Imitation Learning

Introduction

- Imitation Learning
  - Also known as learning by demonstration, apprenticeship learning
- An expert demonstrates how to solve the task
  - Machine can also interact with the environment, but cannot explicitly obtain reward.
  - It is hard to define reward in some tasks.
  - Hand-crafted rewards can lead to uncontrolled behavior
- Three approaches:
  - Behavior Cloning
  - Inverse Reinforcement Learning
  - Generative Adversarial Network

Behavior Cloning

Behavior Cloning

Yes, this is supervised learning.

- Self-driving cars as example

observation

Expert (Human driver): 向前

Machine: 向前

Training data:

$(o_1, \hat{a}_1)$
$(o_2, \hat{a}_2)$
$(o_3, \hat{a}_3)$
……

o_i → NN → a_i ↔ $\hat{a}_i$

Actor

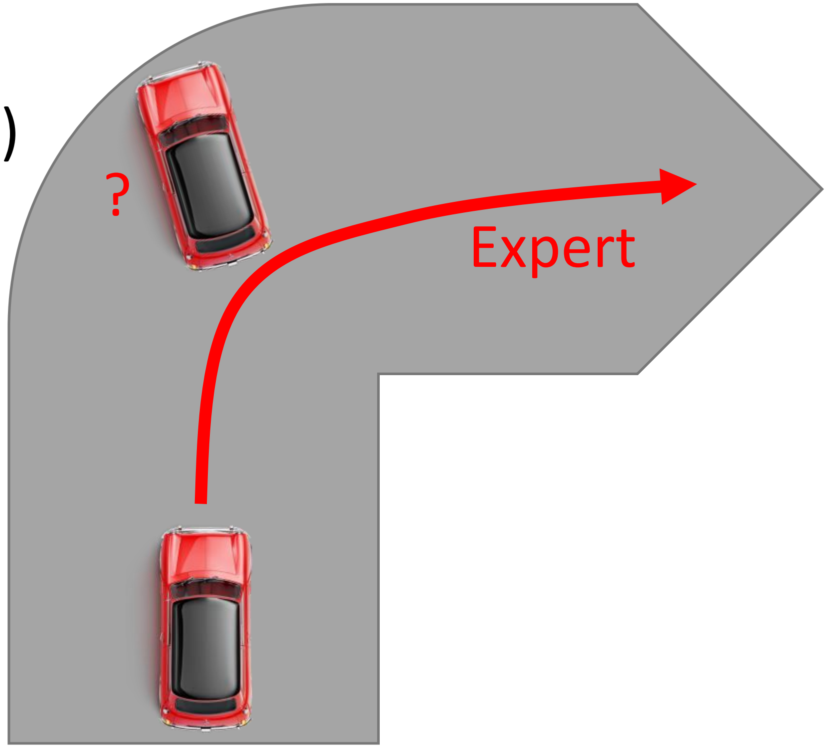
Behavior Cloning

- Problem

Expert only samples limited observation (states)

Let the expert in the states seem by machine

Dataset Aggregation



Behavior Cloning

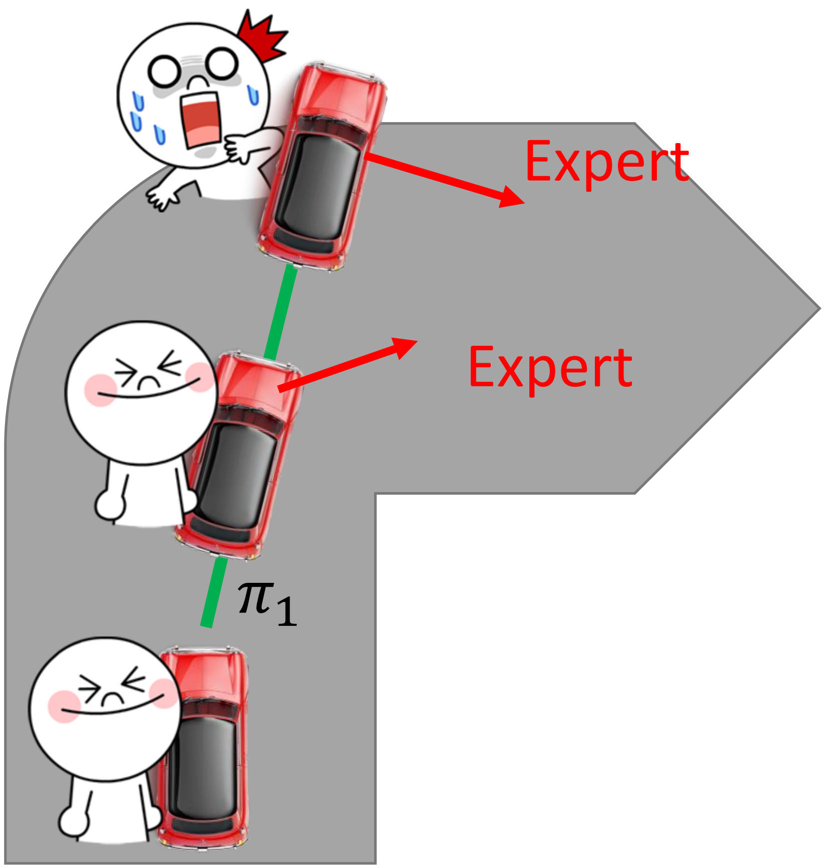
- Dataset Aggregation

  Get actor π_1 by behavior cloning

  Using π_1 to interact with the environment

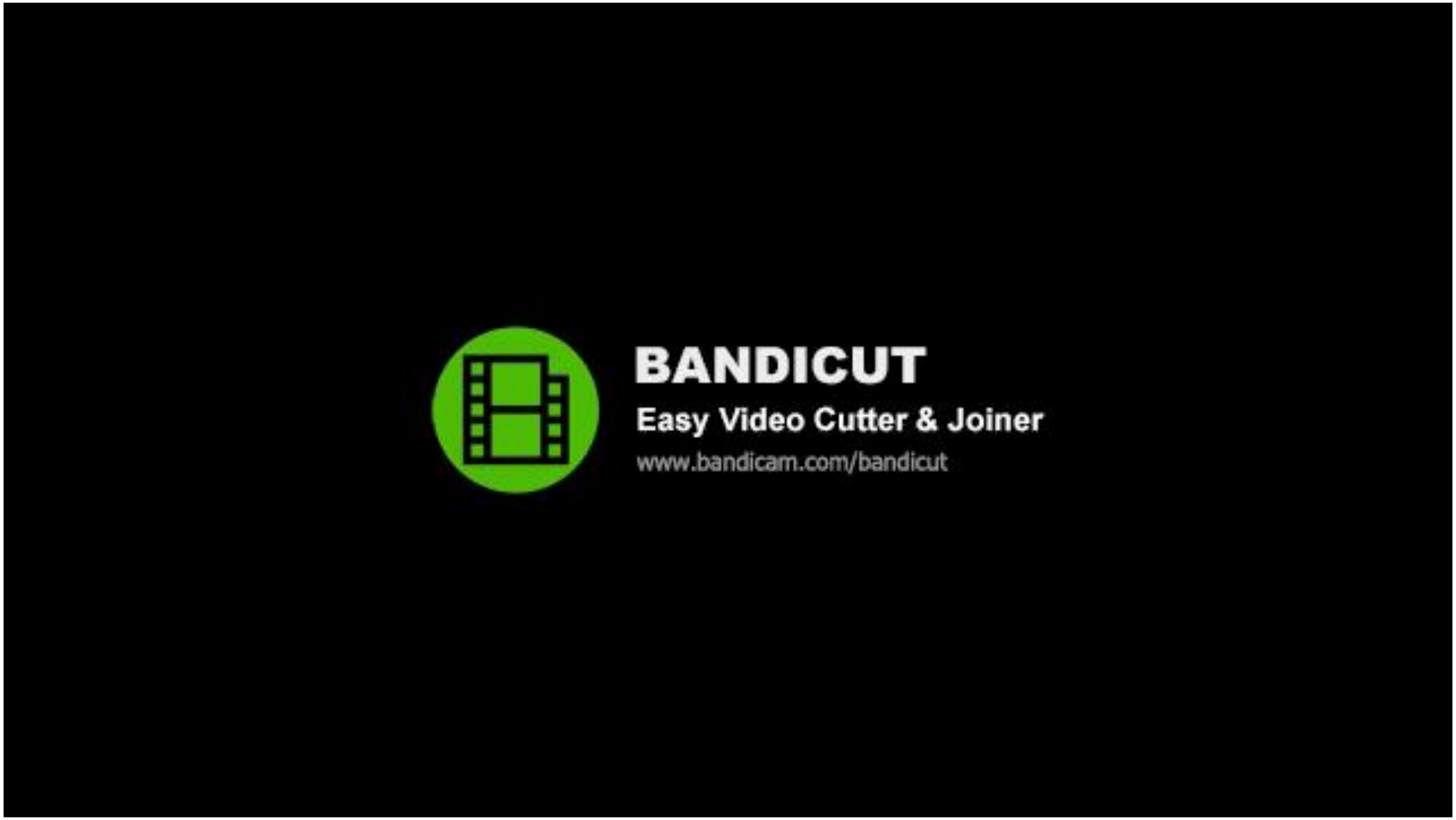
    Ask the expert to label the observation of π_1

