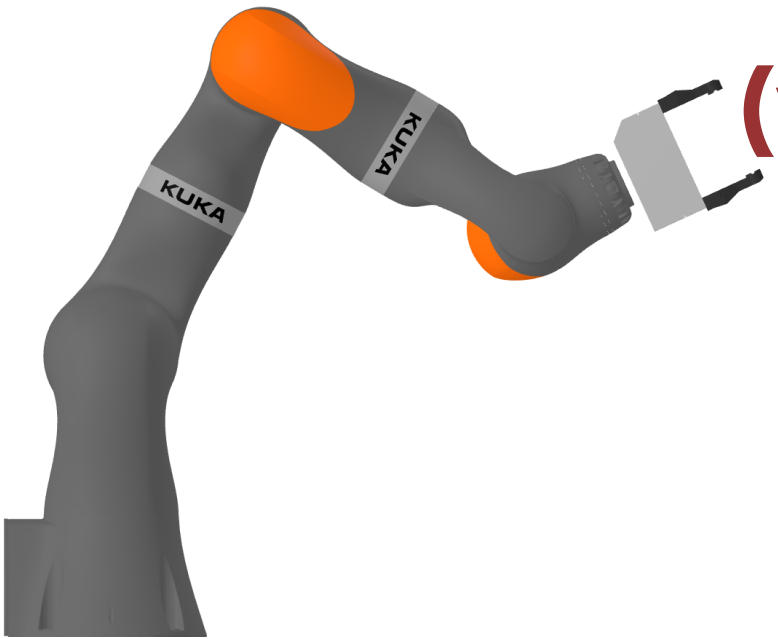


Visuomotor Policies (via Behavior Cloning)

MIT 6.4210/2:

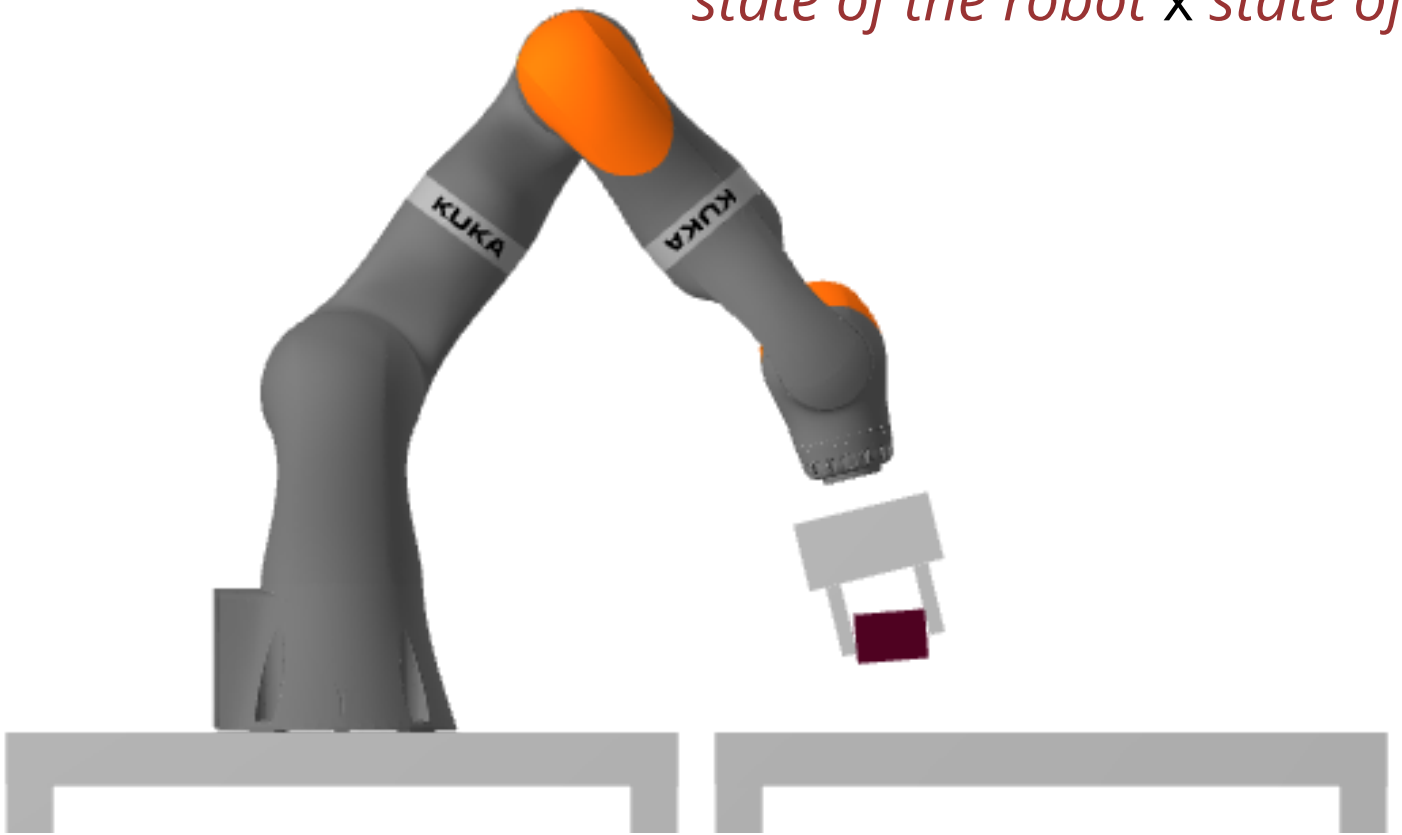
Robotic Manipulation

Fall 2025, Lecture 20

Massachusetts Institute of Technology

policy needs to know

state of the robot x *state of the environment*



https://www.youtube.com/embed/0LJb66aYtG8?start=111&mute=1&enablejsapi=1&controls=0&rel=0&modestbranding=1

https://www.youtube.com/embed/1RbaIo4VdbA? enablejsapi=1&start=112&mute=1

https://www.youtube.com/embed/3qMRaIJ3D3o?start=44&end=98&rel=0&enablejsapi=1

Carnegie Mellon
School of Computer Science

Deep Reinforcement Learning and Control

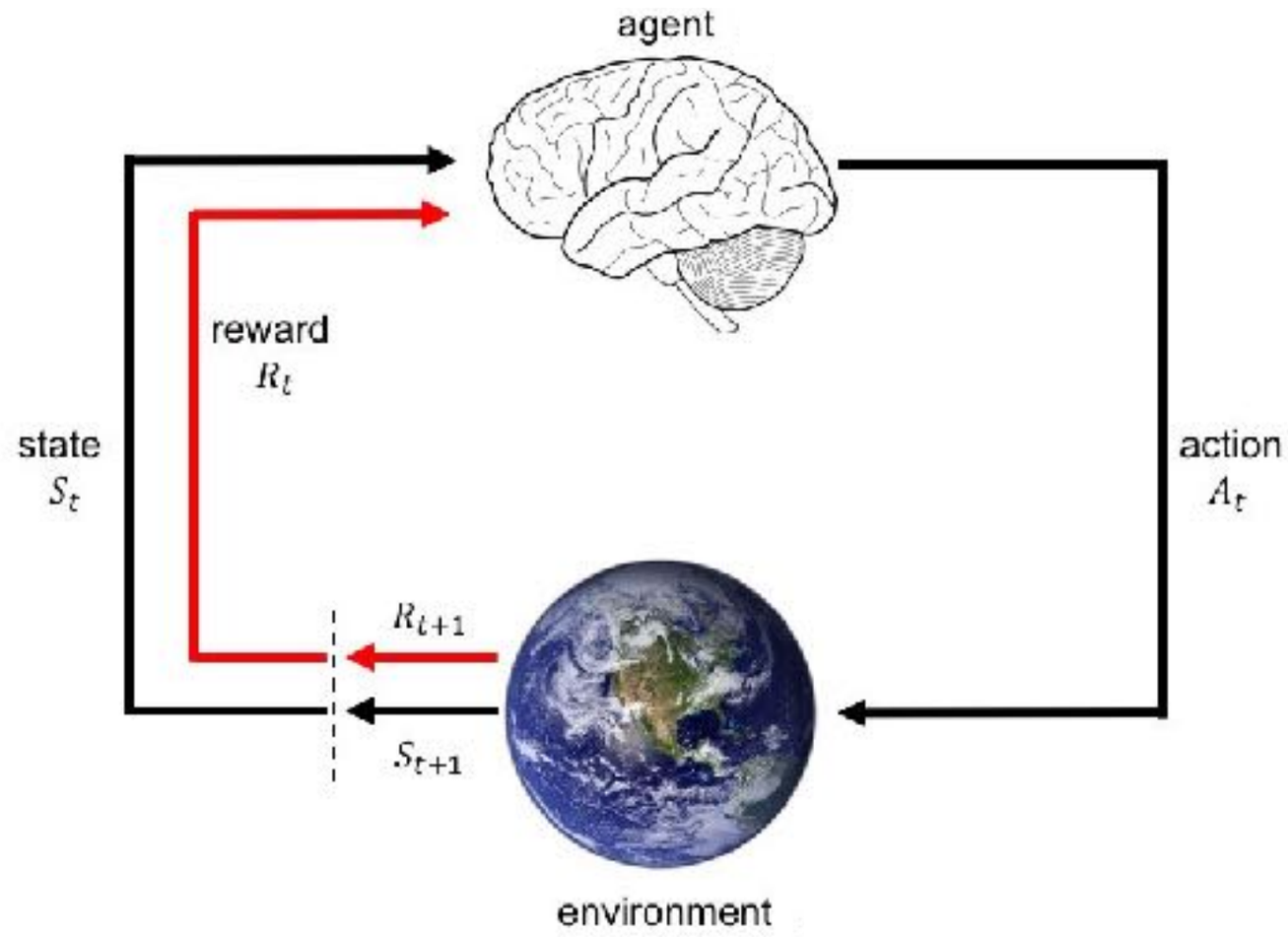
Imitation Learning

Spring 2019, CMU 10-403

Katerina Fragkiadaki



Reinforcement learning



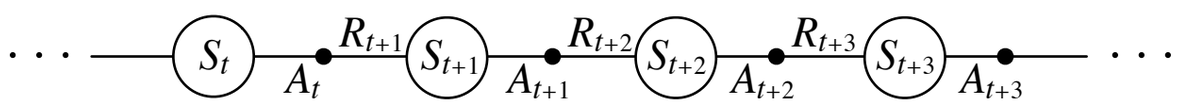
Agent and environment interact at discrete time steps: $t = 0, 1, 2, 3, \ldots$

Agent observes state at step t: $S_t \in \mathcal{S}$

produces action at step t : $A_t \in \mathcal{A}(S_t)$

gets resulting reward: $R_{t+1} \in \mathcal{R} \subset \mathbb{R}$

and resulting next state: $S_{t+1} \in \mathcal{S}^+$



Limitations of Learning by Interaction

- The agent should have the chance to try (and fail) MANY times
- This is impossible when safety is a concern: we cannot afford to fail
- This is also quite impossible in general in real life where each interaction takes time (in contrast to simulation)



Crusher robot

Learning from Demonstration for Autonomous Navigation in Complex Unstructured Terrain, Silver et al. 2010

Imitation Learning (a.k.a. Learning from Demonstrations)

visual imitation





The actions of the teacher need to be inferred from visual sensory input and mapped to the end-effectors to the agent.

Two challenges:

1) visual understanding
2) action mapping, especially when the agent and the teacher do not have the same action space

we will come back to this in a later lecture

kinesthetic imitation





- The teacher takes over the end-effectors of the agent.
- Demonstrated actions can be imitated directly (*cloned*)
- A.k.a. behavior cloning

this lecture!

Imitating Controllers

visual imitation

kinesthetic imitation



- Experts do not need to be humans.
- Machinery that we develop in this lecture can be used for imitating expert policies found through (easier) optimization in a constrained smaller part of the state space.
- Imitation then means distilling knowledge of expert constrained policies into a general policy that can do well in all scenarios the simpler policies do well.

The actions of the teacher need to be inferred from visual sensory input and mapped to the end-effectors to the agent.

Two challenges:

1) visual understanding
2) action mapping, especially when the agent and the teacher do not have the same action space

We will come back to this in a later lecture

- The teacher takes over the end-effectors of the agent.
- Demonstrated actions can be imitated directly (cloned)
- A.k.a. behavior cloning

this lecture!

Notation



Richard Bellman



Lev Pontryagin

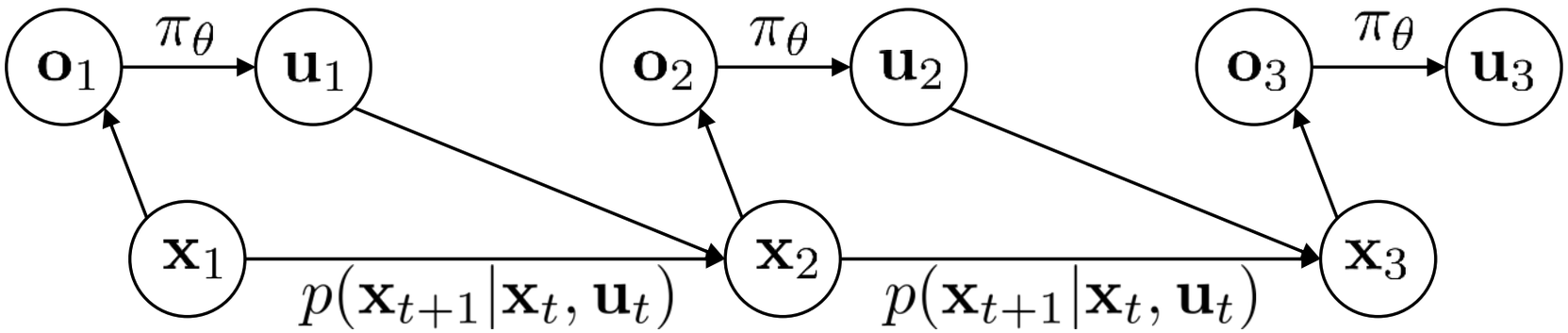
actions	a_t
states	s_t
rewards	r_t
dynamics	$p(s_{t+1} \mid s_t, a_t)$
observations	o_t

actions	u_t
states	x_t
costs	$c(x_t, u_t)$
dynamics	$p(x_{t+1} \mid x_t, u_t)$

Diagram from Sergey Levine

Imitation learning VS Sequence labelling

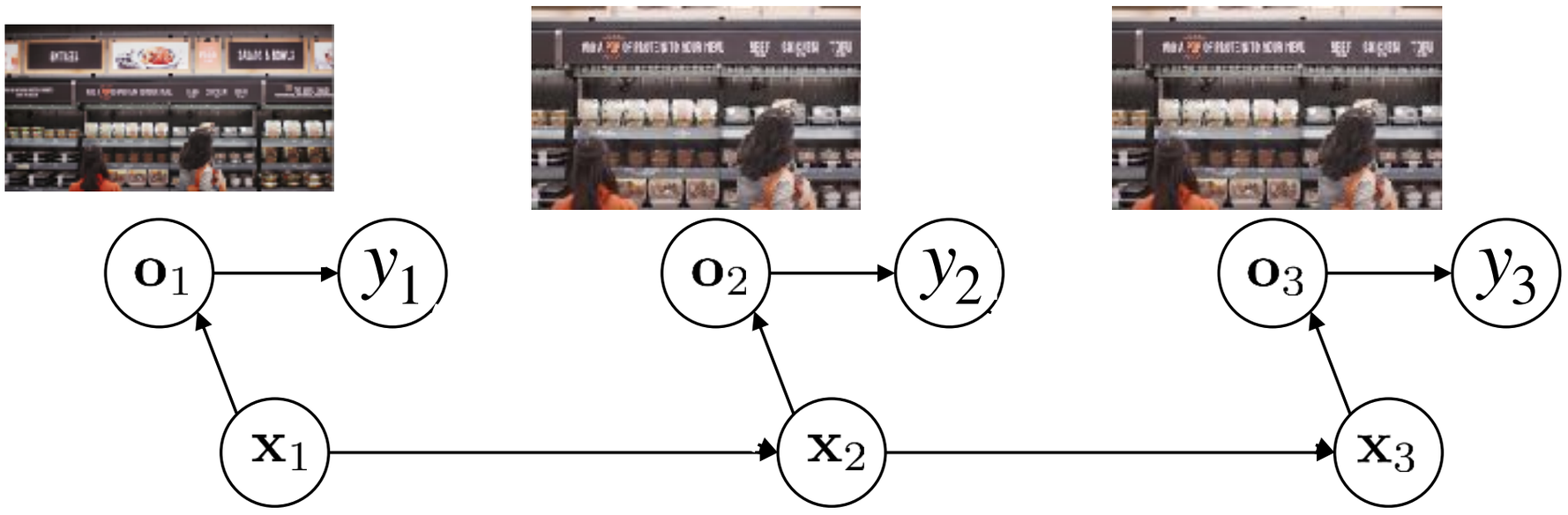
Imitation learning



Training data:

$o_1^1, u_1^1, o_2^1, u_2^1, o_3^1, u_3^1, \ldots.$
$o_1^2, u_1^2, o_2^2, u_2^2, o_3^2, u_3^2, \ldots.$
$o_1^3, u_1^3, o_2^3, u_2^3, o_3^3, u_3^3, \ldots.$

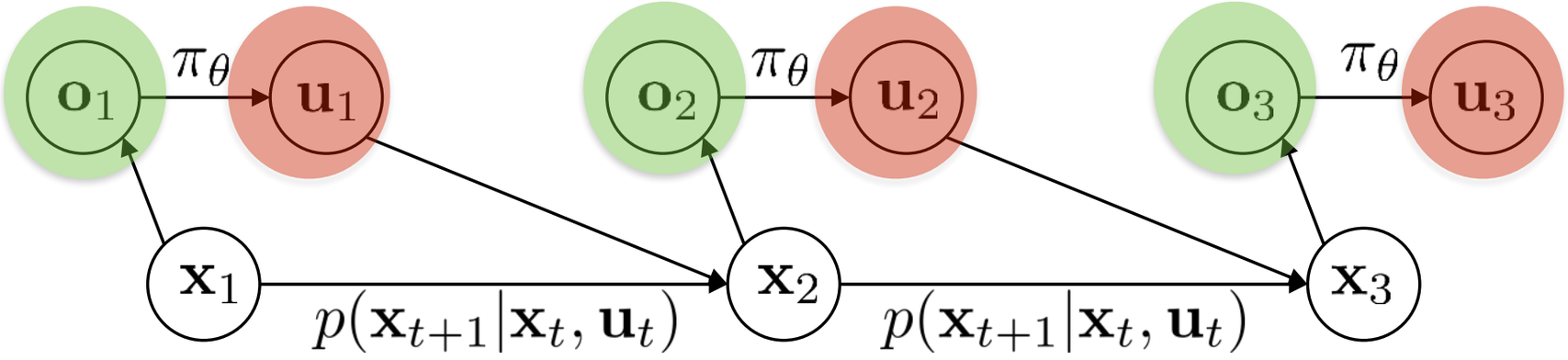
Sequence labelling



y: which product was purchased if any

Imitation learning VS Sequence labelling

Imitation learning



Training data:

