DriveGAIL | Generative Adversarial Imitation Learning For Self-driving Cars

“ 따라쟁이 자율주행 에이전트 제작기 "

정 상 용

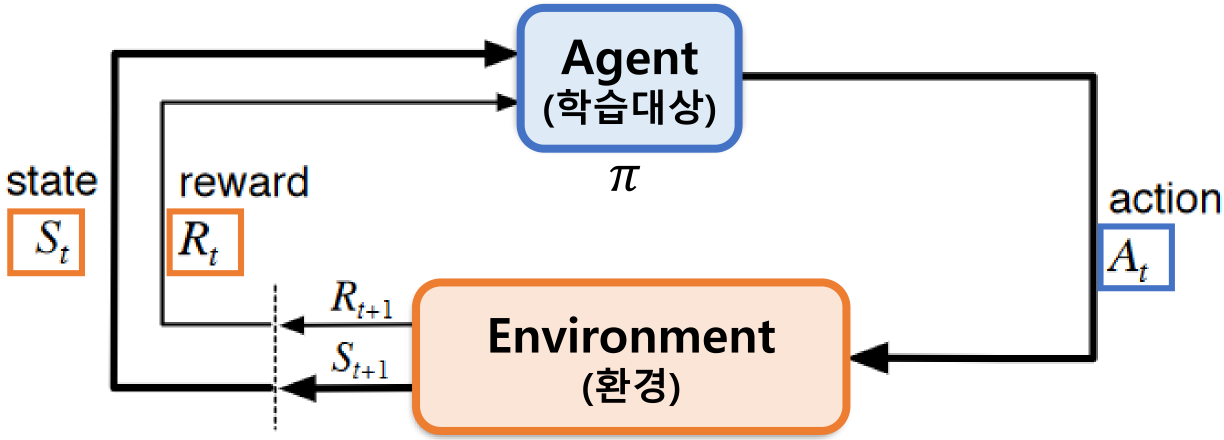
Speaker

- **정 상 용**
- Reinforcement Learning / DLC1
- sangyongjeong@gmail.com

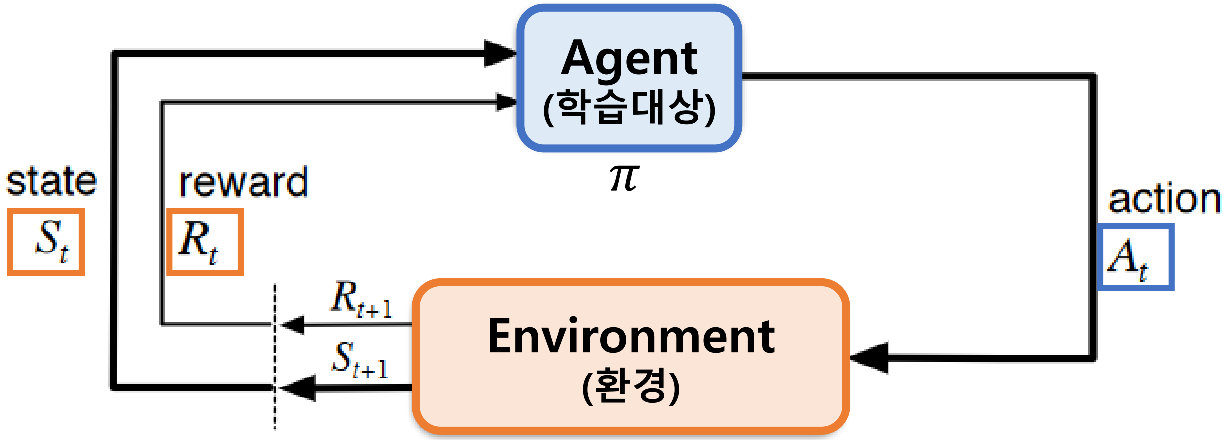
Index

- Imitation Learning
- 적용 알고리즘 소개
- DriveGAIL 구현 및 실험

Background: 강화학습이란?

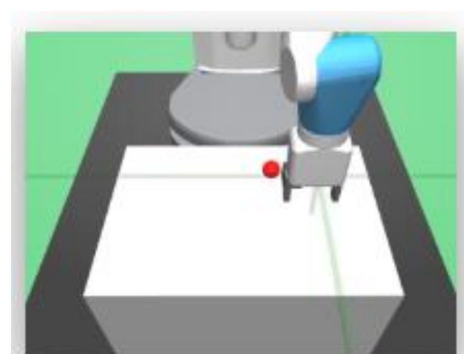


Background: 강화학습이란?



(예)

Breakout-v0
Maximize score in the game Breakout, with screen images as input



FetchPickAndPlace-v0
Lift a block into the air.

source: gym.openai.com

What is the reward?

현실문제에 적용하려면?

→ Reward 정의하기 매우복잡 or 애매함

source: Deep RL Bootcamp Lecture 10B, BAIR

Reward가 필요 없는 'Imitation Learning'

Imitation Learning is a sequential task where the learner tries to mimic an expert's action in order to achieve the best performance.
- 'Global overview of Imitation Learning', Alexandre, 2018

'Imitation Learning(모방 학습)'은 학습자가 최상의 성능을 얻기 위해 전문가의 행동을 모방하는 순차적인 작업

전문가의 행동 = Expert Trajectories

State(Observation)에 따른 Expert의 'Action' 데이터 셋

$$\{(s_0^i, a_0^i, s_1^i, a_1^i, \ldots)\}_{i=1}^{n} \sim \pi_E$$

Reward가 필요 없는 'Imitation Learning'

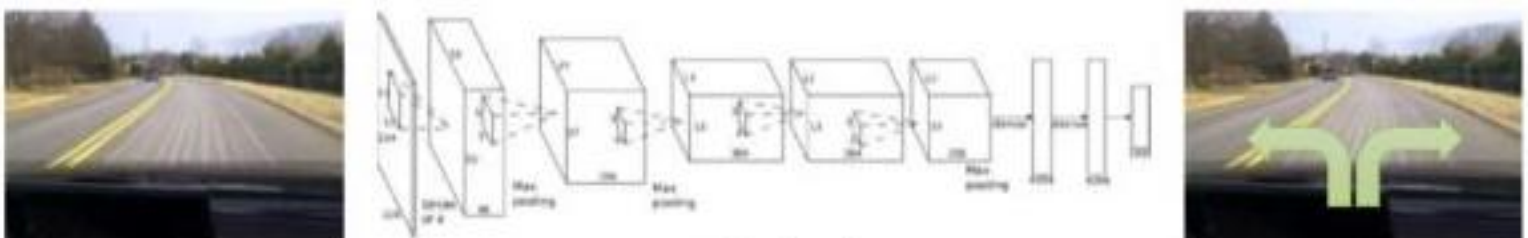
Imitation Learning is a sequential task where the learner tries to mimic an expert's action in order to achieve the best performance.
- 'Global overview of Imitation Learning', Alexandre, 2018

'Imitation Learning(모방 학습)'은 학습자가 최상의 성능을 얻기 위해 전문가의 행동을 모방하는 순차적인 작업

- Behavioral Cloning
- Inverse Reinforcement Learning (IRL)
- Generative Adversarial Reinforcement Learning (GAIL)