  Using new data to train π_2



Behavior Cloning

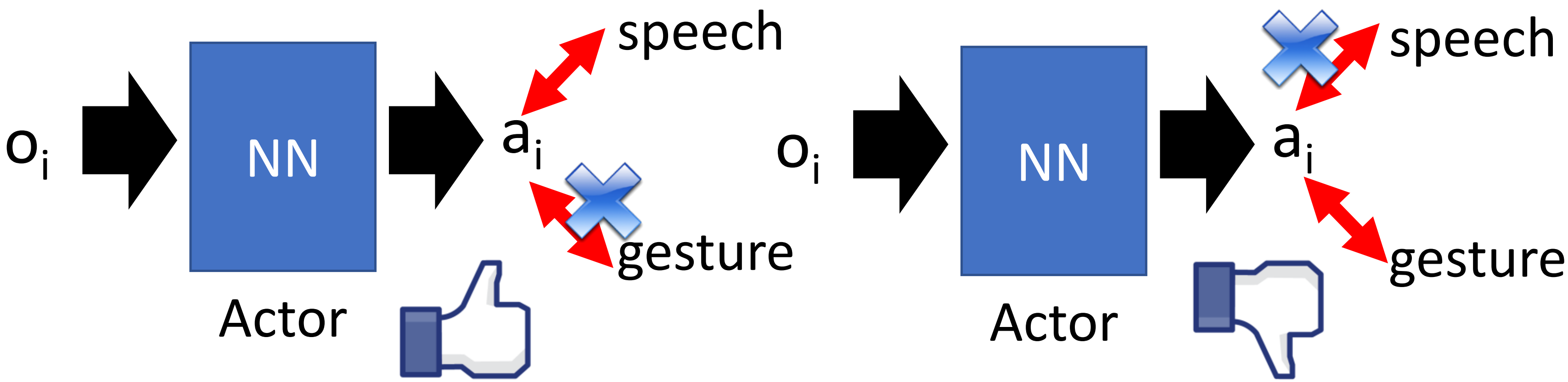
The agent will copy every behavior, even irrelevant actions.



https://www.youtube.com/watch?v=j2FSB3bseek

Behavior Cloning

- Major problem: if machine has limited capacity, it may choose the wrong behavior to copy.



- Some behavior must copy, but some can be ignored.
    - Supervised learning takes all errors equally

Mismatch

- In supervised learning, we expect training and testing data have the same distribution.
- In behavior cloning:
  - Training: $(o, a) \sim \hat{\pi}$ (expert)
    - ***<u>Action a taken by actor influences the distribution of o</u>***
  - Testing: $(o', a') \sim \pi^*$ (actor cloning expert)
    - If $\hat{\pi} = \pi^*$, (o, a) and (o', a') from the same distribution
    - If $\hat{\pi}$ and π^* have difference, the distribution of o and o' can be very different.

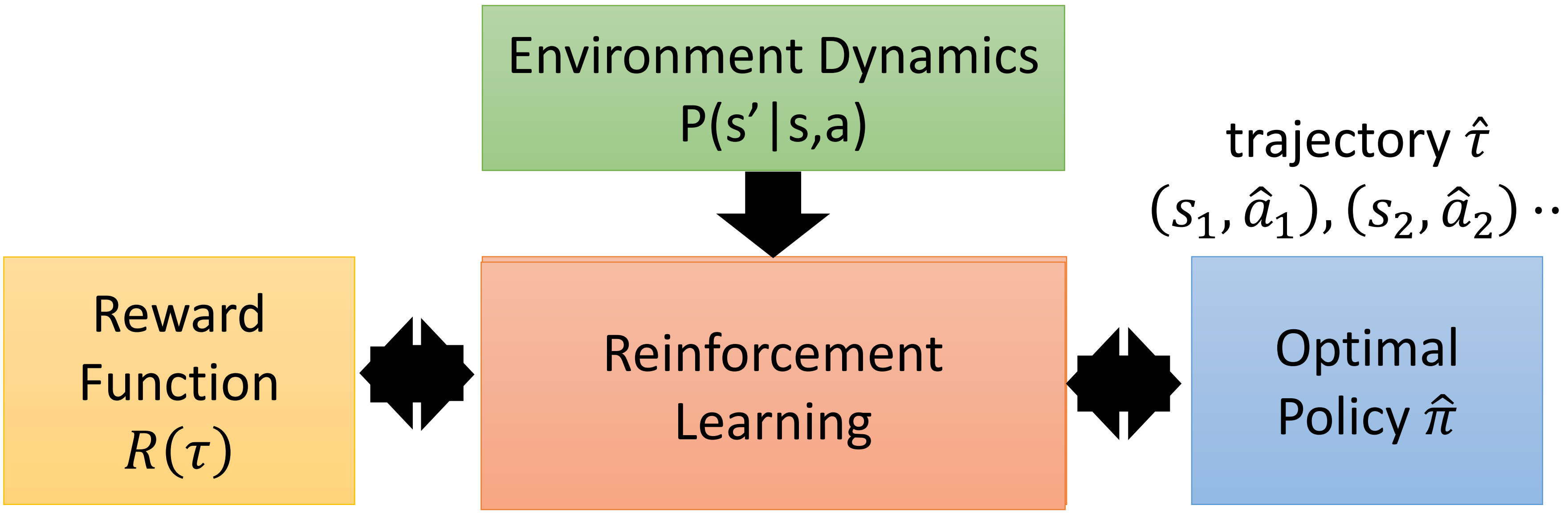
Inverse Reinforcement Learning (IRL)

Also known as inverse optimal control,

inverse optimal planning

Pieter Abbeel and Andrew Y. Ng. "Apprenticeship learning via inverse reinforcement learning“, ICML, 2004

Inverse Reinforcement Learning



- Using the reward function to find a policy π^*
- Modeling reward can be easier. Simple reward function can lead to complex policy.

Inverse Reinforcement Learning

- <u>***Original RL***</u>:
  - given a reward function $R(\tau)$, $R(\tau) = \sum_{t=1}^{T} r(s_t, a_t)$
  - Initialize an actor π
  - In each iteration
    - using π to interact with the environment N times, obtain $\{\tau^1, \tau^2, \cdots, \tau^N\}$

      $\tau = \{s_1, a_1, \boxed{r_1}, s_2, a_2, \boxed{r_2}, \cdots, s_T, a_T, \boxed{r_T}\}$

      $$\bar{R}_\pi = \sum_\tau R(\tau) P(\tau|\pi) \approx \frac{1}{N} \sum_{n=1}^{N} R(\tau^n)$$

      $$R(\tau) = \sum_{t=1}^{T} r_t$$
    - Update π to maximize $\bar{R}_\pi$
  - The actor π is the optimal actor $\hat{\pi}$

Inverse Reinforcement Learning

- ***Inverse RL***:
  - $R(\tau)$ or $r(s,a)$ is to be found
  - Given expert policy $\hat{\pi}$ (Given the trajectories $\{\hat{\tau}_1, \hat{\tau}_2, \cdots, \hat{\tau}_N\}$)
  - The expert policy $\hat{\pi}$ is the actor that can obtain maximum expected reward
  - Find ***reward function*** that fulfills the above statements (explaining expert behavior)

$$\bar{R}_{\hat{\pi}} > \bar{R}_{\pi} \qquad \text{For all other actors } \pi$$

Ring a bell in your mind?

Inverse Reinforcement Learning

Find reward function:

$$\bar{R}_{\hat{\pi}} > \bar{R}_{\pi}$$

For all other actors π

Find policy:

$$\pi^* = arg \max_{\pi} \bar{R}_{\pi}$$

Structured Learning

Training:

$$F(x, \hat{y}) > F(x, y)$$

For all x, for all $y \neq \hat{y}$

Testing (Inference):

$$y^* = arg \max_{y} F(x, y)$$

Review: Structured Perceptron

- **Input**: training data set $\{(x^1, \hat{y}^1), (x^2, \hat{y}^2), \ldots, (x^r, \hat{y}^r), \ldots\}$
- **Output**: weight vector w
- **Algorithm**: Initialize w = 0
  - do
    - For each pair of training example $(x^r, \hat{y}^r)$
      - Find the label $\tilde{y}^r$ maximizing $w \cdot \phi(x^r, y)$

        $$\tilde{y}^r = \arg\max_{y \in Y} w \cdot \phi(x^r, y)$$

      - If $\tilde{y}^r \neq \hat{y}^r$, update w

        $$w \leftarrow w + \phi(x^r, \hat{y}^r) - \phi(x^r, \tilde{y}^r)$$
  - until w is not updated ➡ We are done!

$$F(x, y) = w \cdot \phi(x, y)$$

Can be an issue

Increase $F(x^r, \hat{y}^r)$, decrease $F(x^r, \tilde{y}^r)$

IRL v.s. Structured Perceptron

$$F(x,y) = w \cdot \phi(x,y) \quad \Longleftrightarrow \quad \bar{R}_\pi = w \cdot \phi(\pi)$$

$$\tau = \{s_1, a_1, \quad s_2, a_2, \quad \cdots, s_T, a_T, \quad\} \qquad \phi(\pi)$$

$$\bar{R}_\theta \approx \frac{1}{N}\sum_{n=1}^{N} R(\tau^n) = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T} r_t = w \cdot \boxed{\frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T} f(s_t, a_t)}$$