$o_1^1, u_1^1, o_2^1, u_2^1, o_3^1, u_3^1, \ldots.$
$o_1^2, u_1^2, o_2^2, u_2^2, o_3^2, u_3^2, \ldots.$
$o_1^3, u_1^3, o_2^3, u_2^3, o_3^3, u_3^3, \ldots.$

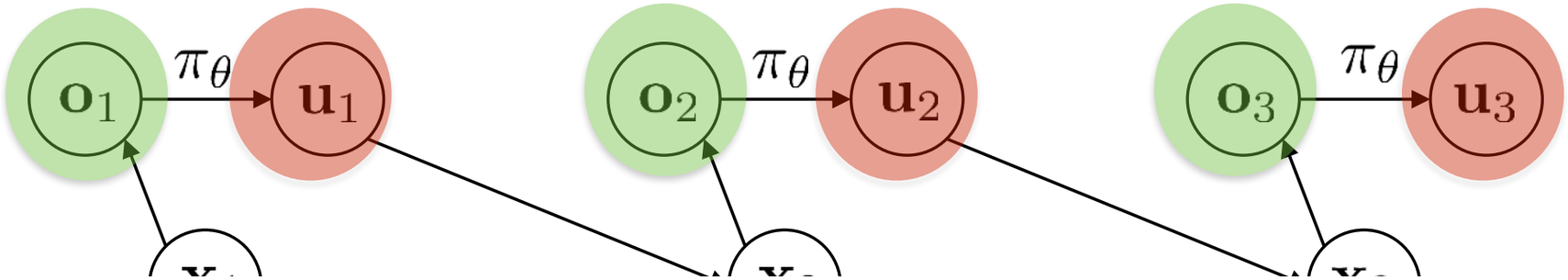
Sequence labelling



y: which product was purchased if any

Imitation learning VS Sequence labelling

Imitation learning



Action interdependence in imitation learning: the actions we predict will influence the data we will see next, and thus, our future predictions.

g data:

$_3^1, u_3^1, \ldots.$
$_3^2, u_3^2, \ldots.$
$_3^3, u_3^3, \ldots.$

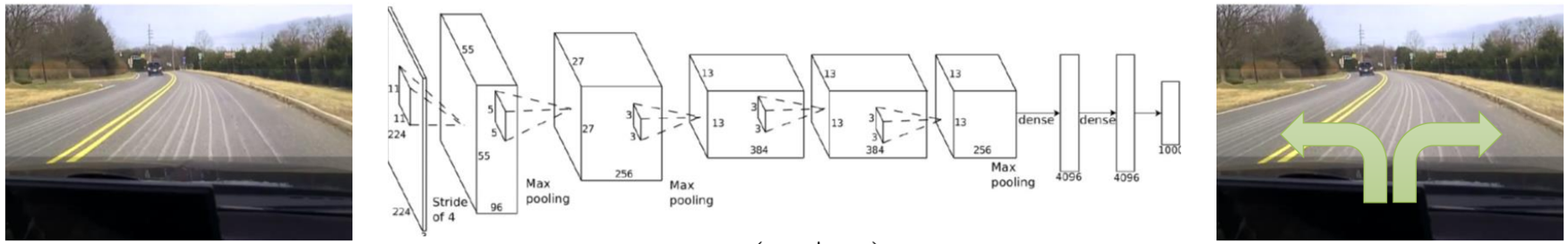
Label interdependence is present in any structured prediction task, e.g, text generation: words we predict influence words we need to predict further down the sentence…



y: which product was purchased if any

Imitation Learning for Driving

Driving policy: a mapping from observations to steering wheel angles

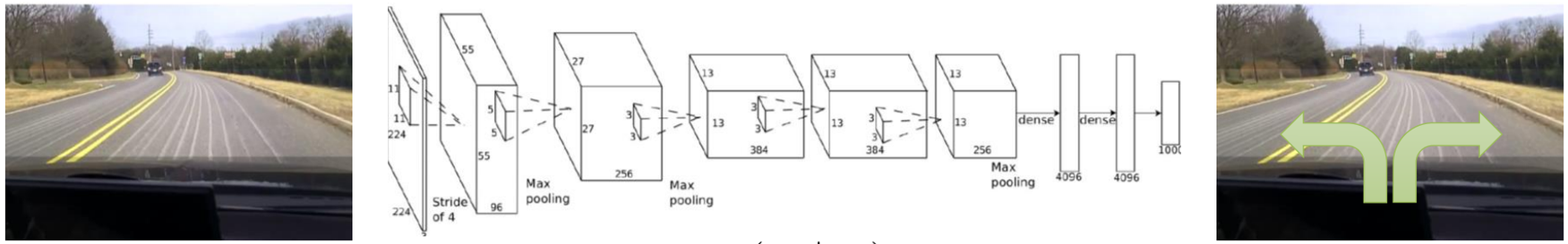


$\mathbf{o}_t$ $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ $\mathbf{u}_t$

End to End Learning for Self-Driving Cars, Bojarski et al. 2016

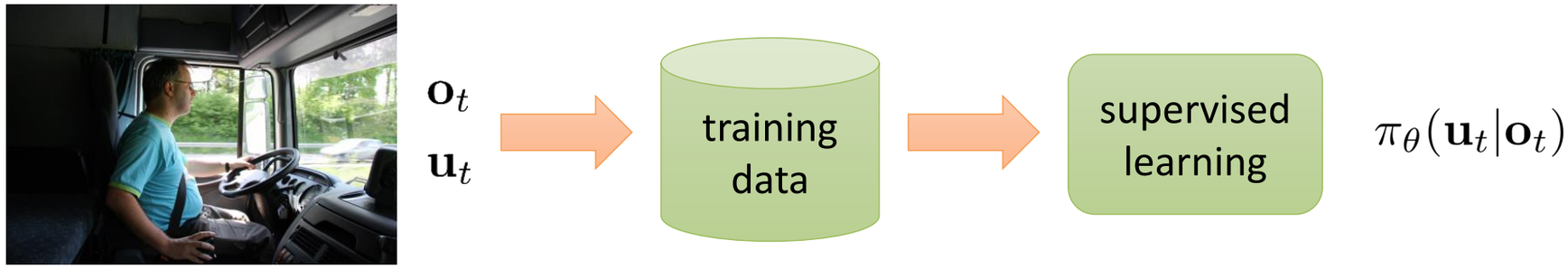
Imitation Learning as Supervised Learning

Driving policy: a mapping from observations to steering wheel angles



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

- Assume actions in the expert trajectories are i.i.d.
- Train a function approximator to map observations to actions at each time step of the trajectory.



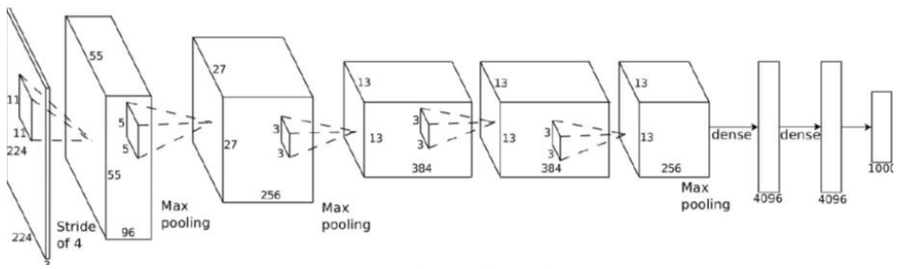
End to End Learning for Self-Driving Cars, Bojarski et al. 2016

What can go wrong?

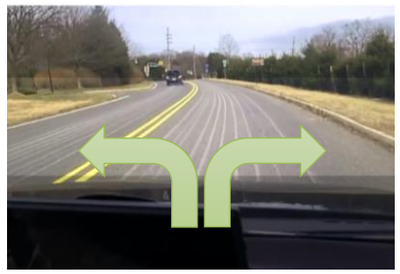
- Compounding errors
  Fix: data augmentation
- Stochastic expert actions
  Fix: stochastic latent variable models, action discretiation, gaussian mixture networks
- Non-markovian observations
  Fix: observation concatenation or recurrent models



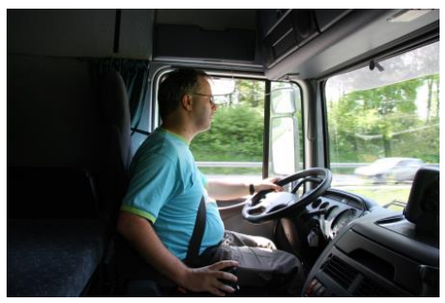
$\mathbf{o}_t$



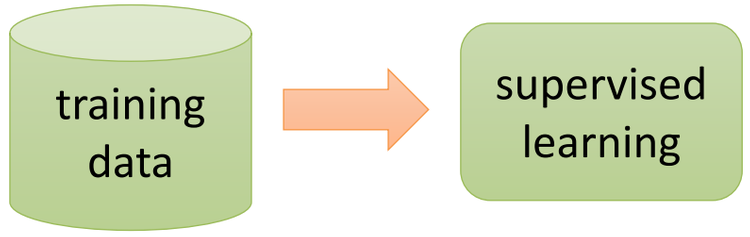
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$



$\mathbf{u}_t$



$\mathbf{o}_t$
$\mathbf{u}_t$



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

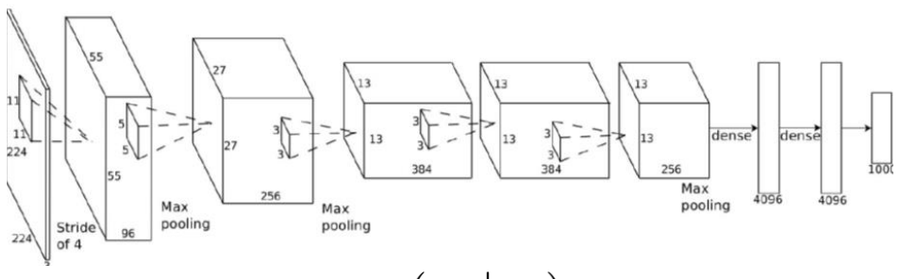
End to End Learning for Self-Driving Cars, Bojarski et al. 2016

What can go wrong?

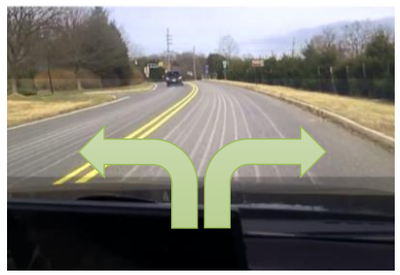
- Compounding errors
  Fix: data augmentation
- Stochastic expert actions
  Fix: stochastic latent variable models, action discretiation, gaussian mixture networks
- Non-markovian observations
  Fix: observation concatenation or recurrent models



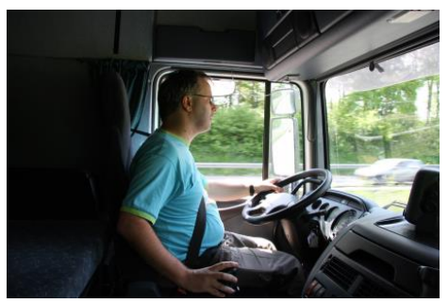
$\mathbf{o}_t$



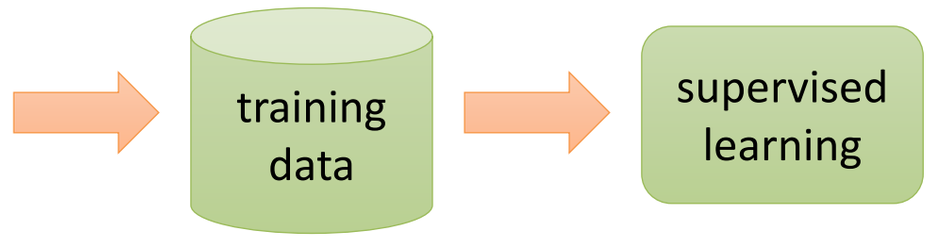
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$



$\mathbf{u}_t$



$\mathbf{o}_t$
$\mathbf{u}_t$

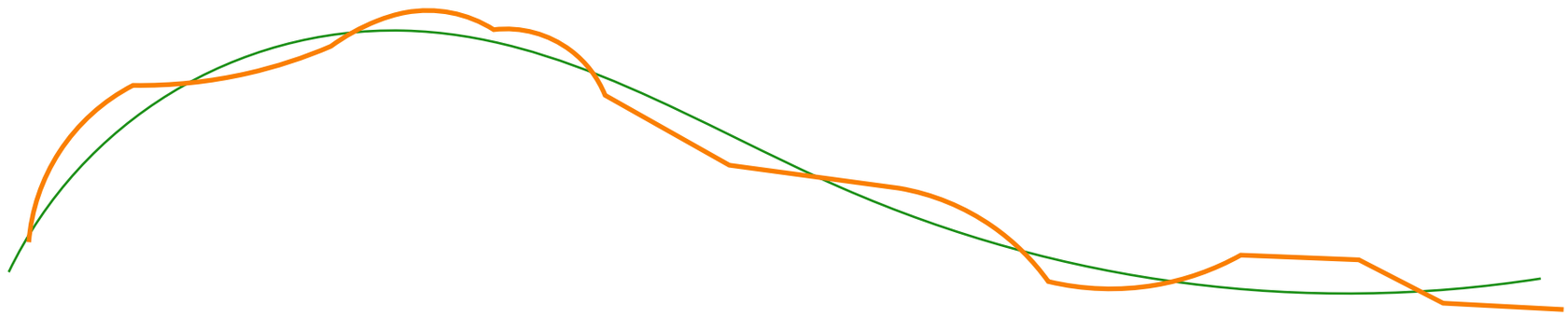


$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

End to End Learning for Self-Driving Cars, Bojarski et al. 2016

Independent in time errors

This means that at each time step t, the agent wakes up on a state drawn from the data distribution of the expert trajectories, and executes an action

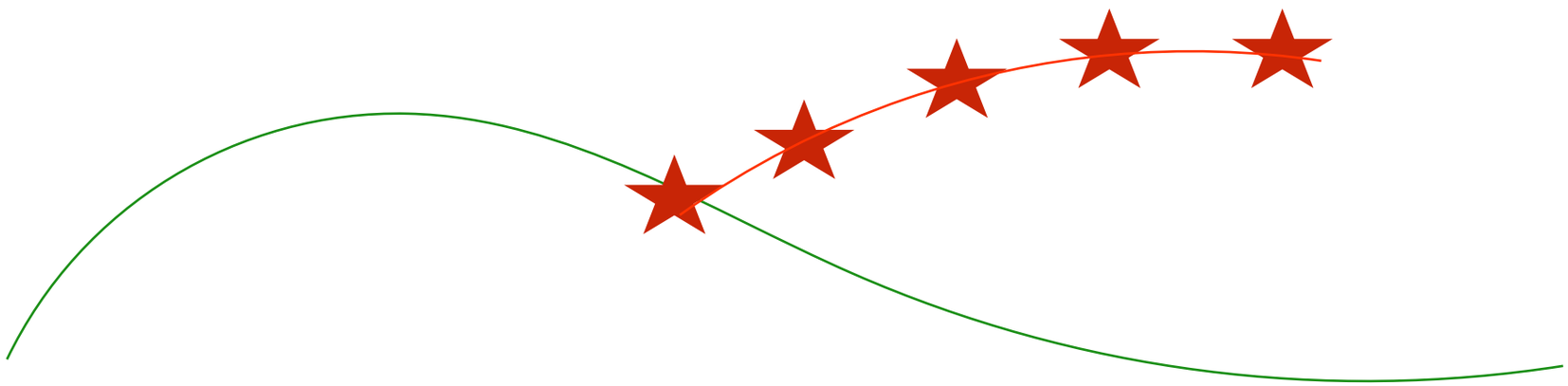


error at time t with probability ε

E[Total errors] ≲ εT

Compounding Errors

This means that at each time step t, the agent wakes up on the state that resulted from executing the action the learned policy suggested in the previous time step.



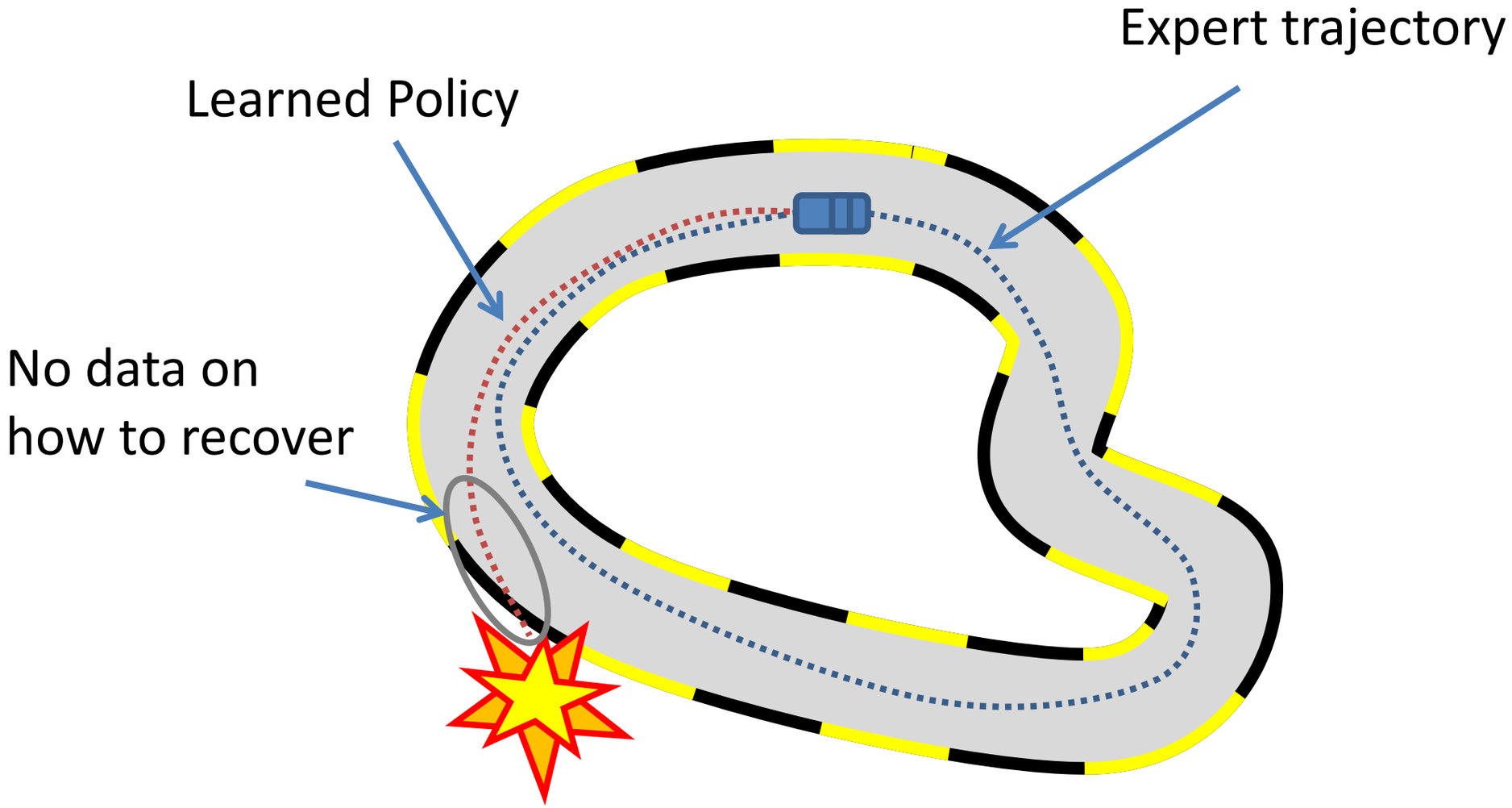
error at time t with probability ε

$$E[\text{Total errors}] \lesssim \varepsilon(T + (T-1) + (T-2) + \ldots + 1) \propto \varepsilon T^2$$

A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning, Ross et al. 2011

Data Distribution Mismatch!

$$p_{\pi^*}(o_t) \neq p_{\pi_\theta}(o_t)$$



Data Distribution Mismatch!

	supervised learning	supervised learning + control (NAIVE)
train	$(x,y) \sim D$	$s \sim d_{\pi^*}$
test	$(x,y) \sim D$	$s \sim d_{\pi}$

SL succeeds when training and test data distributions match, that is a fundamental assumption.