Behavioral Cloning: Supervised learning

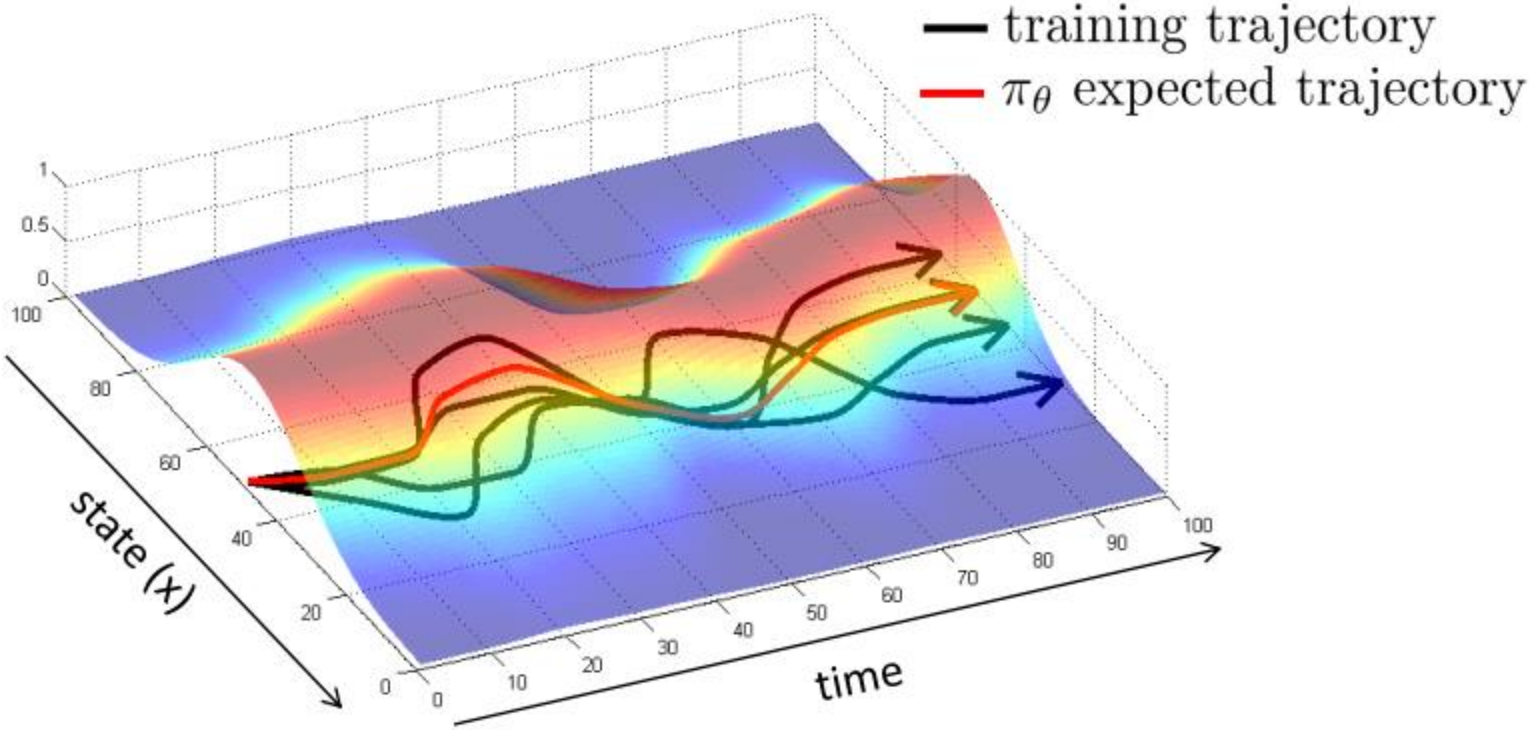
Expert



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

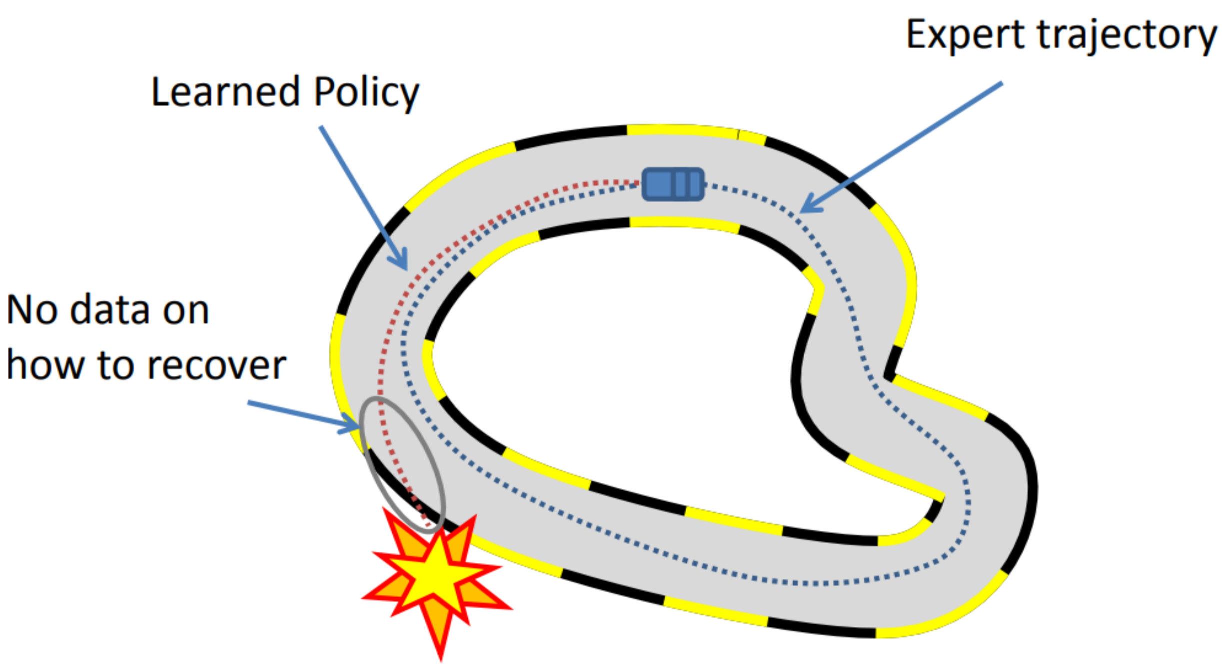
source: Deep Reinforcement Learning, CS294-112, Berkeley

Behavioral Cloning



source: Deep Reinforcement Learning, CS294-112, Berkeley

Behavioral Cloning의 한계



→ **Supervised Learning: 매우 다양한 Trajectory 필요**

→ **사람에 의한 Expert Trajectory 수집의 한계**

source: Deep Reinforcement Learning and Control, CMU

(1) Reward function / feature의 정의

(2) IRL: Function Parameter Update ↔ (3)RL by Reward Function

(예) Maximum Entropy Inverse RL, Ziebart, 2008

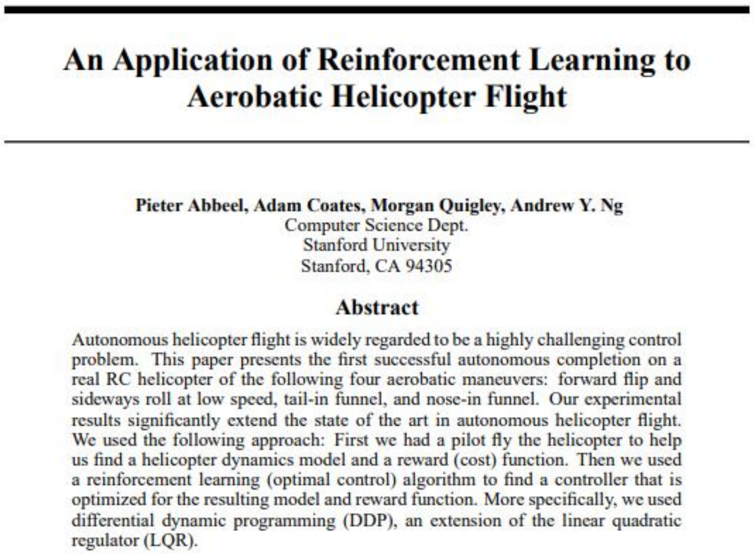
reward parameters

reward function

0. Initialize ψ, gather demonstrations $\mathcal{D}$
1. Solve for optimal policy $\pi(\mathbf{a}|\mathbf{s})$ w.r.t. reward r_ψ
2. Solve for state visitation frequencies $p(\mathbf{s}|\psi)$
3. Compute gradient $\nabla_\psi \mathcal{L} = -\frac{1}{|\mathcal{D}|}\sum_{\tau_d \in \mathcal{D}} \frac{dr_\psi}{d\psi}(\tau_d) - \sum_s p(s|\psi)\frac{dr_\psi}{d\psi}(s)$
4. Update ψ with one gradient step using $\nabla_\psi \mathcal{L}$

IRL 사례_Stanford Autonomous Helicopter

Reward function w/ 24 features



An Application of Reinforcement Learning to Aerobatic Helicopter Flight

Pieter Abbeel, Adam Coates, Morgan Quigley, Andrew Y. Ng
Computer Science Dept.
Stanford University
Stanford, CA 94305

Abstract

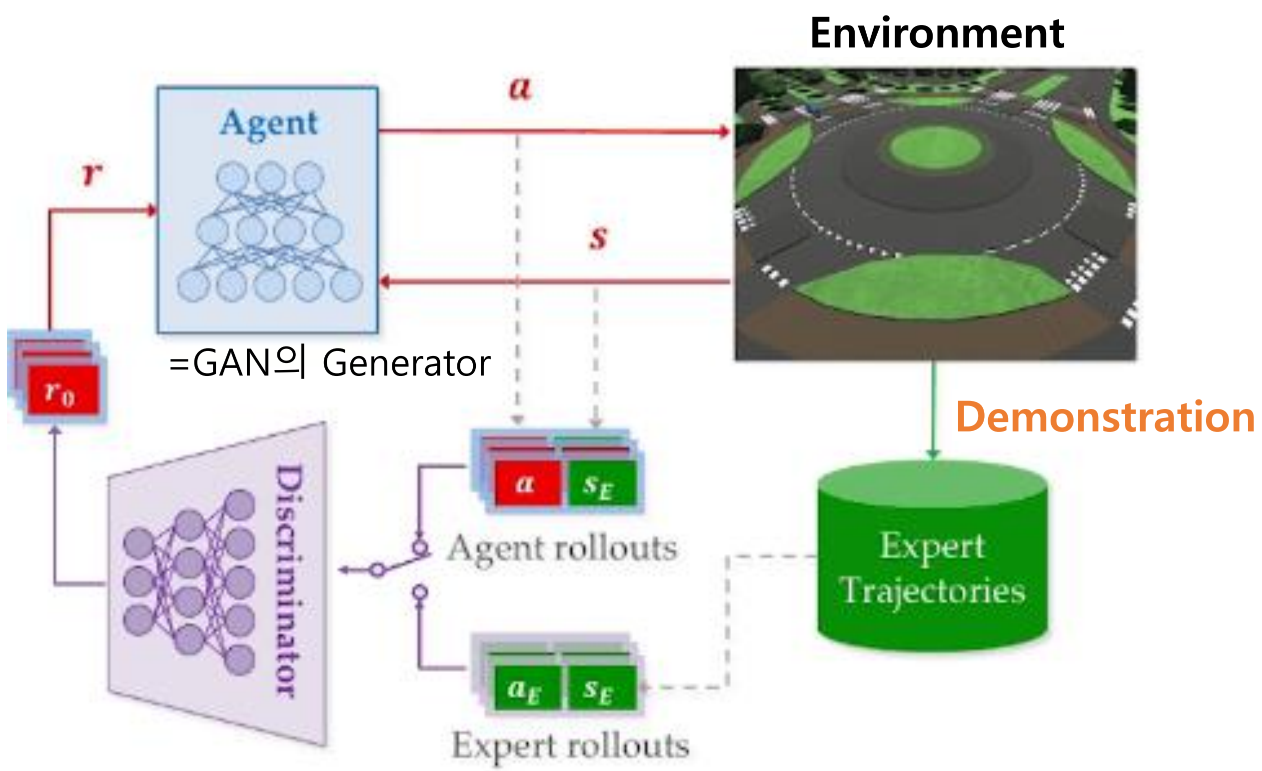
Autonomous helicopter flight is widely regarded to be a highly challenging control problem. This paper presents the first successful autonomous completion on a real RC helicopter of the following four aerobatic maneuvers: forward flip and sideways roll at low speed, tail-in funnel, and nose-in funnel. Our experimental results significantly extend the state of the art in autonomous helicopter flight. We used the following approach: First we had a pilot fly the helicopter to help us find a helicopter dynamics model and a reward (cost) function. Then we used a reinforcement learning (optimal control) algorithm to find a controller that is optimized for the resulting model and reward function. More specifically, we used differential dynamic programming (DDP), an extension of the linear quadratic regulator (LQR).

