TEXAS A&M UNIVERSITY
Engineering

Learning from Demonstrations: Applications to Autonomous UAV Landing and Minecraft

Committee members:
Dr. Srinivas Shakkottai
Dr. Jean Francois Chamberland
Dr. Moble Benedict

Student: Prabhasa Kalkur
Committee Chair: Dr. Dileep Kalathil

Oct 05, 2020

What is imitation learning?

TEXAS A&M UNIVERSITY
Engineering

Learning to imitate from expert behavior

Sample-efficient learning: learn behavior from as little expert data as possible

What is the presentation about?

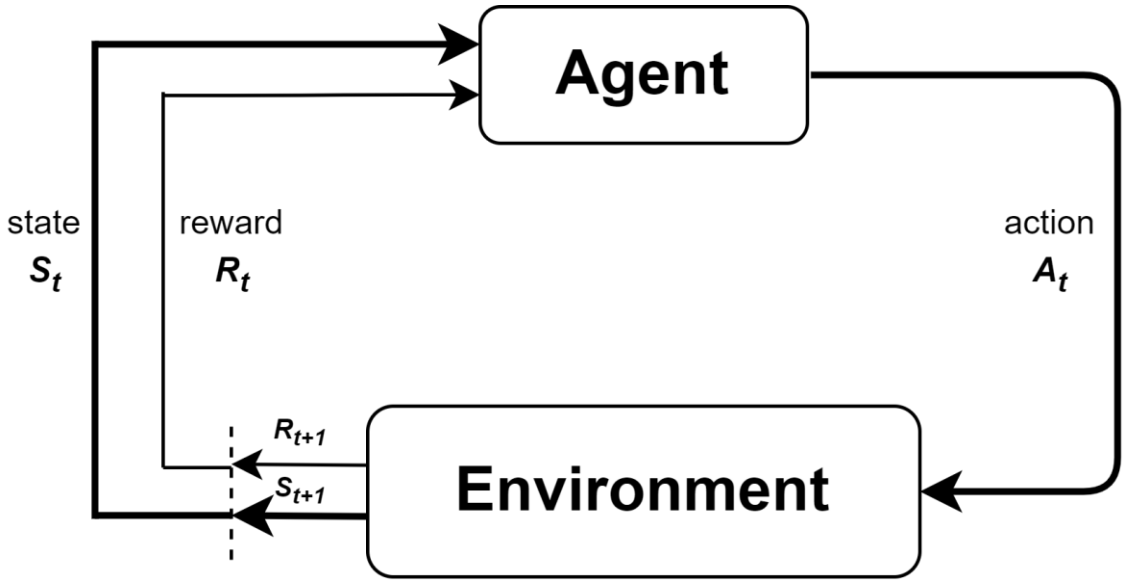
TEXAS A&M UNIVERSITY
Engineering

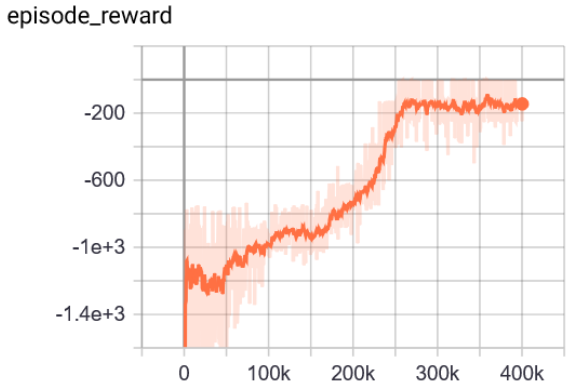
- Motivate the need for **sample-efficient** methods for learning behavior
- Pick Generative Adversarial Imitation Learning (**GAIL**) as our algorithm
- Apply GAIL to the **sparsely-rewarded** task of landing a drone (simulation)
- Discuss potential of sample-efficient learning to solve complex tasks in Minecraft

Reinforcement Learning

TEXAS A&M UNIVERSITY
Engineering

- $s, s' \in S, a \in A$. Consider tuple $[\boldsymbol{S, A}, \boldsymbol{P(s'|s,a)}, \boldsymbol{R(s,a)}, \gamma, H]$, define a policy (model) $\pi : S \rightarrow A$
  - **Reinforcement Learning (RL):** find an optimal π^* that maximizes $\sum_{t=0}^{\infty} \gamma^t R_t$





100 episodes of policy:
95/100 successful
Reward (mean, std): (**-175**, 50)

true reward/cost/score

↑

Performance metrics

↓

Rollouts: (S, A) **from** π^*

Organization of the talk

TEXAS A&M UNIVERSITY
Engineering

1. Need for sample-efficiency
2. Introduction to Imitation Learning
3. Application 1: Autonomous UAV Landing
4. Application 2: Minecraft
5. Conclusions and Future Work

Sections

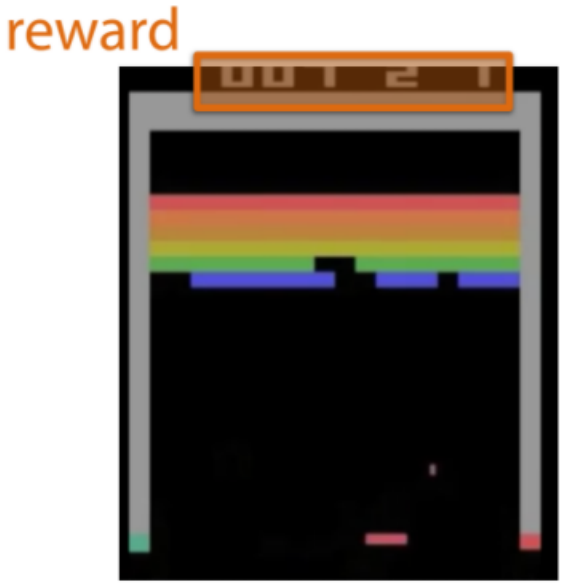
TEXAS A&M UNIVERSITY
Engineering

1. Need for sample-efficiency
2. Introduction to Imitation Learning
3. Application 1: Autonomous UAV Landing
4. Application 2: Minecraft
5. Conclusions and Future Work

Why study imitation learning?

