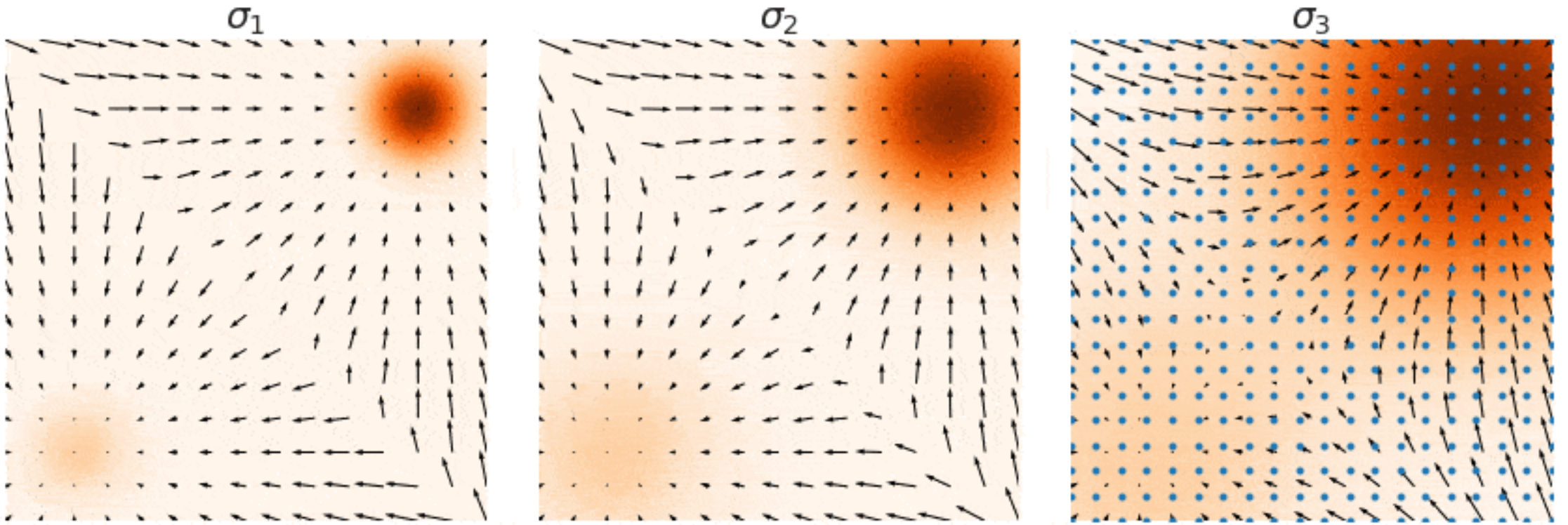
Estimated scores

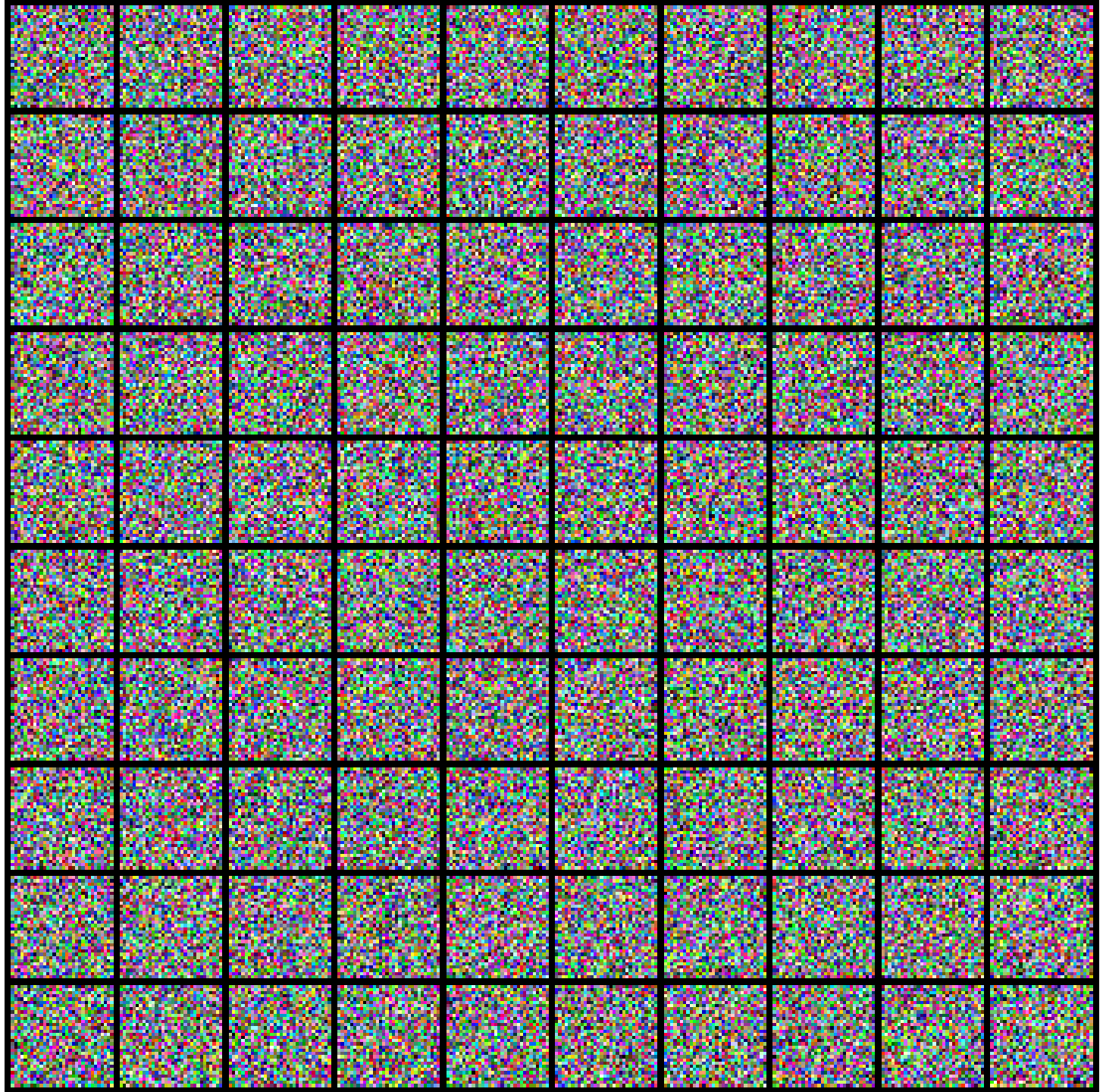


Bigger noise: better scores, worse data dist

$$\sigma_1 < \sigma_2 < \cdots < \sigma_L$$

$$\sigma_1 < \sigma_2 < \cdots < \sigma_L$$

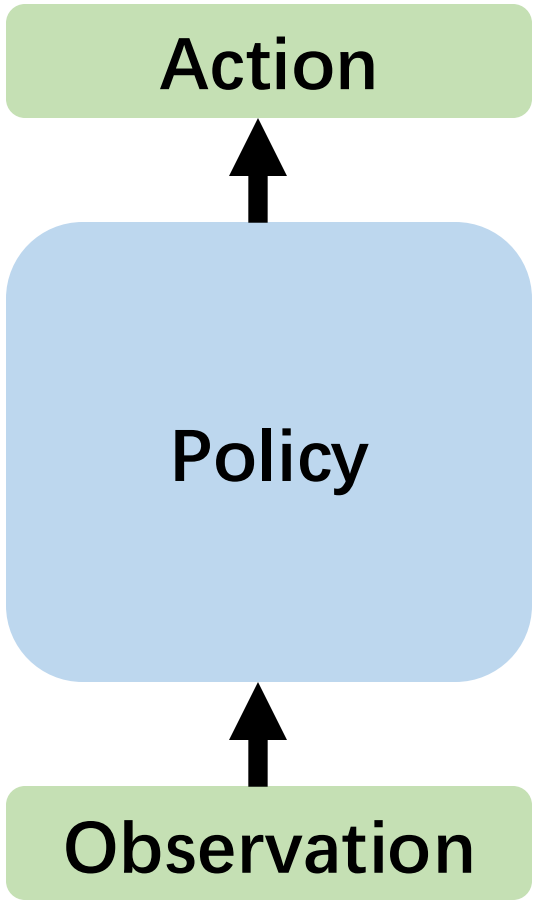




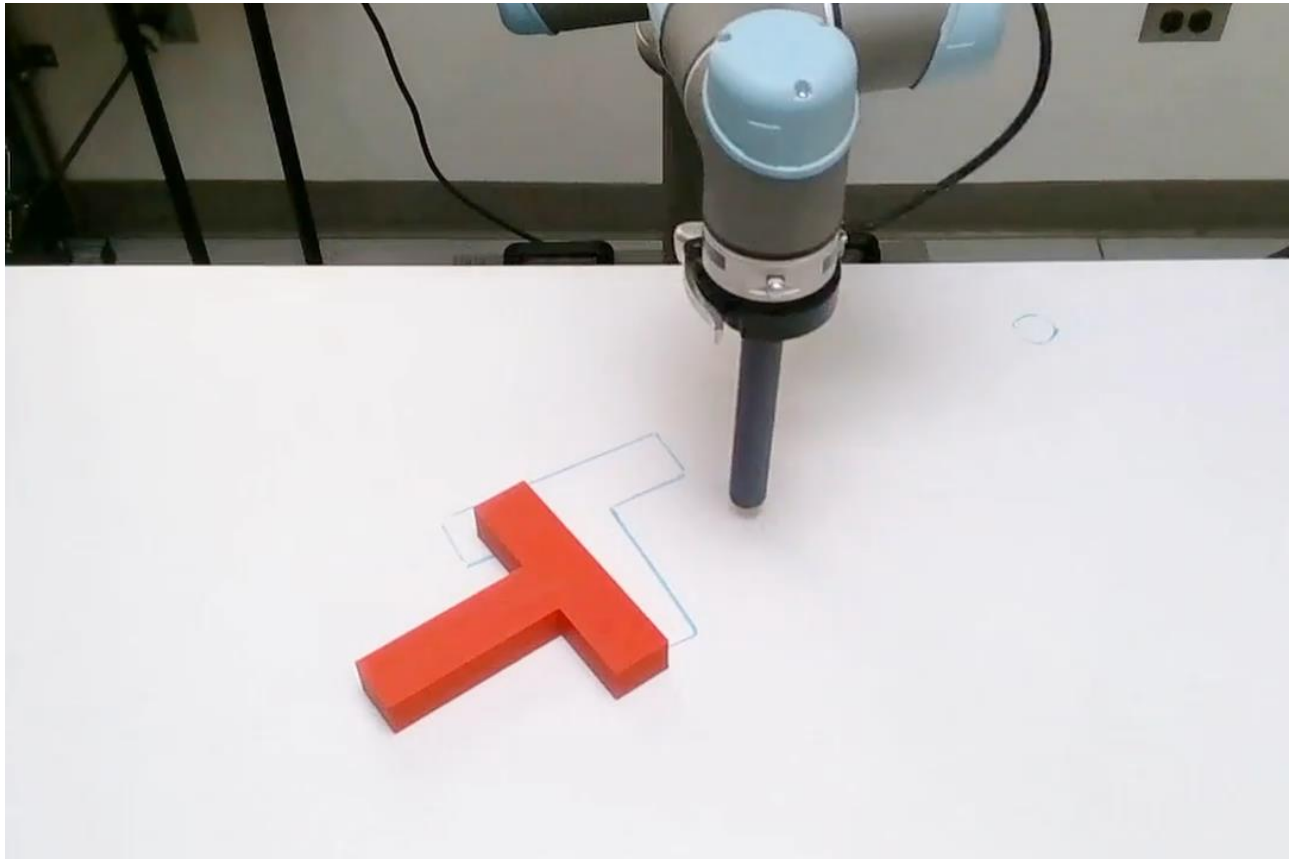
In Lec9

1. Diffusion Models
2. Diffusion in Policy Learning
3. Flow Matching

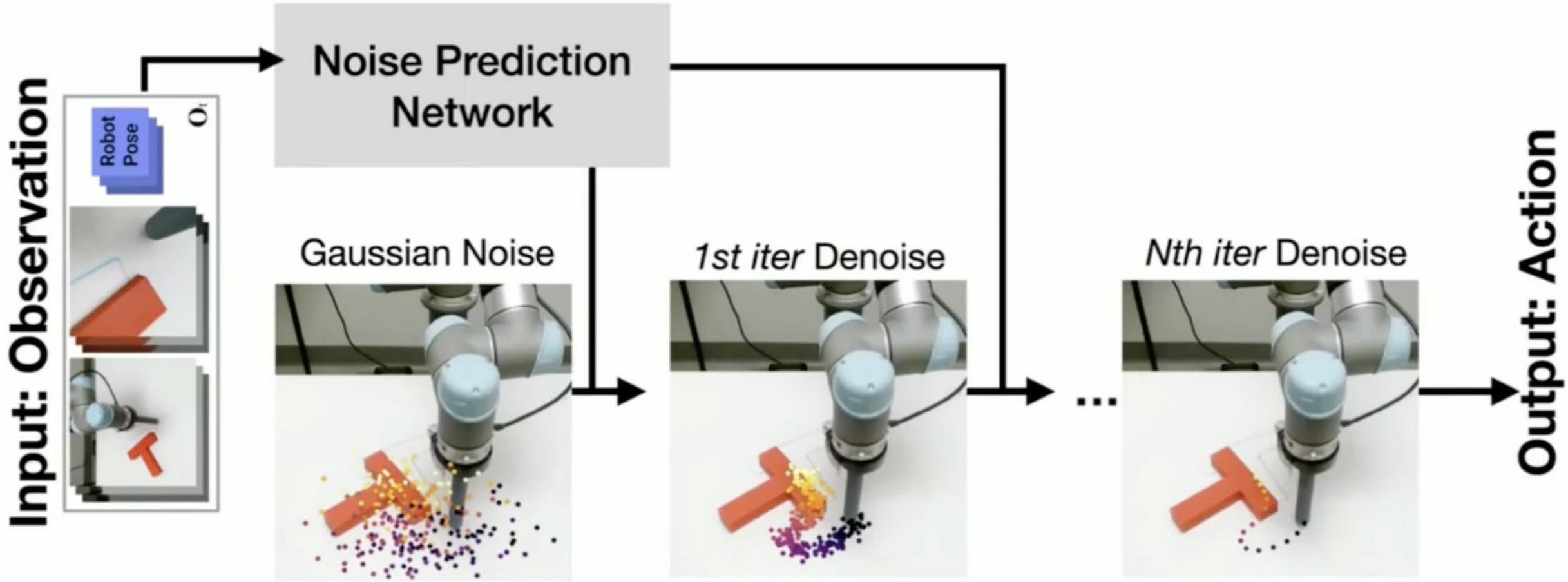
Diffusion Policy



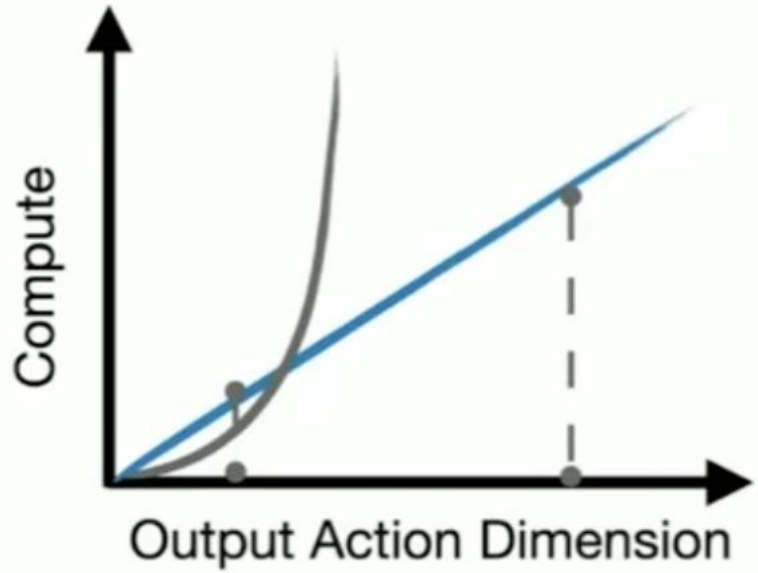
Needs strong expressiveness