Inverse Reinforcement Learning의 한계

- Large Environment, Unknown Dynamics Model에 적용 어려움
- Reward Function, Feature 정의 방법이 성능에 많은 영향
- 다소 비효율적인 학습 절차 (IRL ↔ RL)
- 그러나 한계 극복하며 계속 발전 중

GAIL(Generative Adversarial Imitation Learning)

GAIL = GAN + RL



source: https://www.latentlogic.com/learning-from-demonstration-in-the-wild/

GAIL(Generative Adversarial Imitation Learning)

Algorithm 1 Generative adversarial imitation learning

1: **Input:** Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters θ_0, w_0

2: **for** $i = 0, 1, 2, \ldots$ **do**

(3) Agent에서 Trajectory 도출

3: Sample trajectories $\tau_i \sim \pi_{\theta_i}$

4: Update the discriminator parameters from w_i to w_{i+1} with the gradient

(4) Update Discriminator

$$\hat{\mathbb{E}}_{\tau_i}[\nabla_w \log(D_w(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_w \log(1 - D_w(s,a))] \tag{17}$$

5: Take a policy step from θ_i to θ_{i+1}, using the TRPO rule with cost function $\log(D_{w_{i+1}}(s,a))$. Specifically, take a KL-constrained natural gradient step with

(5) Update Agent's Policy

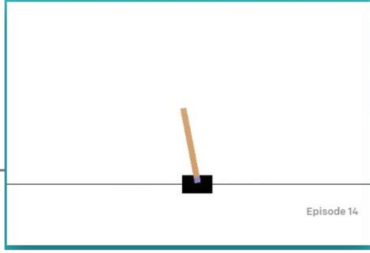
$$\begin{aligned} &\hat{\mathbb{E}}_{\tau_i}\left[\nabla_\theta \log \pi_\theta(a|s) Q(s,a)\right] - \lambda \nabla_\theta H(\pi_\theta), \\ &\text{where } Q(\bar{s},\bar{a}) = \hat{\mathbb{E}}_{\tau_i}[\log(D_{w_{i+1}}(s,a)) \mid s_0 = \bar{s}, a_0 = \bar{a}] \end{aligned} \tag{18}$$

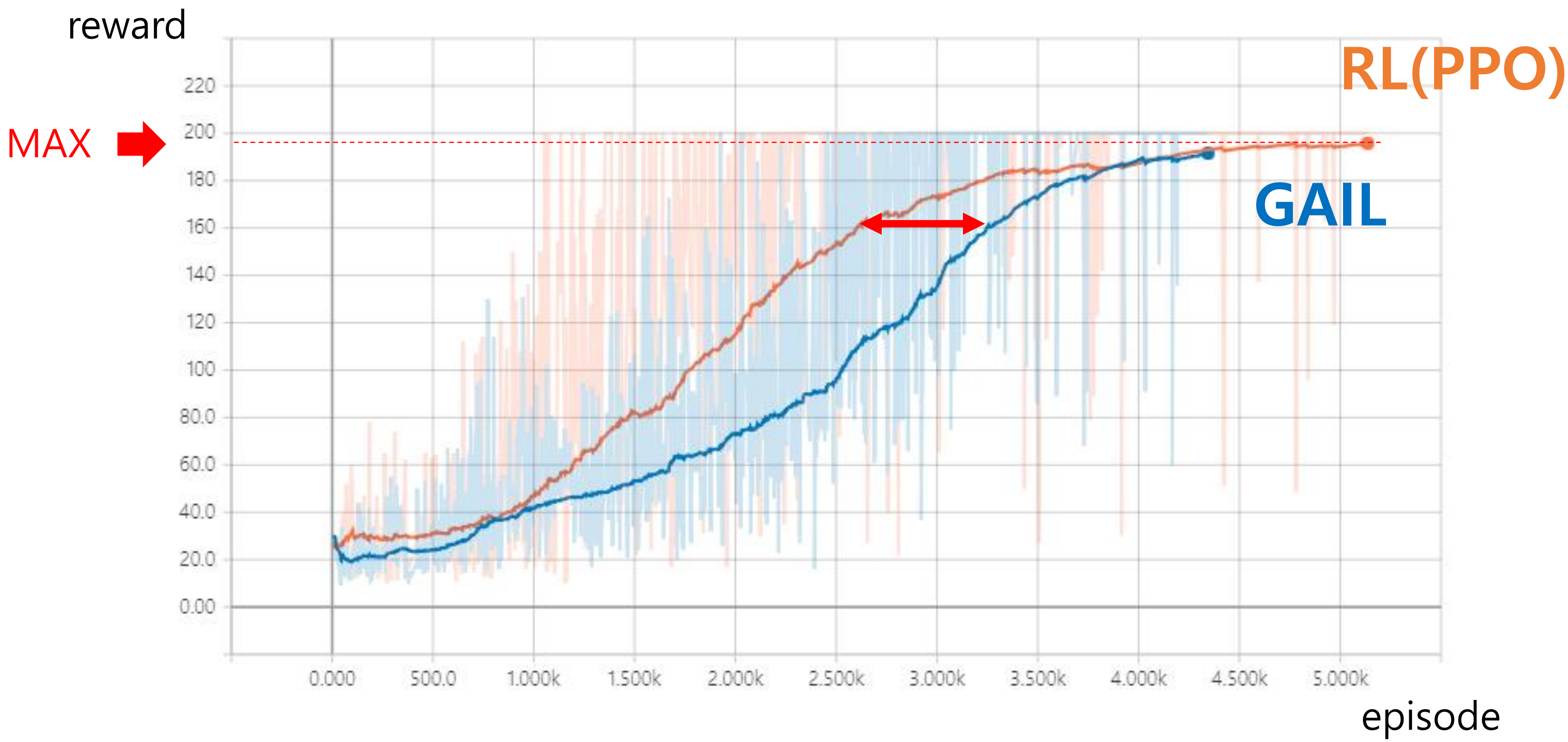
6: **end for**

+ **Cost: Wasserstein Distance**
+ **Optimizer: COCOB**
+ **RL: PPO (Proximal Policy Optimization)**

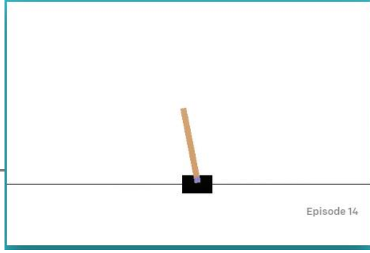
source: https://www.latentlogic.com/learning-from-demonstration-in-the-wild/