Solution: data augmentations

Change $p_{\pi^*}(o_t)$ using demonstration augmentation!! Add examples in expert demonstration trajectories to cover the states/observations points where the agent will land when trying out its own policy. How?

- Synthetically in simulation or by clever hardware
- Interactively with experts in the loop (DAGGER)

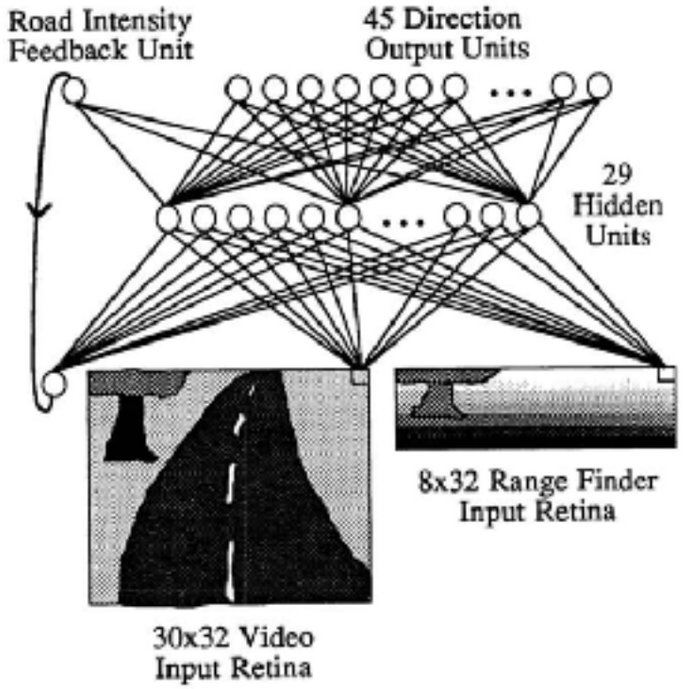
Solution: data augmentations

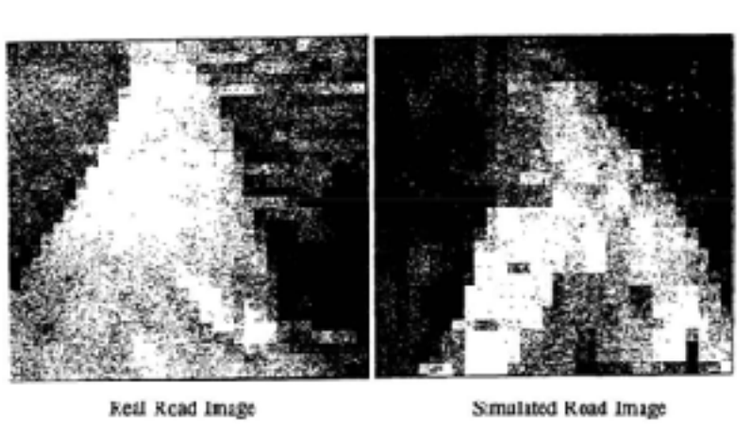
Change the training data distribution $p_{\pi^*}(o_t)$ using demonstration augmentation: add examples in expert demonstration trajectories to cover the states/observations where the agent will land when trying out its own policy.

	supervised learning	supervised learning + control (NAIVE)
train	(x,y) ~ D	s ~ d_{π^*}
test	(x,y) ~ D	s ~ d_{π}

Demonstration Augmentation: ALVINN 1989

Road follower



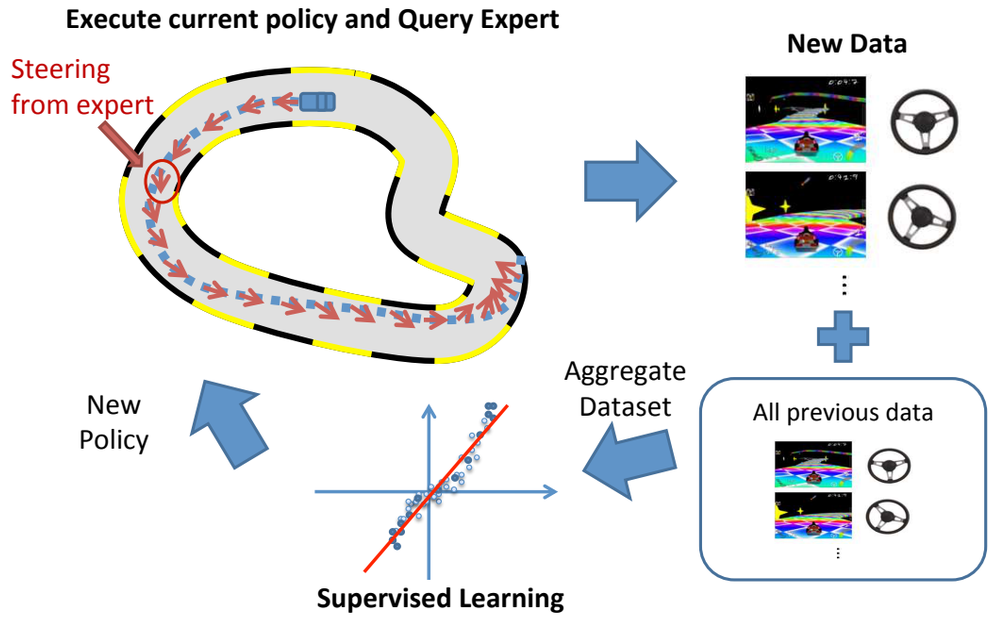


- Using graphics simulator for road images and corresponding steering angle ground-truth
- Online adaptation to human driver steering angle control
- 3 layers, fully connected layers, very low resolution input from camera

> *"In addition, the network must not solely be shown examples of accurate driving, but also how to recover (i.e. return to the road center) once a mistake has been made. Partial initial training on a variety of simulated road images should help eliminate these difficulties and facilitate better performance."*
> *ALVINN: An autonomous Land vehicle in a neural Network", Pomerleau 1989*

DAGGER (in simulation)

Dataset AGGregation: bring learner's and expert's trajectory distributions closer by (asking uman experts to provide) labelling additional data points resulting from applying the current policy



A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning, Ross et al. 2011

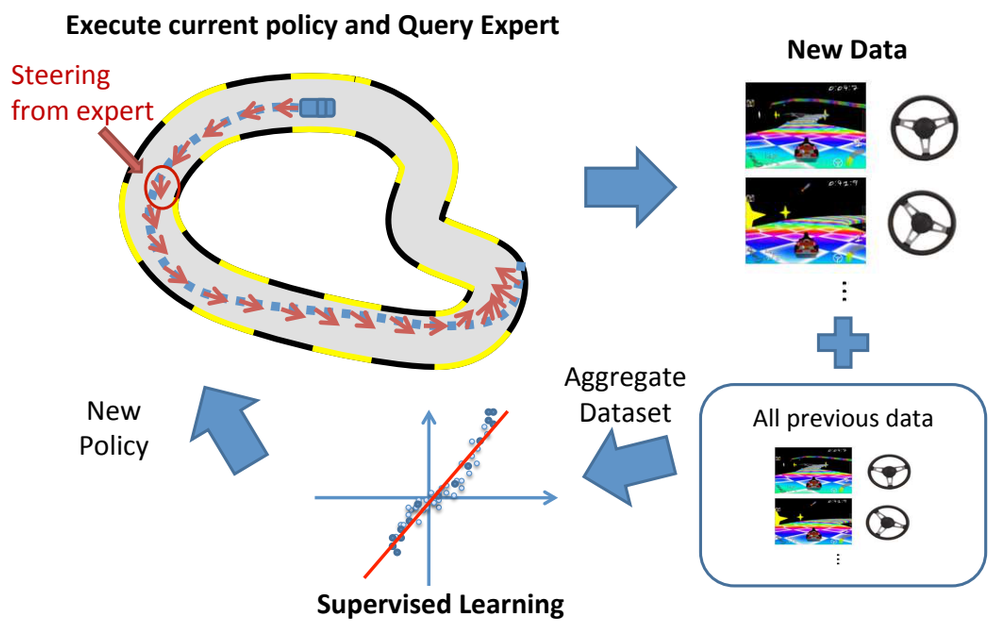
DAGGER (in simulation)

Dataset AGGregation: bring learner's and expert's trajectory distributions closer by (asking uman experts to provide) labelling additional data points resulting from applying the current policy

1. train $\pi_\theta(u_t|o_t)$ from human data $\mathcal{D}_{\pi^*} = \{o_1, u_1, ..., o_N, u_N\}$
2. run $\pi_\theta(u_t|o_t)$ to get dataset $\mathcal{D}_\pi = \{o_1, ..., o_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions u_t
4. Aggregate: $\mathcal{D}_{\pi^*} \leftarrow \mathcal{D}_{\pi^*} \cup \mathcal{D}_\pi$
5. GOTO step 1.

Problems:

- execute an unsafe/partially trained policy
- repeatedly query the expert



A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning, Ross et al. 2011

DAGGER (on a real platform)

Application on drones: given RGB from the drone camera predict steering angles



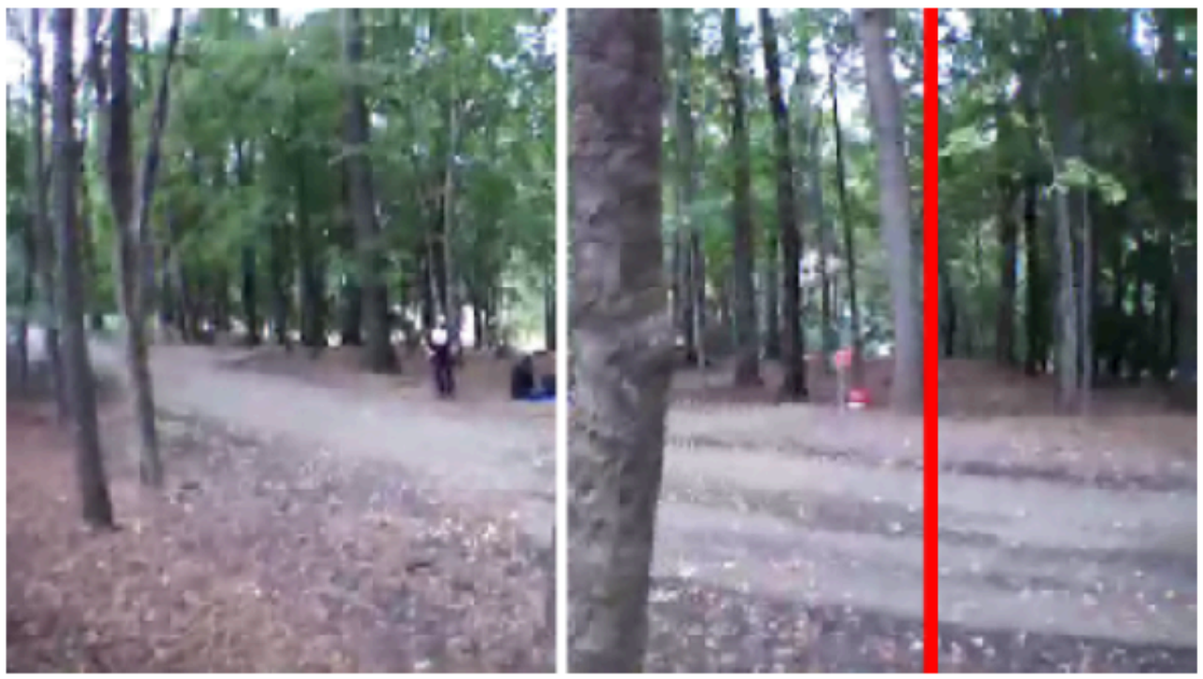
Learning monocular reactive UAV control in cluttered natural environments, Ross et al. 2013

DAGGER (on a real platform)

Application on drones : given RGB from the drone camera predict steering angle

Caveats:

1. It is hard for the expert to provide the right magnitude for the turn without feedback of his own actions! Solution: provide him with visual feedback



Learning monocular reactive UAV control in cluttered natural environments, Ross et al. 2013

DAGGER (on a real platform)

Caveats:

1. Is hard for the expert to provide the right magnitude for the turn without feedback of his own actions! Solution: provide him with his visual feedback

2. The expert's reaction time to the drone's behavior is large, this causes imperfect actions to be commanded. Solution: play-back in slow motion offline and record their actions.

3. Executing an imperfect policy causes accidents, crashes into obstacles. Solution: safety measures which make again the data distribution matching imperfect between train and test, but good enough..

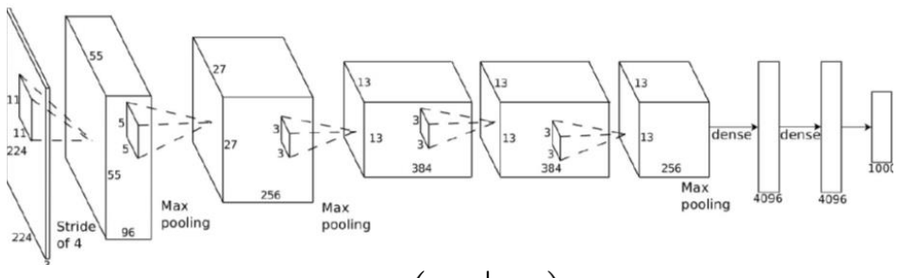
Learning monocular reactive UAV control in cluttered natural environments, Ross et al. 2013

What can go wrong?

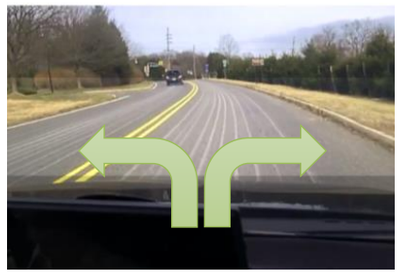
- Compounding errors
  Fix: data augmentation
- Stochastic expert actions
  Fix: stochastic latent variable models, action discretiation, gaussian mixture networks
- Non-markovian observations
  Fix: observation concatenation or recurrent models



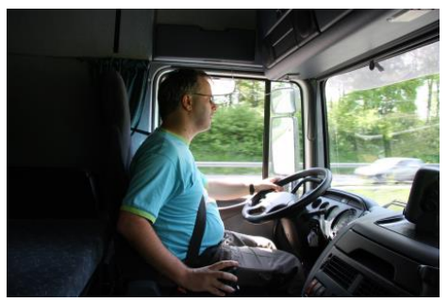
$\mathbf{o}_t$



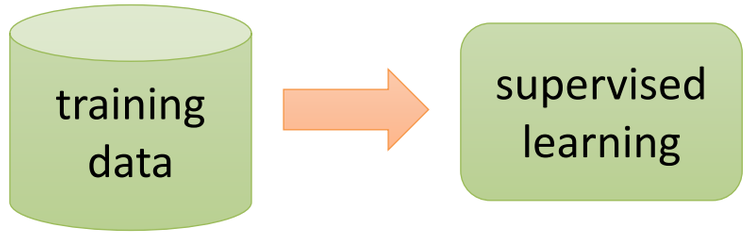
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$



$\mathbf{u}_t$



$\mathbf{o}_t$
$\mathbf{u}_t$



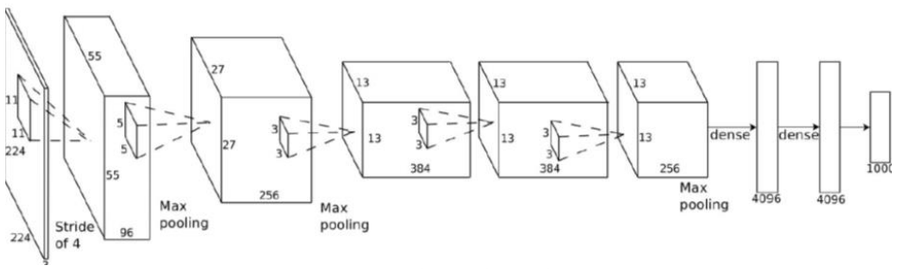
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