Credit: Cheng Chi & his talk @Lerobot


Input: Observation
Robot Pose
O_t
Noise Prediction Network
Gaussian Noise
1st iter Denoise
Nth iter Denoise
...
Output: Action

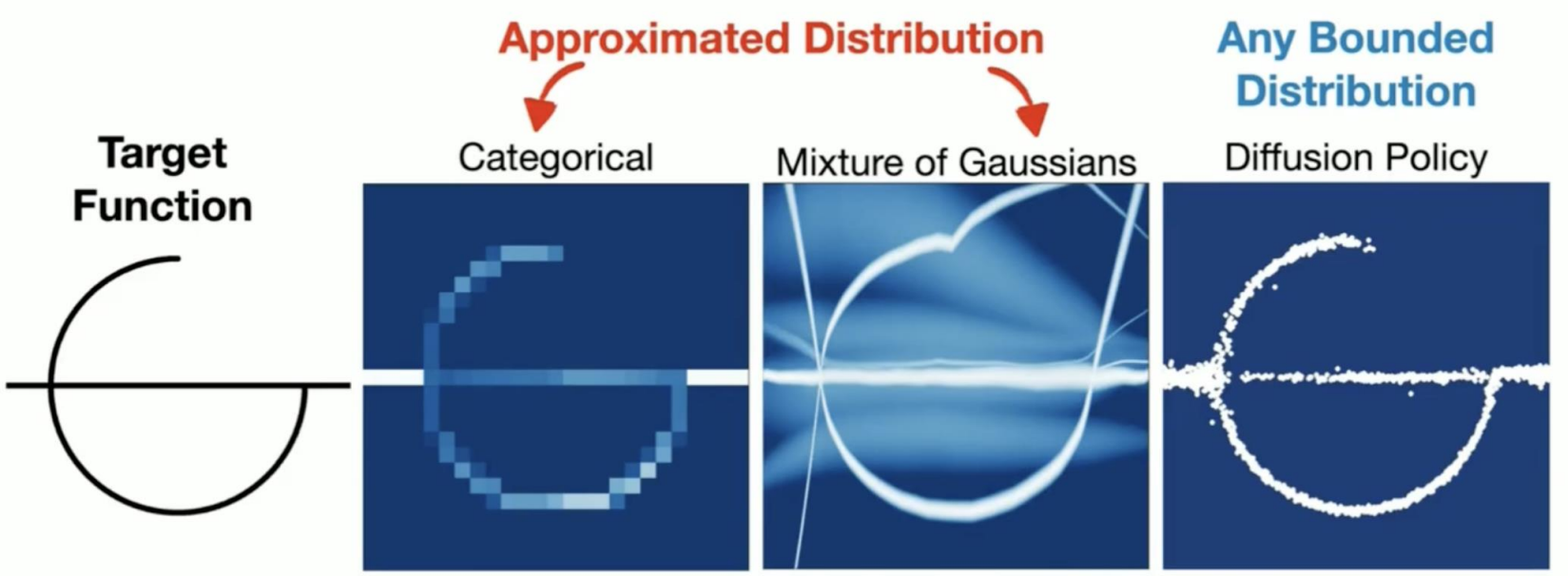
Action Multimodality

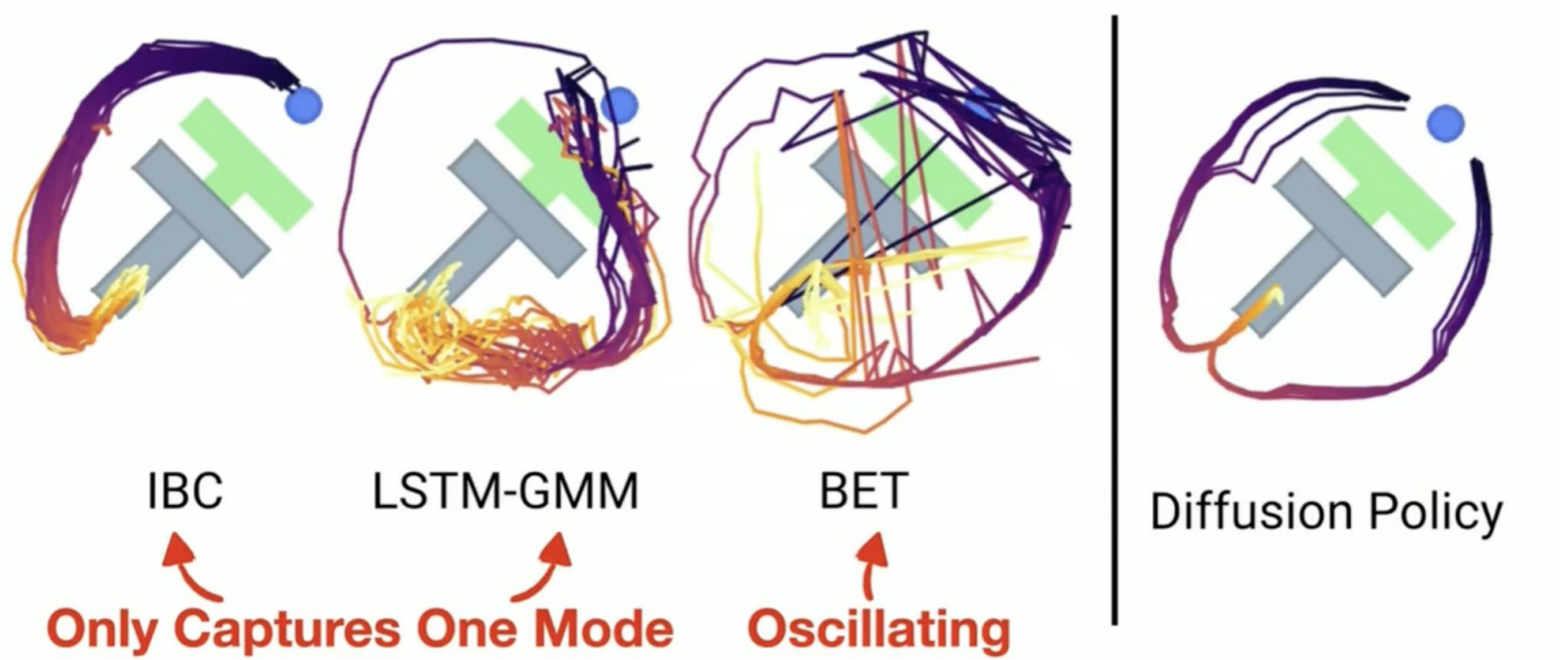


Action Space Scalability

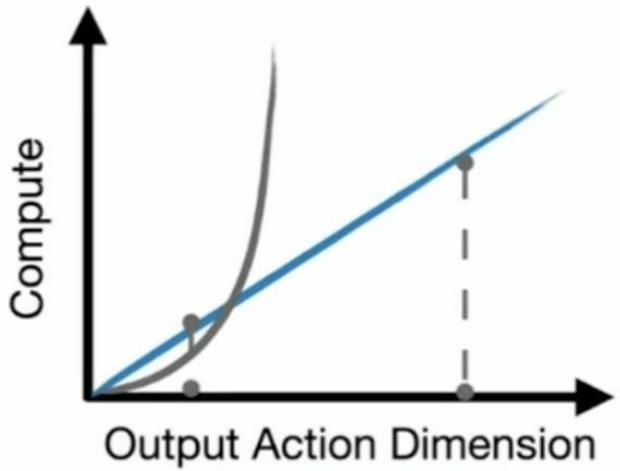
Training Stability

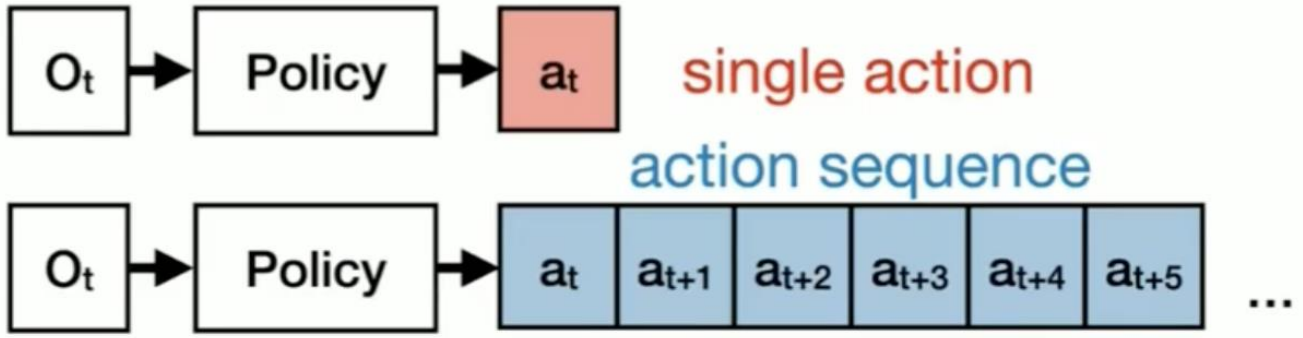
Action Modality

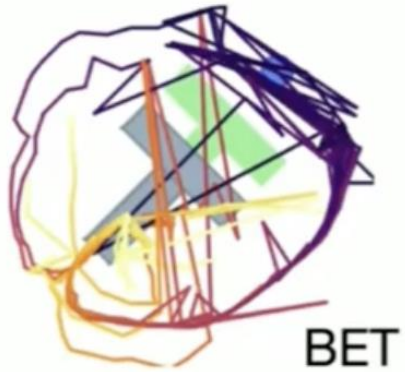




Action Space Scalability







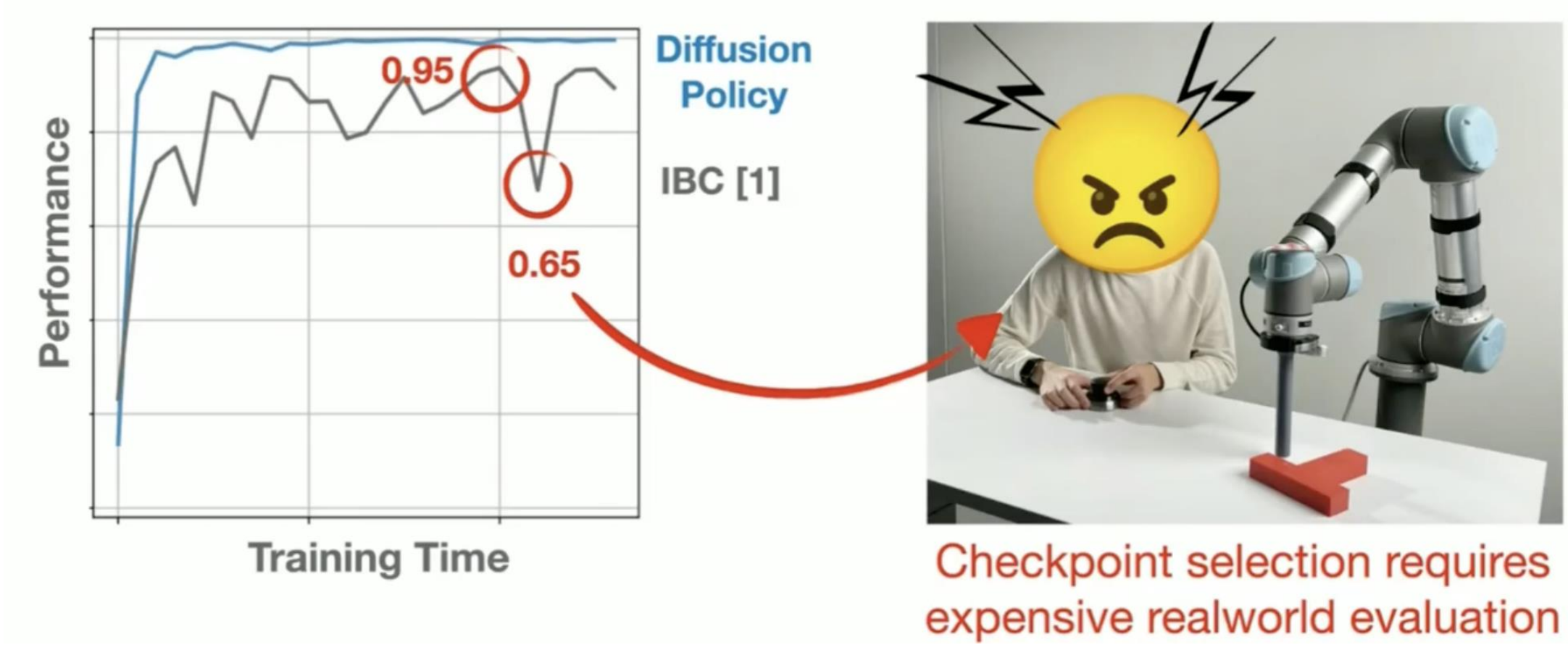
BET [2]