GAIL 실험: CartPole

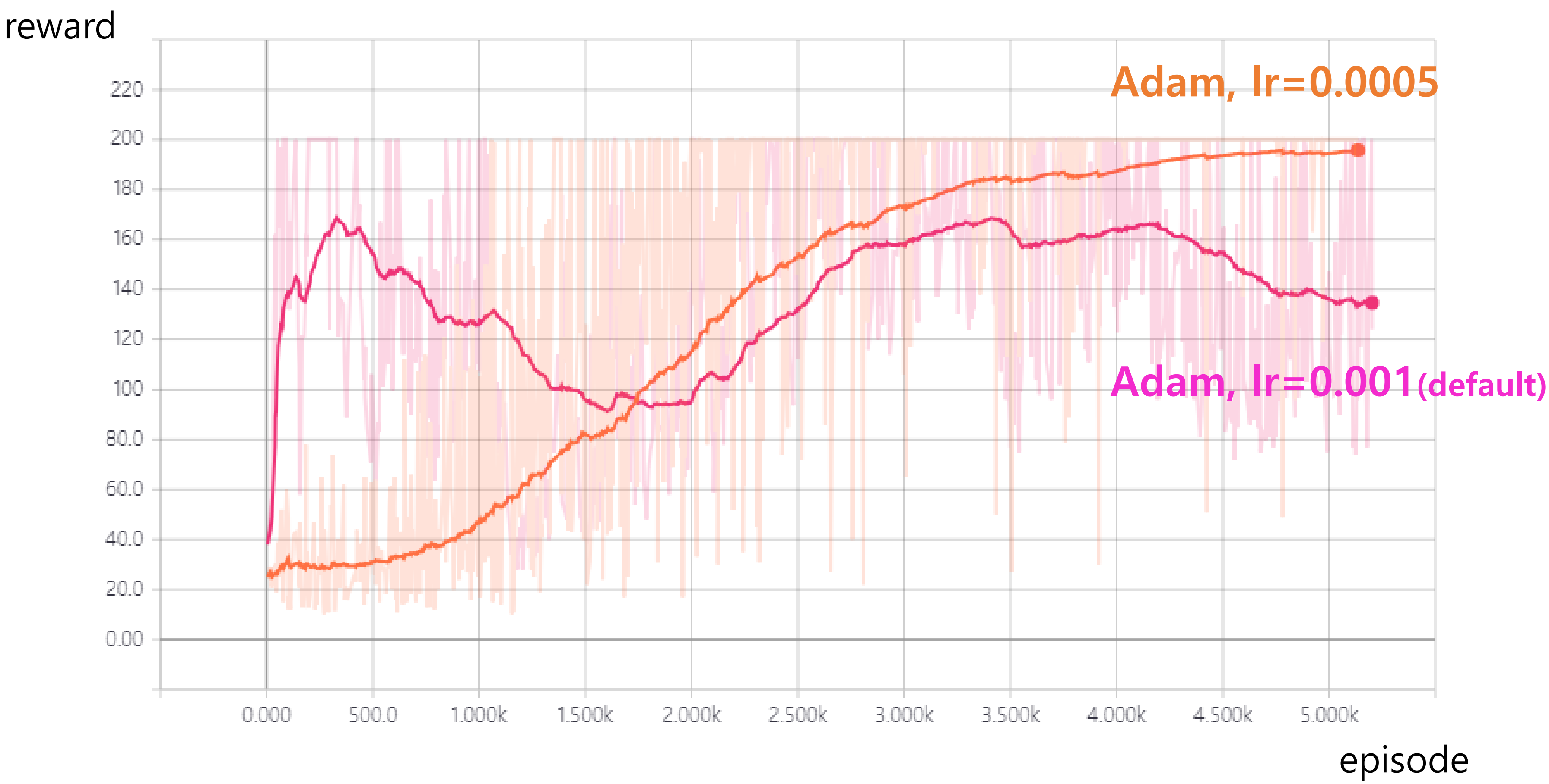




+ Optimizer 적용 실험: PPO-CartPole

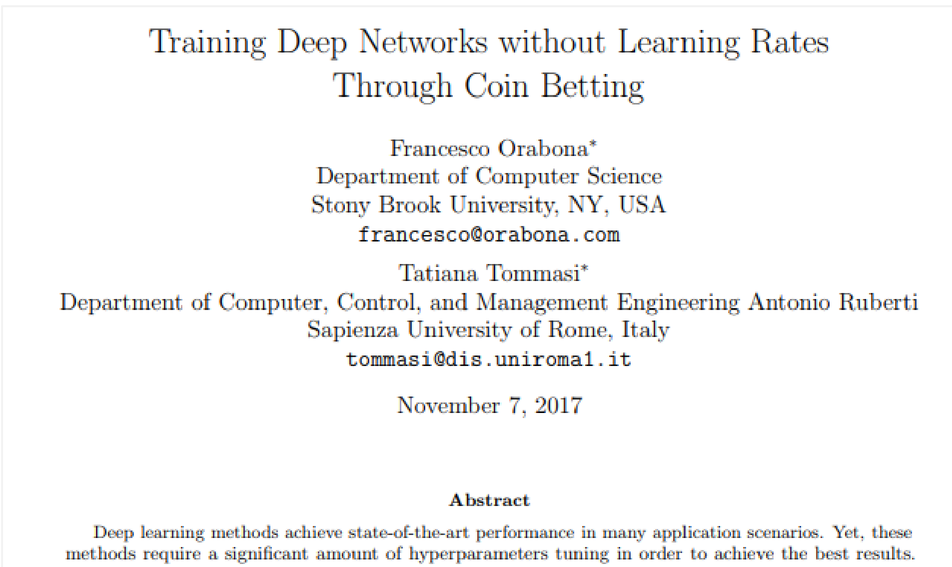


AS-IS



COCOB(‘17): ‘Learning Rate’가 불필요한 Optimizer

COntinuous COin Betting (COCOB)


Training Deep Networks without Learning Rates
Through Coin Betting

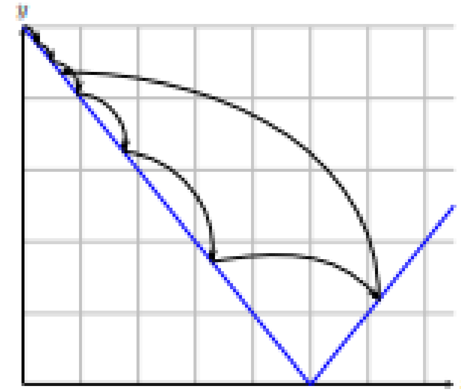
Francesco Orabona*
Department of Computer Science
Stony Brook University, NY, USA
francesco@orabona.com

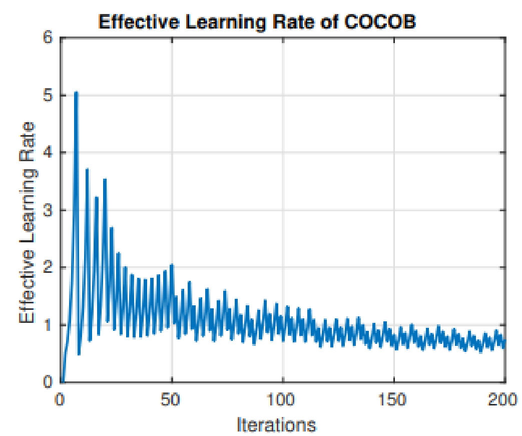
Tatiana Tommasi*
Department of Computer, Control, and Management Engineering Antonio Ruberti
Sapienza University of Rome, Italy
tommasi@dis.uniroma1.it

November 7, 2017

Abstract

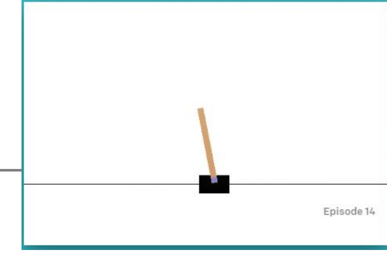
Deep learning methods achieve state-of-the-art performance in many application scenarios. Yet, these methods require a significant amount of hyperparameters tuning in order to achieve the best results.