$r_t = w \cdot f(s_t, a_t)$ w: Parameters $f(s_t, a_t)$: feature vector

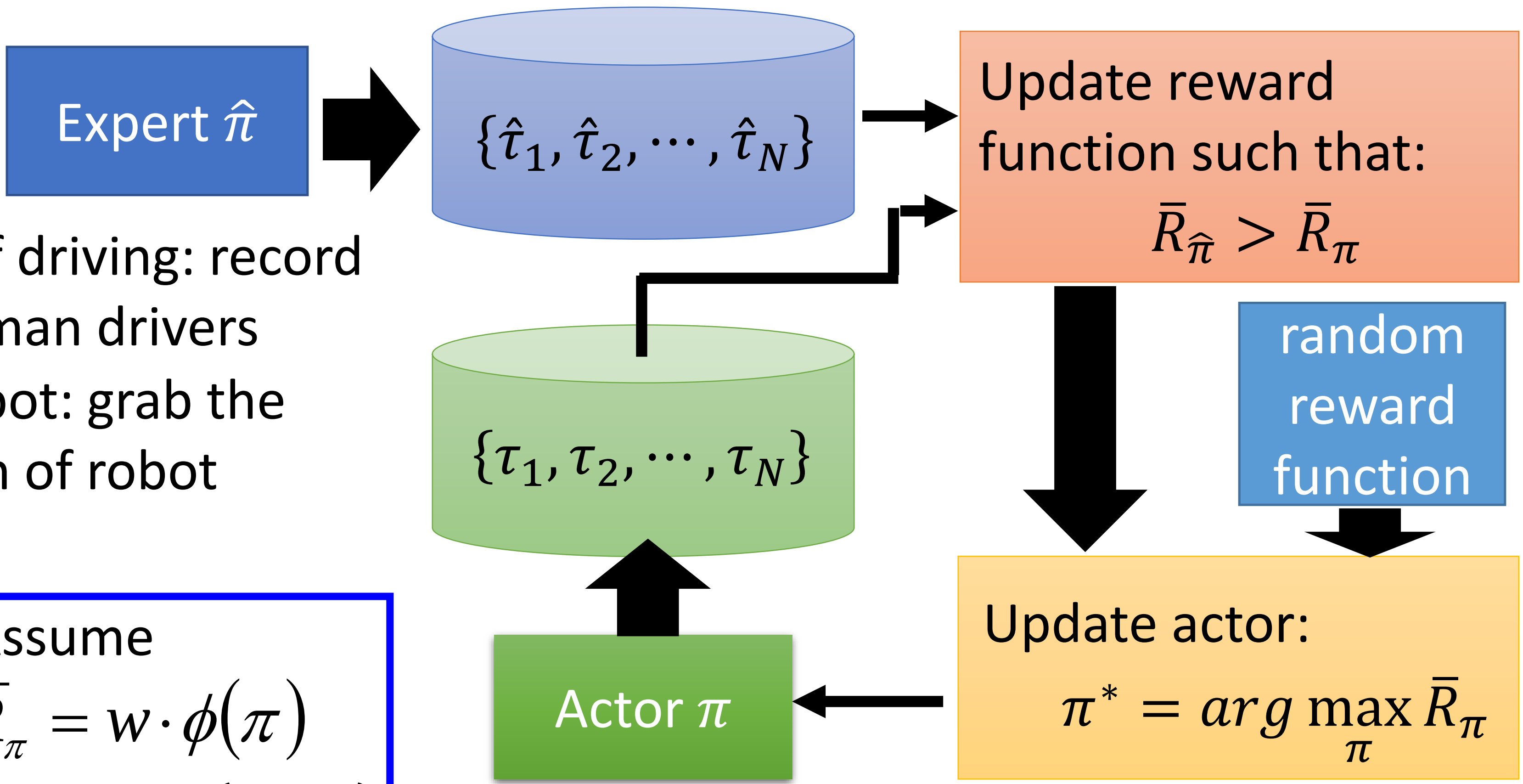
$$\tilde{y} = \arg\max_{y \in Y} F(x,y) \quad \Longleftrightarrow \quad \pi^* = arg\max_{\pi} \bar{R}_\pi$$

This is reinforcement learning.

Framework of IRL

$$\phi(\pi) = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T} f(s_t, a_t)$$