Oscillation between Modes

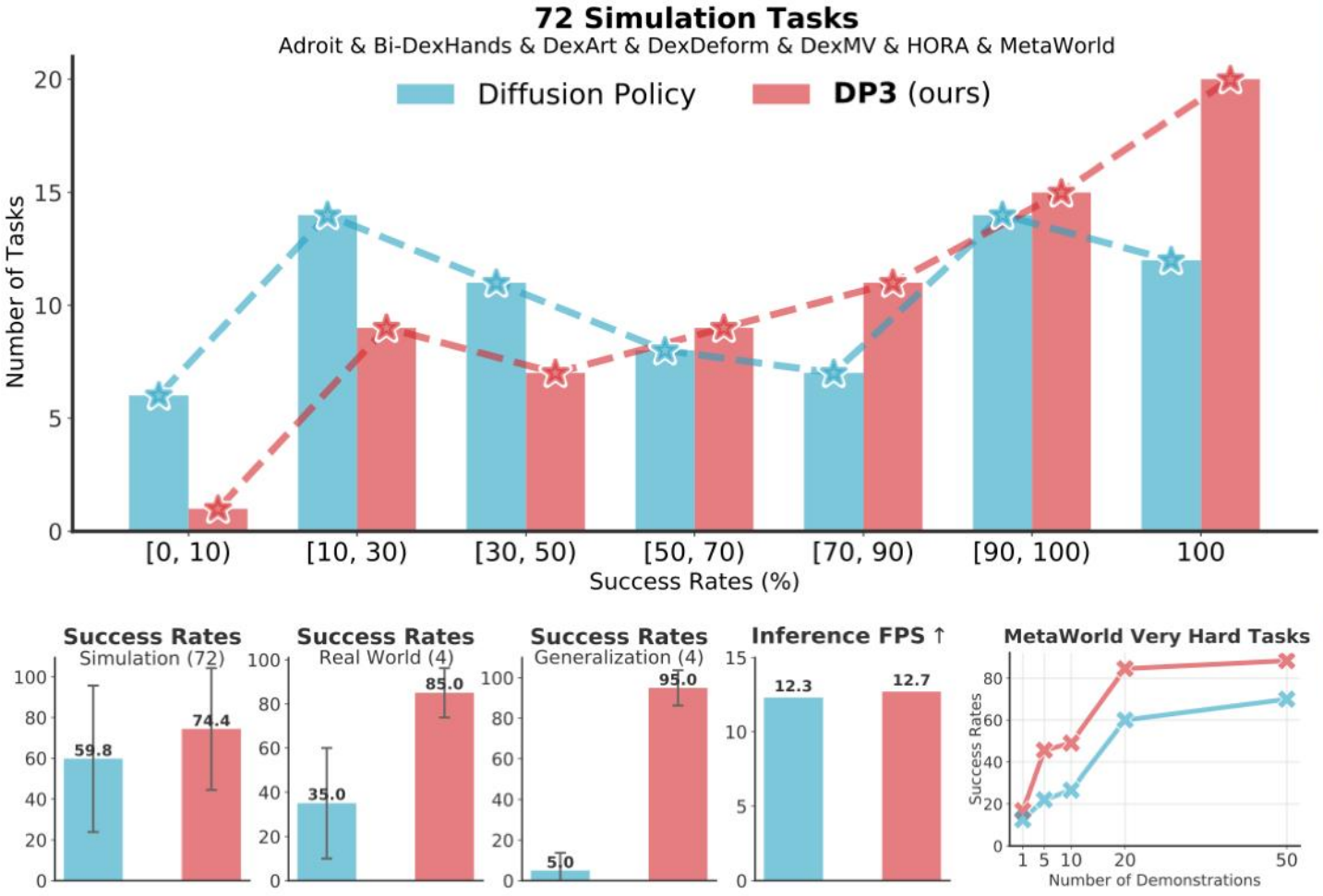
DiffusionPolicy

Temporal Mode Consistency

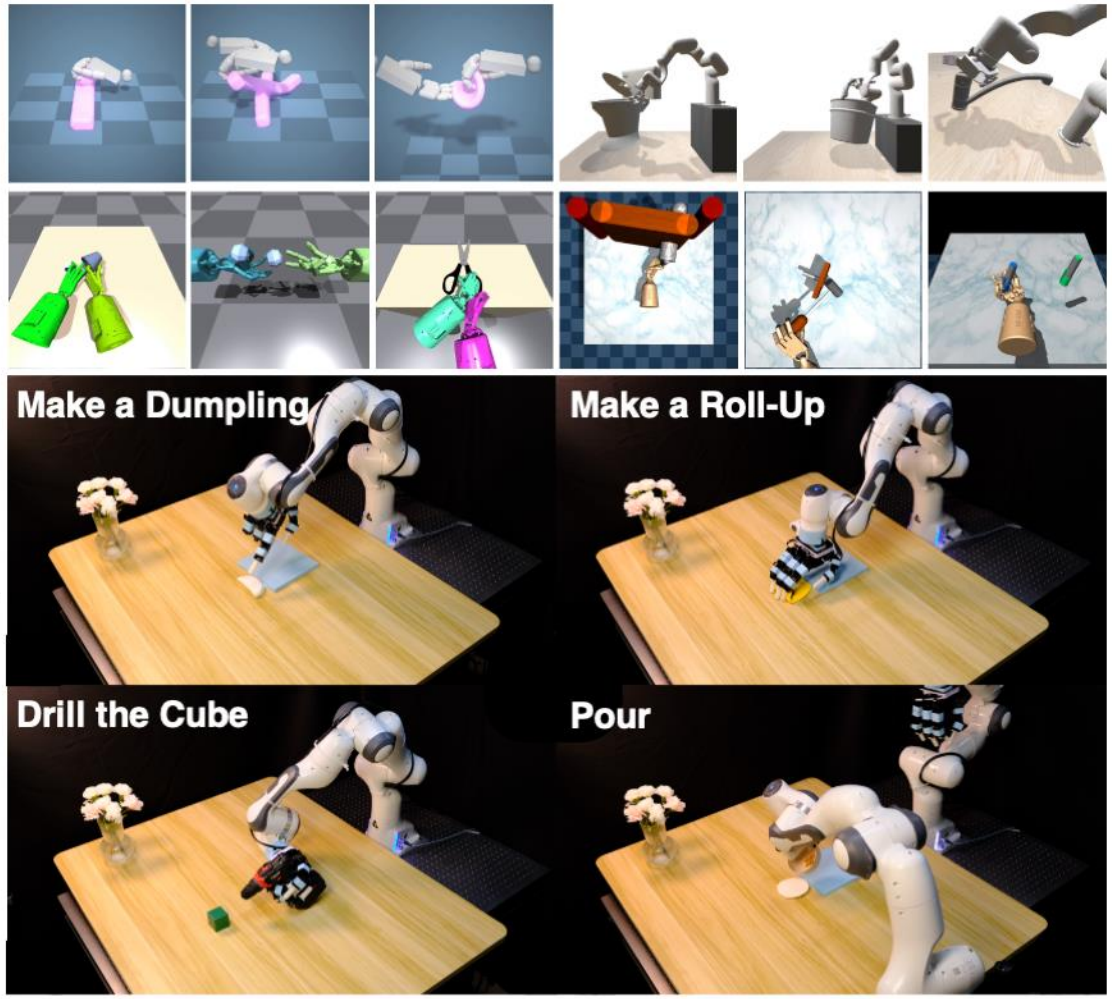
Training Stability



DP3



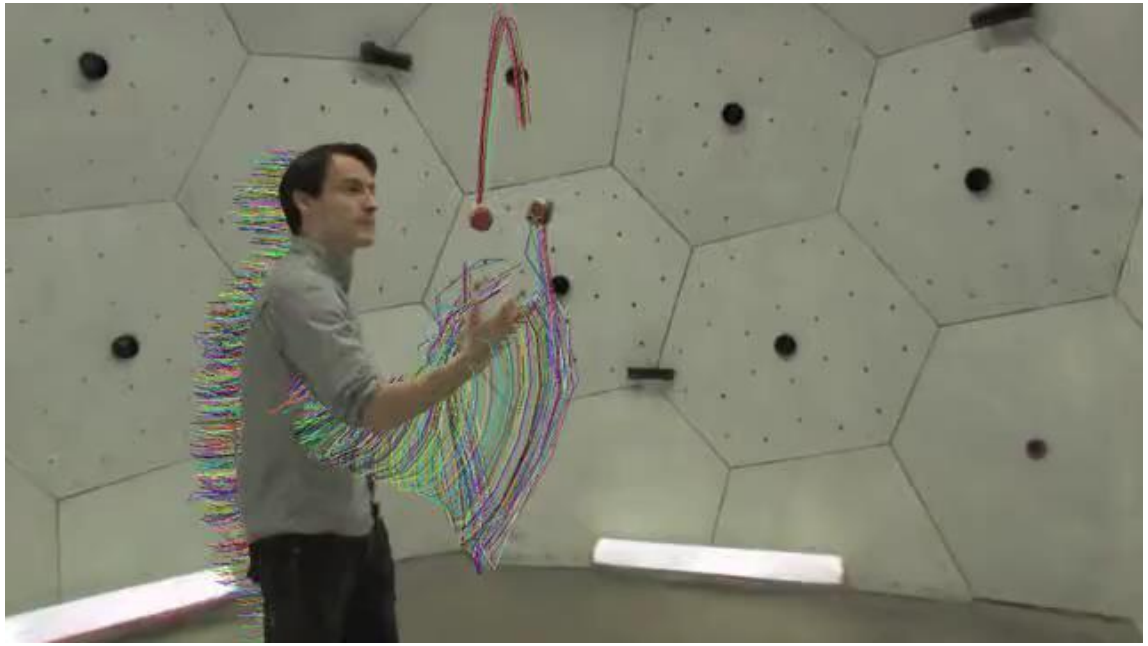
(a) 3D Diffusion Policy (**DP3**) vs. Diffusion Policy: ***Better, Faster, Stronger.***



(b) We evaluate DP3 in diverse simulated and real tasks.

What could be the Missing Point?

The world itself is 3D.



Luiten et al., 2023

3D Diffusion Policy (DP3)

Diffusion Policy + *Very* Simple 3D Representations

Perception: Compact 3D Representations from Point Clouds

(a) Point Cloud Processing

Single-view Point Cloud

Why Simple 3D Representations?