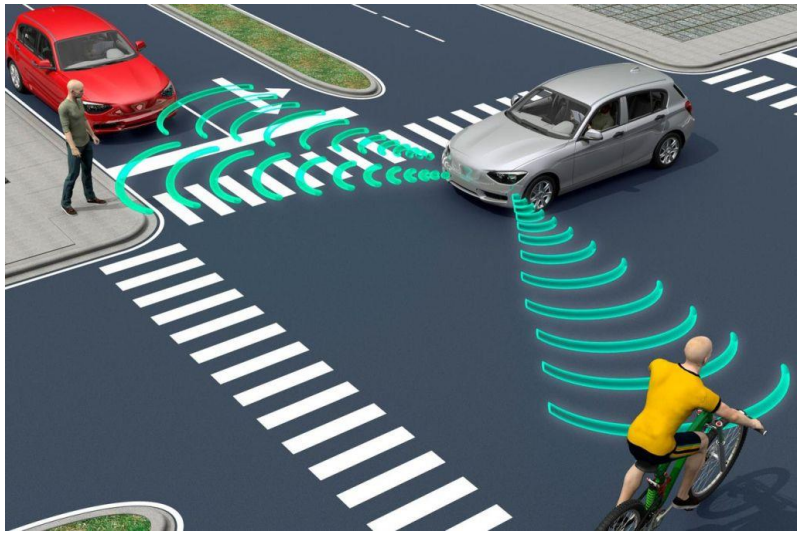
TEXAS A&M UNIVERSITY
Engineering

1. Rewards obvious in computer games: maximize score
   - Not so obvious in real-word scenarios: use a proxy instead



Mnih et al.'15

vs



Why study imitation learning?

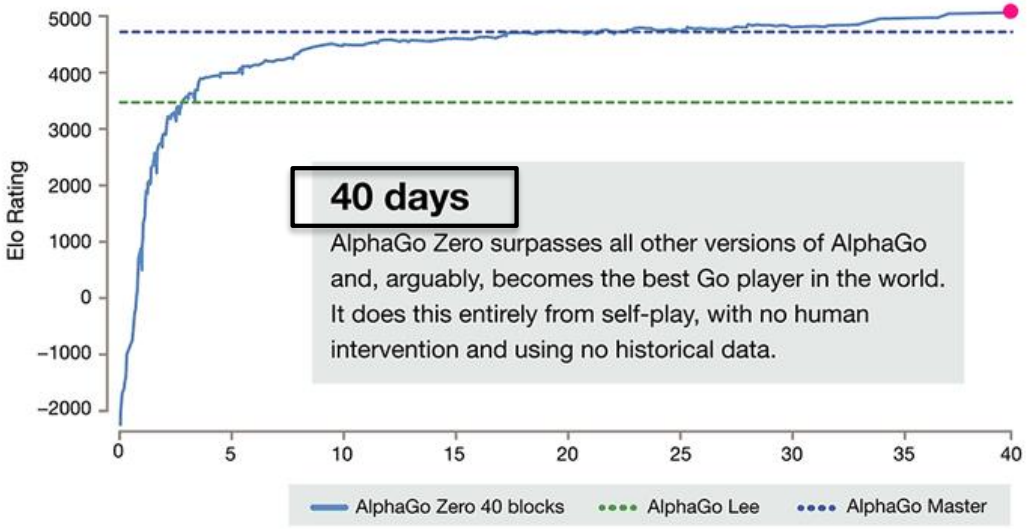
TEXAS A&M UNIVERSITY
Engineering

2. Can be easier to **demonstrate** desired behavior

Why study imitation learning?

TEXAS A&M UNIVERSITY
Engineering

3. Modern Deep-RL requires exponentially increasing number of samples: **sample-inefficient**
   - Challenging for the AI community to reproduce SOTA results



Go: AlphaGo Zero

	OPENAI 1V1 BOT	OPENAI FIVE
CPUs	60,000 CPU cores on Azure	128,000 preemptible CPU cores on GCP
GPUs	256 K80 GPUs on Azure	256 P100 GPUs on GCP
Experience collected	~300 years per day	~180 years per day (~900 years per day counting each hero separately)
Size of observation	~3.3 kB	~36.8 kB
Observations per second of gameplay	10	7.5
Batch size	8,388,608 observations	1,048,576 observations
Batches per minute	~20	~60

Dota 2: OpenAI Five

Levine et al. "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection."

Why study imitation learning?

TEXAS A&M UNIVERSITY
Engineering

3. Modern Deep-RL requires exponentially increasing number of samples
   - **Not practical, especially when env samples are expensive, and compute is limited**
   - One approach: use sample-efficient methods like Imitation Learning

Many competitions trying to promote compute and sample-efficient learning:

- **NeurIPS 2019:** Game of Drones
- **NeurIPS 2019 & 2020:** MineRL Challenge

Why study imitation learning?