$$w \rightarrow w + \phi(\hat{\pi}) - \phi(\pi)$$



Self driving: record human drivers

Robot: grab the arm of robot

By Reinforcement learning

Assume

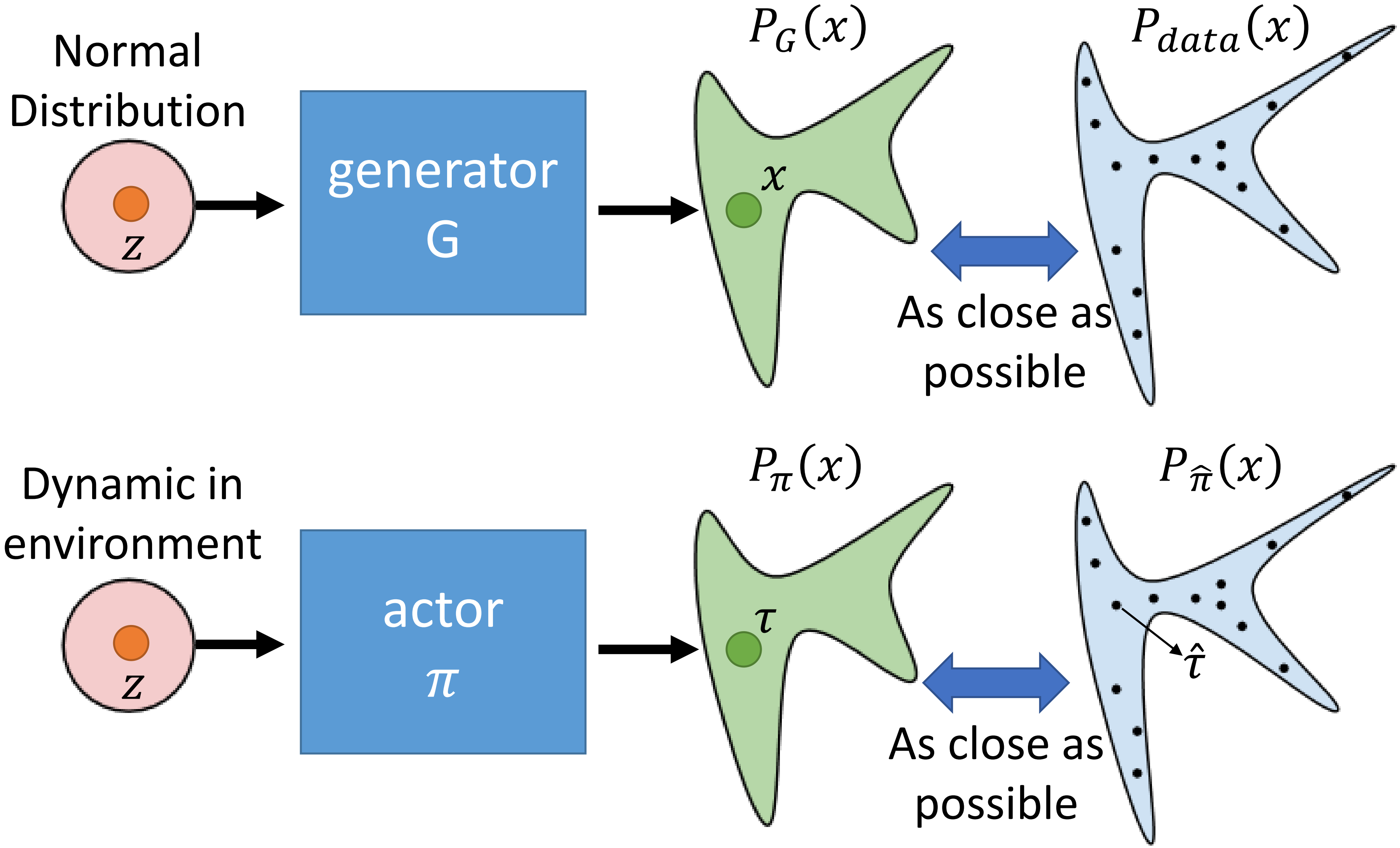
$\bar{R}_{\pi} = w \cdot \phi(\pi)$

$r_t = w \cdot f(s_t, a_t)$

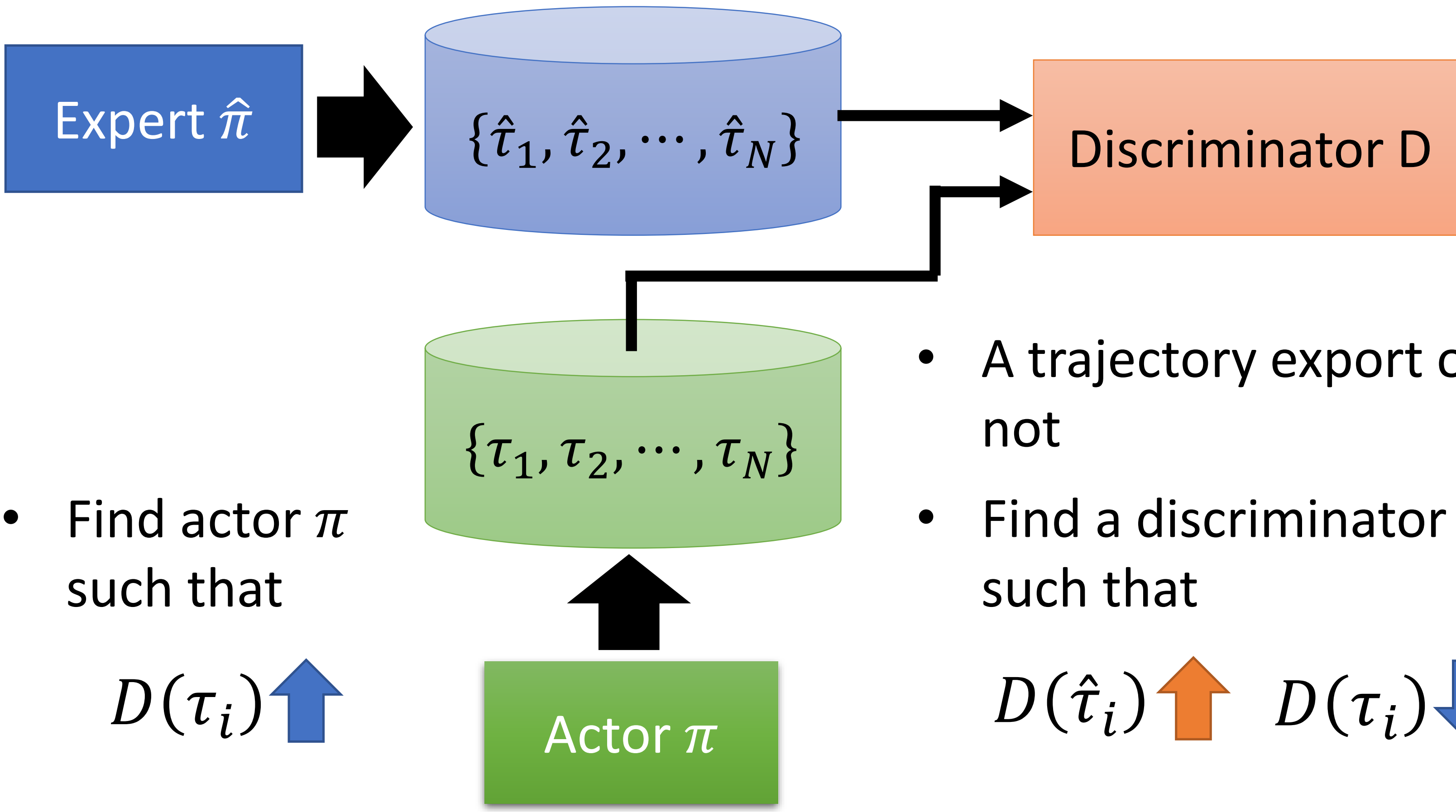
GAN for Imitation Learning

Jonathan Ho and Stefano Ermon. "Generative adversarial imitation learning“, NIPS, 2016

GAN v.s. Imitation Learning



GAN for Imitation Learning



- A trajectory export or not
- Find a discriminator such that

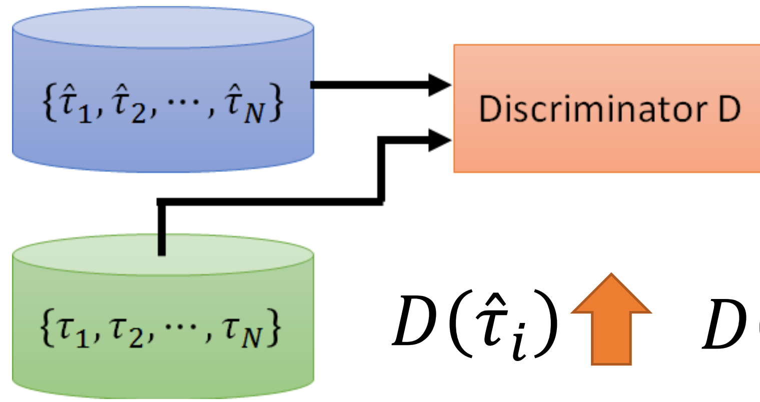
$$D(\hat{\tau}_i)\uparrow \quad D(\tau_i)\downarrow$$

- Find actor π such that

$$D(\tau_i)\uparrow$$

GAN for Imitation Learning

- Discriminator



$D(\hat{\tau}_i)$ $D(\tau_i)$

$\tau = \{s_1, a_1, s_2, a_2, \cdots, s_T, a_T\}$

$D(\tau)$

$D(\hat{\tau}_i)$ $D(\tau_i)$

(s,a) from expert → $d(s,a)$

(s,a) from actor → $d(s,a)$

GAN for Imitation Learning

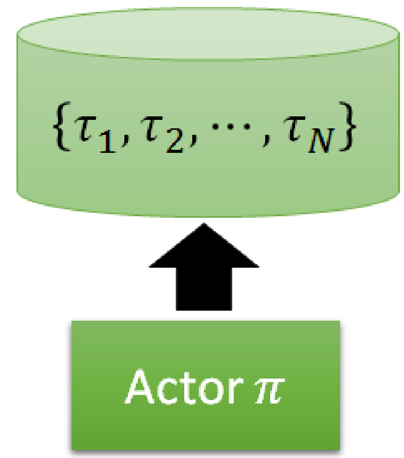
- Generator

$\tau = \{s_1, a_1, s_2, a_2, \cdots, s_T, a_T\}$

$$D(\tau) = \frac{1}{T}\sum_{t=1}^{T} d(s_t, a_t)$$

- Find actor π such that

$D(\tau_i)$



$$\theta^\pi \leftarrow \theta^\pi + \eta \nabla_{\theta^\pi} E_\pi[D(\tau)] \Rightarrow \theta^\pi \leftarrow \theta^\pi + \eta \sum_{i=1}^{N} D(\tau_i) \nabla_{\theta^\pi} logP(\tau_i|\pi)$$

policy gradient

Each step in the same trajectory can have different values.

Given discriminator D

Using π to interact with the environment to obtain $\{\tau_1, \tau_2, \cdots, \tau_N\}$

If D(τ_i) is large, increase $P(\tau_i|\pi)$; otherwise, decrease $P(\tau_i|\pi)$

Algorithm

- Input: expert trajectories $\{\hat{\tau}_1, \hat{\tau}_2, \cdots, \hat{\tau}_N\}$
- Initialize discriminator D and actor π
- In each iteration:
  - Using actor to obtain trajectories $\{\tau_1, \tau_2, \cdots, \tau_N\}$
  - Update discriminator parameters: Increase $D(\hat{\tau}_i)$, decrease $D(\tau_i)$

$$D(\tau) = \frac{1}{T}\sum_{t=1}^{T} \underbrace{d(s_t, a_t)}_{\text{reward}}$$

Find the reward function that expert has larger reward.

  - Update actor parameters: Increase $D(\tau_i)$

$$\theta^\pi \leftarrow \theta^\pi + \eta \sum_{i=1}^{N} D(\tau_i) \nabla_{\theta^\pi} logP(\tau_i|\pi)$$

Find the actor maximizing reward by reinforcement learning

Recap: Sentence Generation & Chat-bot

Sentence Generation

Expert trajectory:
床 前 明 月 光

(o_1, a_1): (“<BOS>”,”床”)

(o_2, a_2): (“床”,”前”)

(o_3, a_3): (“床前”,”明”)

⋮ ⋮

Chat-bot

Expert trajectory:
input: how are you
Output: I am fine

(o_1, a_1): (“input, <BOS>”,”I”)

(o_2, a_2): (“input, I”, ”am”)

(o_3, a_3): (“input, I am”, ”fine”)

⋮ ⋮

Maximum likelihood is behavior cloning. Now we have better approach like SeqGAN.