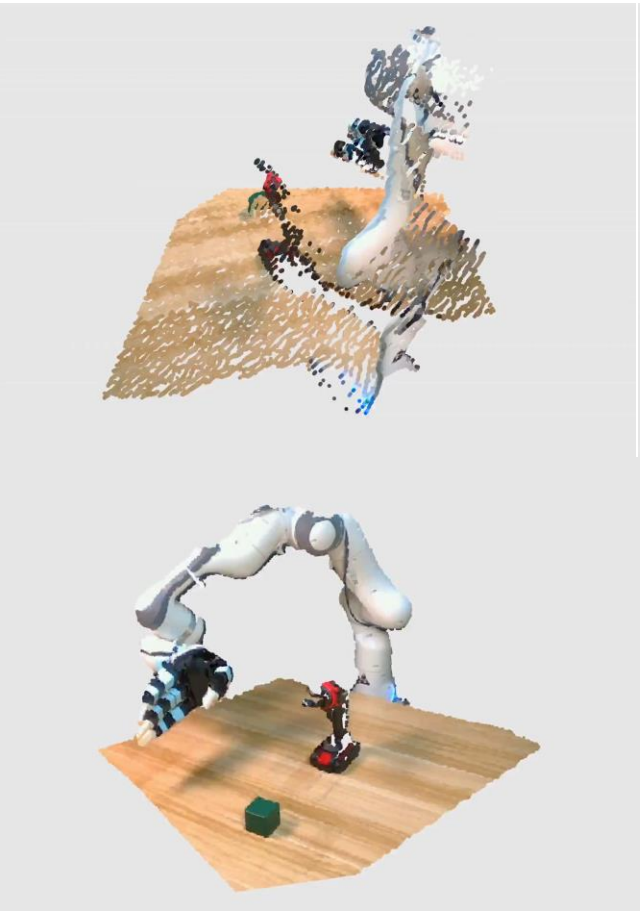
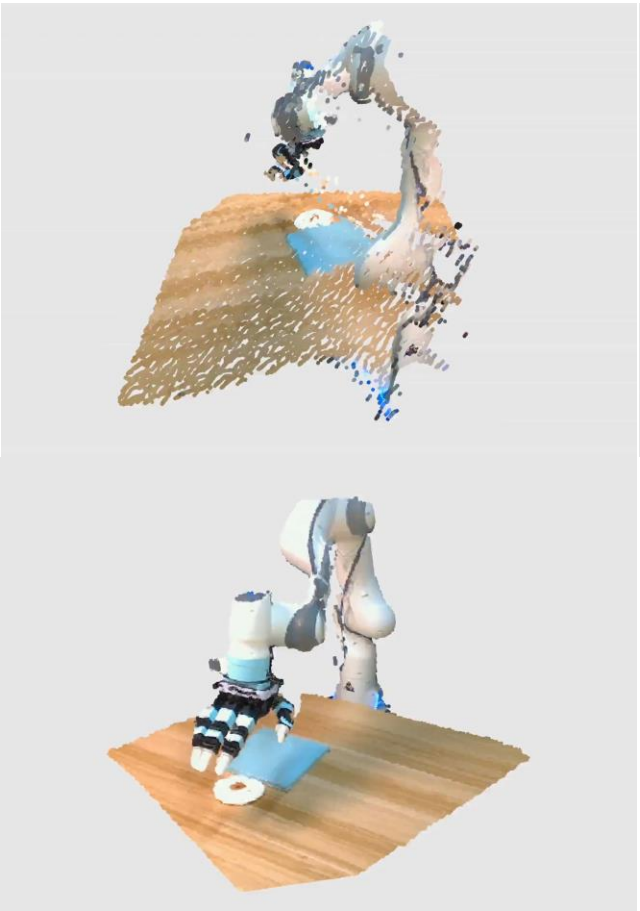
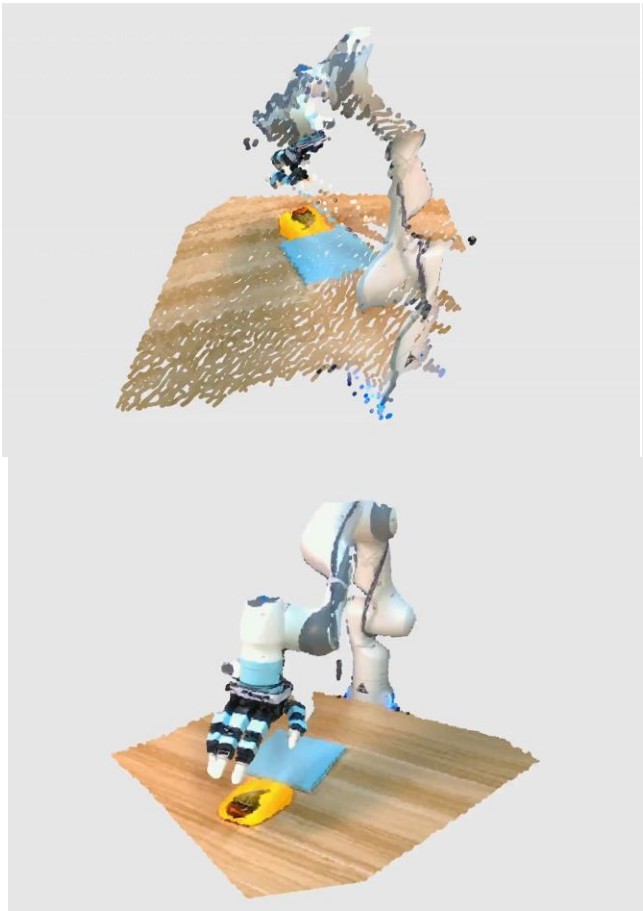
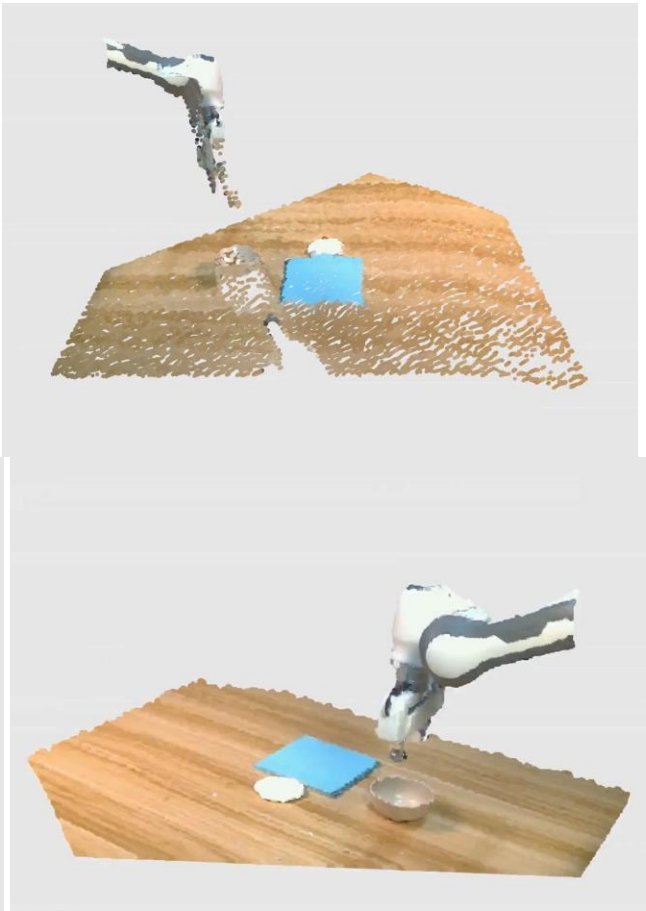
EXP1: Point Cloud ≈ Oracle State > Voxel/Depth/Image

Repr.	H	D	P	A	DA	SP	Average
Oracle State	99±2	61±2	**44±3**	94±1	**72±7**	91±8	76.8
Point cloud	**100±0**	**62±4**	43±6	**99±1**	69±4	**97±4**	**78.3**
Image	48±17	50±5	25±4	15±1	43±7	63±3	40.7
Depth	39±15	49±1	12±3	15±4	15±2	62±3	32.0
RGB-D	57±14	47±5	14±2	15±3	14±1	61±3	34.7
Voxel	54±5	33±3	18±2	10±2	17±1	62±6	32.3

EXP2: DP3 Encoder >> Other Point Encoders (Why?)

Encoders	H	D	P	A	DA	SP	Average
DP3 Encoder	**100±0**	**62±4**	**43±6**	**99±1**	**69±4**	**97±4**	**78.3**
PointNet	46±8	34±8	14±4	0±0	0±0	0±0	15.7
PointNet++	0±0	0±0	13±3	0±0	0±0	0±0	2.2
PointNeXt	0±0	0±0	14±3	0±0	0±0	0±0	2.3
Point Transformer	0±0	0±0	6±5	0±0	0±0	0±0	1.0
PointNet++ (pre-trained)	5±9	19±12	17±6	0±0	0±0	0±0	6.8
PointNeXt (pre-trained)	0±0	36±13	17±6	0±0	0±0	0±0	8.8

Evaluate DP3 on Challenging Real-World Tasks

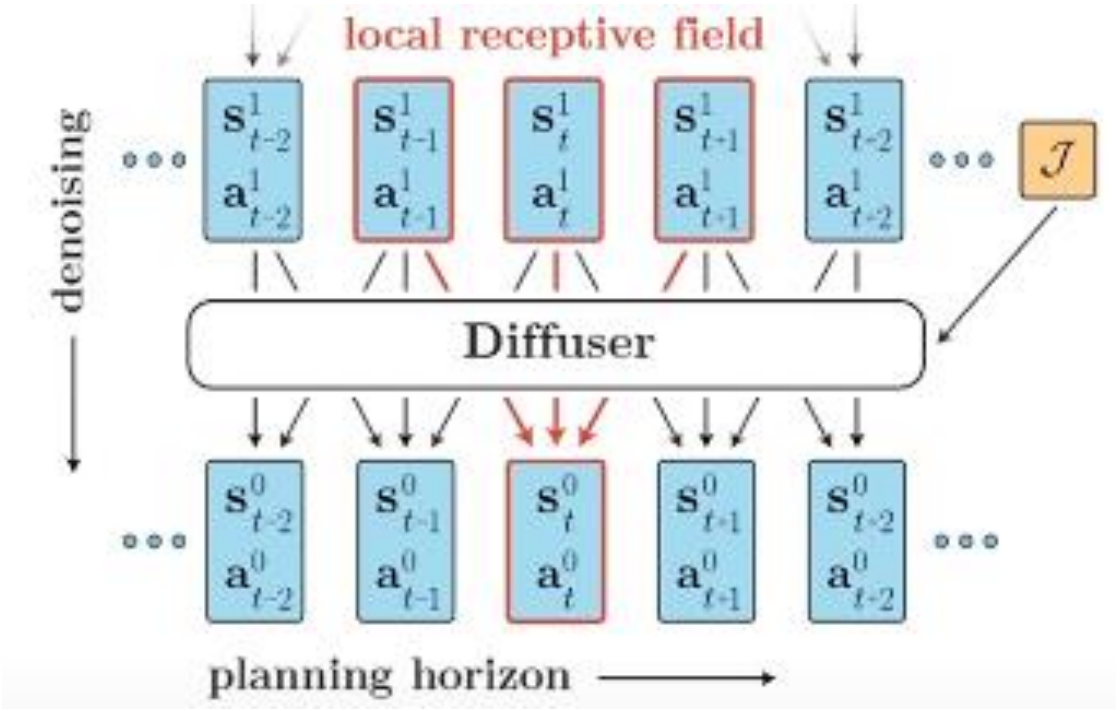
Drill	Dumpling	Roll	Pour
			

Diffusion Planner

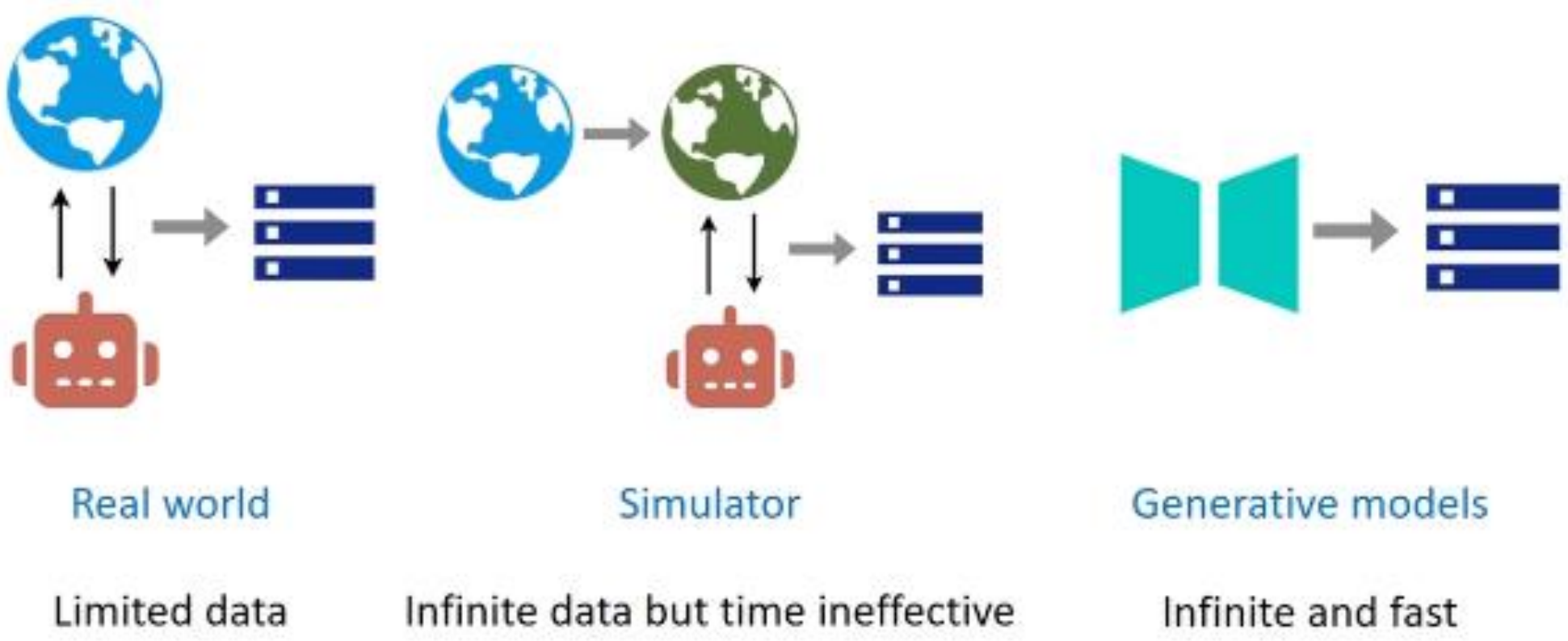
- Diffusion is just a data generator. So basically you have a lot of choices.
- Given s predict s’
- Given s predict a
- Given s predict s, a
- Given s predict s, a, r
- Given s predict a series of the previous variables

Why is diffusion better at all?

- Key insight:
  - Action chunking: regularize locally, propagate globally
  - If we have an end goal, things would be much easier!



Diffusion for Data Generation!