TEXAS A&M UNIVERSITY
Engineering

4. How humans and animals fundamentally learn behavior

Picture credits: Sapana

Sections

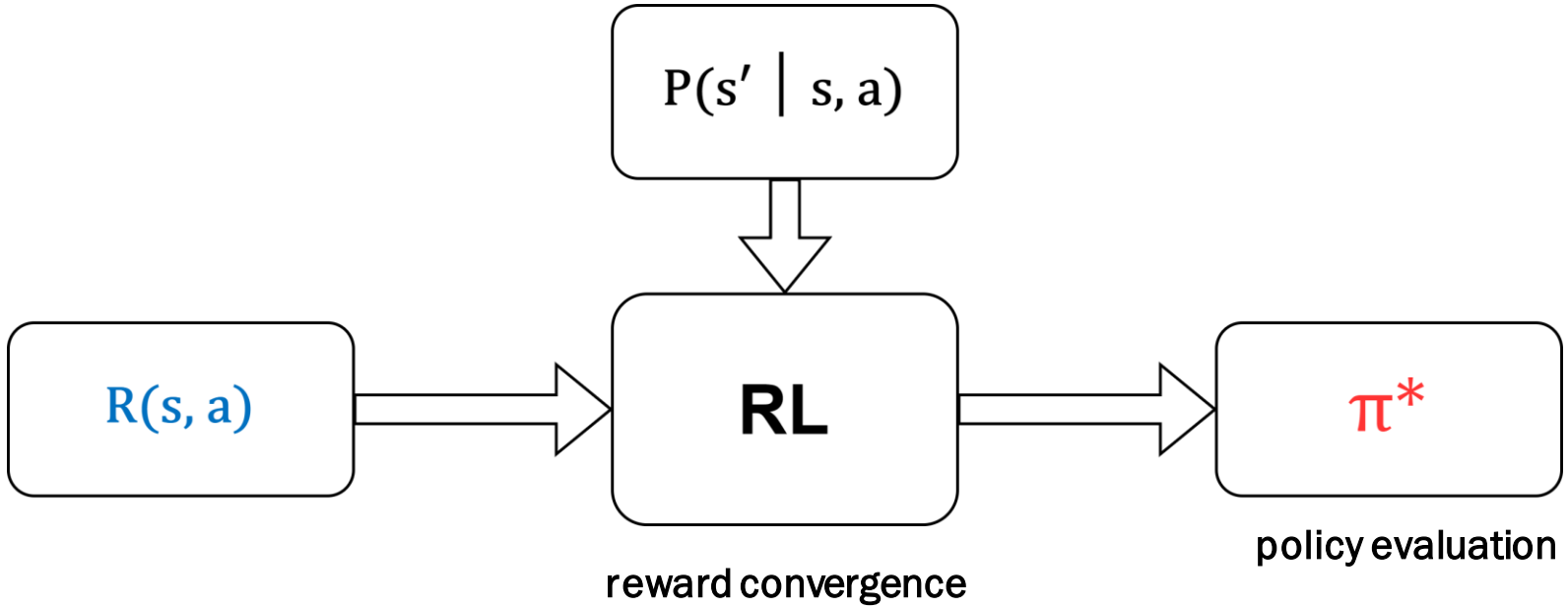
TEXAS A&M UNIVERSITY
Engineering

1. Need for sample-efficiency
2. Introduction to Imitation Learning
3. Application 1: Autonomous UAV Landing
4. Application 2: Minecraft
5. Conclusions and Future Work