Examples of Recent Study

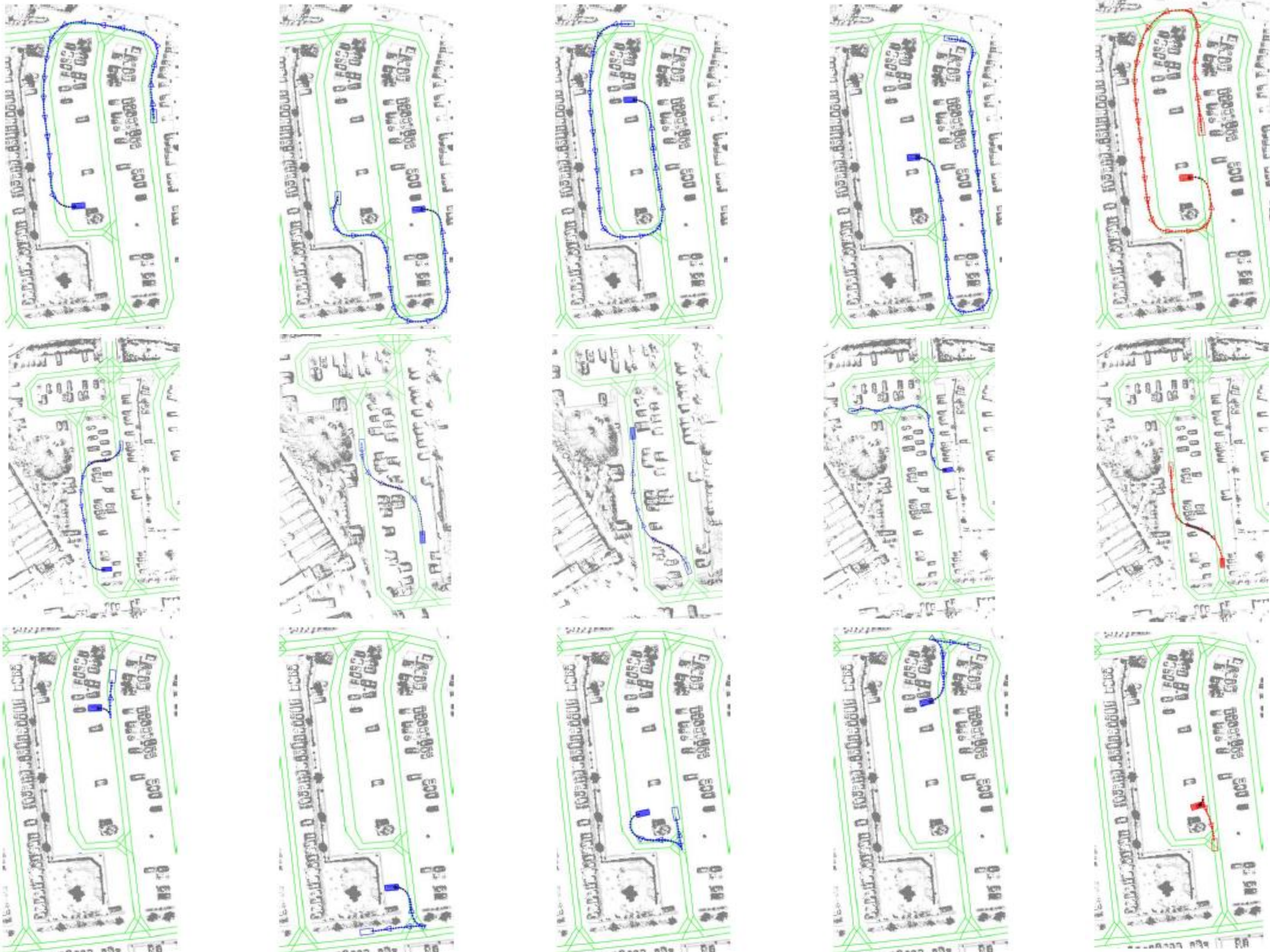
Robot

Chelsea Finn, Sergey Levine, Pieter Abbeel, " Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization", ICML, 2016

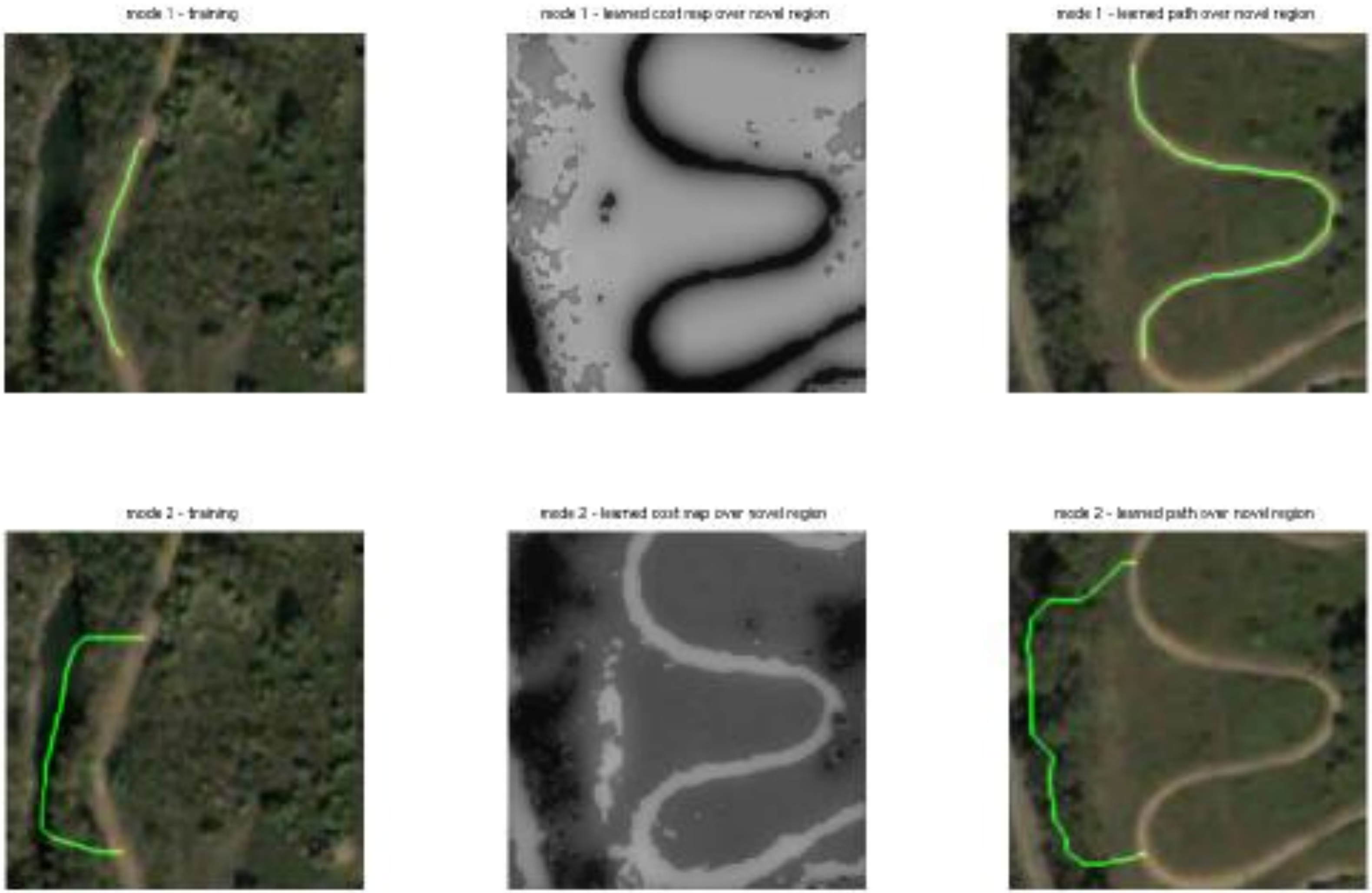
http://rll.berkeley.edu/gcl/

Parking Lot Navigation

- Reward function:
    - Forward vs. reverse driving
    - Amount of switching between forward and reverse
    - Lane keeping
    - On-road vs. off-road
    - Curvature of paths



Path Planning



Third Person Imitation Learning

- Ref: Bradly C. Stadie, Pieter Abbeel, Ilya Sutskever, "Third-Person Imitation Learning", arXiv preprint, 2017

First Person

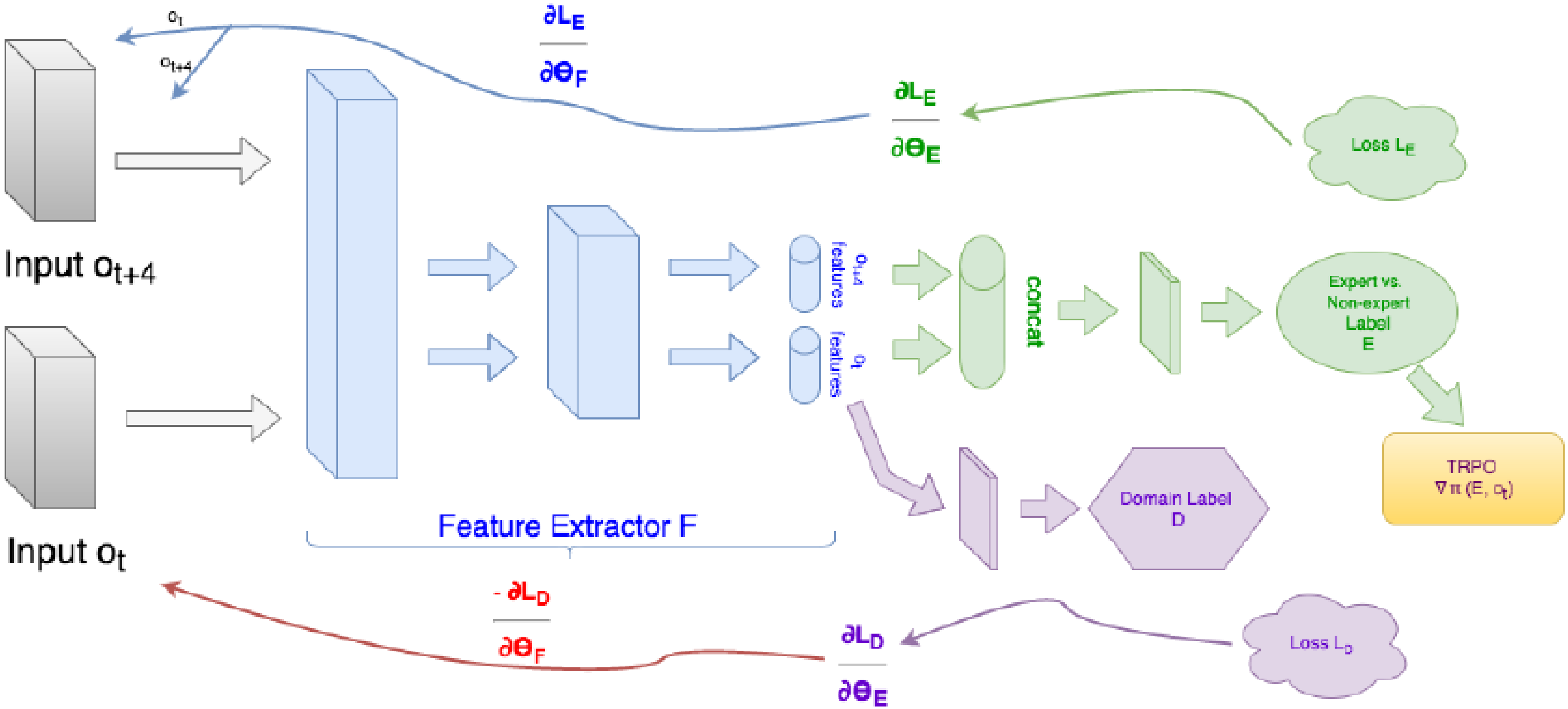
http://lasa.epfl.ch/research_new/ML/index.php