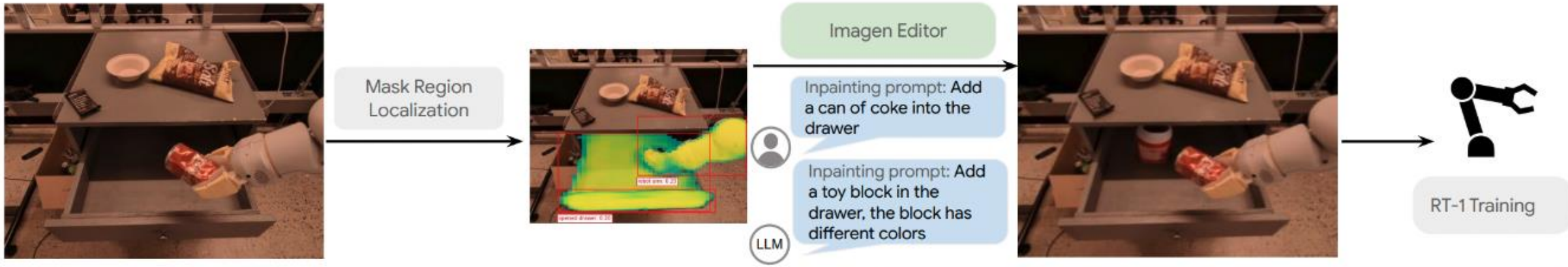
Credit: Weinan Zhang

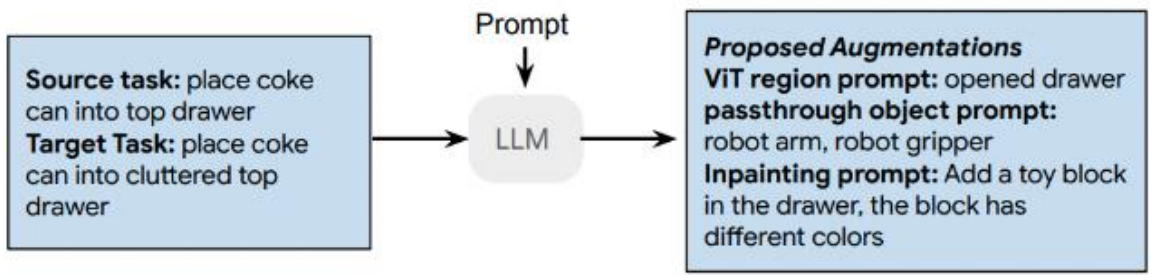
Scaling Robot Learning with Semantically Imagined Experience

Tianhe Yu Ted Xiao Austin Stone Jonathan Tompson
Anthony Brohan Su Wang Jaspiar Singh Clayton Tan Dee M
Jodilyn Peralta Brian Ichter Karol Hausman Fei Xia

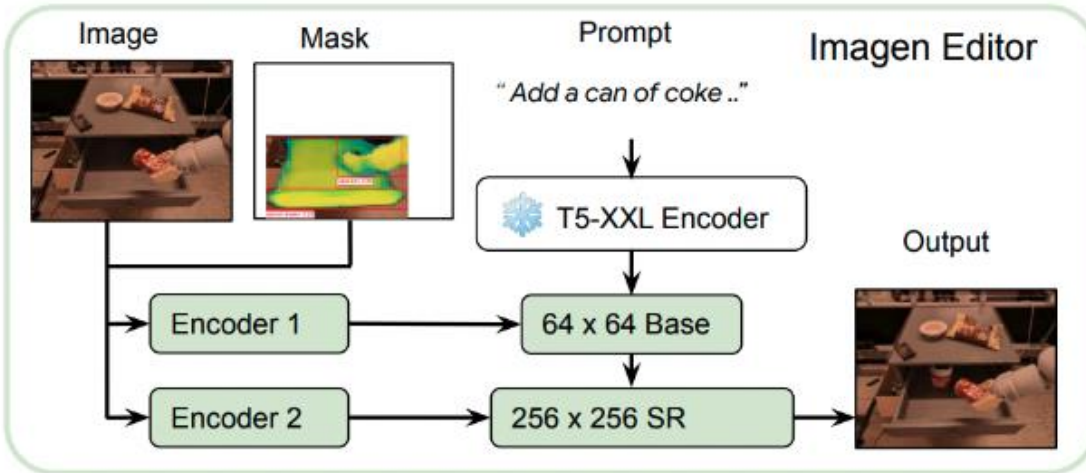
13 robots,17 months,
to collect 130k demonstrations

+ Generative models?





LLM-assisted Augmentation Prompt Generation




Robotics @ Google

ROSIE Augmented Data

Real-world Rollout

robot putting objects in drawer → *robot putting objects in sink*

In Lec9

1. Diffusion Models
2. Diffusion in Policy Learning
3. Flow Matching

π_0: Our First Generalist Policy

Published October 31, 2024
Email research@physicalintelligence.company
Paper π_0.pdf

VLMs effectively transfer semantic knowledge from the web, but they are trained to output only discrete language tokens. Dexterous robot manipulation requires π_0 to output motor commands at a high frequency, up to 50 times per second. To provide this level of dexterity, we developed a novel method to augment pre-trained VLMs with continuous action outputs via flow matching, a variant of diffusion models. Starting from diverse robot data and a VLM pre-trained on Internet-scale data, we train our vision-language-action flow matching model, which we can then post-train on high-quality robot data to solve a range of downstream tasks.

Imagine a ball in the 3D space

- Starting position at x_0
- Speed $\boldsymbol{u}_t(\boldsymbol{x})$
-

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{u}_t(\boldsymbol{x}), \quad \boldsymbol{\phi}_0(\boldsymbol{x}) = \boldsymbol{x}$$

http://xyfjason.top/blog-main/2024/06/22/Flow-Matching/

Now imagine a distribution of balls

- There initial distribution p_0 at $t = 0$.
- There final distribution p_1 at $t = 1$.
- If p_0 is very easy to sample, and p_1 is the target distribution.
- Then everything is solved!
- **Continuous Normalizing Flow (CNF)**

Probability Path

$$p : [0,1] \times \mathbb{R}^d \to \mathbb{R}_{>0}$$

- $p_t(x)$ is the probability density at time t and position x.
- $t \mapsto p_t$ is the path
- $\int p_t(\mathbf{x})\mathrm{d}\mathbf{x} = 1$ It is continuous. It is normalized.

Continuity equation

- $\mathbf{u} : [0, 1] \times \mathbb{R}^d \to \mathbb{R}^d$ velocity field.
- $\mathbf{u}_t(\mathbf{x})$ is the velocity at time t and position x.
- The continuity equation:

$$\frac{\partial}{\partial t} p_t(\mathbf{x}) + \nabla \cdot (p_t(\mathbf{x}) \mathbf{u}_t(\mathbf{x})) = 0$$

- Intuition:
  - First term: particles amount changes
  - Second term: particles flow in/out
- Question:
  - Is p a one-to-one mapping to u?
  - Considering adding a Solenoidal Vector Field to u?

Flow

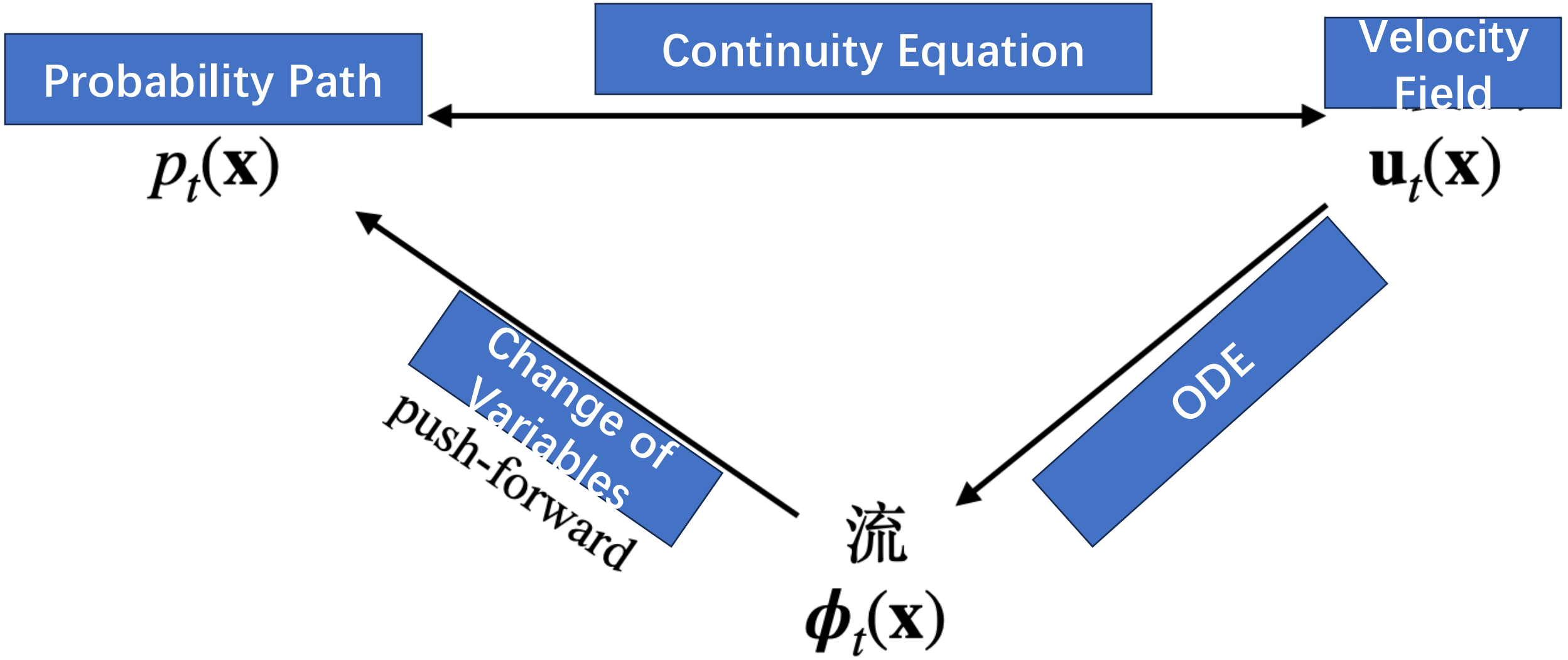
- The trajectory of particles following u is called flow $\boldsymbol{\phi} : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$
- $$\frac{\partial}{\partial t}\boldsymbol{\phi}_t(\mathbf{x}) = \mathbf{u}_t\left(\boldsymbol{\phi}_t(\mathbf{x})\right)$$
$$\boldsymbol{\phi}_0(\mathbf{x}) = \mathbf{x}$$
- you can do the integral
$$\phi_t(\mathbf{x}) - \mathbf{x} = \int_0^t \mathbf{u}_s\left(\phi_s(\mathbf{x})\right)\mathrm{d}s$$

Change of variables formula

- $\boldsymbol{\phi}_t^{-1}(\mathbf{x})$ is the starting position of the particles at position x and time t.
- Following change of variables formula:

$$p_t(\mathbf{x}) = p_0\left(\boldsymbol{\phi}_t^{-1}(\mathbf{x})\right)\left|\det\left[\frac{\partial \boldsymbol{\phi}_t^{-1}}{\partial \mathbf{x}}(\mathbf{x})\right]\right|$$

- Push-forward equation


Probability Path
Continuity Equation
Velocity Field
$p_t(\mathbf{x})$
$\mathbf{u}_t(\mathbf{x})$
Change of Variables
push-forward
ODE
流
$\phi_t(\mathbf{x})$

Flow Matching

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{t, p_t(\mathbf{x})} \left[\left\| \mathbf{v}_t^\theta(\mathbf{x}) - \mathbf{u}_t(\mathbf{x}) \right\|^2 \right]$$

- But $u_t(x)$ is unknown.
- But we might be able to know the velocity given samples.

Conditional Flow Matching

- $p_t(\mathbf{x} \mid \mathbf{x}_1)$ conditional probability path.
- Then we have:

$$p_t(\mathbf{x}) = \int p_t(\mathbf{x} \mid \mathbf{x}_1) q(\mathbf{x}_1) \mathrm{d}\mathbf{x}_1 = \mathbb{E}_{q(\mathbf{x}_1)} \left[p_t(\mathbf{x} \mid \mathbf{x}_1) \right]$$

- $\mathbf{u}_t(\mathbf{x} \mid \mathbf{x}_1)$ conditional velocity field.

$$\mathbf{u}_t(\mathbf{x}) = \int \mathbf{u}_t(\mathbf{x} \mid \mathbf{x}_1) \frac{p_t(\mathbf{x} \mid \mathbf{x}_1) q(\mathbf{x}_1)}{p_t(\mathbf{x})} \mathrm{d}\mathbf{x}_1 = \mathbb{E}_{p(\mathbf{x}_1 \mid \mathbf{x})} \left[\mathbf{u}_t(\mathbf{x} \mid \mathbf{x}_1) \right]$$

$$\begin{aligned}
\frac{\partial p_t(\mathbf{x})}{\partial t} &= \int \left(\frac{\partial}{\partial t} p_t(\mathbf{x} \mid \mathbf{x}_1) \right) q(\mathbf{x}_1) \mathrm{d}\mathbf{x}_1 \\
&= -\int \nabla \cdot \left(p_t(\mathbf{x} \mid \mathbf{x}_1) \mathbf{u}_t(\mathbf{x} \mid \mathbf{x}_1) \right) q(\mathbf{x}_1) \mathrm{d}\mathbf{x}_1 \\
&= -\nabla \cdot \left(\int p_t(\mathbf{x} \mid \mathbf{x}_1) \mathbf{u}_t(\mathbf{x} \mid \mathbf{x}_1) q(\mathbf{x}_1) \mathrm{d}\mathbf{x}_1 \right) \\
&= -\nabla \cdot \left(p_t(\mathbf{x}) \mathbf{u}_t(\mathbf{x}) \right)
\end{aligned}$$

CFM Loss

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t, q(\mathbf{x}_1), p_t(\mathbf{x}|\mathbf{x}_1)} \left[\left\| \mathbf{v}_t^\theta(\mathbf{x}) - \mathbf{u}_t\left(\mathbf{x} \mid \mathbf{x}_1\right) \right\|^2 \right]$$

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{t, p_t(\mathbf{x})} \left[\left\| \mathbf{v}_t^\theta(\mathbf{x}) \right\|^2 \right] + \mathbb{E}_{t, p_t(\mathbf{x})} \left[\left\| \mathbf{u}_t(\mathbf{x}) \right\|^2 \right] - 2\mathbb{E}_{t, p_t(\mathbf{x})} \left[\left\langle \mathbf{v}_t^\theta(\mathbf{x}), \mathbf{u}_t(\mathbf{x}) \right\rangle \right]$$

$$\begin{aligned}
\mathbb{E}_{p_t(\mathbf{x})} \left[\left\langle \mathbf{v}_t^\theta(\mathbf{x}), \mathbf{u}_t(\mathbf{x}) \right\rangle \right] &= \int p_t(\mathbf{x}) \left\langle \mathbf{v}_t^\theta(\mathbf{x}), \int \mathbf{u}_t\left(\mathbf{x} \mid \mathbf{x}_1\right) \frac{p_t\left(\mathbf{x} \mid \mathbf{x}_1\right) q\left(\mathbf{x}_1\right)}{p_t(\mathbf{x})} \mathrm{d}\mathbf{x}_1 \right\rangle \mathrm{d}\mathbf{x} \\
&= \int \left\langle \mathbf{v}_t^\theta(\mathbf{x}), \int \mathbf{u}_t\left(\mathbf{x} \mid \mathbf{x}_1\right) p_t\left(\mathbf{x} \mid \mathbf{x}_1\right) q\left(\mathbf{x}_1\right) \mathrm{d}\mathbf{x}_1 \right\rangle \mathrm{d}\mathbf{x} \\
&= \iint \left\langle \mathbf{v}_t^\theta(\mathbf{x}), \mathbf{u}_t\left(\mathbf{x} \mid \mathbf{x}_1\right) \right\rangle p_t\left(\mathbf{x} \mid \mathbf{x}_1\right) q\left(\mathbf{x}_1\right) \mathrm{d}\mathbf{x}_1 \, \mathrm{d}\mathbf{x} \\
&= \mathbb{E}_{q(\mathbf{x}_1), p_t(\mathbf{x}|\mathbf{x}_1)} \left[\left\langle \mathbf{v}_t^\theta(\mathbf{x}), \mathbf{u}_t\left(\mathbf{x} \mid \mathbf{x}_1\right) \right\rangle \right]
\end{aligned}$$

$$\mathcal{L}'_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,q(\mathbf{x}_1),p_t(\mathbf{x}|\mathbf{x}_1)} \left[\left\| \mathbf{v}_t^\theta(\mathbf{x}) \right\|^2 - 2 \left\langle \mathbf{v}_t^\theta(\mathbf{x}), \mathbf{u}_t\left(\mathbf{x} \mid \mathbf{x}_1\right) \right\rangle \right]$$