End to End Learning for Self-Driving Cars, Bojarski et al. 2016

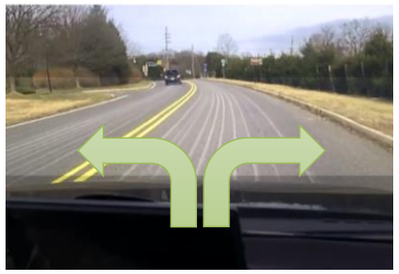
Non-markovian observations



$\mathbf{o}_t$



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



$\mathbf{u}_t$

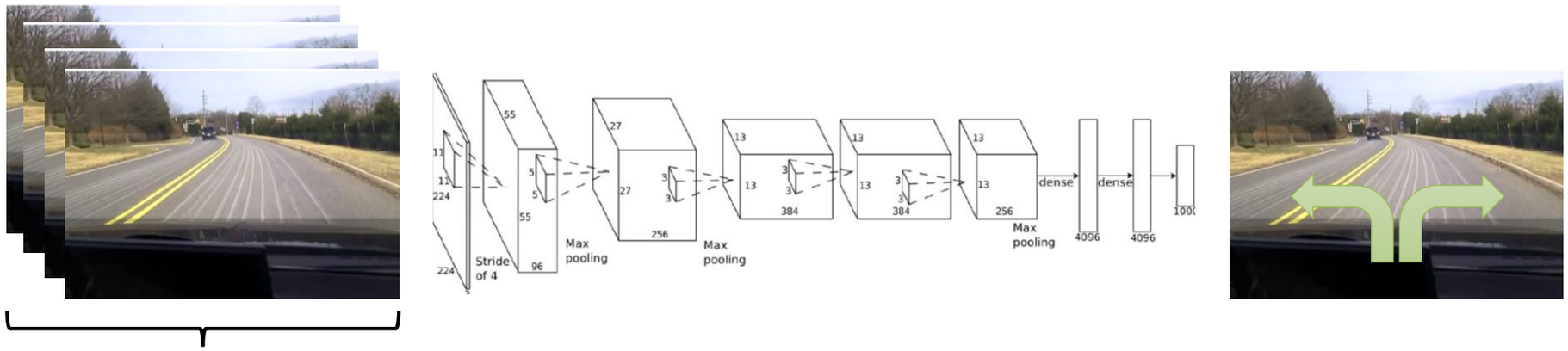
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

behavior depends only
on current observation

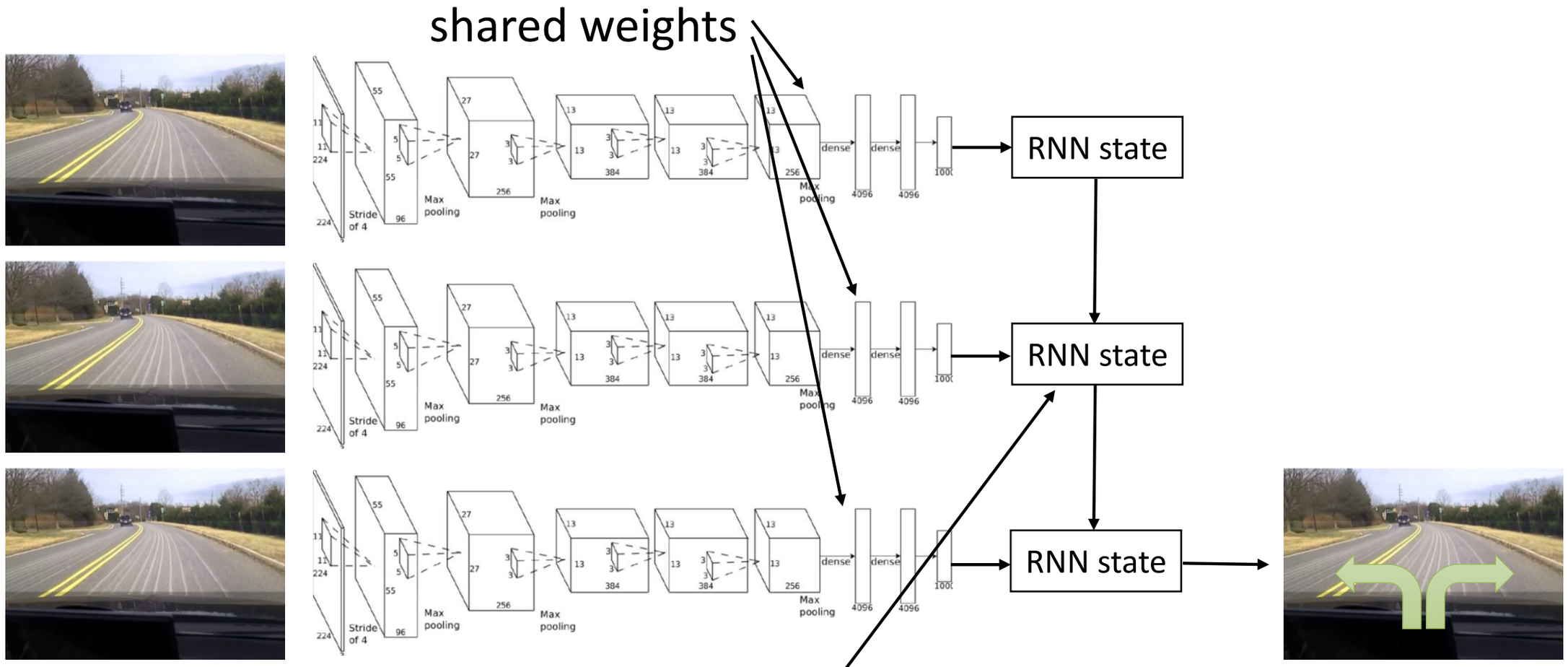
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_1, \ldots, \mathbf{o}_t)$

behavior depends on
all past observations

Fix 1: concatenate observations



Fix 2: use recurrent networks



Typically, LSTM cells work better here

Diagram from Sergey Levine

Recurrent Neural Networks (RNNs)

- RNNs tie the weights at each time step
- Condition the neural network on all previous inputs
- In principle, any interdependencies can be modeled across time steps.
- In practice, limitations from SGD training, capacity, initialization etc.

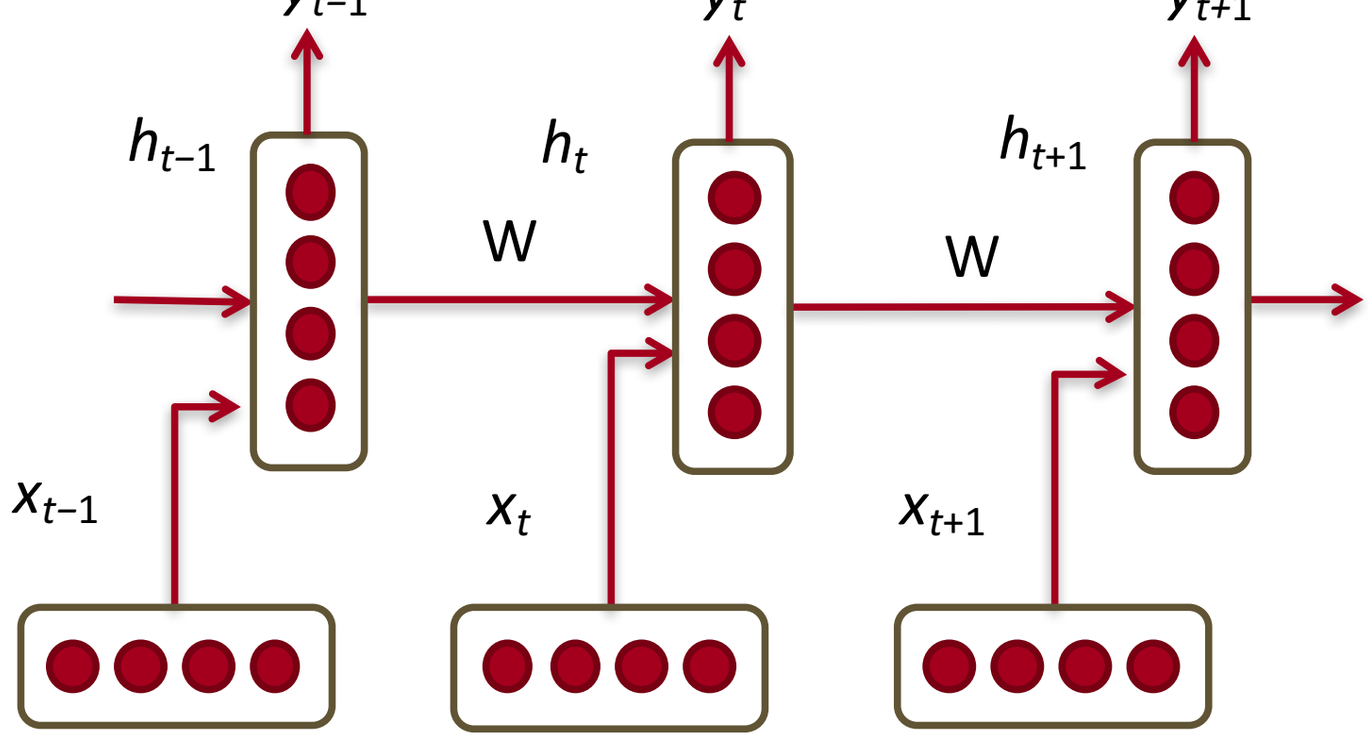


Diagram from Richard Socher

Recurrent Neural Network (single hidden layer)

- Given list of **vectors**: $x_1, ..., x_{t-1}, x_t, x_{t+1}, ..., x_T$
- At a single time step:

$$h_t = \sigma\left(W^{(hh)} h_{t-1} + W^{(hx)} x_{[t]}\right)$$

$$\hat{y}_t = \text{softmax}\left(W^{(S)} h_t\right)$$

(in case of discrete labels)

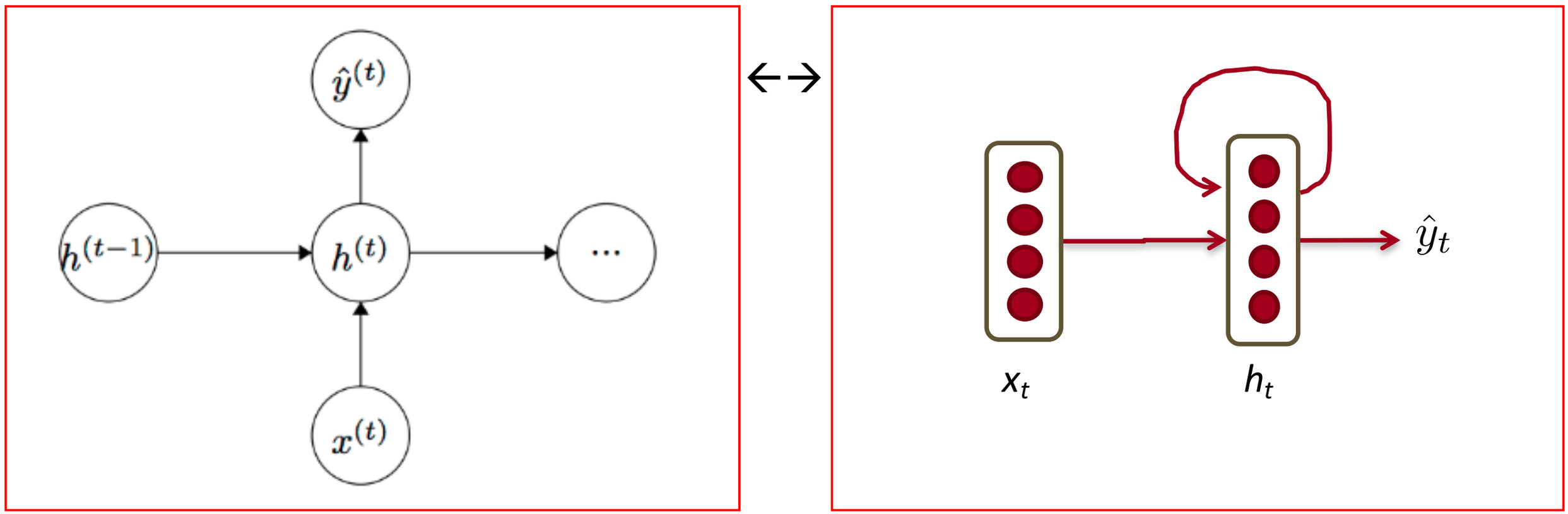
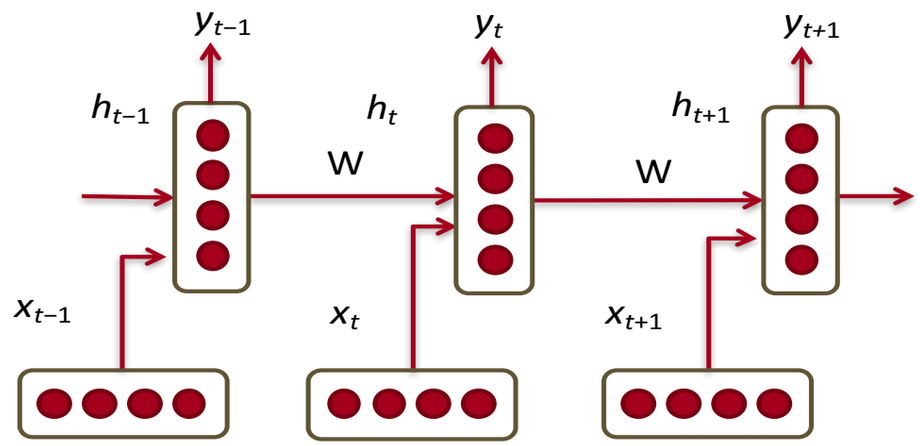


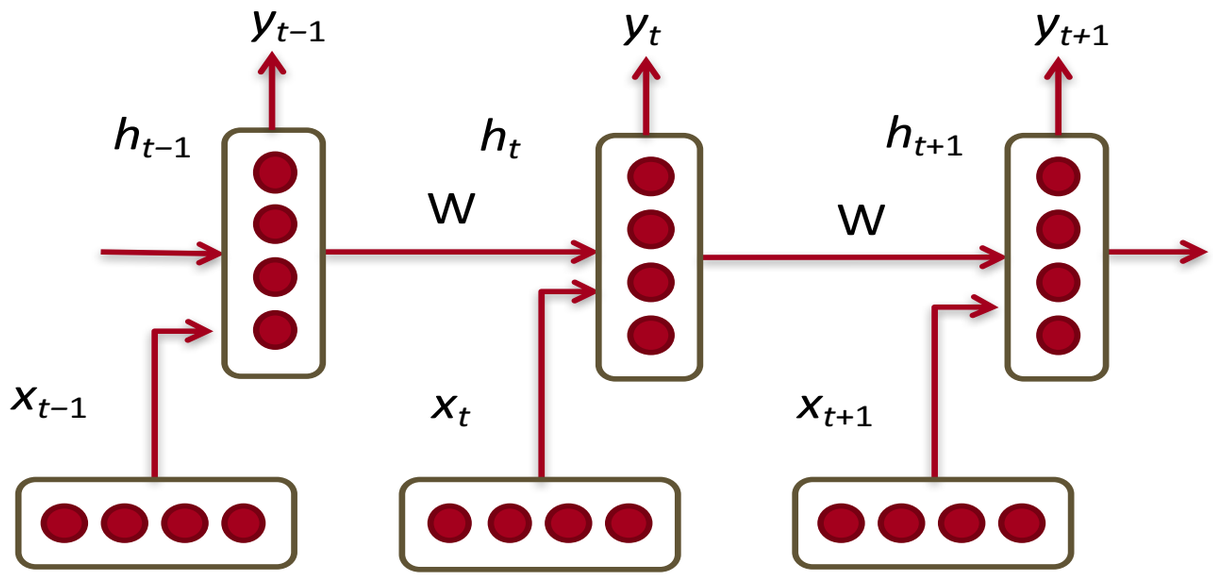
Diagram from Richard Socher

Recurrent Neural Networks

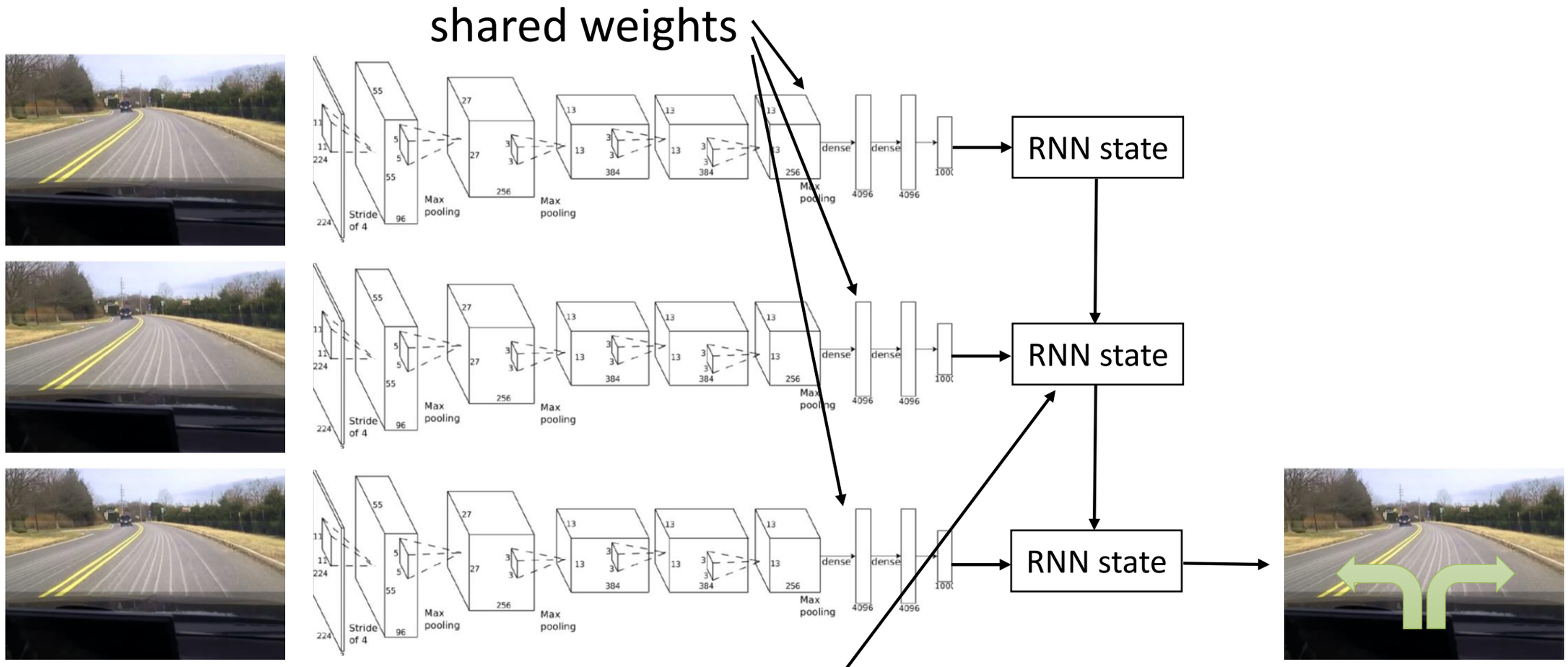
For sequence labelling problems, actions of the labelling policies are y_t, e.g., part of speech tags



For sequence generation, actions of the labelling policies are $y_t = x_{t+1}$, e.g., word in answer generation $\hat{P}(x_{t+1} = v_j | x_t, ..., x_1) = \hat{y}_{t,j}$



Fix 2: use recurrent networks

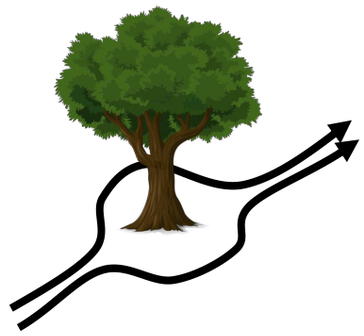


Typically, LSTM cells work better here

Diagram rom Sergey Levine

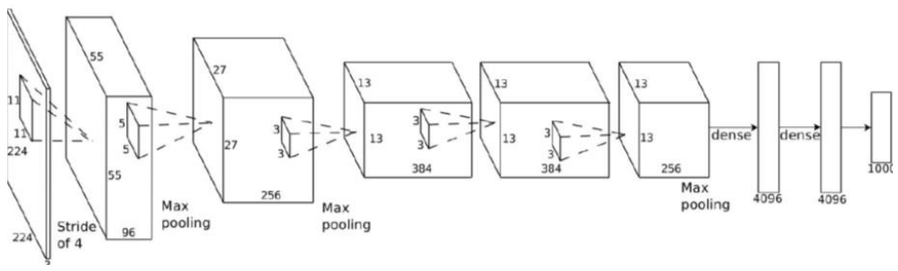
What can go wrong?