Third Person

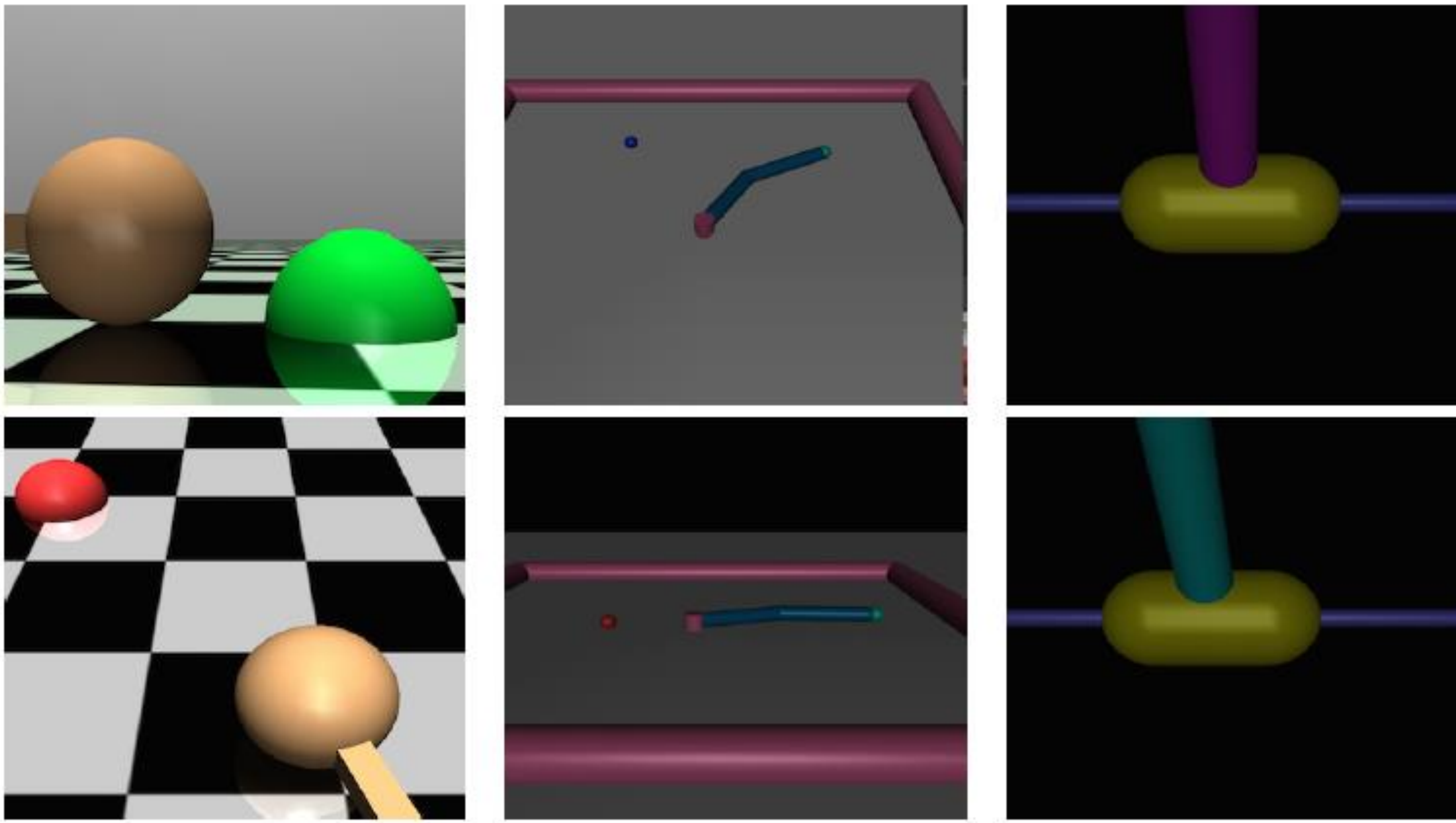
https://kknews.cc/sports/q5kbb8.html

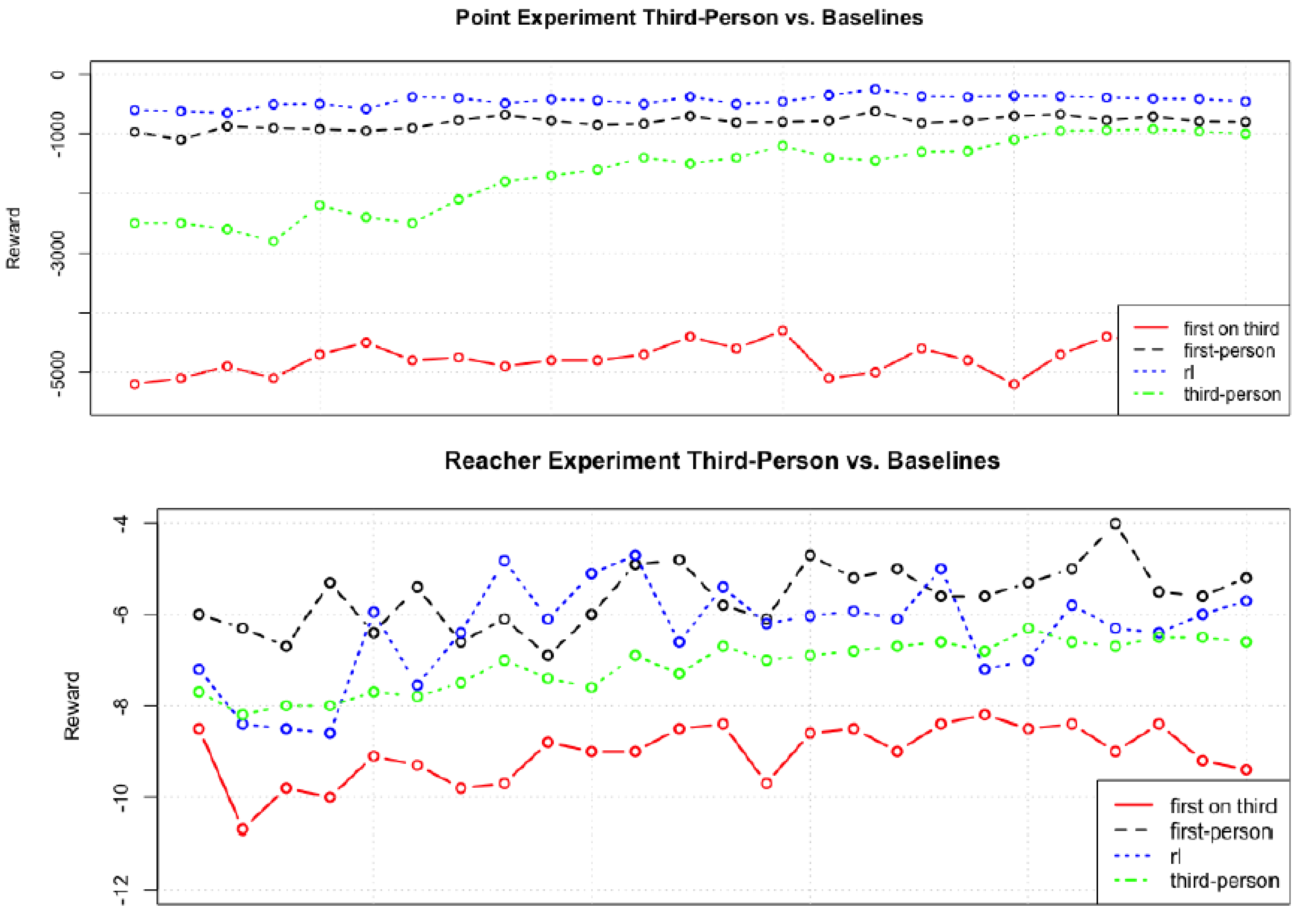
http://sc.chinaz.com/Files/pic/icons/1913/%E6%9C%BA%E5%99%A8%E4%BA%BA%E5%9B%BE%E6%A0%87%E4%B8%8B%E8%BD%BD34.png