RL algorithms

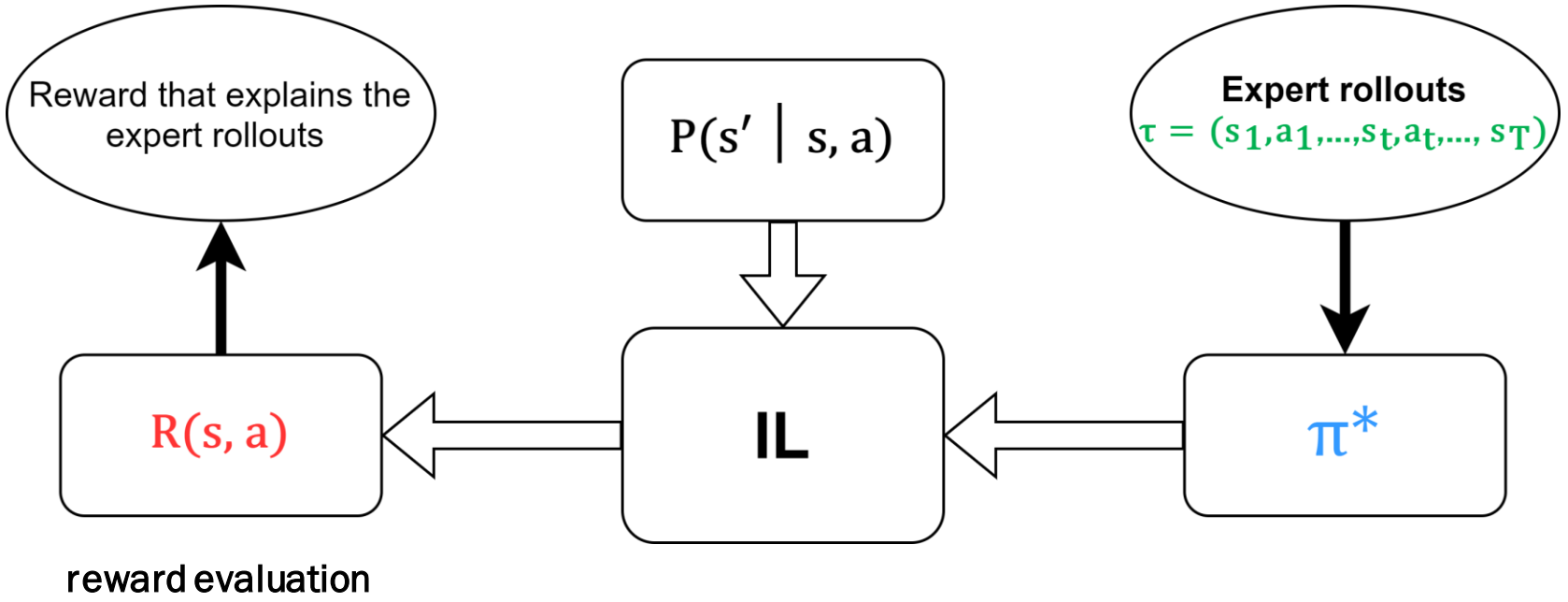
TEXAS A&M UNIVERSITY
Engineering

- $s, s' \in S, a \in A$. For MDP $[\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{P}(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a}), \boldsymbol{R}(\boldsymbol{s}, \boldsymbol{a}), \gamma]$, define a policy $\boldsymbol{\pi}: S \rightarrow A$
  - **Goal:** find an optimal $\boldsymbol{\pi}^*$ that maximizes $\sum_{t=0}^{\infty} \boldsymbol{\gamma}^t \boldsymbol{R}_t$
  - **Metric:** (i) Reward convergence, (ii) Policy evaluation (testing)



IL algorithms

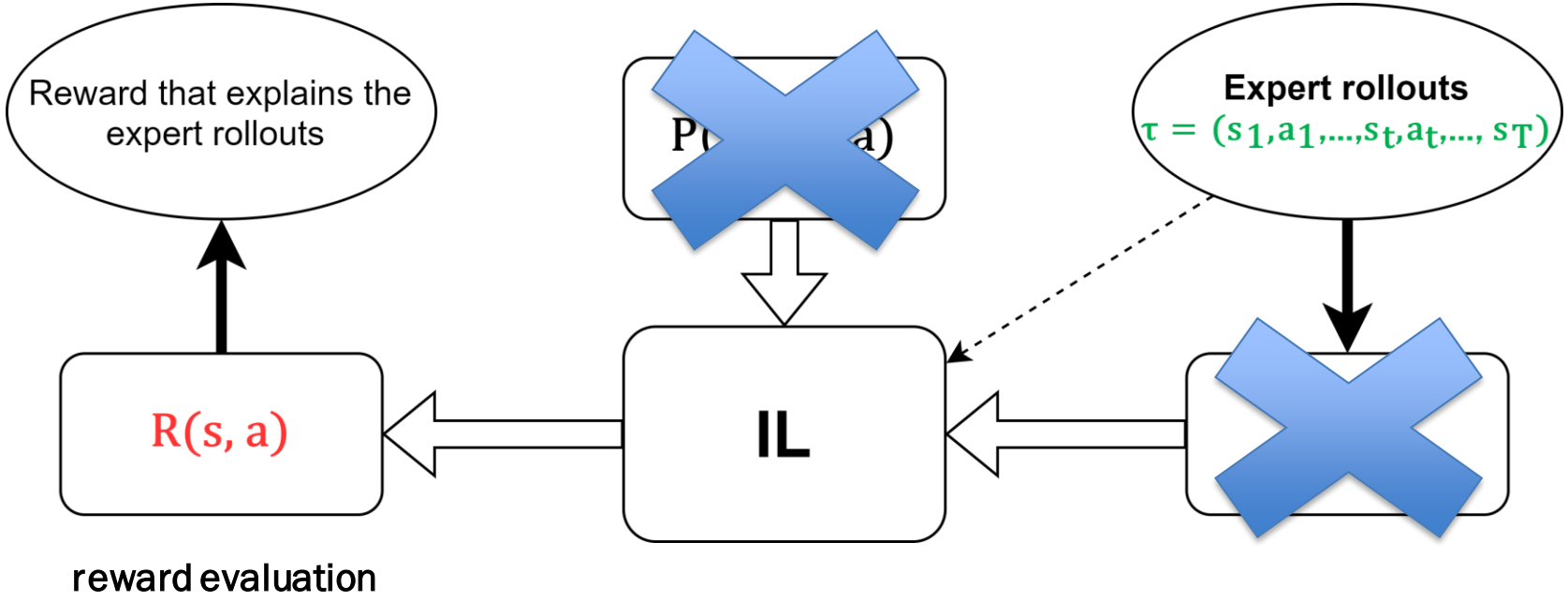
- $s, s' \in S, a \in A$. For MDP $[\boldsymbol{S, A, P(s'|s,a), R(s,a)}, \gamma]$, define a policy $\boldsymbol{\pi}: S \rightarrow A$
  - **Goal:** given $\boldsymbol{\tau = (s_0, a_0, s_1, a_1, \ldots, s_t, at, \ldots, s_T)}$ generated from a $\boldsymbol{\pi^*}$, extract its $\boldsymbol{R(s,a)}$
  - **Metric:** Reward evaluation (?)