- Compounding errors
  - Fix: data augmentation
- Stochastic expert actions
  - Fix: stochastic latent variable models, action discretiation, gaussian mixture networks
- Non-markovian observations
  - Fix: observation concatenation or recurrent models

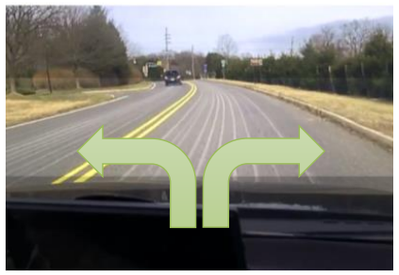




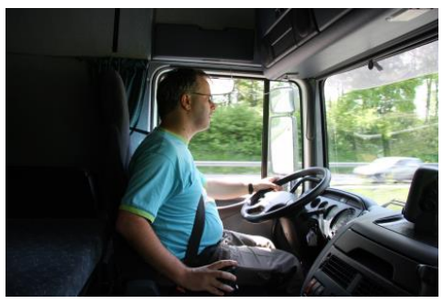
$\mathbf{o}_t$



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

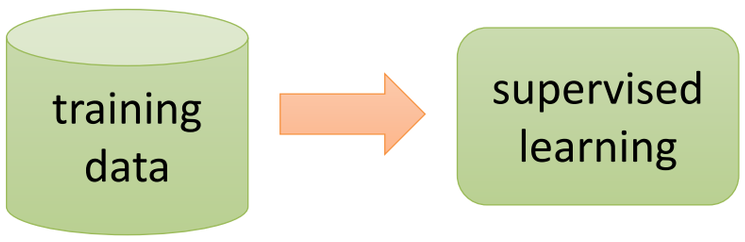


$\mathbf{u}_t$



$\mathbf{o}_t$

$\mathbf{u}_t$

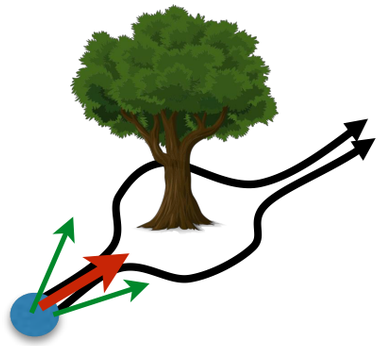


$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

End to End Learning for Self-Driving Cars, Bojarski et al. 2016

Regression fails under multimodality

The answer that minimizes the mean square error is the average which is not a valid prediction



groundtruth streering angles

predicted streering angles

Stochastic expert actions: Fixes

- Discretize the action space and use a classifier (e.g., softmax output and cross-entropy loss)
- Use gaussian mixture model as an output layer, mixture components weights, means and variances are parametrized at the output of a neural net, minimize GMM loss, (e.g., Handwriting generation Graves 2013)
- Stochastic neural networks (later lecture)

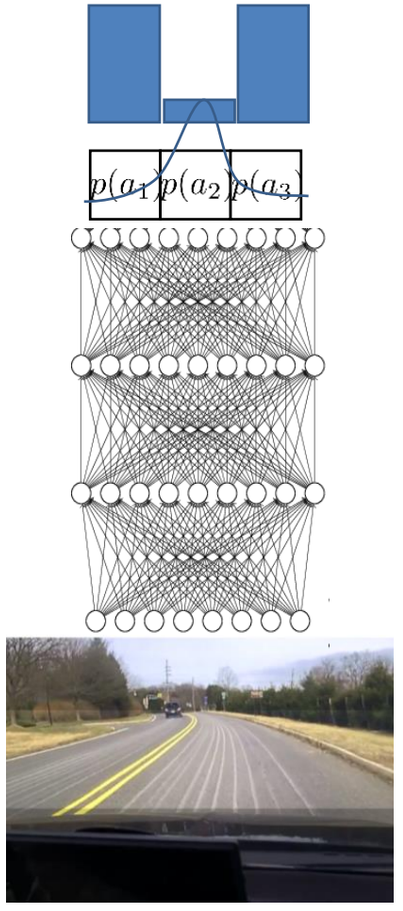
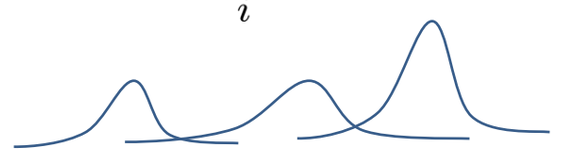


Diagram from Sergey Levine

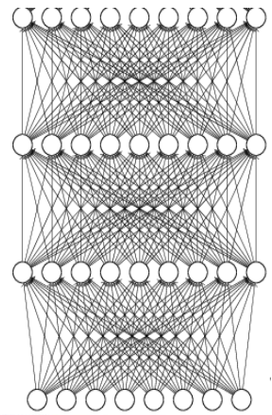
Stochastic expert actions: Fixes

- Discretize the action space and use a classifier (e.g., softmax output and cross-entropy loss)
- Use gaussian mixture model as an output layer, mixture components weights, means and variances are parametrized at the output of a neural net, minimize GMM loss, (e.g., Handwriting generation Graves 2013)
- Stochastic neural networks (later lecture)

$$\pi(\mathbf{a}|\mathbf{o}) = \sum_i w_i \mathcal{N}(\mu_i, \Sigma_i)$$



$$w_1, \mu_1, \Sigma_1, \ldots, w_N, \mu_N, \sigma_N$$



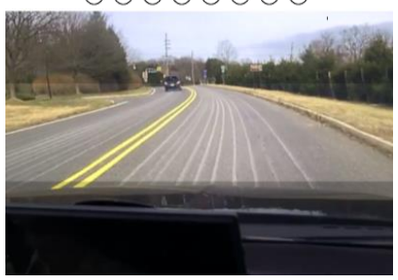


Diagram from Sergey Levine

Deep Imitation Learning for Complex Manipulation Tasks from Virtual Reality Teleoperation

Tianhao Zhang*[12], Zoe McCarthy*[1], Owen Jow[1], Dennis Lee[1], Xi Chen[12], Ken Goldberg[1], Pieter Abbeel[1-4]

***Abstract*— Imitation learning is a powerful paradigm for robot skill acquisition. However, obtaining demonstrations suitable for learning a policy that maps from raw pixels to actions can be challenging. In this paper we describe how consumer-grade Virtual Reality headsets and hand tracking hardware can be used to naturally teleoperate robots to perform complex tasks. We also describe how imitation learning can learn deep neural network policies (mapping from pixels to actions) that can acquire the demonstrated skills. Our experiments showcase the effectiveness of our approach for learning visuomotor skills.**

I. Introduction

Imitation learning is a class of methods for acquiring skills by observing demonstrations (see, e.g., [1], [2], [3] for surveys). It has been applied successfully to a wide range of domains in robotics, for example to autonomous driving [4], [5], [6], autonomous helicopter flight [7], gesturing [8],



Fig. 1: Virtual Reality teleoperation in action

2017

https://www.youtube.com/embed/kZlg0QvKkQQ?mute=1&rel=0&enablejsapi=1

https://www.youtube.com/embed/OD2UbZNw9sQ?enablejsapi=1

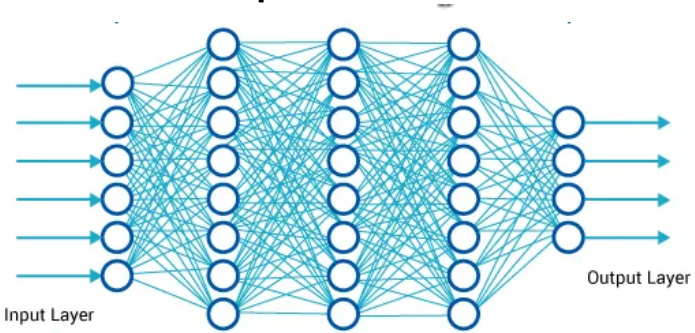
https://www.youtube.com/embed/XHIWa1LRgbI?enablejsapi=1


Stanford
ARTIFICIAL
INTELLIGENCE
twitter: @tonyzzhao

Visuomotor policies

perception network

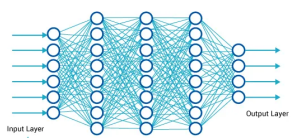
(often pre-trained)



$\mathbb{R}^{640 \times 480 \times 3}$ x history

z **learned state representation**

policy network



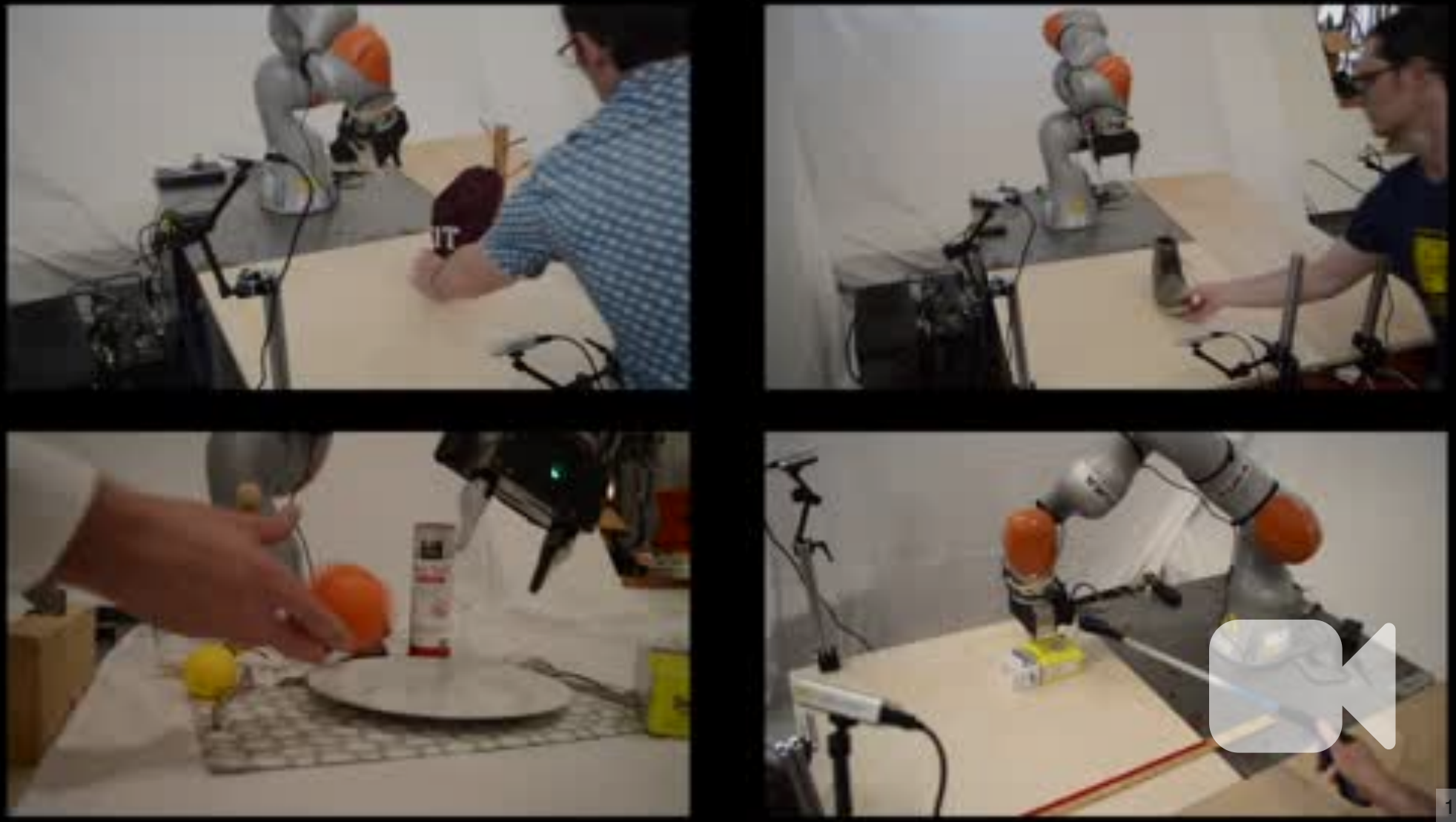
$\approx \mathbb{R}^7$

actions

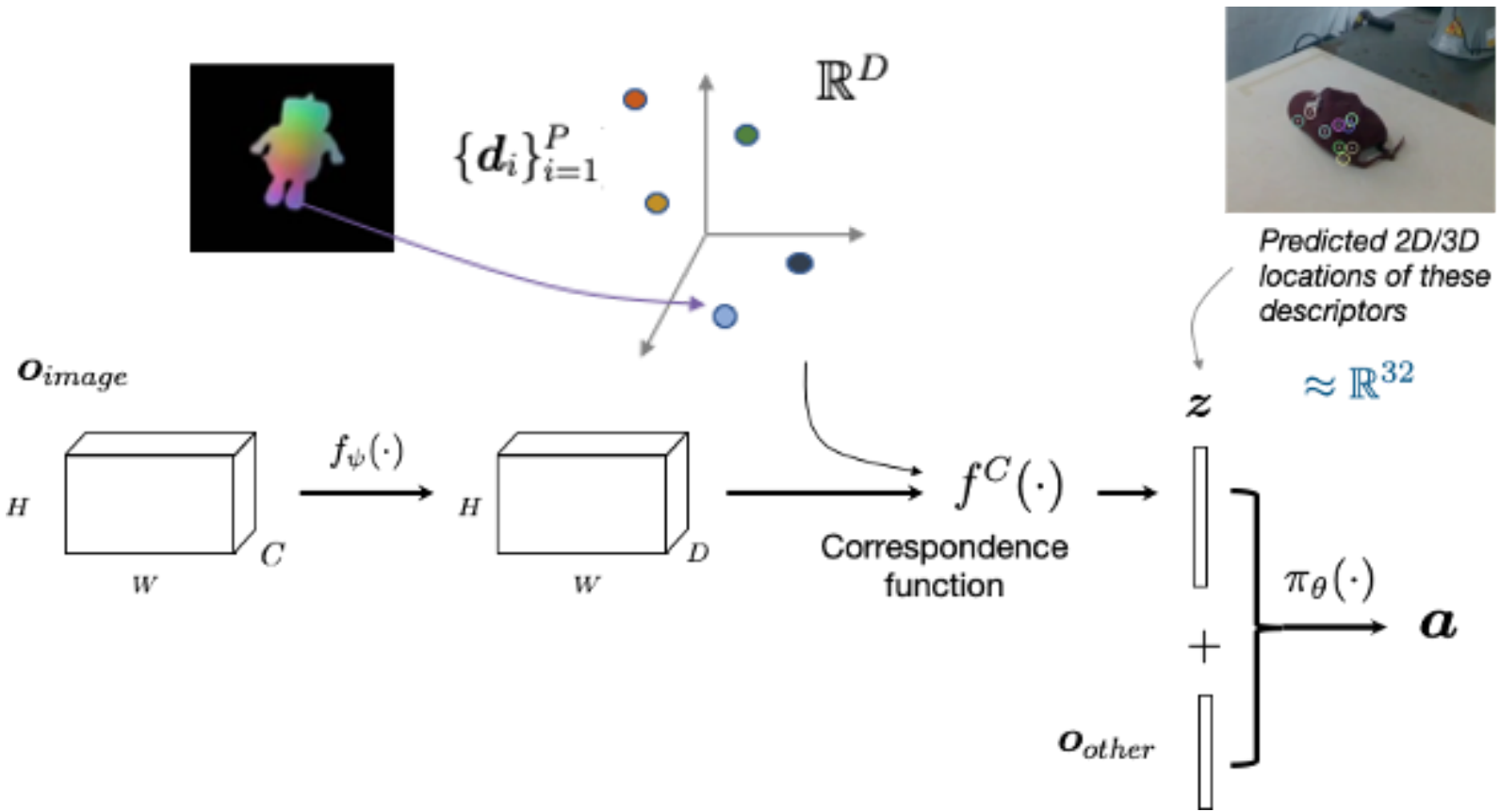
other robot sensors

$\approx \mathbb{R}^{13}$

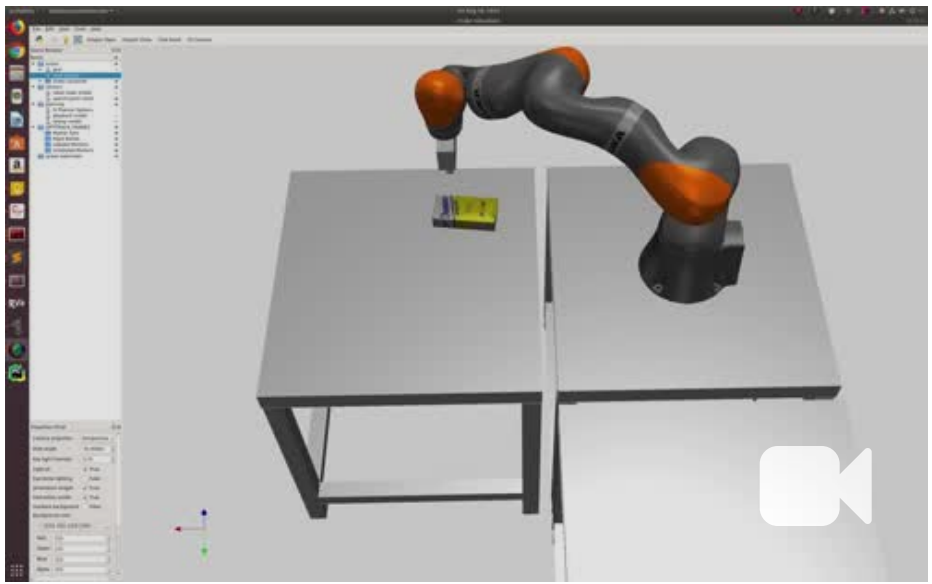
Levine*, Finn*, Darrel, Abbeel, JMLR 2016