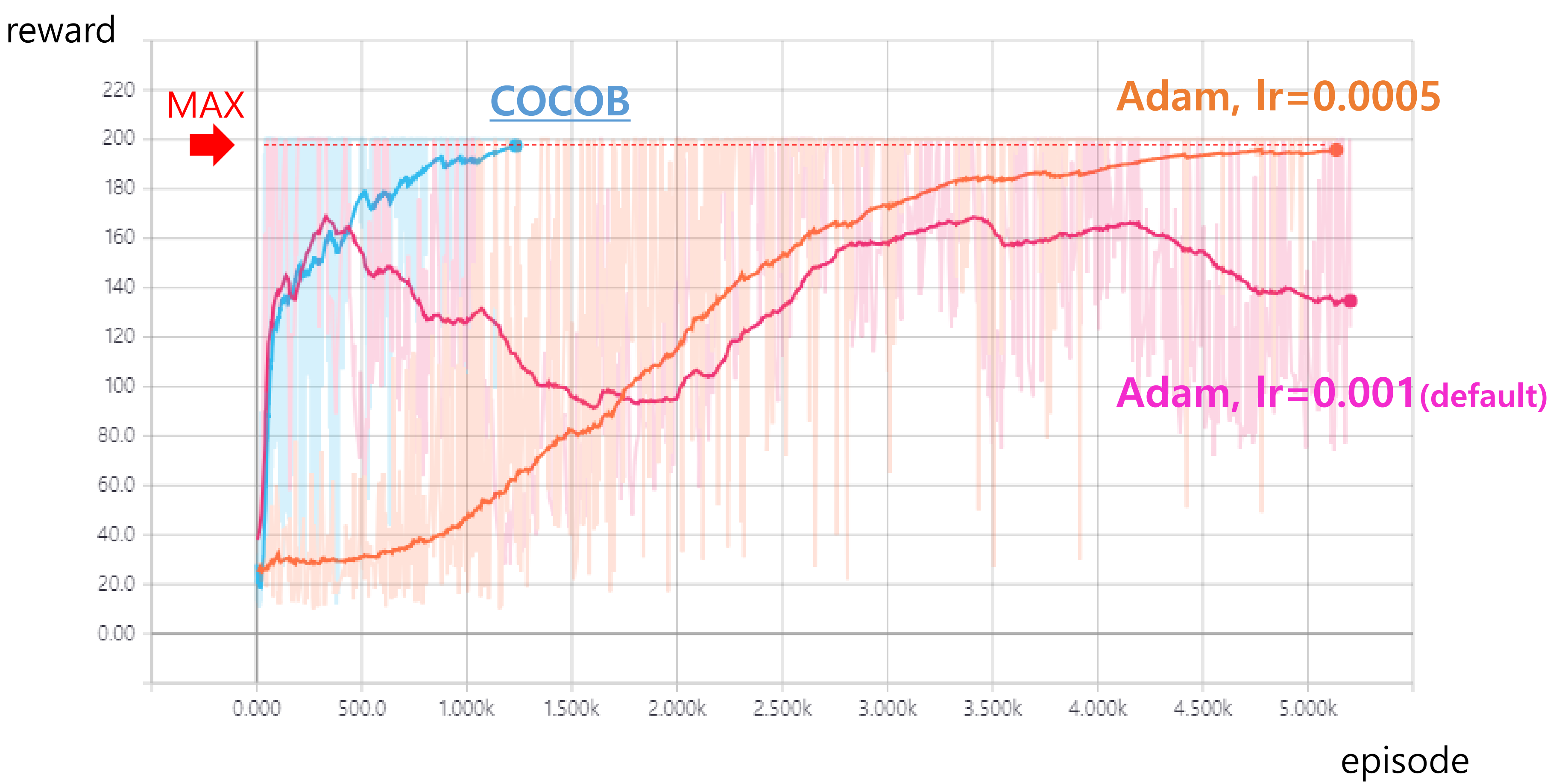




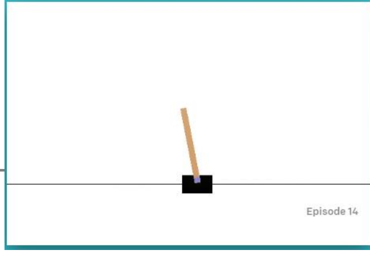
source: https://arxiv.org/abs/1705.07795