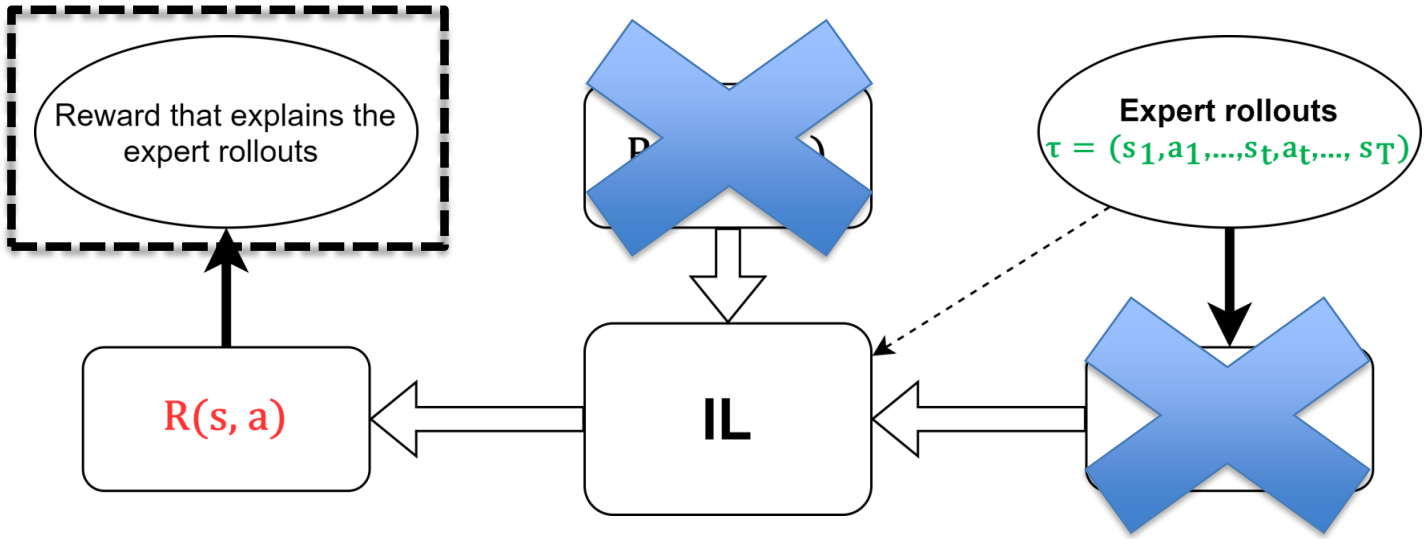


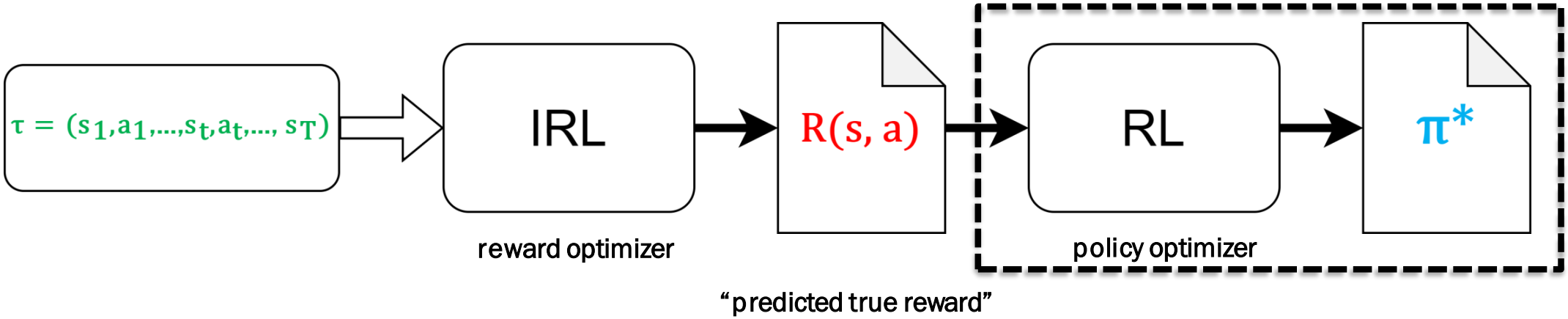
Flowchart credits: Sapana

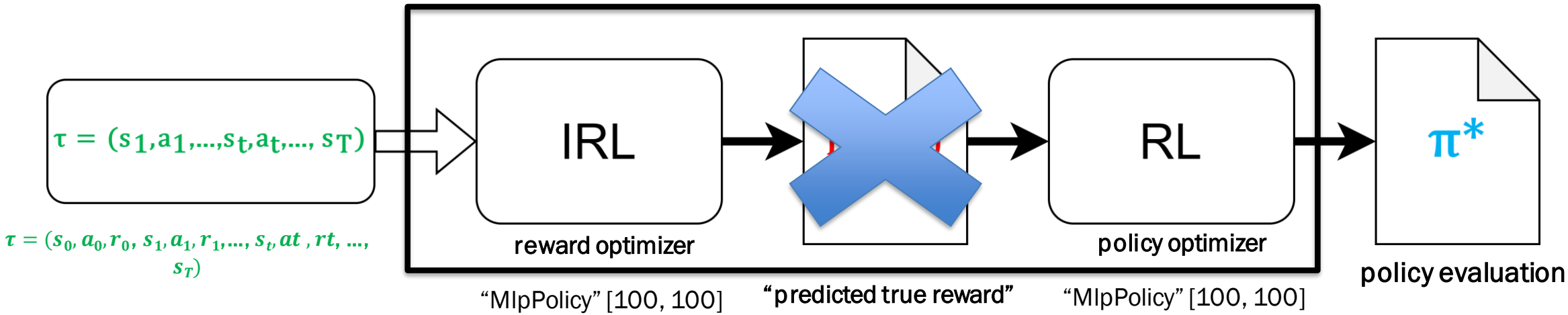
IL algorithms

- $s, s' \in S, a \in A$. For MDP $[\boldsymbol{S, A}, \boldsymbol{P(s'|s,a)}, \boldsymbol{R(s,a)}, \gamma]$, define a policy $\boldsymbol{\pi}: S \rightarrow A$
  - **Goal:** given $\boldsymbol{\tau = (s_0, a_0, s_1, a_1, \ldots, s_t, at, \ldots, s_T)}$ generated from a $\boldsymbol{\pi^*}$, extract its $\boldsymbol{R(s,a)}$
  - **Metric:** Reward evaluation (?)