Idea: Use small set of dense descriptors



Imitation learning setup



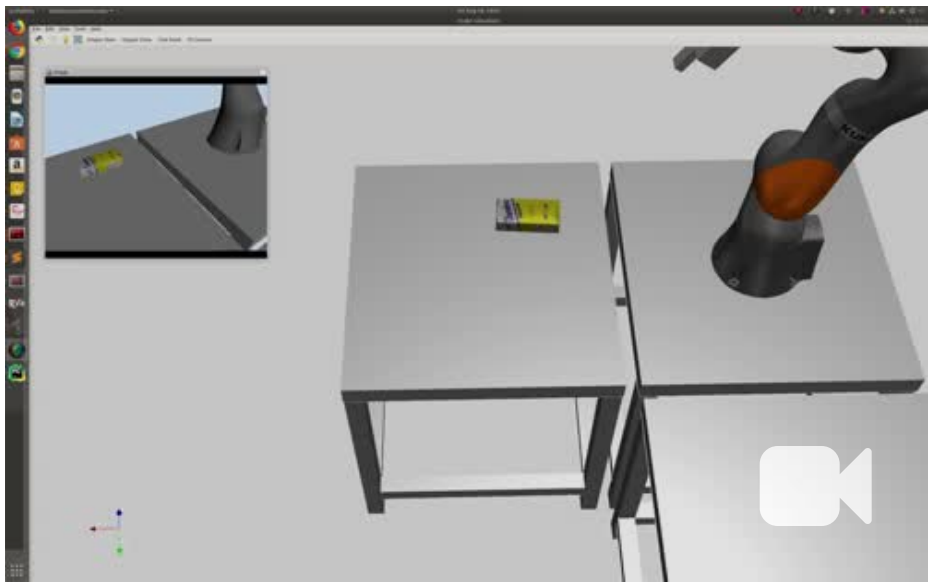
from hand-coded policies in sim



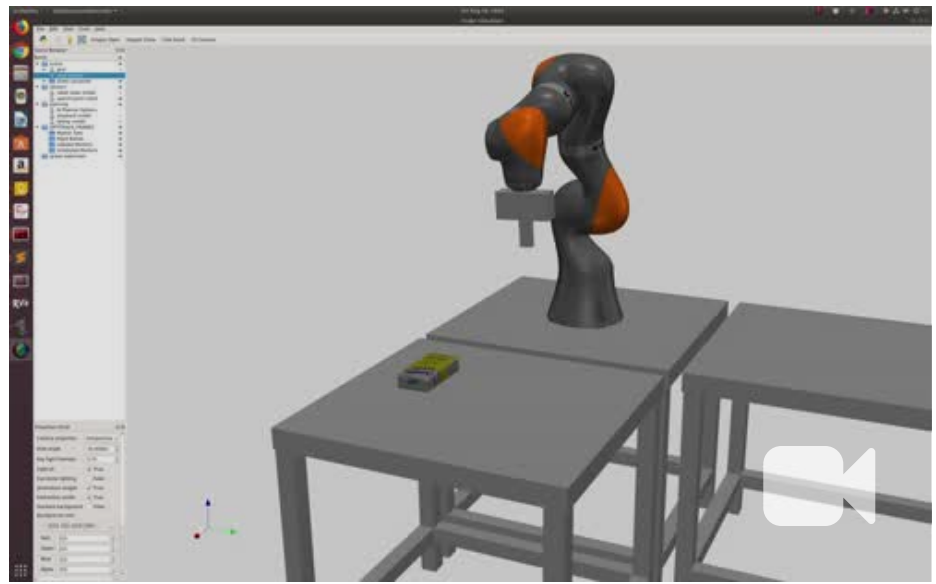
and teleop on the real robot

Standard "behavior-cloning" objective + data augmentation

Simulation experiments

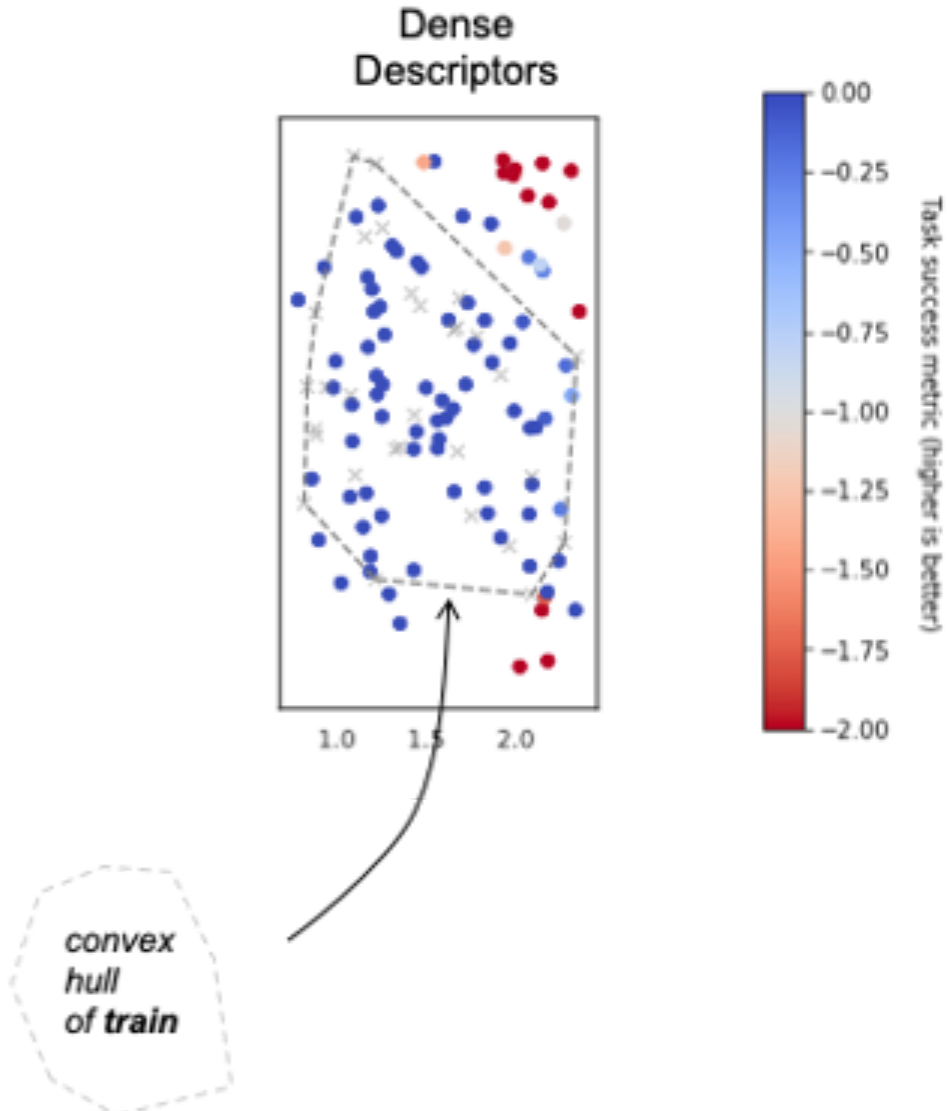


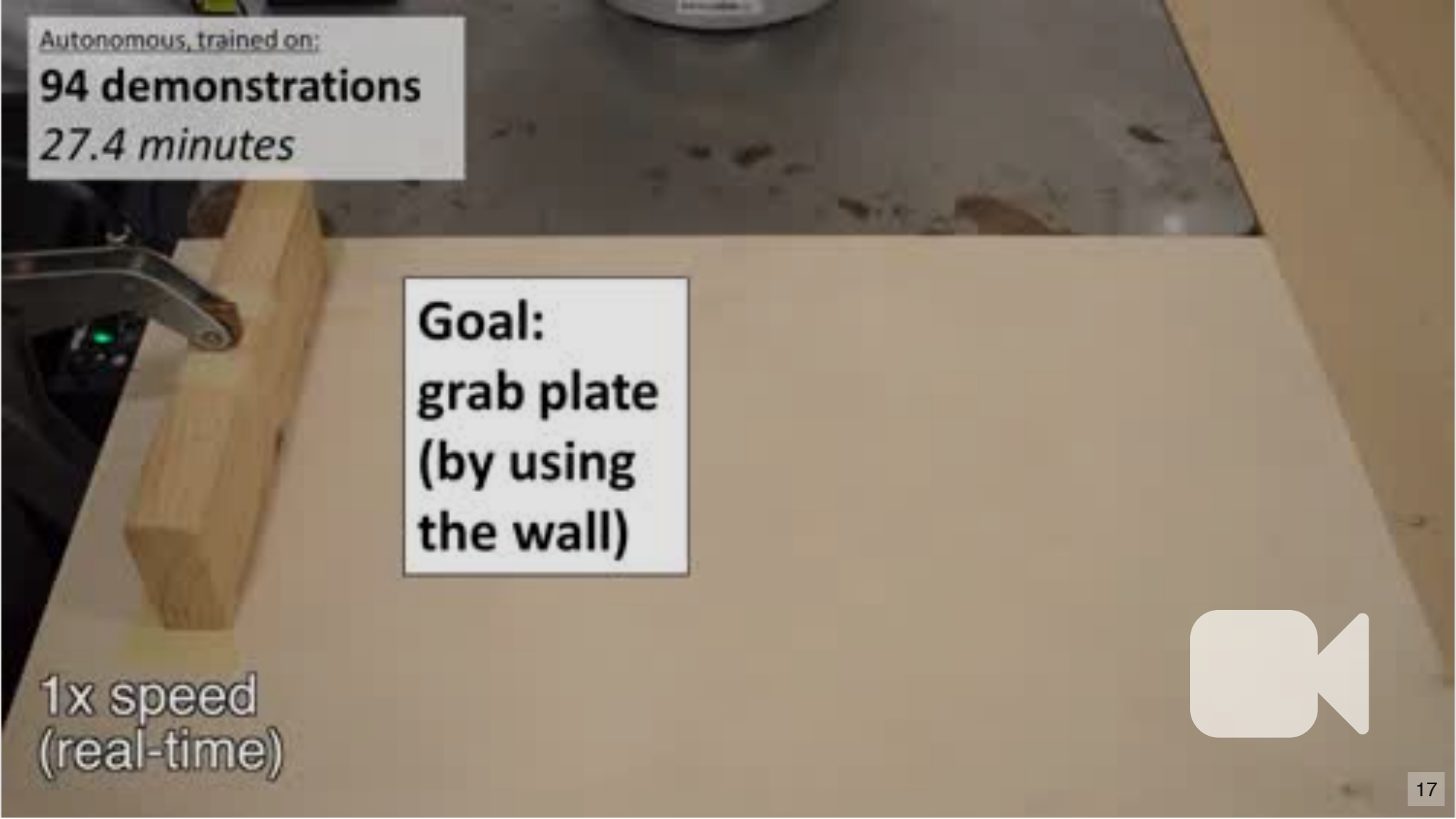
"push box"



"flip box"

Policy is a small LSTM network (~100 LSTMs)


Dense
Descriptors
0.00
−0.25
−0.50
−0.75
−1.00
−1.25
−1.50
−1.75
−2.00
Task success metric (higher is better)
1.0
1.5
2.0
convex
hull
of train


Autonomous, trained on:
94 demonstrations
27.4 minutes
Goal:
grab plate
(by using
the wall)
1x speed
(real-time)



Vincent Vanhoucke

Oct 27 · 4 min read · Member-only · Listen

Shifting Winds in Robot Learning Research

Reinforcement Learning is dead, long live Reinforcement Learning!

"And then ... BC methods started to get good. Really good. So good that our best manipulation system today mostly uses BC, with a sprinkle of Q learning on top to perform high-level action selection. Today, less than 20% of our research investments is on RL, and the research runway for BC-based methods feels more robust."

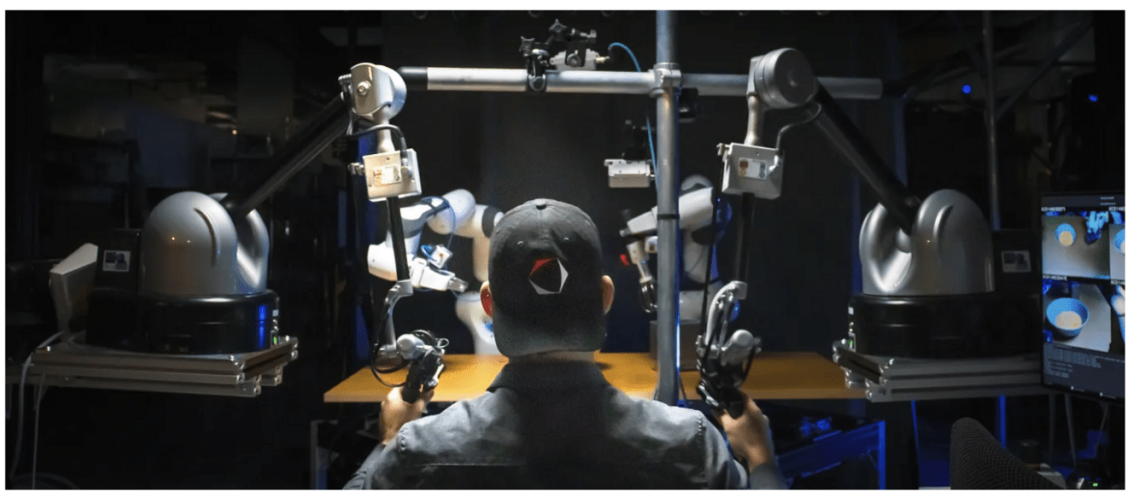
TRI's Robots Learn New Manipulation Skills in an Afternoon. Here's How.