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,q(\mathbf{x}_1),p_t(\mathbf{x}|\mathbf{x}_1)} \left[\left\| \mathbf{v}_t^\theta(\mathbf{x}) - \mathbf{u}_t\left(\mathbf{x} \mid \mathbf{x}_1\right) \right\|^2 \right]$$

General form of conditional velocity field

$$p_t\left(\mathbf{x} \mid \mathbf{x}_1\right) = \mathcal{N}\left(\mathbf{x}; \boldsymbol{\mu}_t\left(\mathbf{x}_1\right), \sigma_t^2\left(\mathbf{x}_1\right)\mathbf{I}\right)$$

$$p_0\left(\mathbf{x} \mid \mathbf{x}_1\right) = p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mathbf{0}, \mathbf{I})$$

$$p_1\left(\mathbf{x} \mid \mathbf{x}_1\right) = \mathcal{N}\left(\mathbf{x}; \mathbf{x}_1, \sigma_{\min}^2\mathbf{I}\right)$$

$$\boldsymbol{\psi}_t(\mathbf{x}) = \sigma_t\left(\mathbf{x}_1\right)\mathbf{x} + \boldsymbol{\mu}_t\left(\mathbf{x}_1\right)$$

$$
\begin{aligned}
p_0\left(\boldsymbol{\psi}_t^{-1}(\mathbf{x}) \mid \mathbf{x}_1\right)\left|\det\left[\frac{\partial \boldsymbol{\psi}_t^{-1}}{\partial \mathbf{x}}(\mathbf{x})\right]\right| &= p\left(\boldsymbol{\psi}_t^{-1}(\mathbf{x})\right)\left|\det\left[\frac{\partial \boldsymbol{\psi}_t^{-1}}{\partial \mathbf{x}}(\mathbf{x})\right]\right| \\
&= \mathcal{N}\left(\frac{\mathbf{x}-\boldsymbol{\mu}_t\left(\mathbf{x}_1\right)}{\sigma_t\left(\mathbf{x}_1\right)} ; \mathbf{0}, \mathbf{I}\right)\left|\frac{1}{\sigma_t^d\left(\mathbf{x}_1\right)}\right| \\
&= \mathcal{N}\left(\mathbf{x} ; \boldsymbol{\mu}_t\left(\mathbf{x}_1\right), \sigma_t^2\left(\mathbf{x}_1\right) \mathbf{I}\right) \\
&= p_t\left(\mathbf{x} \mid \mathbf{x}_1\right)
\end{aligned}
$$

$$
\frac{\partial}{\partial t} \boldsymbol{\psi}_t(\mathbf{x})=\mathbf{u}_t\left(\boldsymbol{\psi}_t(\mathbf{x}) \mid \mathbf{x}_1\right)
$$

$$
\begin{aligned}
\mathcal{L}_{\mathrm{CFM}}(\theta) &= \mathbb{E}_{t, q\left(\mathbf{x}_1\right), p\left(\mathbf{x}_0\right)}\left[\left\|\mathbf{v}_t^\theta\left(\boldsymbol{\psi}_t\left(\mathbf{x}_0\right)\right)-\mathbf{u}_t\left(\boldsymbol{\psi}_t\left(\mathbf{x}_0\right) \mid \mathbf{x}_1\right)\right\|^2\right] \\
&= \mathbb{E}_{t, q\left(\mathbf{x}_1\right), p\left(\mathbf{x}_0\right)}\left[\left\|\mathbf{v}_t^\theta\left(\boldsymbol{\psi}_t\left(\mathbf{x}_0\right)\right)-\frac{\partial}{\partial t} \boldsymbol{\psi}_t\left(\mathbf{x}_0\right)\right\|^2\right] \\
&= \mathbb{E}_{t, q\left(\mathbf{x}_1\right), p\left(\mathbf{x}_0\right)}\left[\left\|\mathbf{v}_t^\theta\left(\boldsymbol{\psi}_t\left(\mathbf{x}_0\right)\right)-\sigma_t^{\prime}\left(\mathbf{x}_1\right) \mathbf{x}_0-\boldsymbol{\mu}_t^{\prime}\left(\mathbf{x}_1\right)\right\|^2\right]
\end{aligned}
$$

Another way to learn from demonstrations: Inverse RL

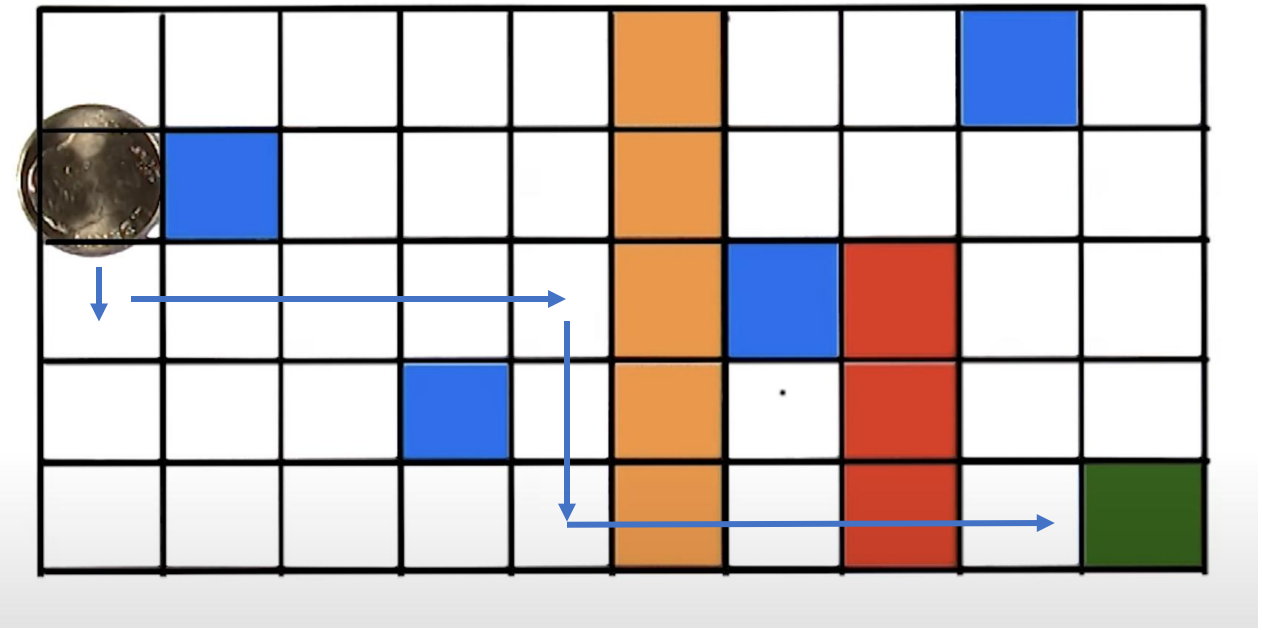
- Instead of learning policies from demonstrations
- Can we learn about the reward function?
- Can we then use these reward function to train an RL agent?
- We will learn these in future lectures if time permits ☺

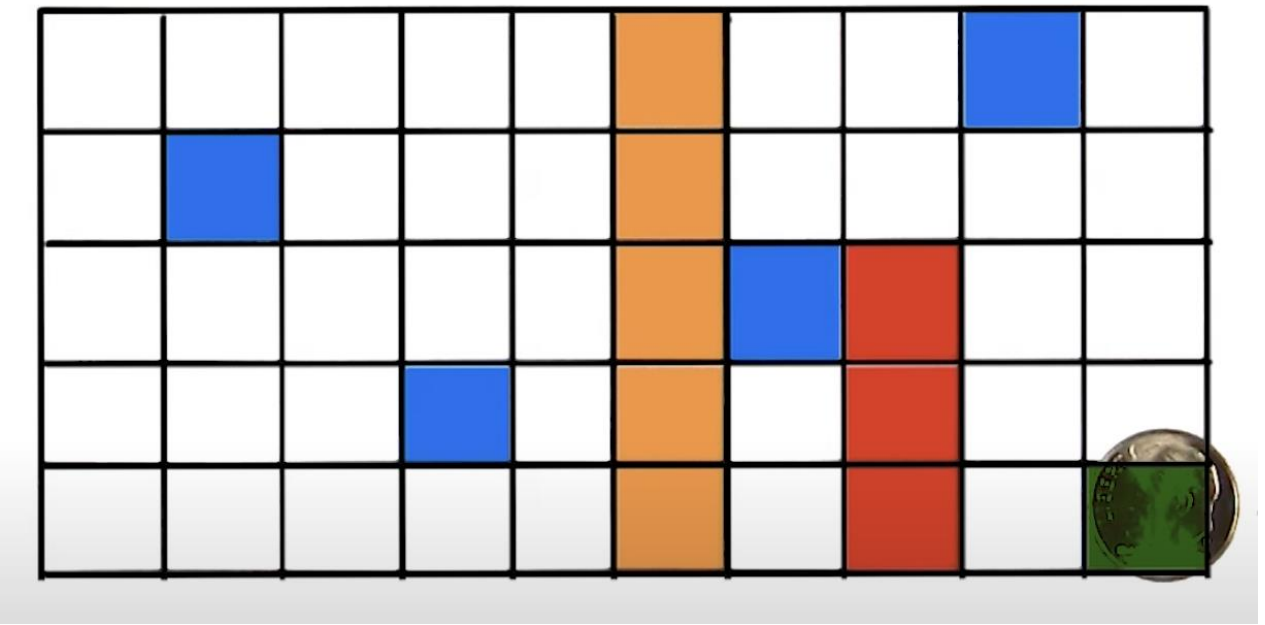
In Lec9

1. Diffusion Models
2. Diffusion in Policy Learning
3. Flow Matching

Recitation starts!

Understand IRL through an Example





- What can you infer?

Source: Udacity

Why do we need inverse RL?

- For science!
  - Model animal and human behaviors

Dynamic Inverse Reinforcement Learning for Characterizing Animal Behavior

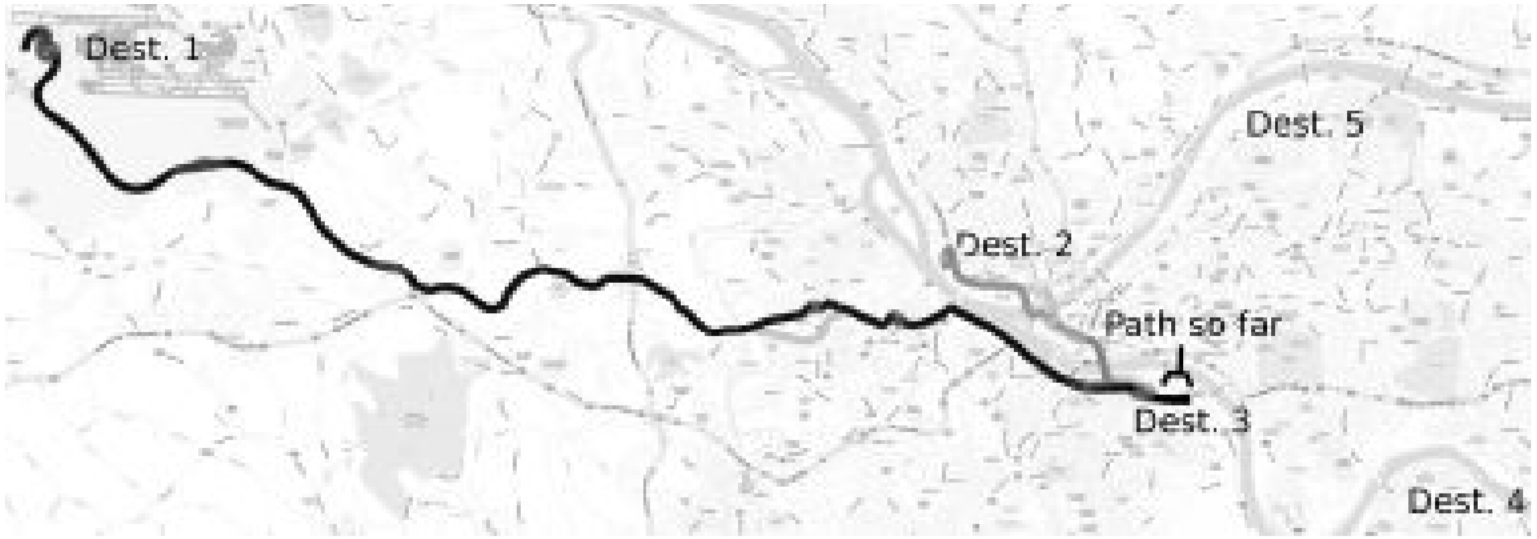
Zoe C. Ashwood[1,2,*] **Aditi Jha**[1,3,*] **Jonathan W. Pillow**[1]

[1]Princeton Neuroscience Institute, Princeton University
[2]Dept. of Computer Science, Princeton University
[3]Dept. of Electrical and Computer Engineering, Princeton University
{zashwood, aditijha, pillow}@princeton.edu

- Imitation
  - Presupposition: reward function provides the most succinct and transferable definition of the task
- Modeling of other agents, both adversarial and cooperative