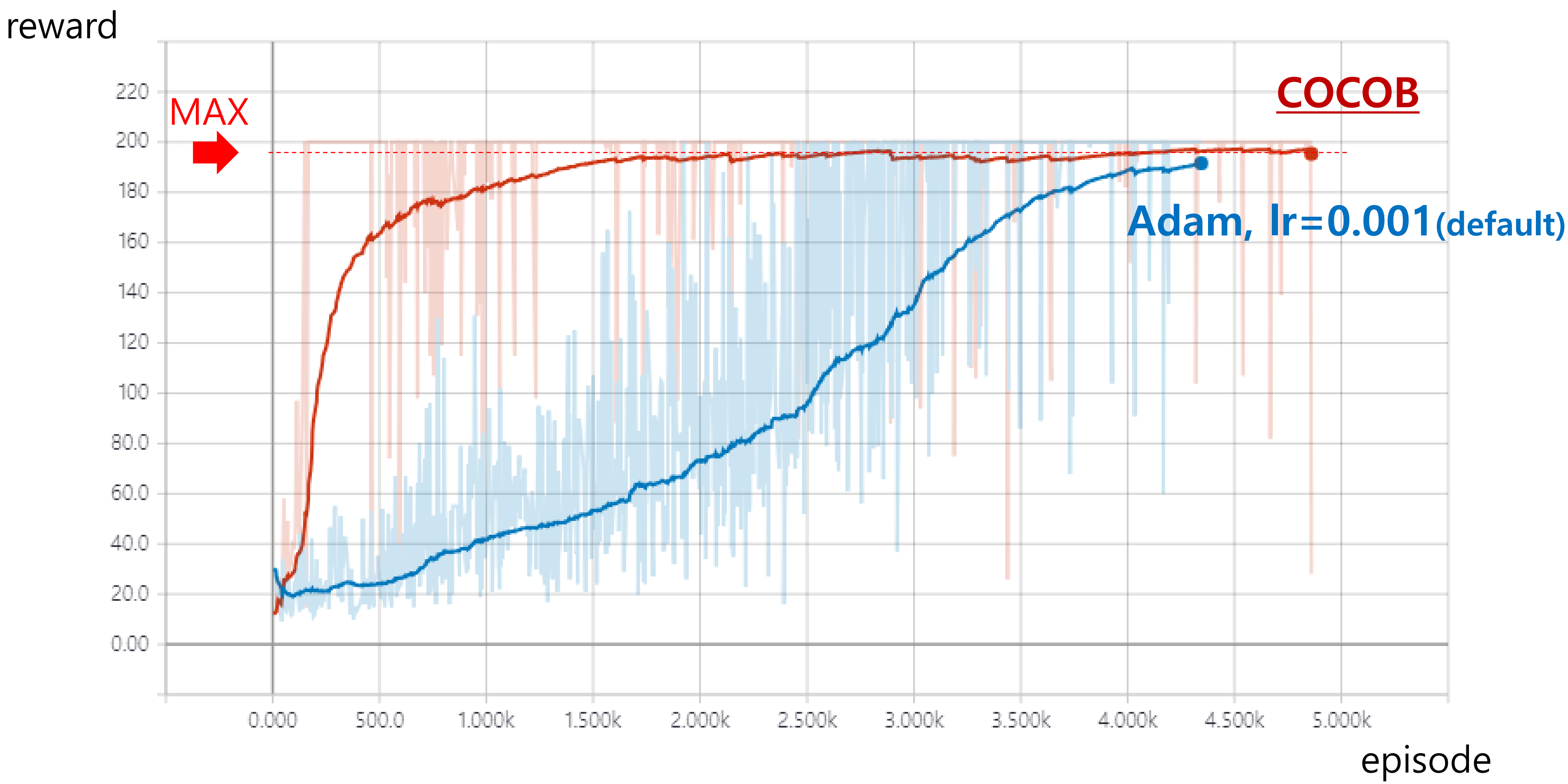
COCOB 실험: PPO – CartPole



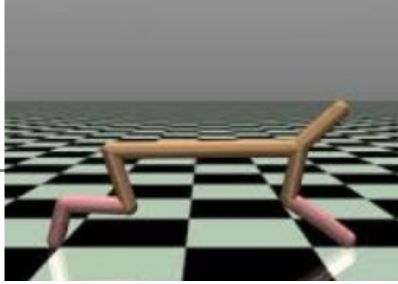


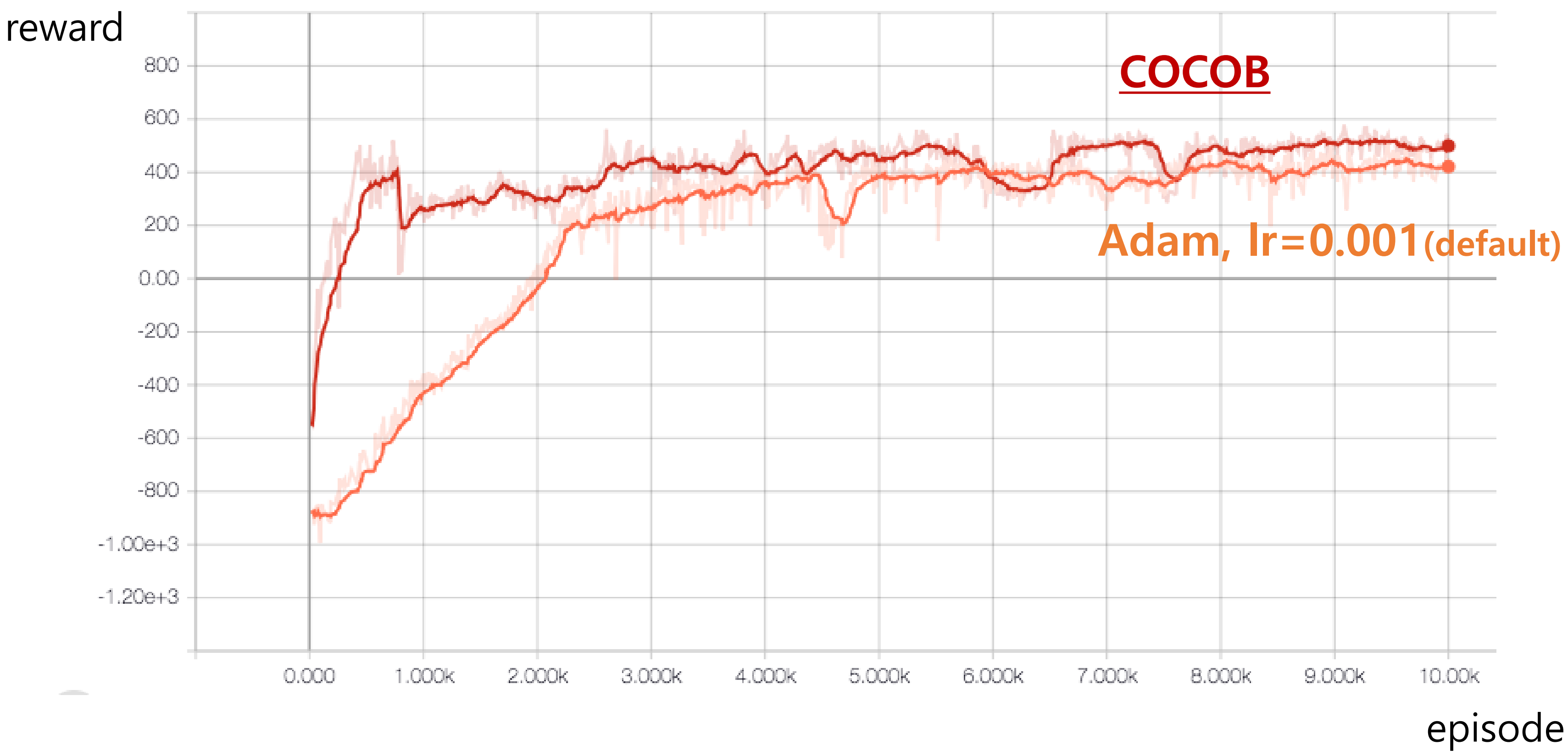
COCOB 실험: GAIL – CartPole



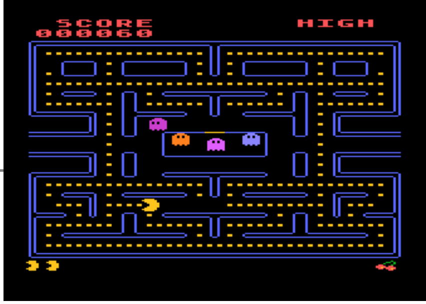


COCOB 실험: GAIL – Mujoco HalfCheetah

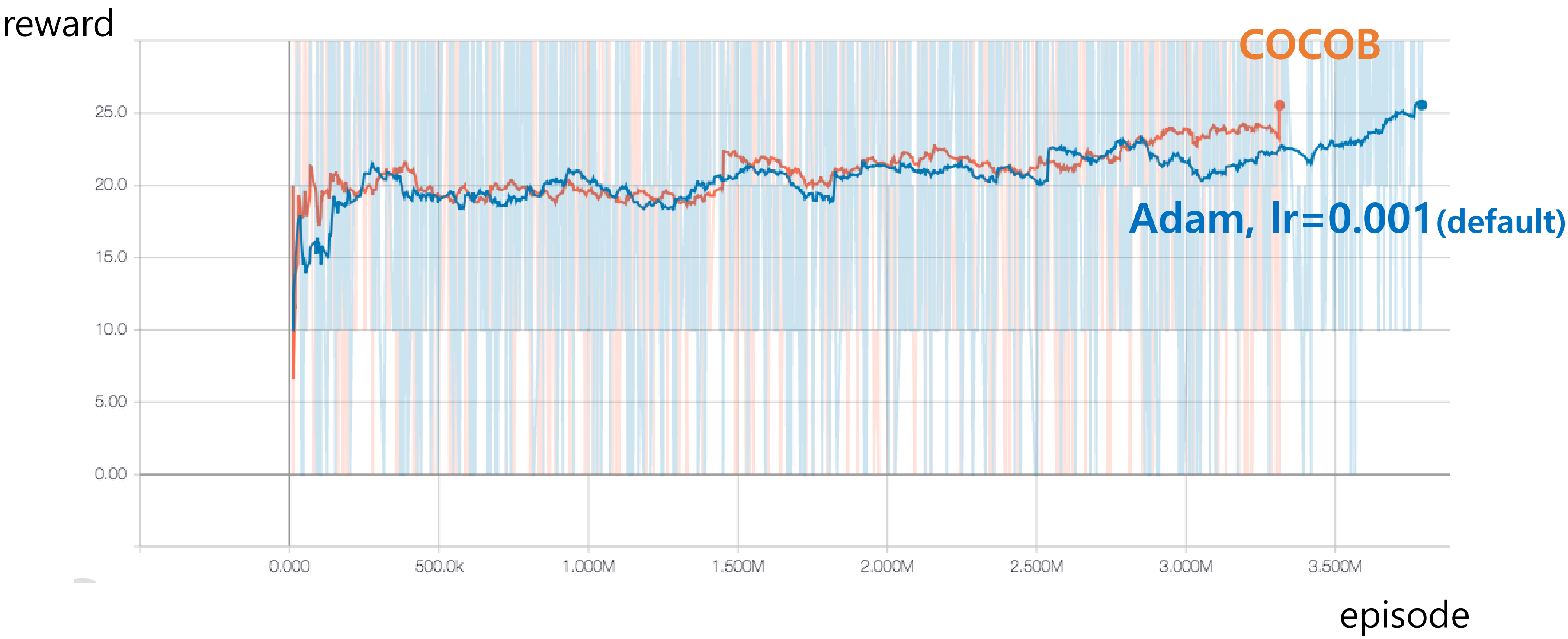




COCOB 실험: DQN – Atari Packman



DQN(Deep Q-Network)

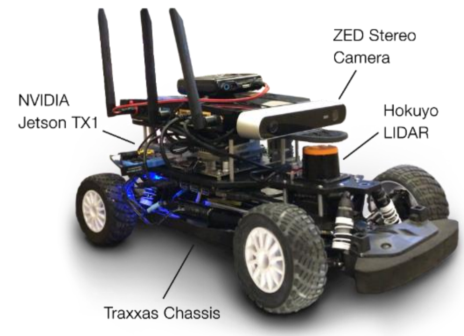


DriveGAIL 구현

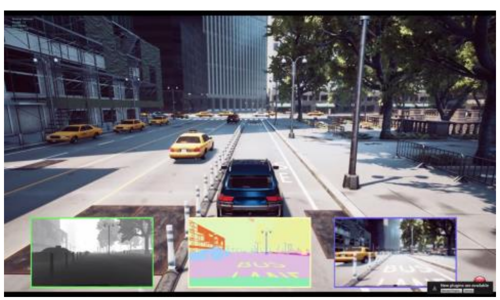
- Agent가 GAIL 을 통해 <u>자율주행 시뮬레이터를 운전</u>하도록 학습시킴
- Environment 개발이 용이한 환경 선택

TORCS

Udacity Simulator



Beaverworks Summer Institute



AirSim

Amazon DeepRacer

source: https://augustt198.github.io/bwsi-report/

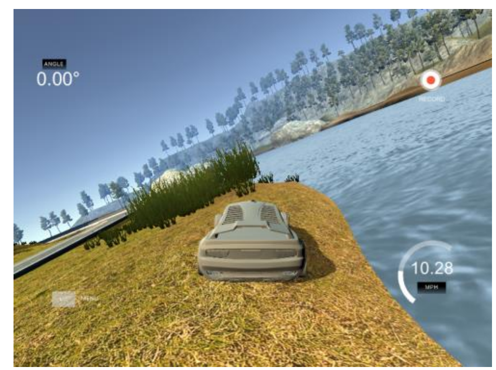
Udacity Self-driving Car Simulator

도로 주행

- 다양한 OS 지원 / Unity기반
- 자유도가 높은 환경
- 장애물 충돌 감지 물리엔진
- 트랙길이: 약 750m

다리

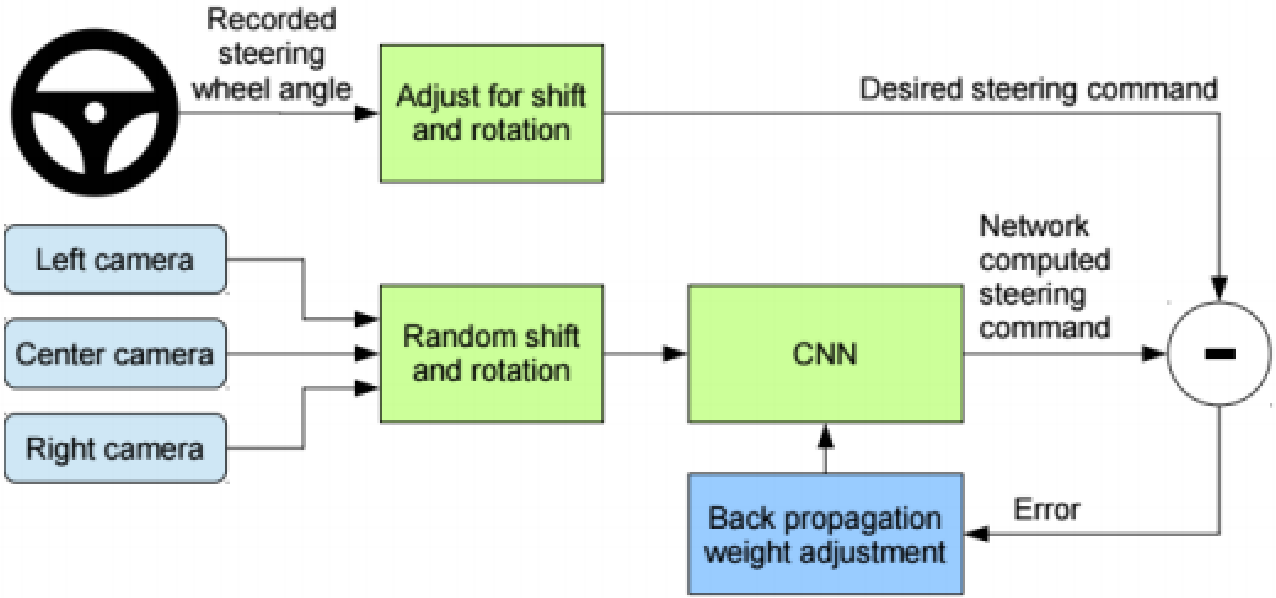
언덕



호수

Simulator의 기본 매커니즘

‘End to End Learning for Self-Driving Cars‘, NVidia, 2016



Simulator Setteing

Observation: - Left / Center / Right 전방카메라

- 회전각 / 현재속도(최대: 30.19 mph)