By: Siyuan Feng, Ben Burchfiel, Toffee Albina, and Russ Tedrake

Toyota Research Institute · Follow

Published in Toyota Research Institute · 11 min read · Sep 14

156 2



Medium blog post

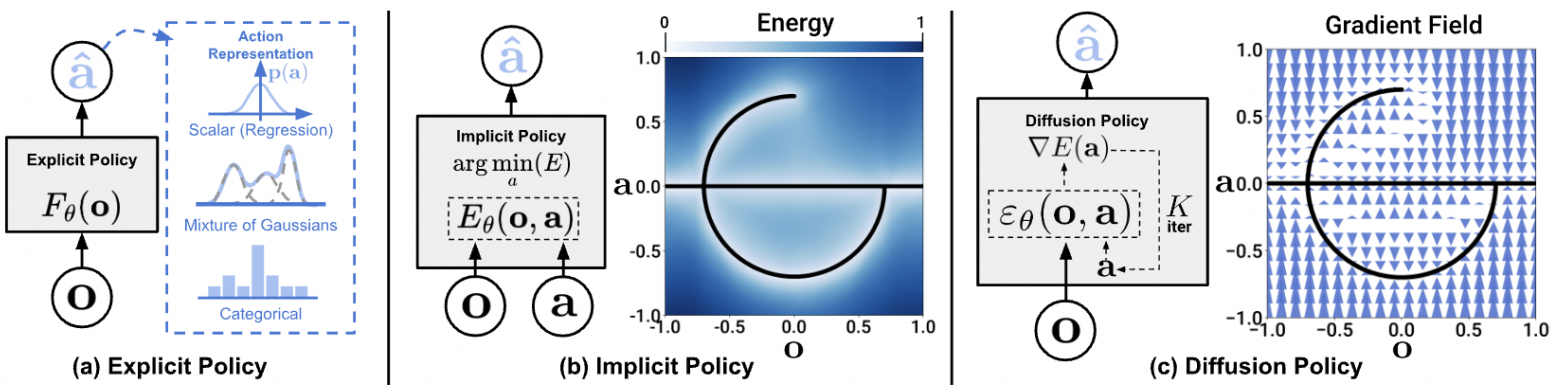
COLUMBIA ENGINEERING

TOYOTA RESEARCH INSTITUTE

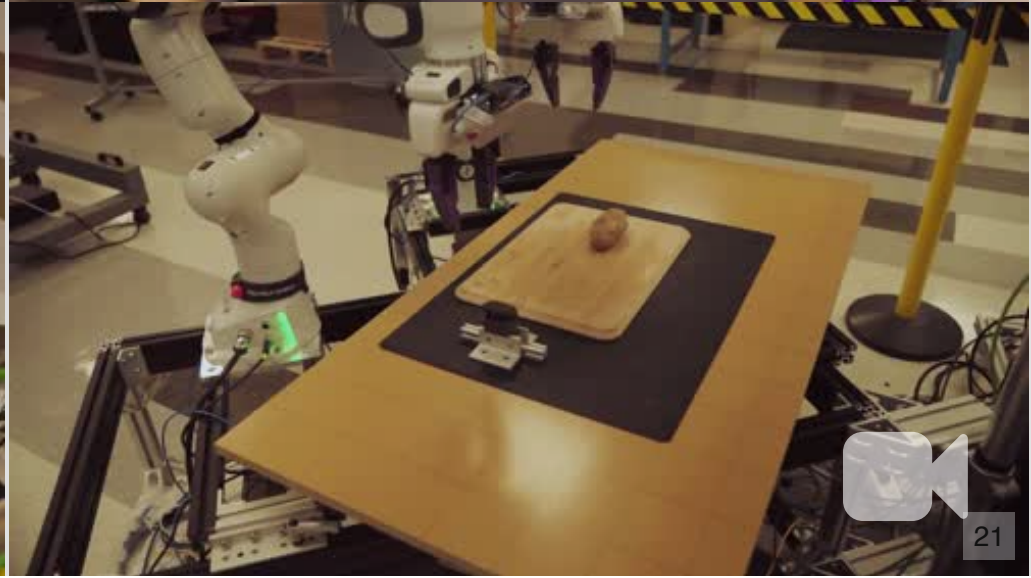
Massachusetts Institute of Technology

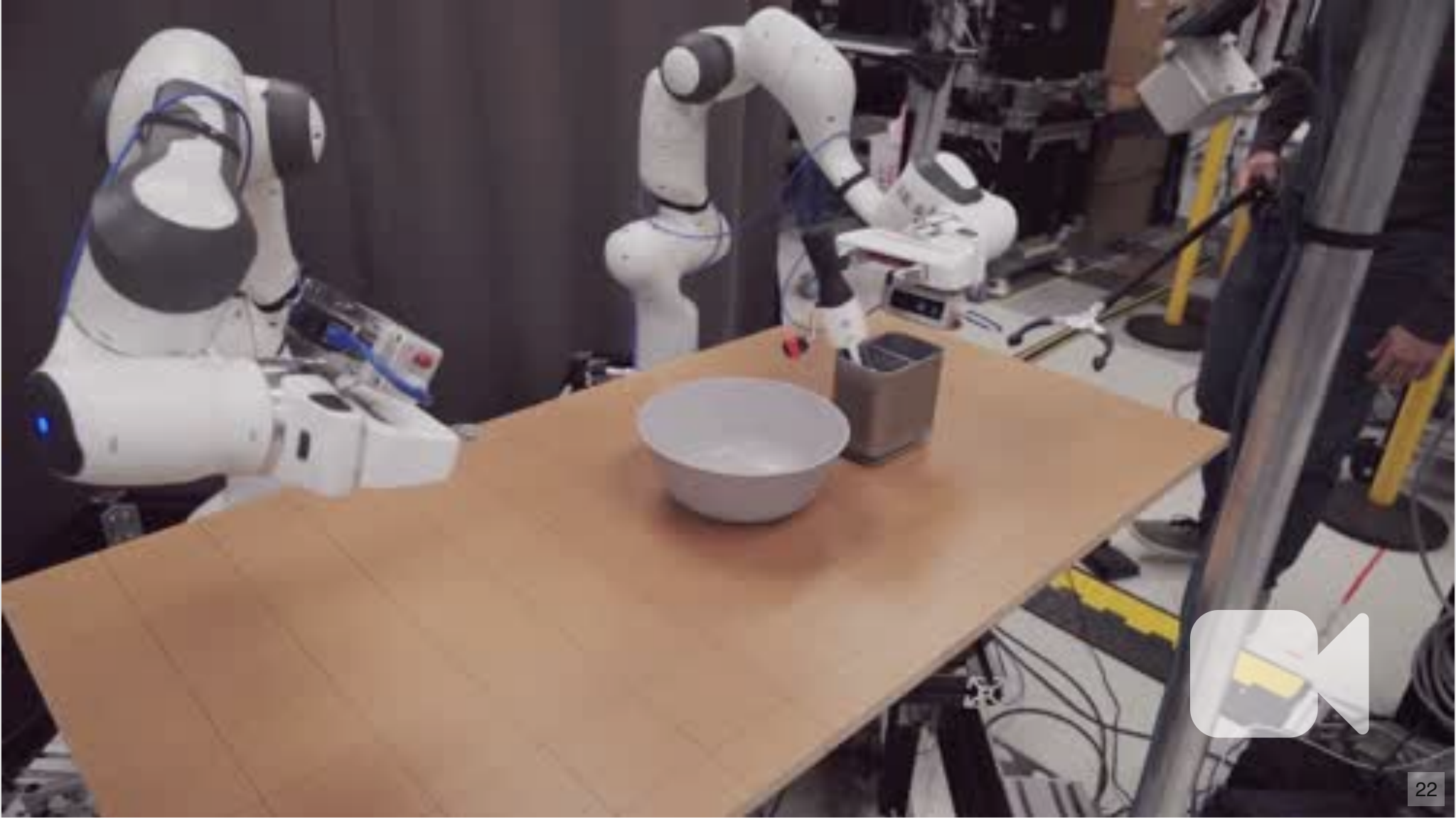
Diffusion Policy

Visuomotor Policy Learning via Action Diffusion

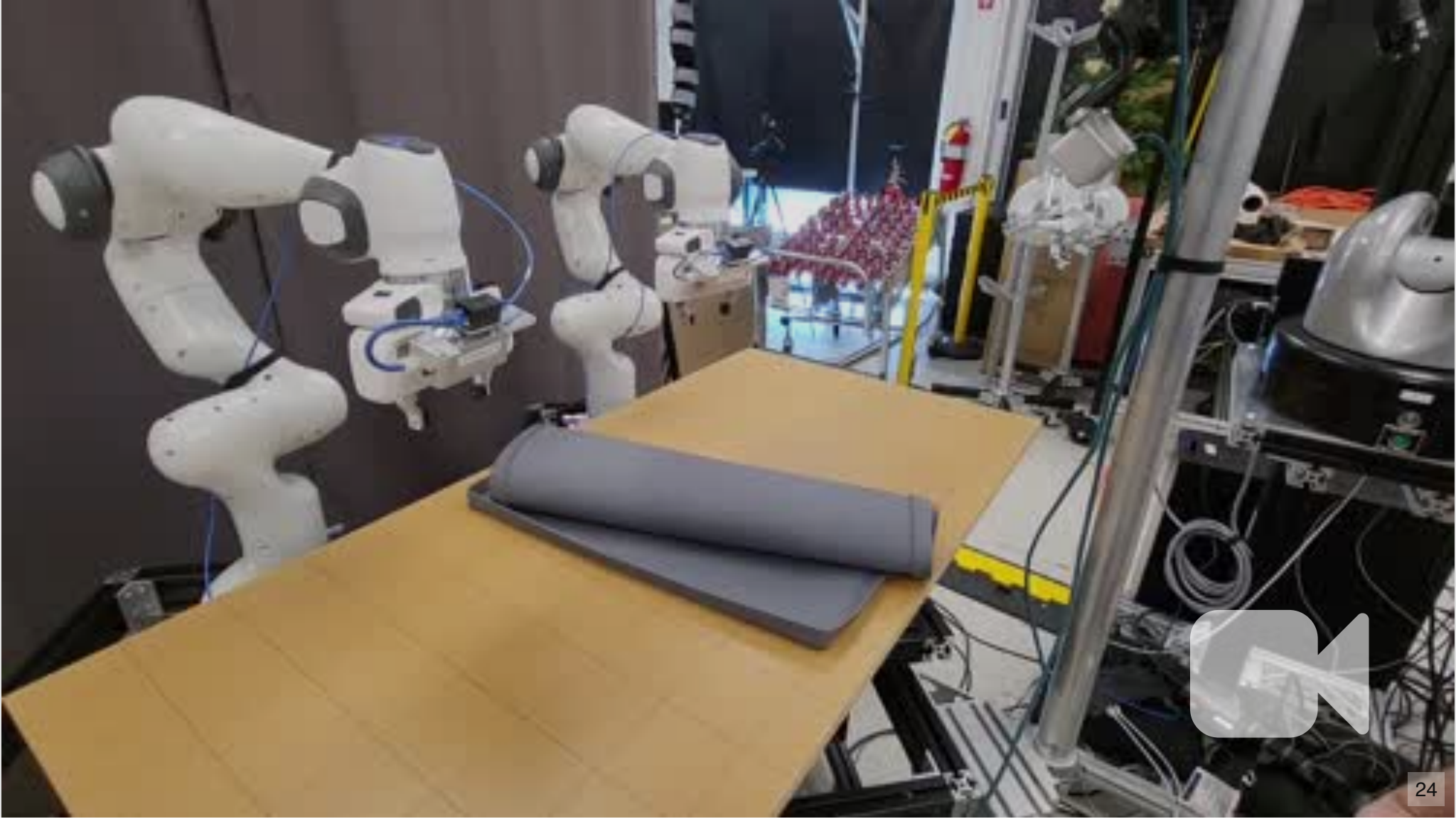










Denoising diffusion models (generative AI)

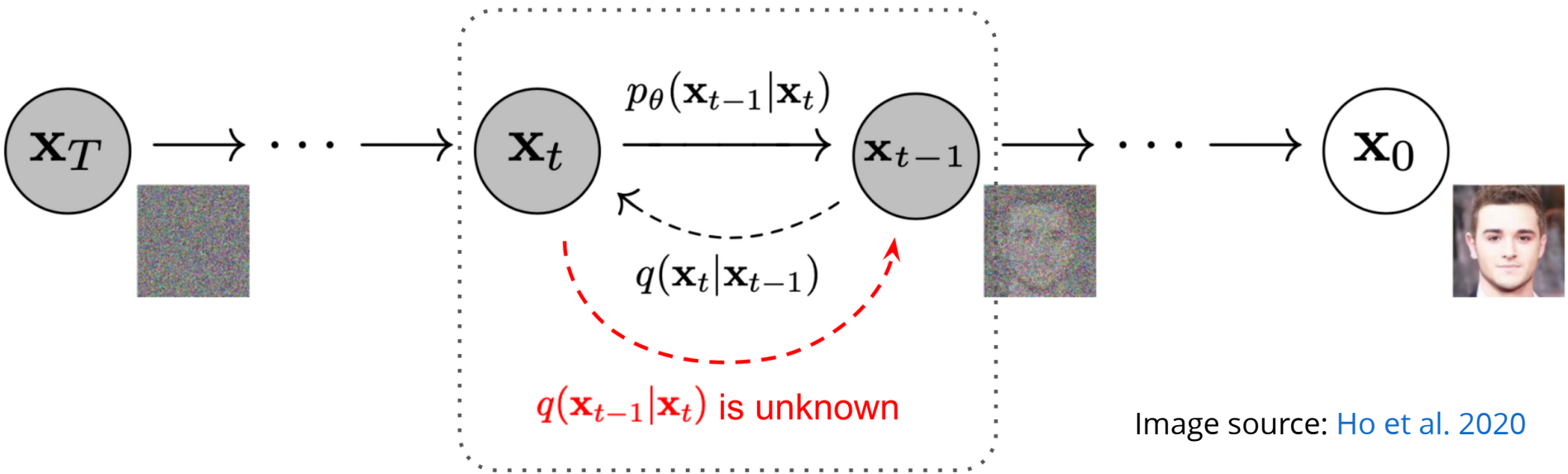


Image source: Ho et al. 2020

Denoiser can be conditioned on additional inputs, u: $p_\theta(x_{t-1}|x_t, u)$

A derministic interpretation (manifold hypothesis)

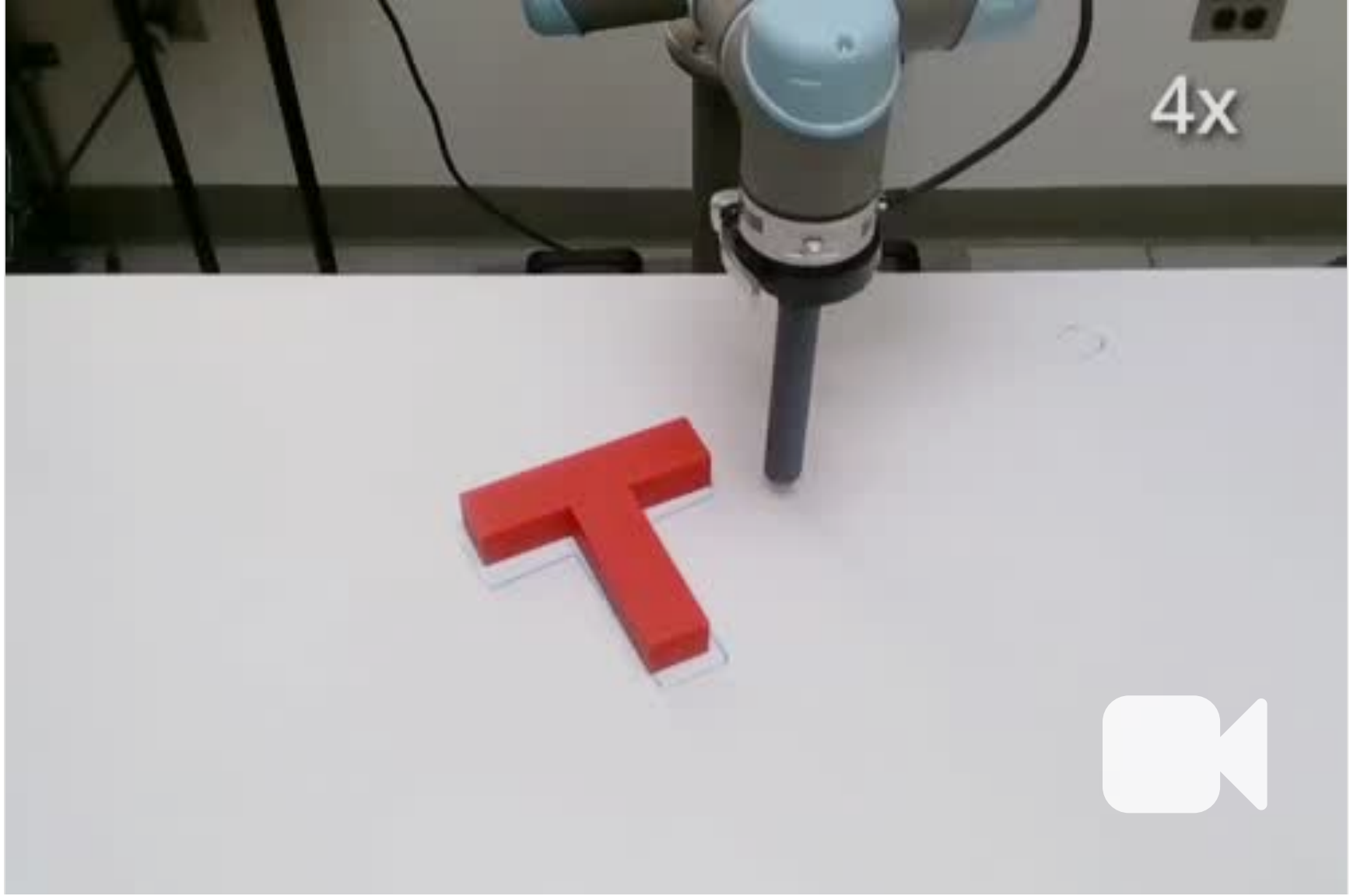
Denoising approximates the *projection onto the data manifold*;
approximating the gradient of the distance to the manifold

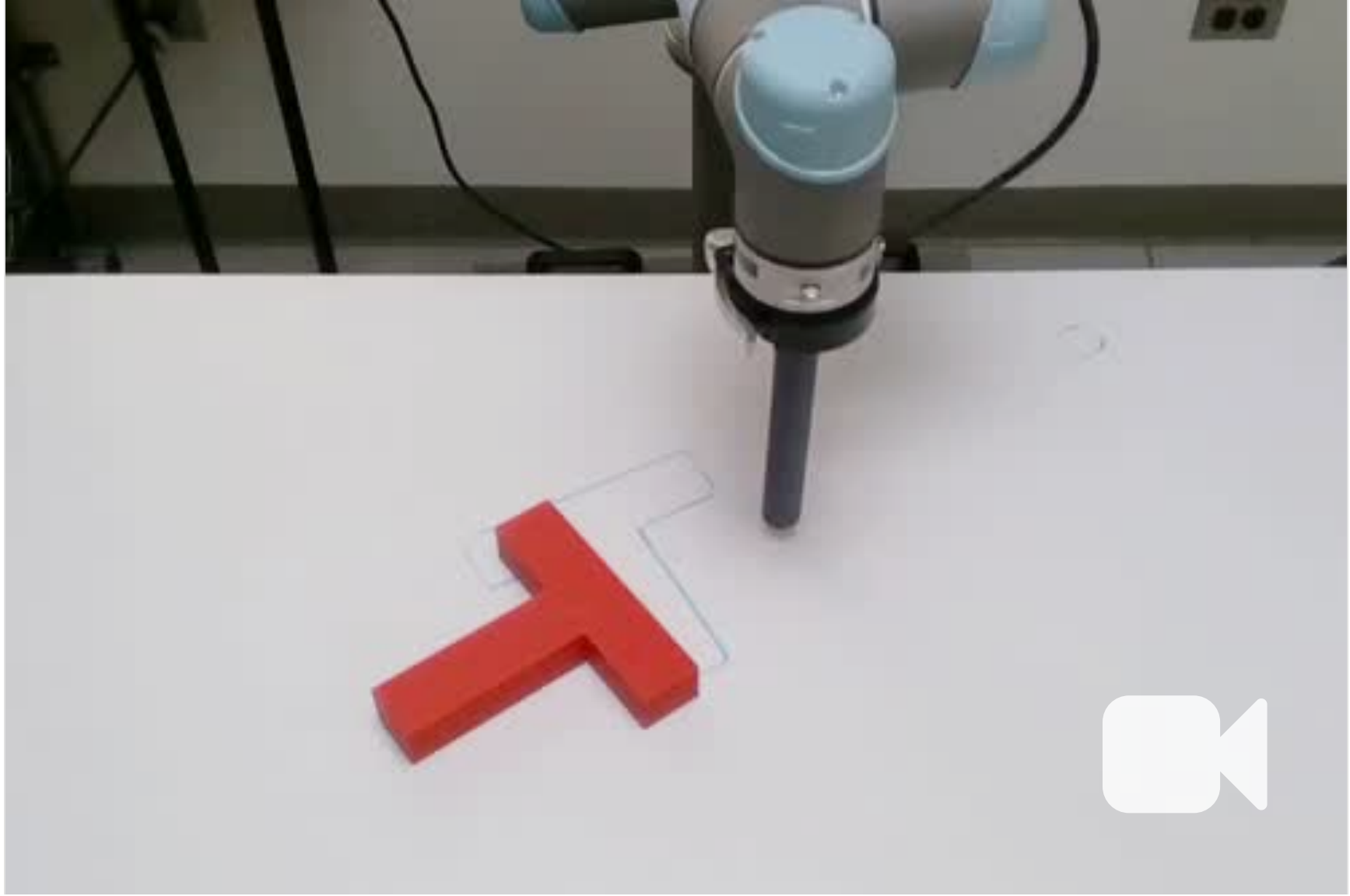
Interpreting and Improving Diffusion Models Using the Euclidean Distance Function

---

Frank Permenter
Toyota Research Institute
Cambridge, MA 02139
frank.permenter@tri.global

Chenyang Yuan
Toyota Research Institute
Cambridge, MA 02139
chenyang.yuan@tri.global