Third Person Imitation Learning



Third Person Imitation Learning

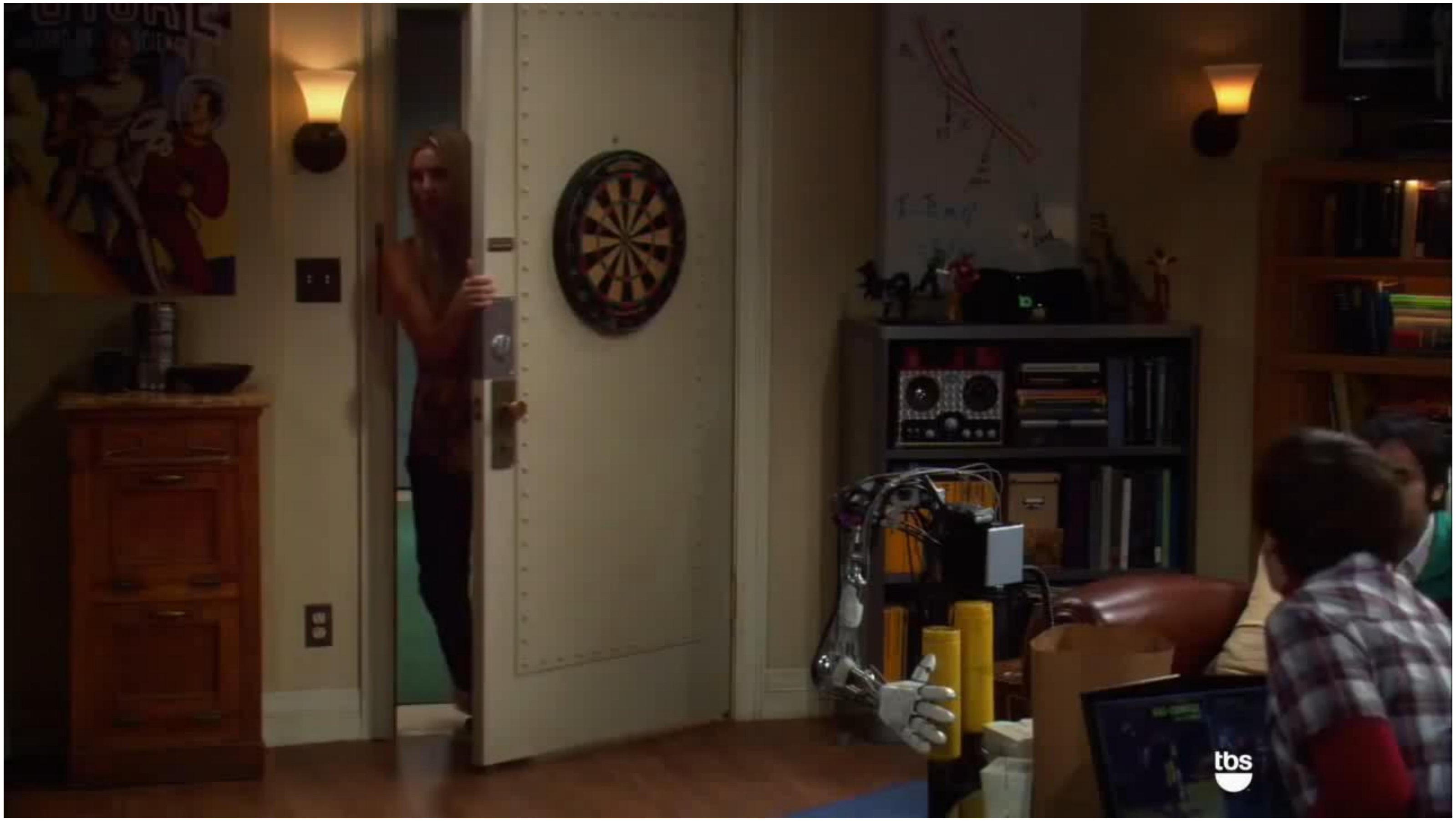




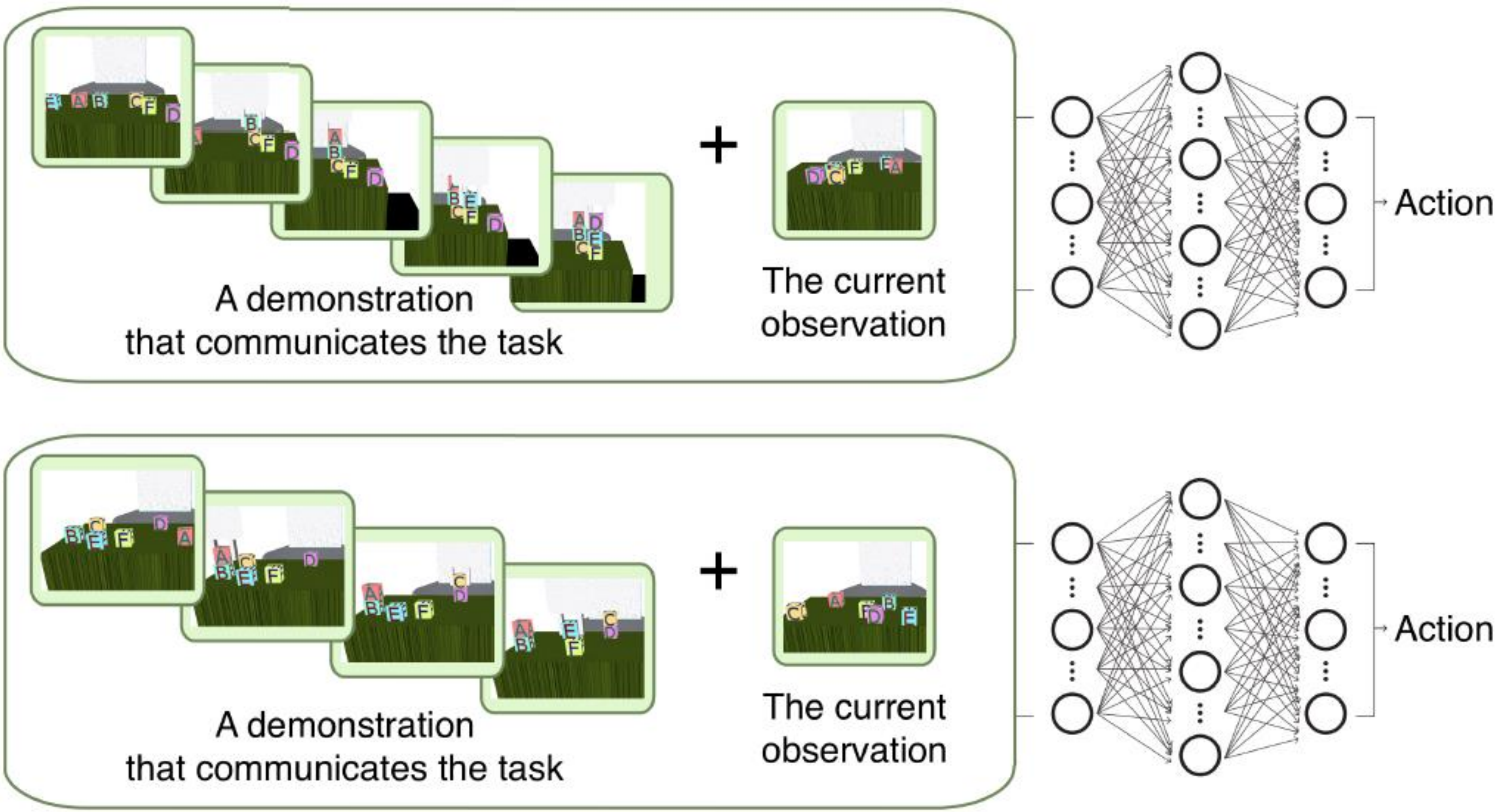
Point Experiment Third-Person vs. Baselines
Reward
0
-1000
-3000
-5000
first on third
first-person
rl
third-person
Reacher Experiment Third-Person vs. Baselines
Reward
-4
-6
-8
-10
-12
first on third
first-person
rl
third-person