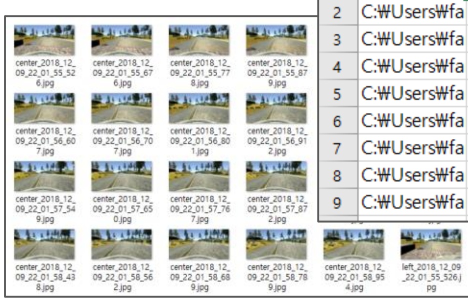
Action: - 회전각 / 목표속도

320 pixel

160 pixel

Saved Trajectories

JPG



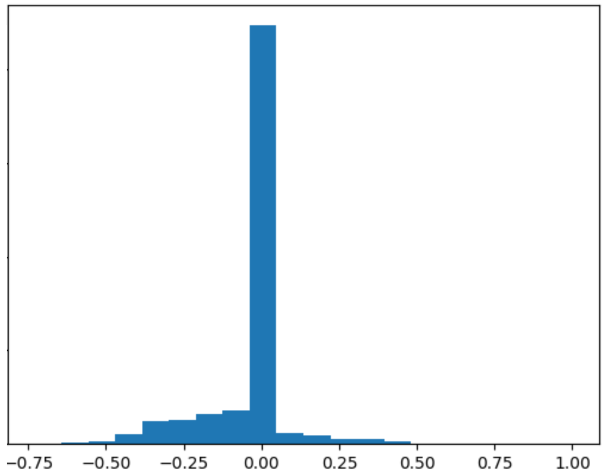
	A	B	C	D	E	F	G
1	C:₩Users₩fa	C:₩Users₩fa	C:₩Users₩far	0	0	0	10.30848
2	C:₩Users₩fa	C:₩Users₩fa	C:₩Users₩far	0	0	0	10.14176
3	C:₩Users₩fa	C:₩Users₩fa	C:₩Users₩far	0	0	0	10.03891
4	C:₩Users₩fa	C:₩Users₩fa	C:₩Users₩far	0	0.3031699	0	10.28023
5	C:₩Users₩fa	C:₩Users₩fa	C:₩Users₩far	0	0.6790172	0	10.94729
6	C:₩Users₩fa	C:₩Users₩fa	C:₩Users₩far	0	1	0	11.99148
7	C:₩Users₩fa	C:₩Users₩fa	C:₩Users₩far	0.3104184	1	0	13.02503
8	C:₩Users₩fa	C:₩Users₩fa	C:₩Users₩far	0	1	0	14.69397
9	C:₩Users₩fa	C:₩Users₩fa	C:₩Users₩far	0	1	0	17.2507

CSV

DriveGAIL: Trajectory 전처리

트랙 특성 상 좌회전과 직진이 많음

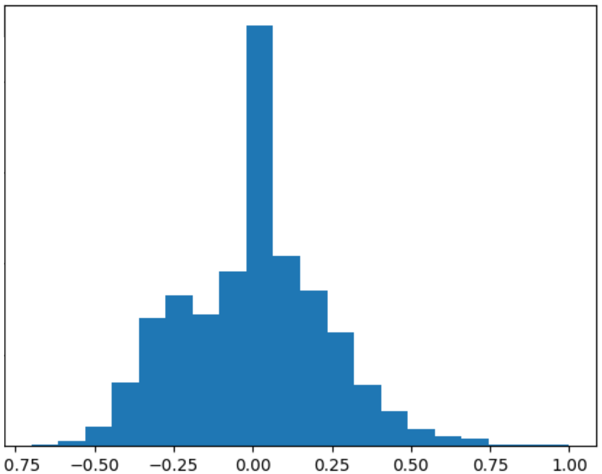
Action



좌회전 직진 우회전

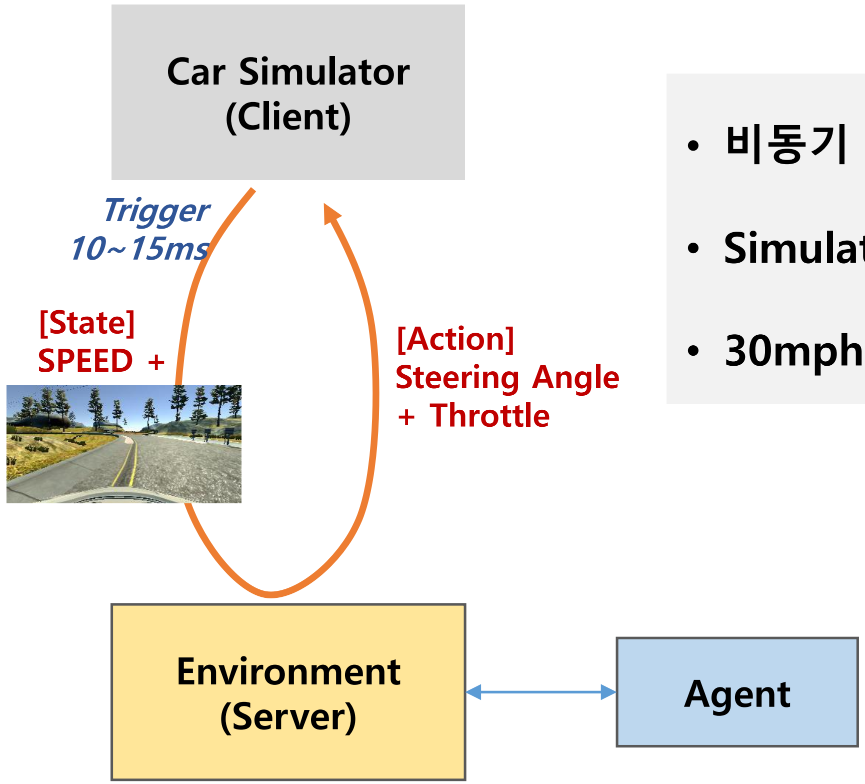
Action Imbalance 해결

Action



좌회전 직진 우회전

DriveGAIL: Environment 개발

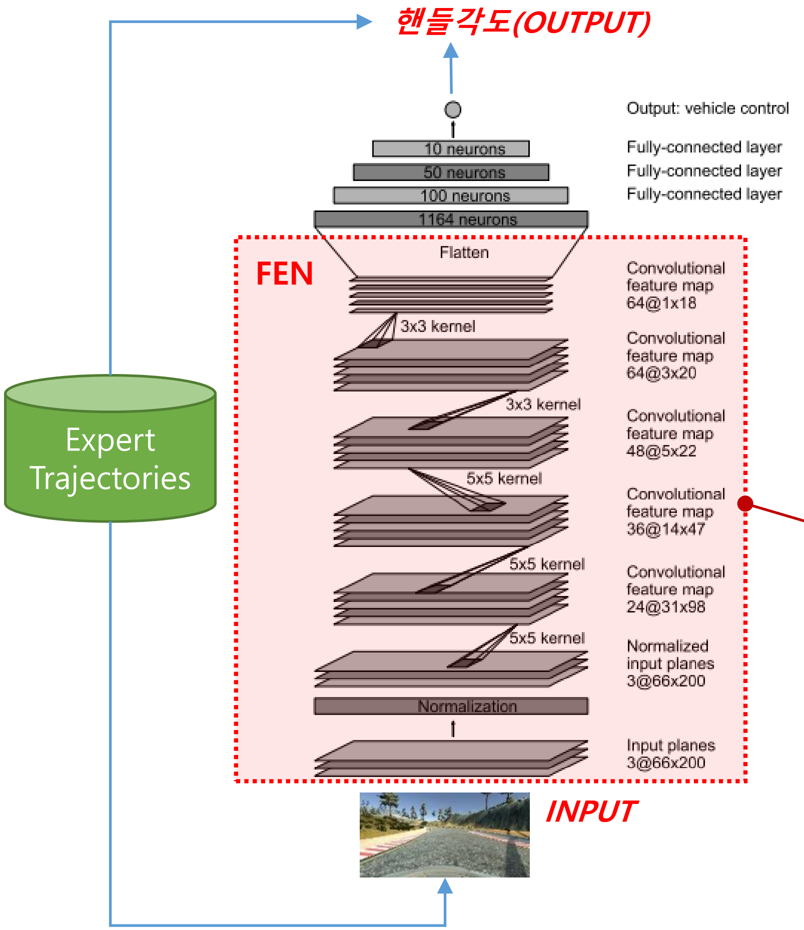


- 비동기 방식 대응을 위한 별도 Environment 개발
- Simulator의 State 전송 → Agent의 Action 리턴
- 30mph 이하일 때 한 Episode 종료

이미지를 Observation Feature로: FEN

FEN(Feature Extraction Network)