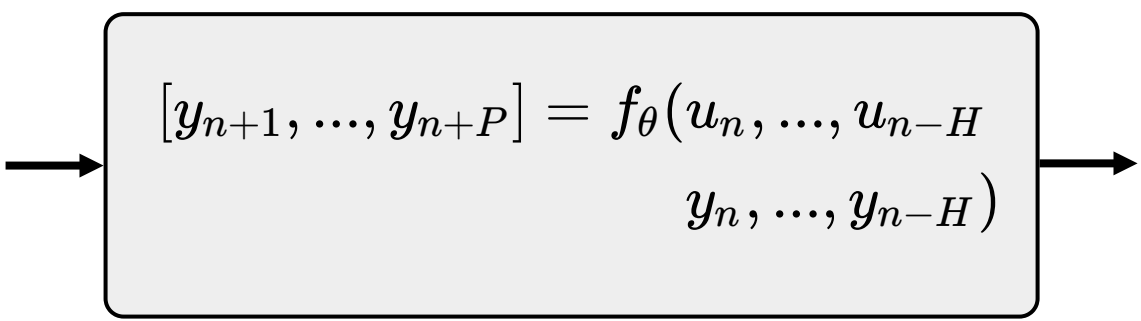

4x



Representing dynamic output feedback



"Diffusion Policy" is an auto-regressive (ARX) model with *forecasting*



H is the length of the history,

P is the length of the prediction

Conditional denoiser produces the forecast, conditional on the history

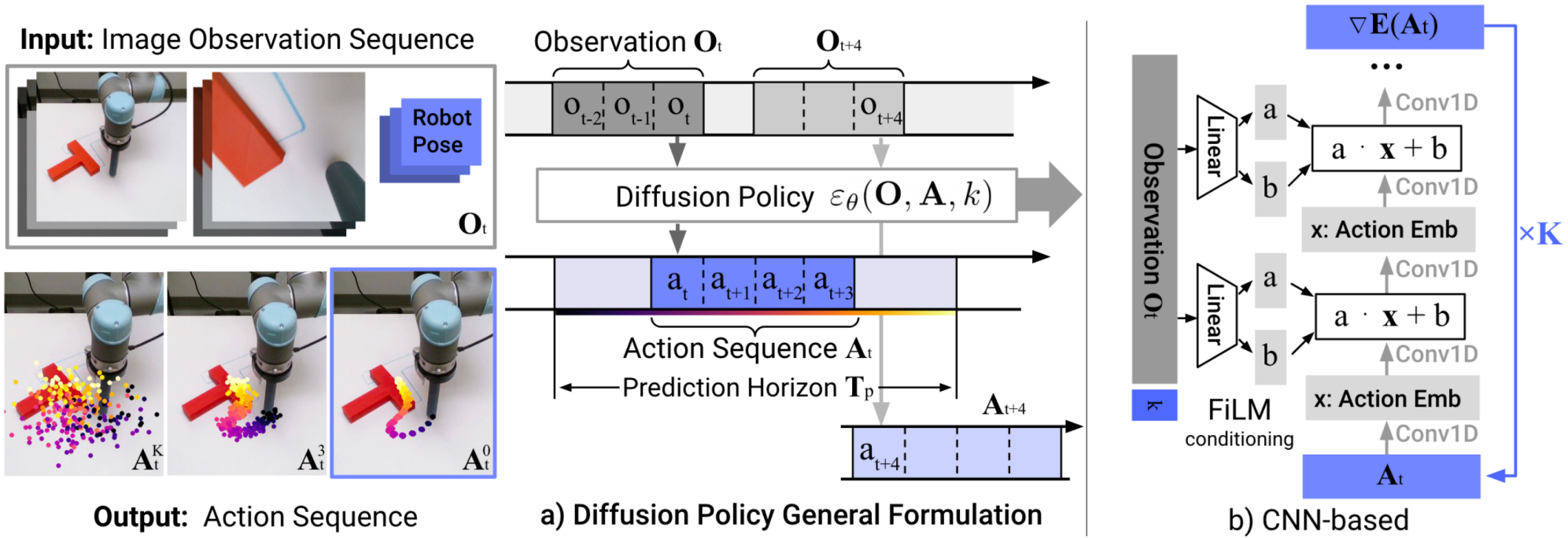
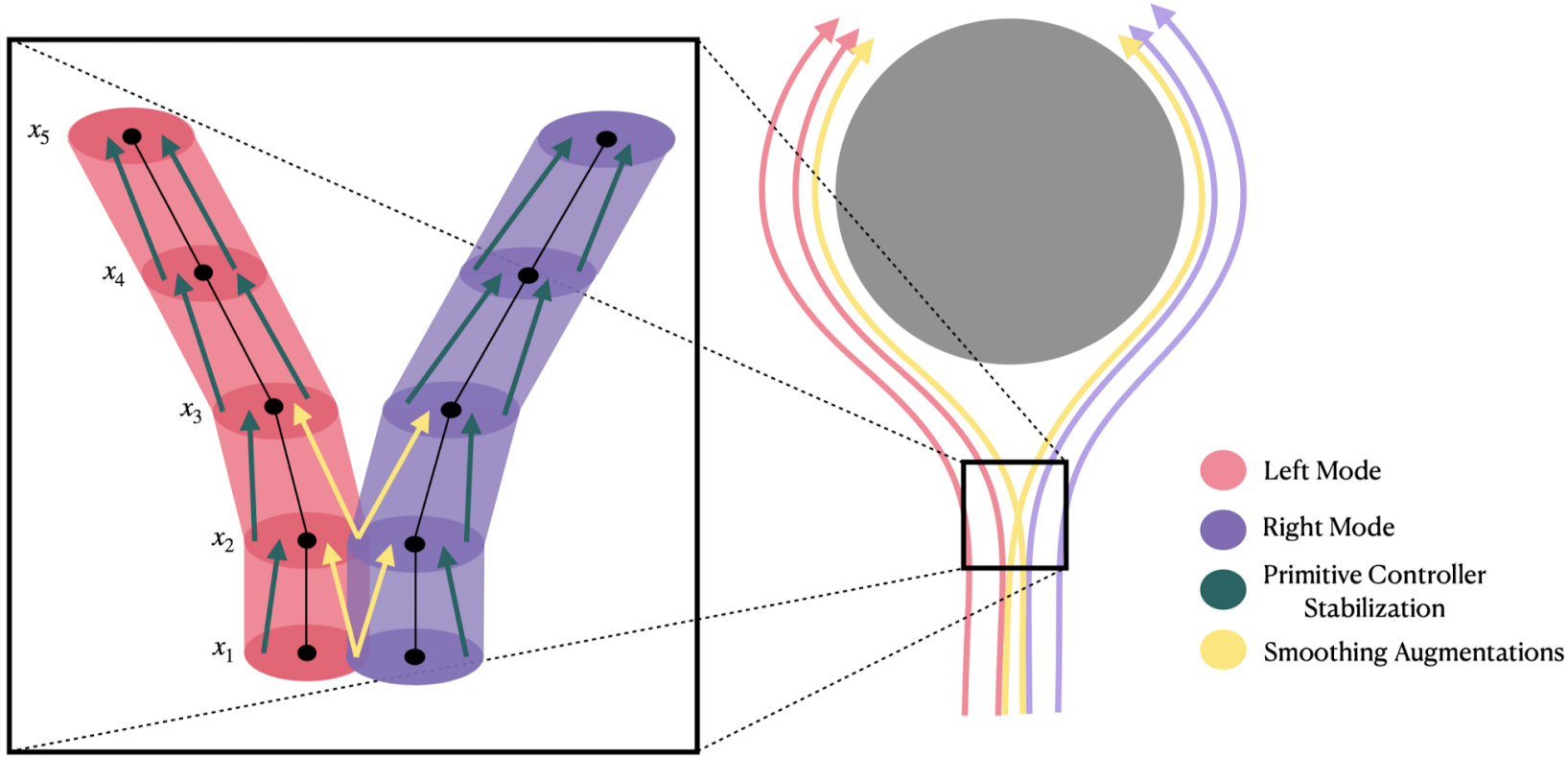


Image backbone: ResNet-18 (pretrained on ImageNet)
Total: 110M-150M Parameters
Training Time: 3-6 GPU Days ($150-$300)

Learns a distribution (score function) over actions

e.g. to deal with "multi-modal demonstrations"

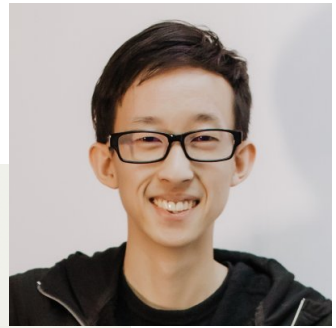




Provable Guarantees for Generative Behavior Cloning: Bridging Low-Level Stability and High-Level Behavior

Adam Block* Ali Jadbabaie Daniel Pfrommer† Max Simchowitz‡ Russ Tedrake

Andy Zeng's MIT CSL Seminar, April 4, 2022



"Dirty Laundry"

Symptoms of a larger problem

The not-so-secret recipe to making a rockstar behavior cloning demo on **real** robots

Step 1. collect your own "expert" data and don't trust anyone else to make it perfect

Step 2. avoid "no action" data so your policy doesn't just sit there

Step 3. It's not working? Collect more data until "extrapolation" becomes "interpolation"

Step 4. Train and test on the same day because your setup might change tomorrow

Mostly because we don't have a lot of data

Andy's slides.com presentation

Why (Denoising) Diffusion Models?

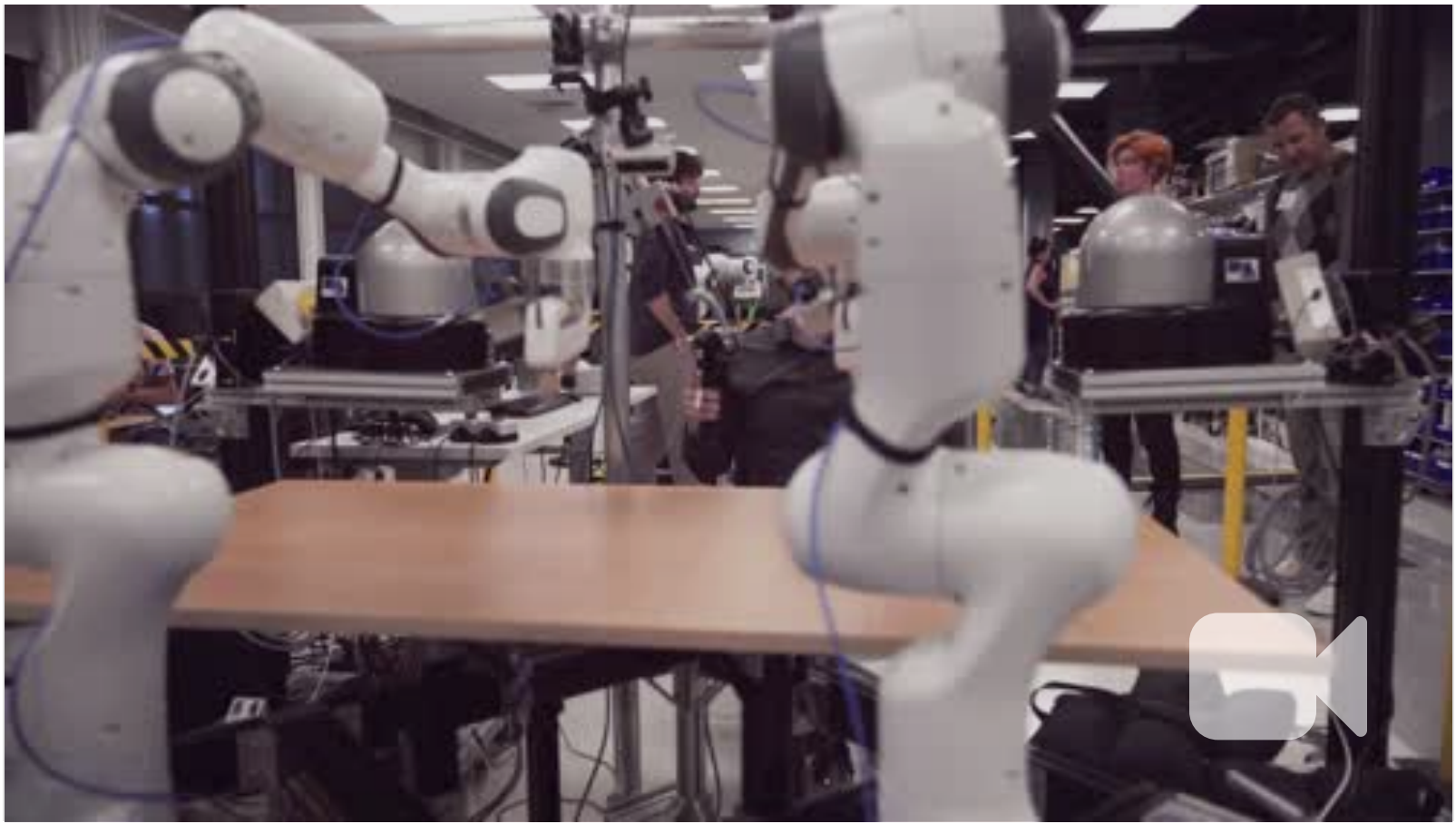
- High capacity + great performance
- Small number of demonstrations (typically ~50)
- Multi-modal (non-expert) demonstrations
- Training stability and consistency
  - no hyper-parameter tuning
- Generates high-dimension *continuous* outputs
  - vs categorical distributions (e.g. RT-1, RT-2)
  - Action-chunking transformers (ACT)
- Solid mathematical foundations (score functions)
- Reduces nicely to the simple cases (e.g. LQG / Youla)

Enabling technologies

Haptic Teleop Interface



Excellent system identification / robot control



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

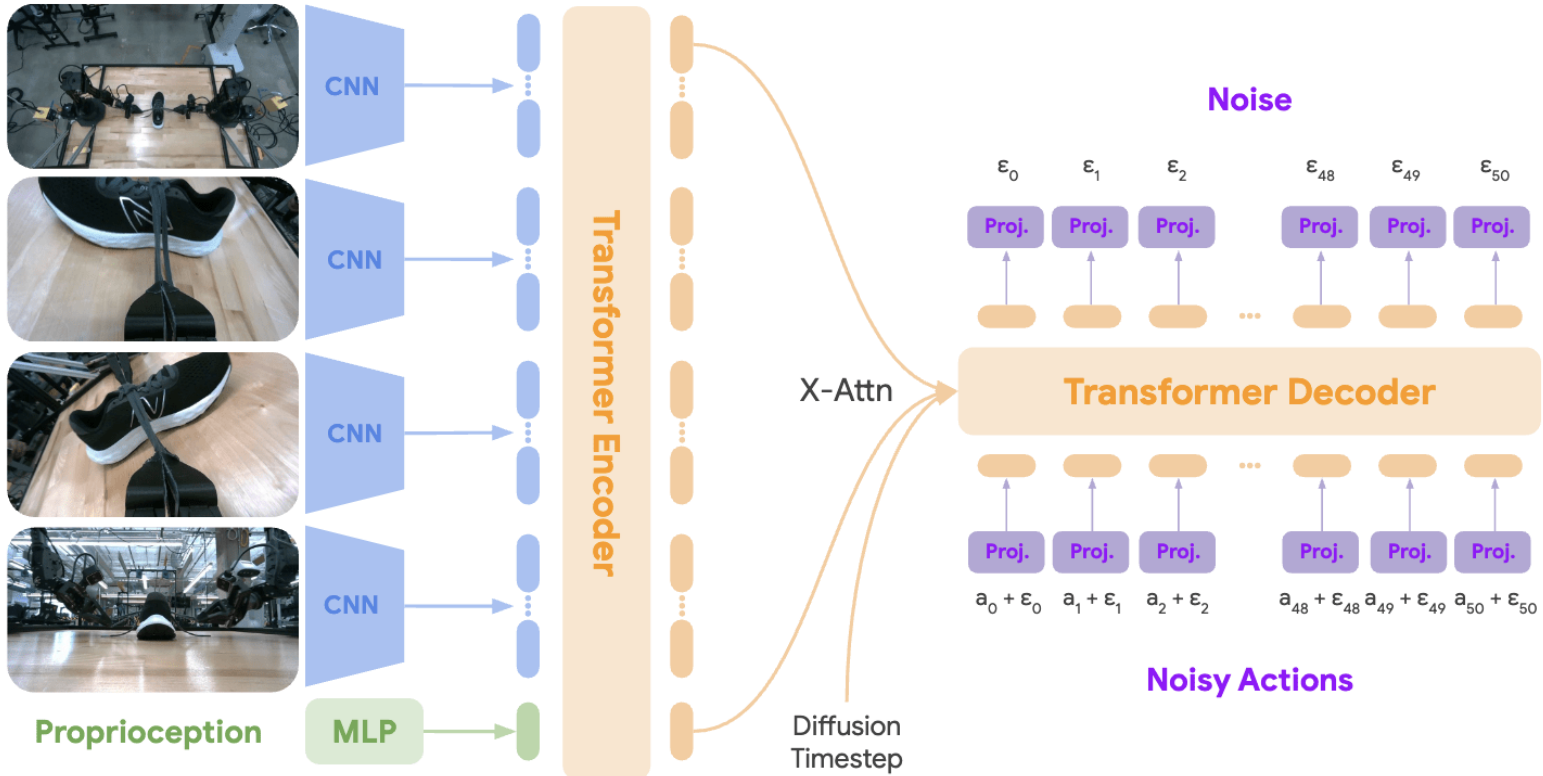


Figure 2: ***Policy Architecture.*** Each camera view is embedded with a ResNet50 [46]. The Transformer Encoder embeds the observations and produces latent embeddings. The Transformer Decoder takes in noisy actions and cross-attends to the latent embeddings produced by the encoder, outputting the predicted noise. The decoder portion runs 50 times during inference to iteratively denoise an action chunk.

https://aloha-unleashed.github.io/assets/LaceMessy.mp4

https://aloha-unleashed.github.io/assets/KitchenStack4x.mp4

https://aloha-unleashed.github.io/assets/ShirtRed.mp4

Training details. We train our models with JAX [51] using 64 TPUv5e chips with a data parallel mesh.

We use a batch size of 256 and train for 2M steps (about 265 hours of training).

Data Collection. Protocol documents (See Appendix B.1) outline instructions for both how to use the robots, and specific instructions for the task being performed.

This allows continuous data collection by a pool of 35 operators without oversight by researchers. Using this protocol, we collect over **26k episodes** for 5 real tasks, on **10 different robots** in 2 different buildings over the course of **8 months**

https://aloha-unleashed.github.io/assets/kitchen_protocol.pdf

Task	Success Rate	Number of Demonstrations
ShirtEasy	75%	8658 (5345 Easy; 3313 Messy)
ShirtMessy	70%	
LaceEasy	70%	5133 (2212 Easy; 2921 Messy)
LaceMessy	40%	
FingerReplace	75%	5247
GearInsert-1	95%	4005
GearInsert-2	75%	
GearInsert-3	40%	
RandomKitchen-Bowl	95%	3198 (216 In-Domain)
RandomKitchen-Bowl+Cup	65%	
RandomKitchen-Bowl+Cup+Fork	25%	

Table 1: Success rates and number of demonstrations for 5 real tasks, with separate models for each task. Shirt and Lace tasks each have a single model trained on 8658 and 5133 demonstrations, respectively, and are separately evaluated on Easy and Messy variants of the tasks. The GearInsert and RandomKitchen model evaluations are broken down by total progress on the task.

Scaling Up



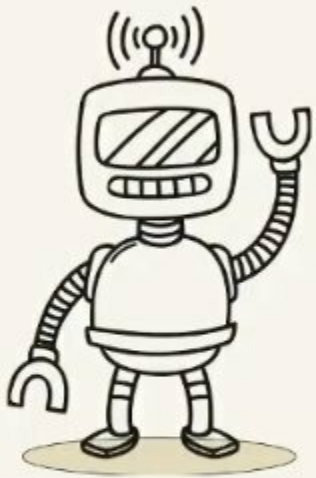
- I've discussed training *one* skill
- Wanted: few shot generalization to new skills
  - multitask, language-conditioned policies
  - connects beautifully to internet-scale data

- Big Questions:
  - How do we get data?
  - What are the scaling laws?

The future of robotics and AI?

Different Deep Learning Frameworks

- PyTorch
- JAX/FLAX
- TensorFlow
- ...



The Field Of Embodied Intelligence

Different Model Architectures

- Diffusion
- Flow Matching
- Autoregressive
- ...