IRL in Urban Navigation

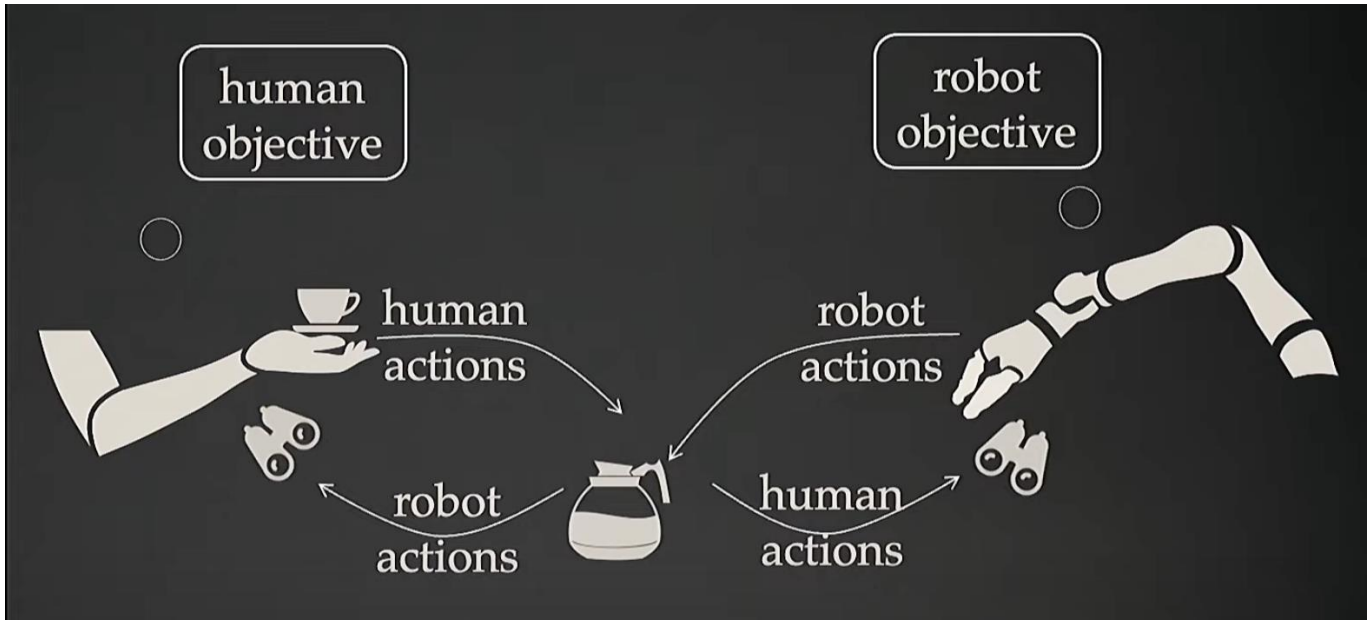
- Infer destination based of partial navigation trajectory

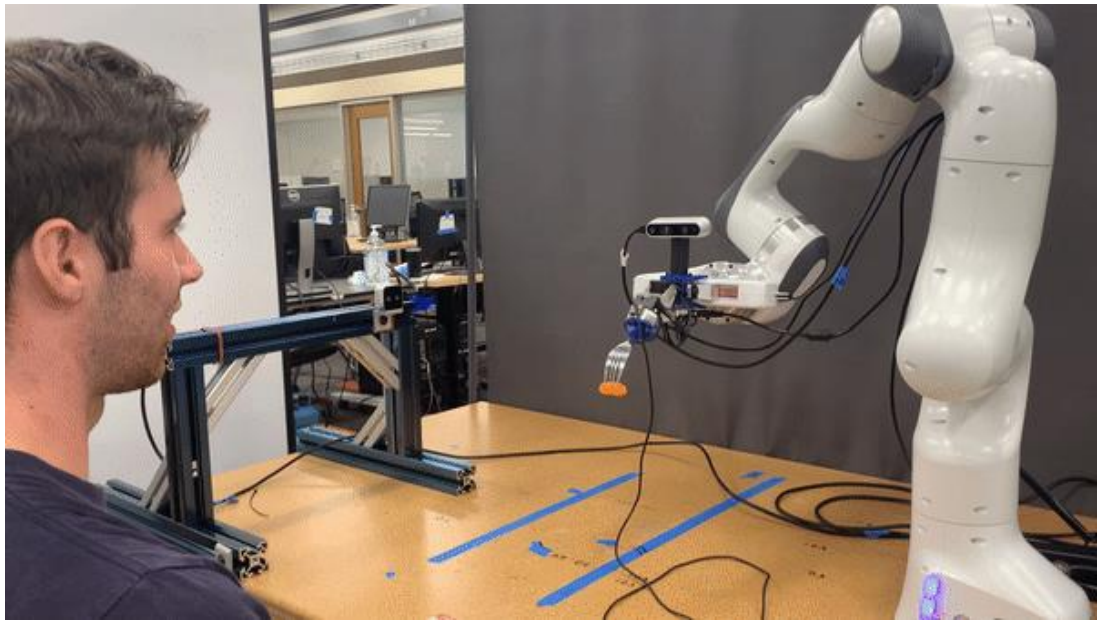


Maximum Entropy Inverse Reinforcement Learning

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

IRL in Human-Robot Interaction (HRI)





Source: Talk from Anca Dragan

IRL v.s. BC

Which has the most succinct description: π^* v.s. R^*?

- It depends.
  - But for RL and planning, reward can be better.
  - From a human video, it is hard to tell what is the optimal policy but easier to tell about the reward.

Basic Principle

- Find a reward function R^* which explains the expert behavior.
- Find R^* such that

$$\mathbf{E}\left[\sum_{t=0}^{\infty}\gamma^t R^*(s_t) \mid \pi^*\right] \geq \mathbf{E}\left[\sum_{t=0}^{\infty}\gamma^t R^*(s_t) \mid \pi\right] \quad \forall \pi$$

- Challenges:
  - R=0 is always a solution. Reward function ambiguity!
  - We observe trajectories rather than policies. How to compute the left-hand side?
  - Expert has to be optimal.
  - It is not possible to enumerate policies.

Feature-Based Reward Function

$$\textbf{Let } R(s) = w^\top \phi(s), \textbf{ where } w \in \Re^n, \textbf{ and } \phi : S \to \Re^n.$$

$$\begin{aligned} \mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi\right] &= \mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t w^\top \phi(s_t) \mid \pi\right] \\ &= w^\top \mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t) \mid \pi\right] \\ &= w^\top \mu(\pi) \end{aligned}$$

- μ is the feature expectation.
- Recall that $\mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t R^*(s_t) \mid \pi^*\right] \geq \mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t R^*(s_t) \mid \pi\right] \quad \forall \pi$

$$\textbf{Find } w^* \textbf{ such that } w^{*\top} \mu(\pi^*) \geq w^{*\top} \mu(\pi) \quad \forall \pi$$

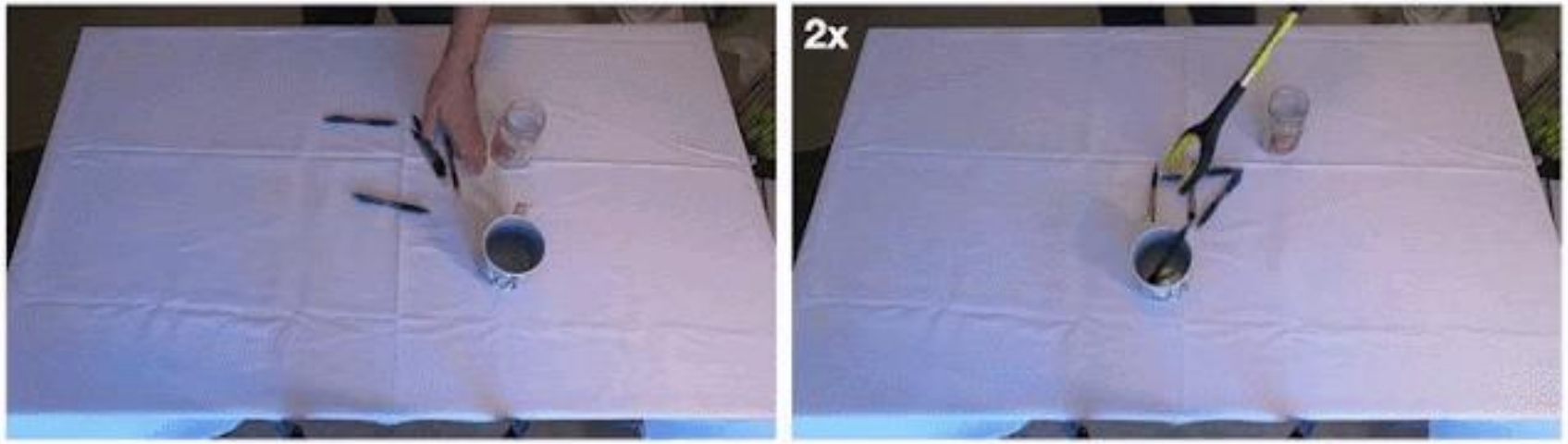
What can you do if demos are videos?

- Follow the idea of IRL
  - Reward learning from videos!
- How can you achieve that?

Source: google ai blog

What are the challenges?

- Different domain!

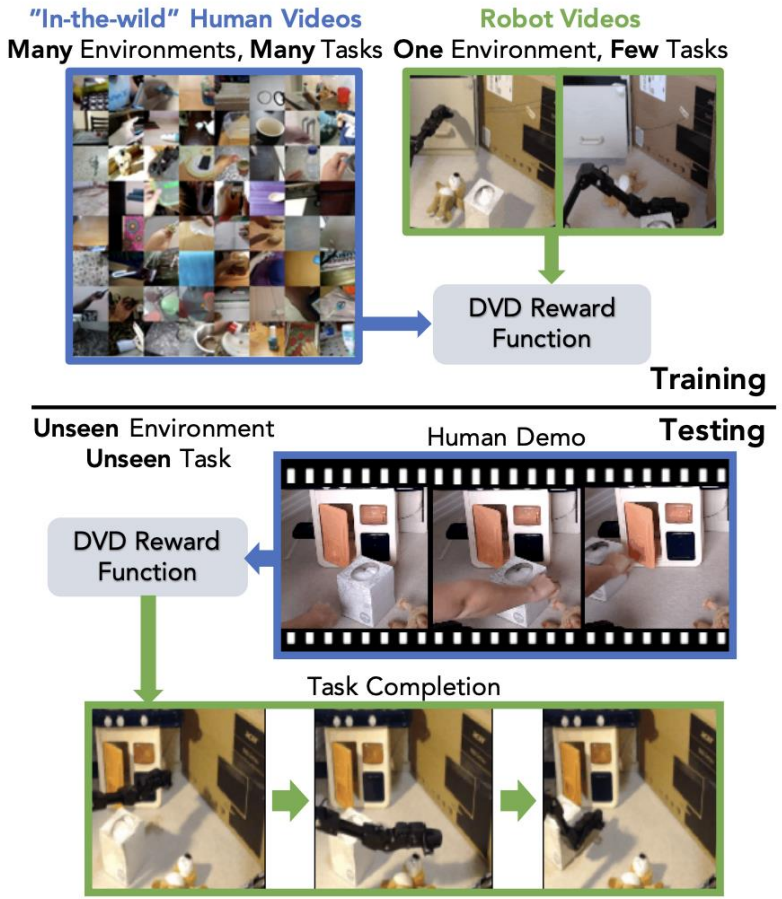


Learning Generalizable Robotic Reward Functions from "In-The-Wild" Human Videos

Annie S. Chen, Suraj Nair, Chelsea Finn
Stanford University

- Domain-agnostic Discriminator
  - A neural network that learns similarity Among tasks.
    + Positive samples, same task
    - Negative samples, different task

    Tasks are from human or robot.
- Video Predictor
  - Serves as a dyanmics model



Learning Generalizable Robotic Reward Functions from "In-The-Wild" Human Videos

Annie S. Chen, Suraj Nair, Chelsea Finn
Stanford University

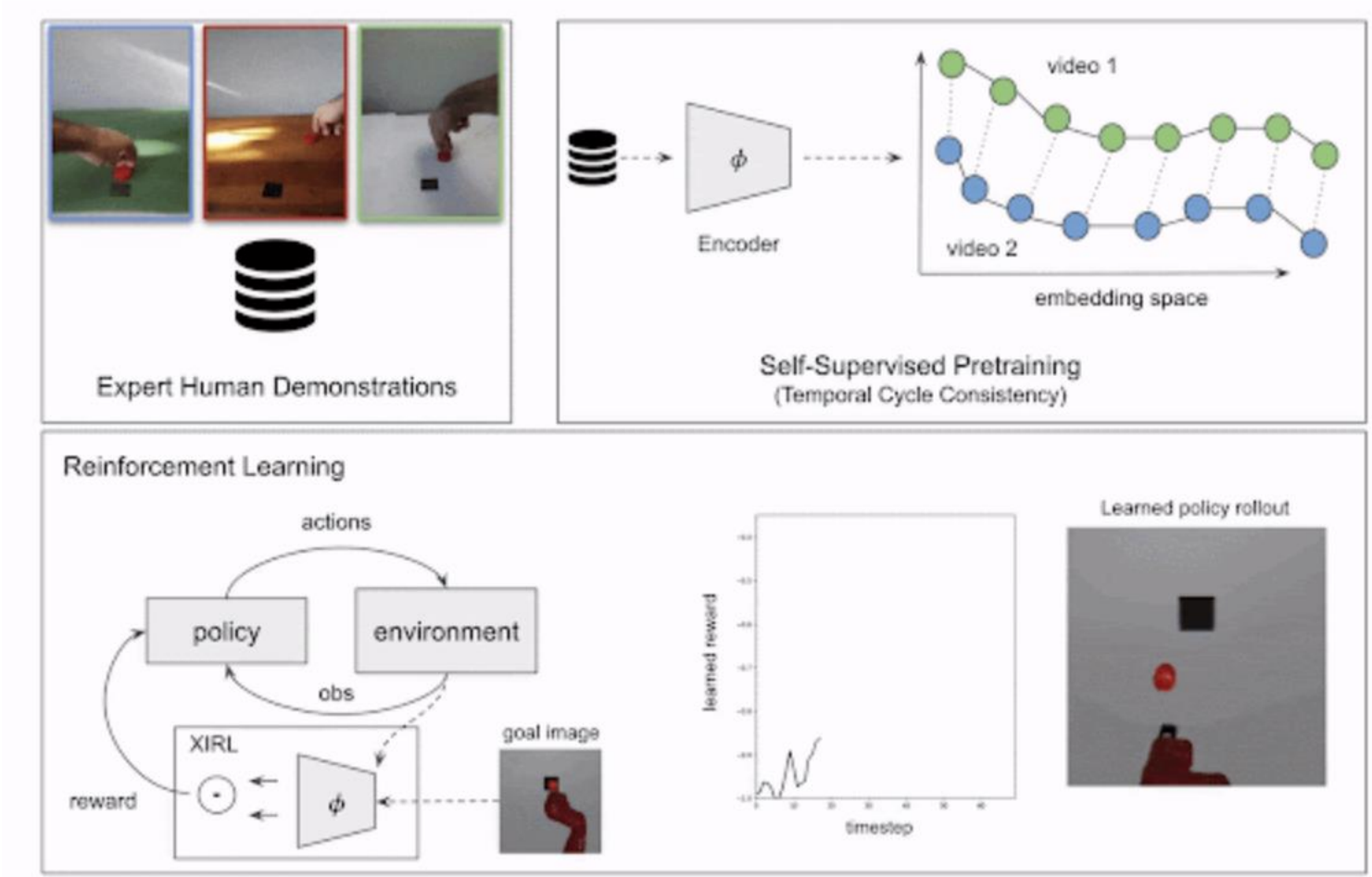
- Algorithm

Algorithm 1 DOMAIN-AGNOSTIC VIDEO DISCRIMINATOR (DVD)