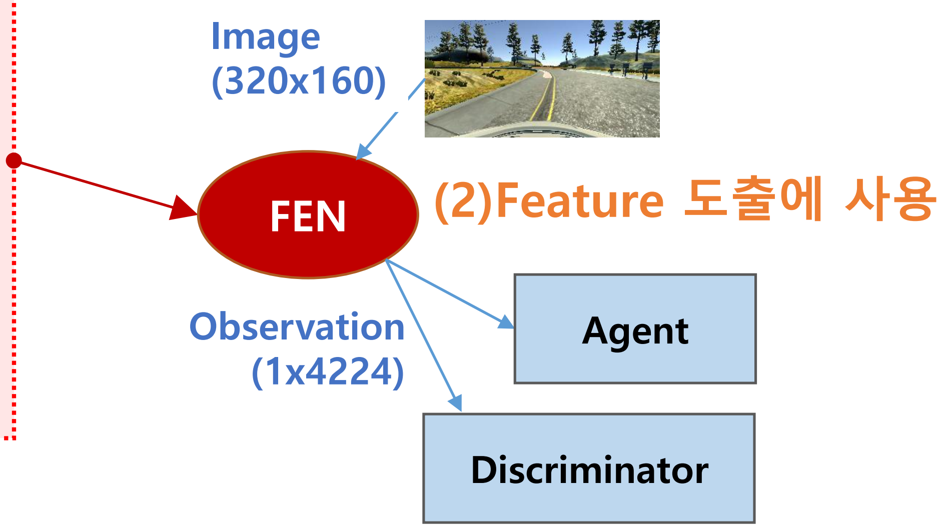
(1)Supervised Learning 학습



- **<u>Convolution Layer의 Feature</u>**를 **<u>State</u>**로 사용
- 전방 카메라 이미지의 Feature 추출 네트워크

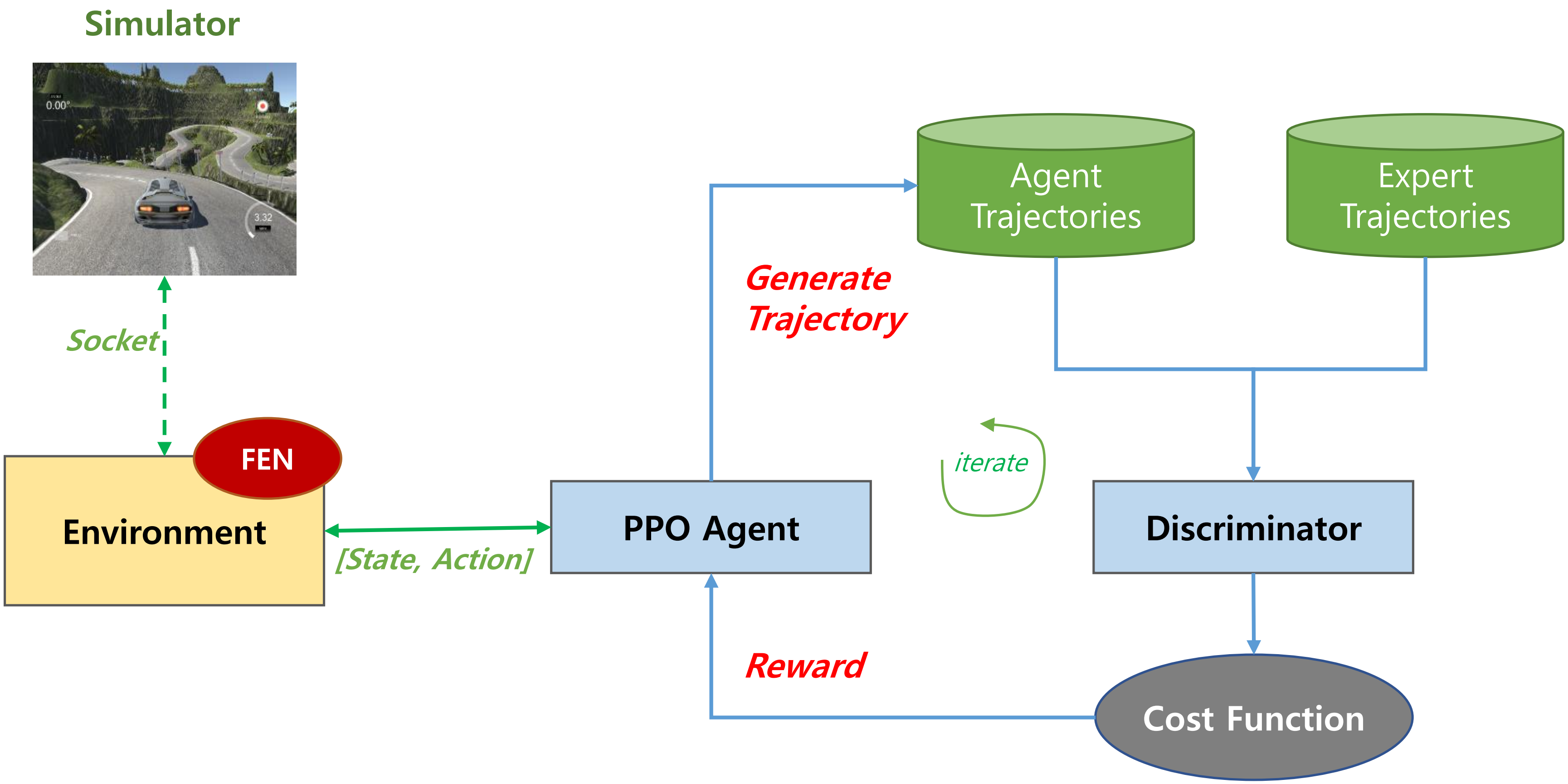
  Expert Trajectory를 <u>Behavioral Cloning</u> 통해

  Supervised Learning 방식으로 학습

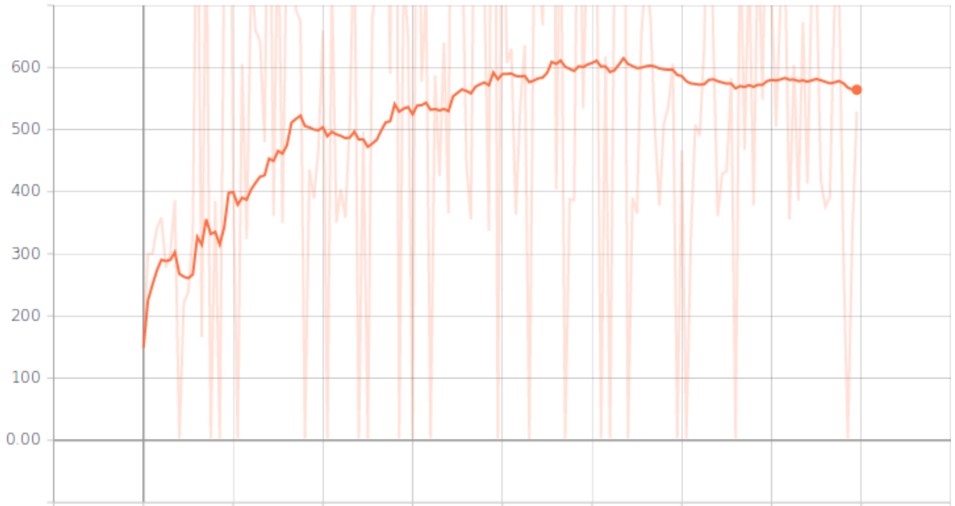


Reference: End to End Learning for Self-Driving Cars, Mariusz(nVidia), 2016

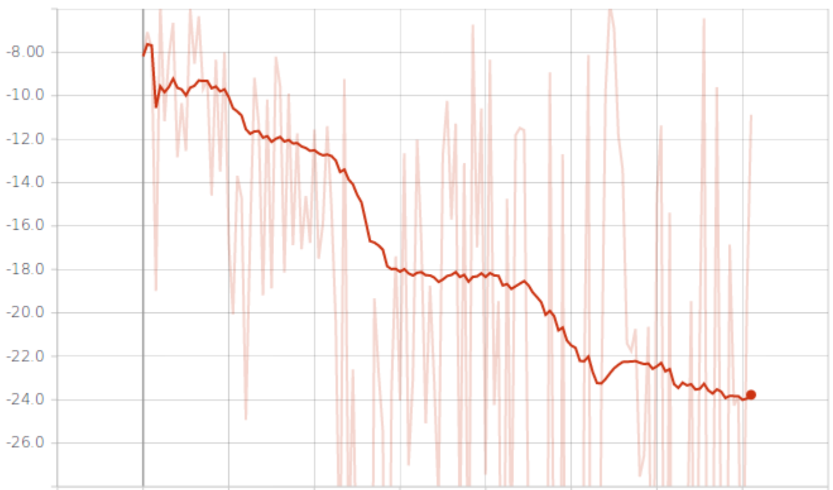
DriveGAIL: Architecture



DriveGAIL: 실험 결과



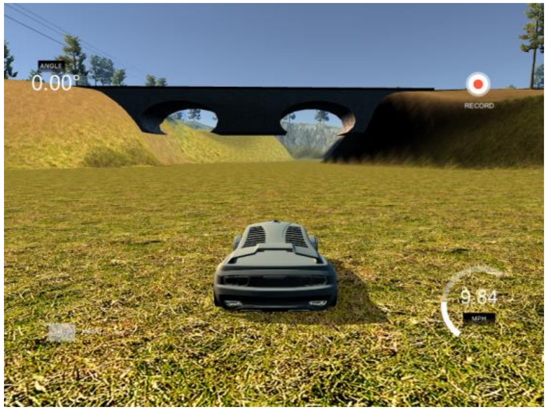
이동거리 (최대 600m)



loss

DriveGAIL: 아쉬운 점

- 코스 완주는 실패 (80% 성공)
- GAIL의 다소 불안정한 성능
- 불완전한 시뮬레이터와 물리엔진
- 부정확한 Episode 기준: 이동거리 Only



잠수 타는 중

I believe I can fly!

Q & A