Reward that explains the expert rollouts
Expert rollouts
$\tau = (s_1, a_1, \ldots, s_t, a_t, \ldots, s_T)$
R(s, a)
IL

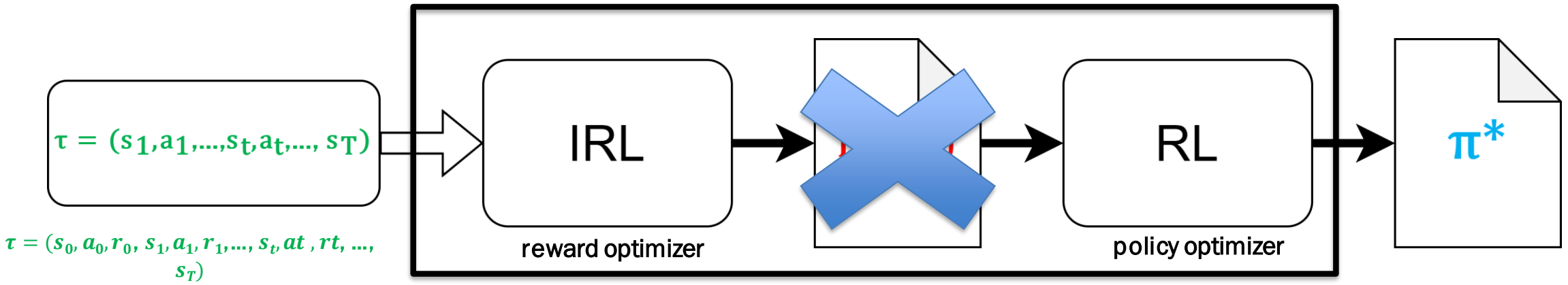

$\tau = (s_1, a_1, \ldots, s_t, a_t, \ldots, s_T)$
IRL
R(s, a)
RL
π^*
reward optimizer
policy optimizer
"predicted true reward"



100 demos of expert:
99/100 successful
Reward (mean, std): (**-165, 40**)

100 episodes of policy:
95/100 successful
Reward (mean, std): (**-175, 50**)

Generative Adversarial Imitation Learning

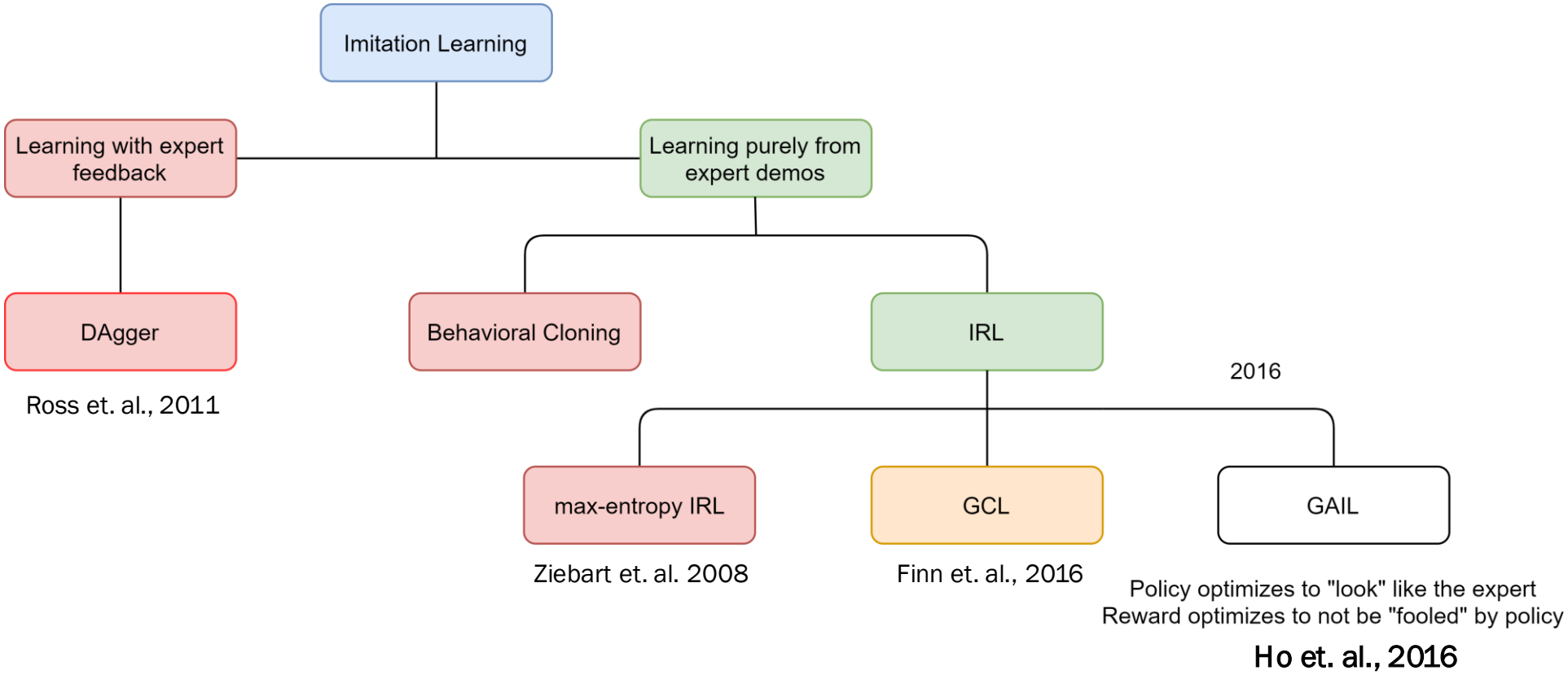


100 demos of expert:
99/100 successful
Reward (mean, std): (**-165, 40**)