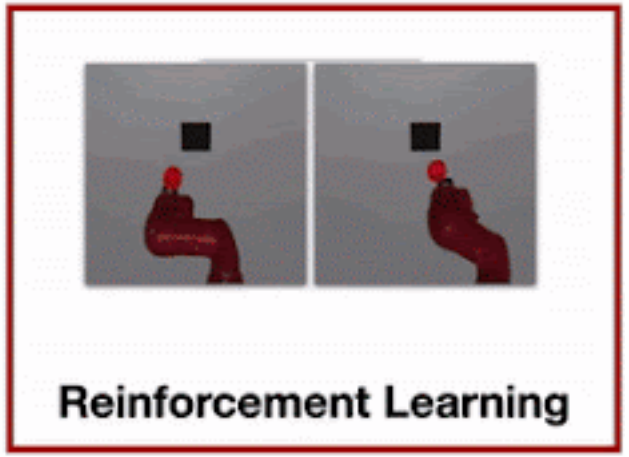
// Training DVD
Require: $\mathcal{D}^h$ human demonstration data for N tasks $\{\mathcal{T}_n\}$
Require: $\mathcal{D}^r$ robot demonstration data for M tasks $\{\mathcal{T}_m\} \subseteq \{\mathcal{T}_n\}$
Require: Pre-trained video encoder f_{enc}
Randomly initialize θ
while training **do**
    Sample anchor video $d_i \in \mathcal{D}^h \cup \mathcal{D}^r$
    Sample positive video $d_i' \in \{\mathcal{D}^h_{\mathcal{T}_i}\} \cup \{\mathcal{D}^r_{\mathcal{T}_i}\} \setminus d_i$
    Sample negative video $d_j \in \{\mathcal{D}^h_{\mathcal{T}_j}\} \cup \{\mathcal{D}^r_{\mathcal{T}_j}\} \forall j \neq i$
    Update $\mathcal{R}_\theta$ with d_i, d_i', d_j according to Eq. 1
// Planning Conditioned on Video Demo
Require: Trained reward function $\mathcal{R}_\theta$ & video prediction model p_ϕ
Require: Human video demo d_i for task $\mathcal{T}_i$
for trials $1, ..., n$ **do**
    Sample $\{a^{1:G}_{1:H}\}$ & get predictions $\{\tilde{s}^g_{1:H}\} \sim \{p_\phi(s_0, a^g_{1:H})\}$
    Step $a^*_{1:H}$ which maximizes $\mathcal{R}_\theta(\tilde{s}^g_{1:H}, d_i)$

XIRL:
Cross-embodiment Inverse Reinforcement Learning

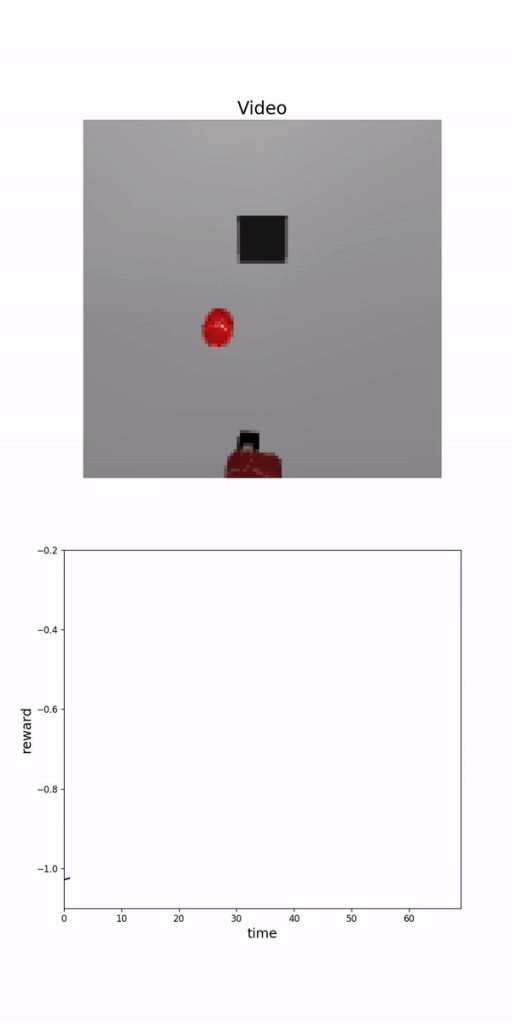
Kevin Zakka[1,3*], **Andy Zeng**[2], **Pete Florence**[2], **Jonathan Tompson**[2],
Jeannette Bohg[1], **and Debidatta Dwibedi**[2]
[1]Stanford University, [2]Robotics at Google, [3]UC Berkeley

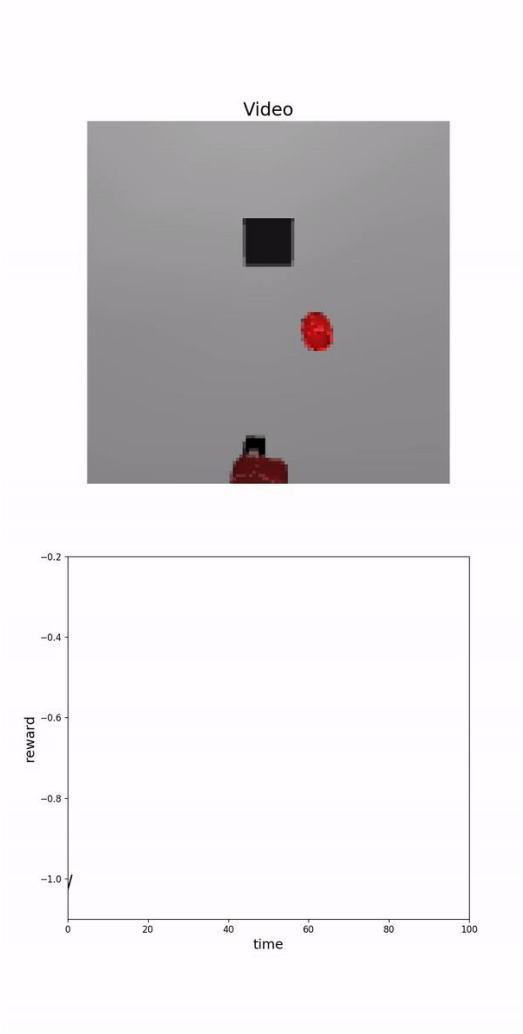


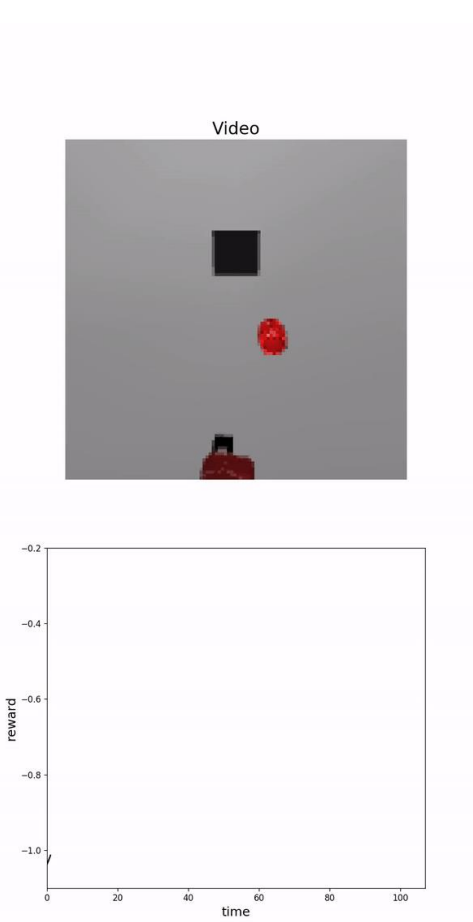
From IRL to RL



The XIRL-learned reward

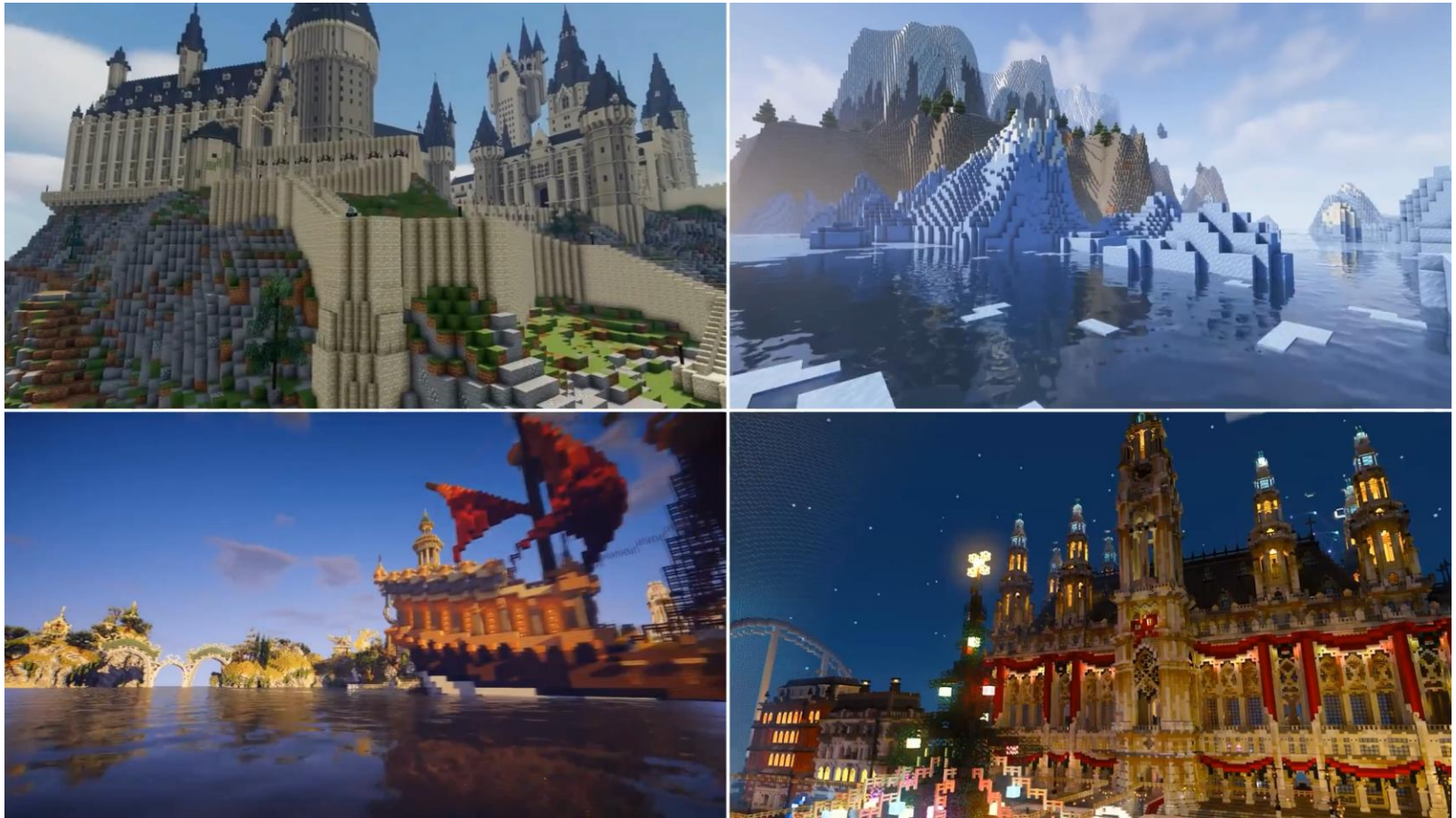






MINEDOJO: Building Open-Ended Embodied Agents with Internet-Scale Knowledge

Linxi Fan[1], **Guanzhi Wang**[2*], **Yunfan Jiang**[3*], **Ajay Mandlekar**[1], **Yuncong Yang**[4], **Haoyi Zhu**[5], **Andrew Tang**[4], **De-An Huang**[1], **Yuke Zhu**[1 6†], **Anima Anandkumar**[1 2†]
[1]NVIDIA, [2]Caltech, [3]Stanford, [4]Columbia, [5]SJTU, [6]UT Austin
[*]Equal contribution [†]Equal advising
https://minedojo.org



Learning Rewards from Language