One-shot Imitation Learning

- How to teach robots? https://www.youtube.com/watch?v=DEGbtjTOIB0



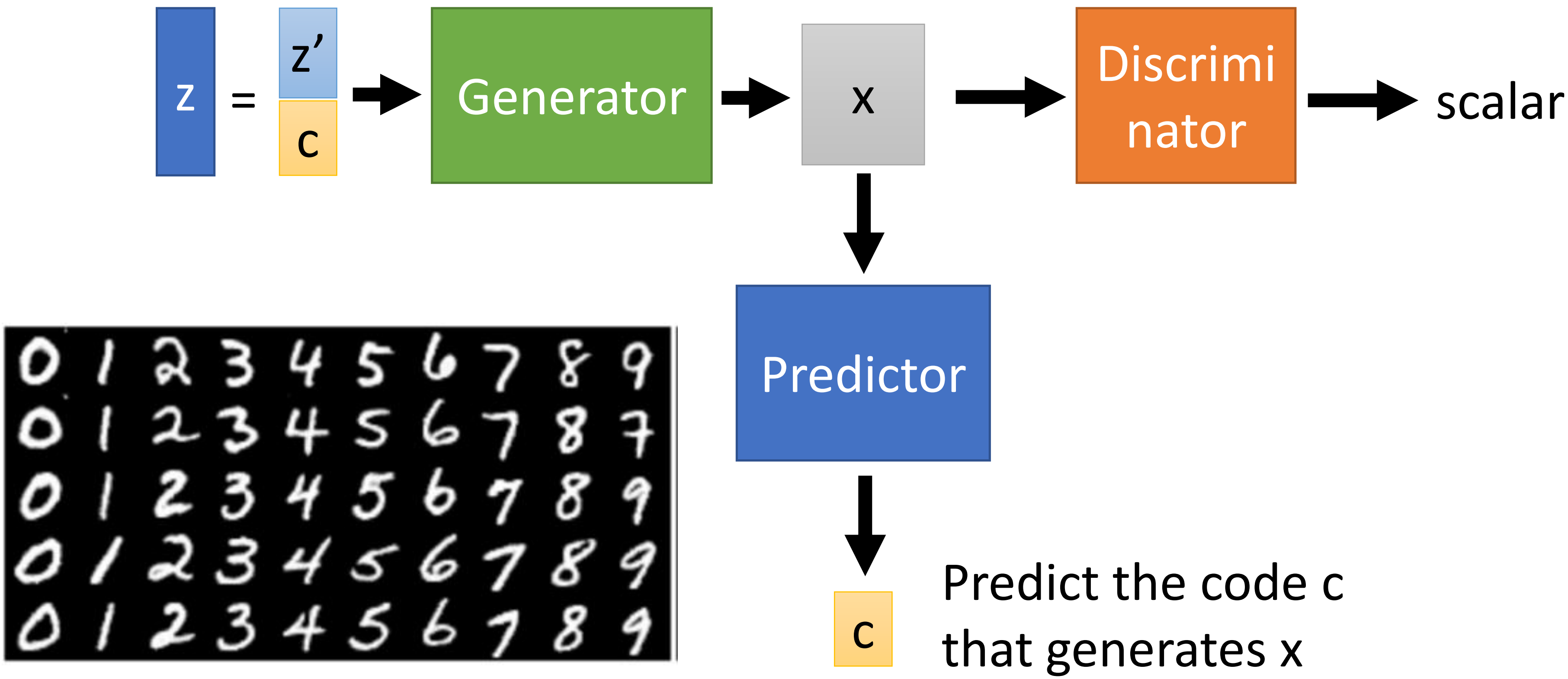
One-shot Imitation Learning



Unstructured Demonstration

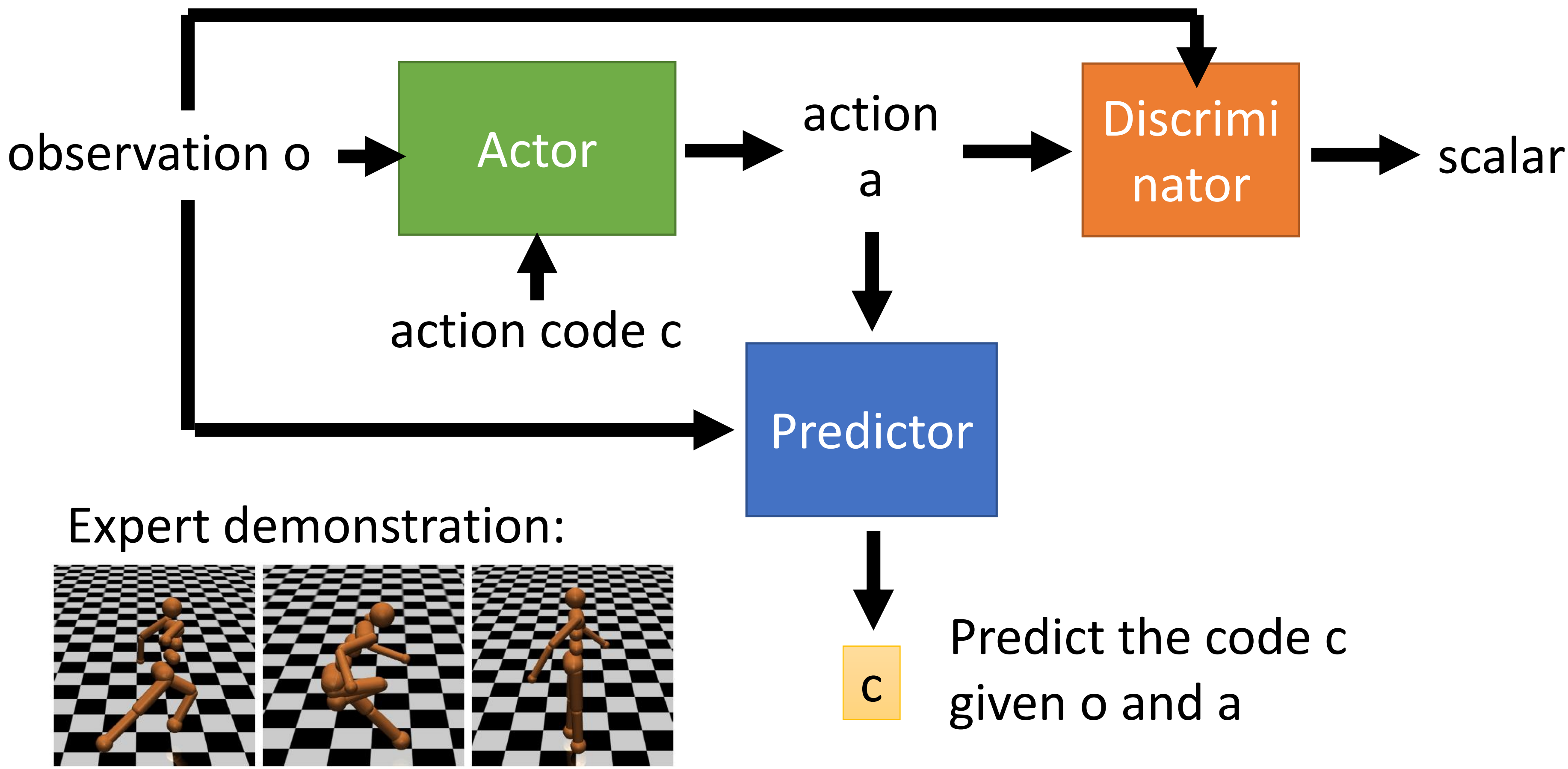
- Review: InfoGAN

Karol Hausman, Yevgen Chebotar, Stefan Schaal, Gaurav Sukhatme, Joseph Lim, Multi-Modal Imitation Learning from Unstructured Demonstrations using Generative Adversarial Nets, arXiv preprint, 2017

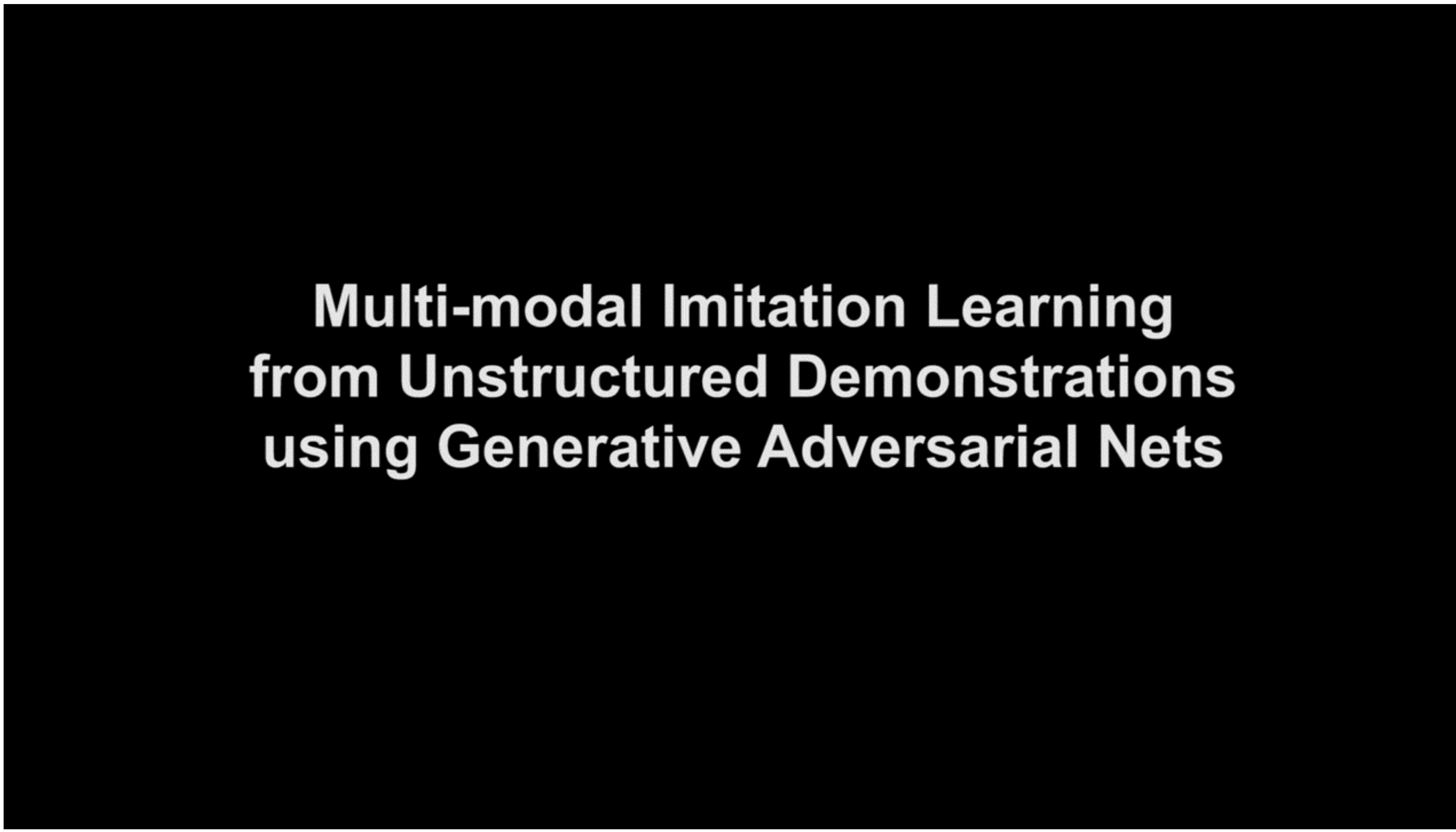


Unstructured Demonstration

- The solution is similar to info GAN



Unstructured Demonstration



https://www.youtube.com/watch?v=tpEgL1AASYk

http://www.commitstrip.com/en/2017/06/07/ai-inside/?