100 episodes of policy:
95/100 successful
Reward (mean, std): (**-175, 50**)

Generative Adversarial Imitation Learning

Imitation Learning approaches



Generative Adversarial Imitation Learning (GAIL) is the **SOTA IL algorithm**

IL Challenges: addressed by GAIL

TEXAS A&M UNIVERSITY
Engineering

- Under-defined rewards: many functions to choose from (Max-entropy IRL ☑)
- Reward evaluation: Explicit RL for each update (GCL, GAIL ☑)
- Unknown environment dynamics (GCL, GAIL ☑)
- *Suboptimal expert: can result in undesired behavior (GAIL ☑)

*Optimal expert: (most) set of demonstrations displaying desired behavior

Some questions…

TEXAS A&M UNIVERSITY
Engineering

1. How does imitation accuracy scale with problem dimensionality and demo data?
2. How ‘smooth’ are the learned policies compared to the expert policy?
3. Can behaviors with sparse rewards be learned? At what cost?
4. Can GAIL imitate suboptimal experts? At what cost?
5. Can GAIL generalize?

Let us learn how to imitate a simple control task: balance an inverted pendulum!

Problem setup

TEXAS A&M UNIVERSITY
Engineering

Train RL -> rollout **expert** -> Train GAIL -> **policy** evaluation (test)

Goal: GAIL should be able to 'imitate' expert (optimal/suboptimal?)

Discuss: imitation accuracy, sample efficiency, effect of reward quality on learning

- **Expert trajectories / rollout / demonstrations:** sample demos [5, 10, 20]
- **Policy evaluation / rollout / testing:** Check policy performance for 100 episodes
- **Task solved each episode:** True reward for 100 consecutive episodes during training

Tools

TEXAS A&M UNIVERSITY
Engineering

- **RL library:** Stable Baselines 2.10
- **Framework:** TensorFlow 1.14
- **Hyperparameters (HPs):** RL Baselines Zoo, etc.
- **Performance metrics (learned reward vs episodes, test scores):** Tensorboard 1.14, W&B 0.10

RL/IL Algorithms

- **SAC** – Soft Actor-Critic (optimal experts)
- **TRPO** – Trust Region Policy Optimization (policy optimizer for GAIL)
- **BC** - Behavioral Cloning* (comparison with GAIL)

*with policy: "MlpPolicy" [100, 100], optimizer: Adam, batch size: 256, train-val: 70-30

OpenAI Gym and MuJoCo

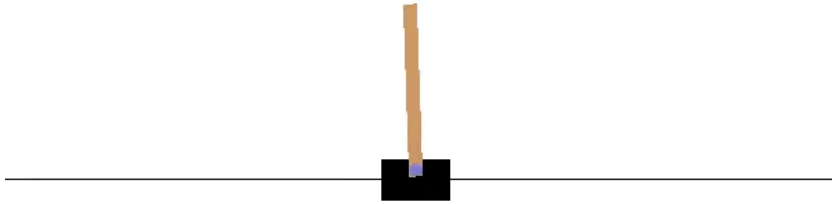
TEXAS A&M UNIVERSITY
Engineering

- **Gym:** "Toolkit for developing and comparing reinforcement learning algorithms"
- Platform for teaching agents to perform simulated tasks under a true reward
- E.g. Atari games, Robotic manipulation, control tasks

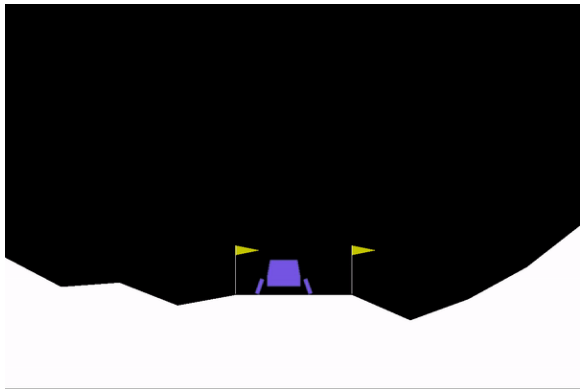
- **MuJoCo:** "A physics engine that does very detailed, efficient simulations with contacts"
- E.g. Continuous control tasks like hopping, walking, or running

- **Why is this important?** Standard benchmark tasks for testing RL, IL algorithms

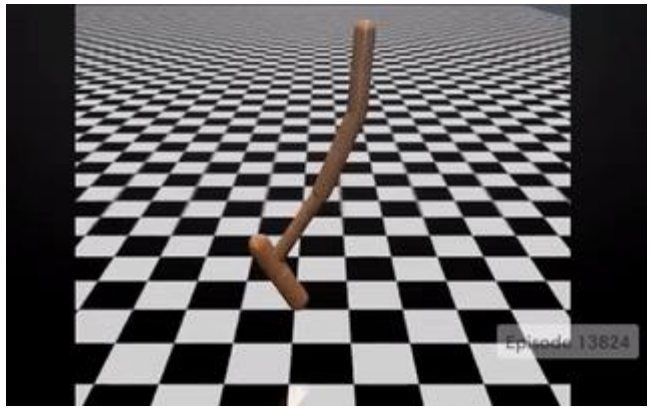
Pendulum-v0



CartPole-v1



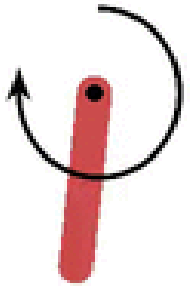
LunarLanderCts-v2



Hopper-v2

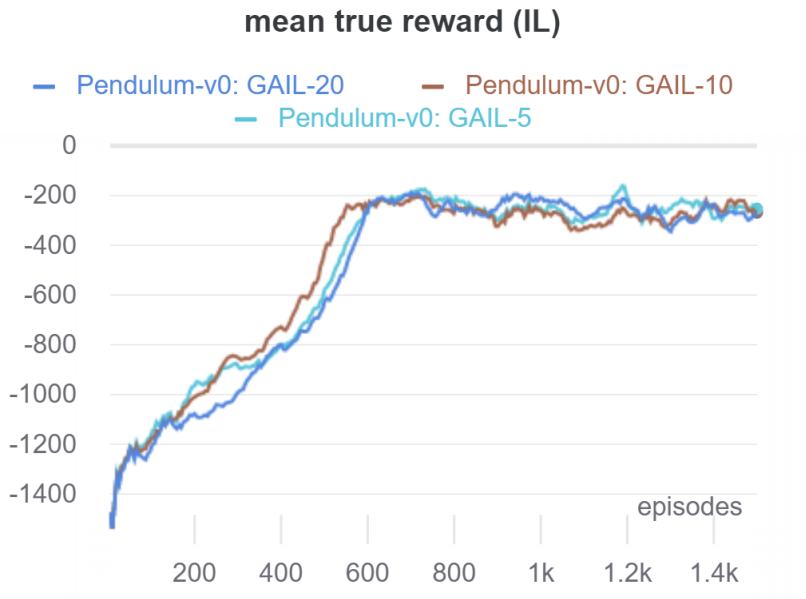
The Pendulum-v0 environment

Properties	Description
State space (cts, dim = 3)	Cosine, sine of angle θ [-1, 1], θ_0 [-8, 8]
Action space (cts, dim = 1)	Joint effort [-2, 2]
Reward	- (θ^2 + 0.1*θ_0^2 + 0.001*action^2), **dense**
Termination / Horizon	200 steps, **finite**
Solved / learned task	defined as **-200** mean reward over 100 consecutive episodes of training
Expert Trajectories for IL	**[5, 10, 20]** with reward (mean, var): **(-147, 84)**

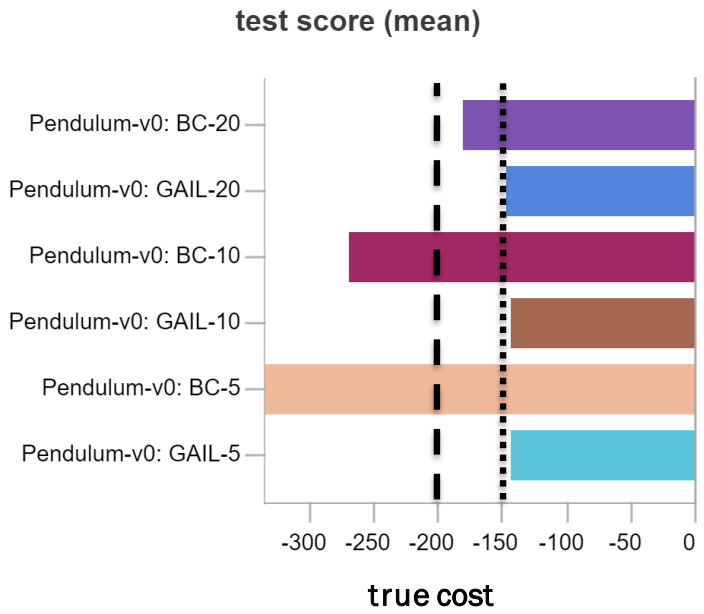


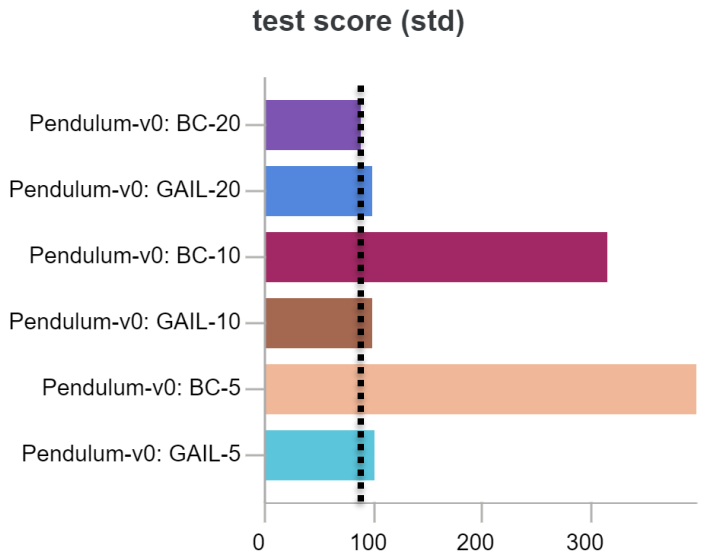
Pendulum-v0: GAIL and BC

TEXAS A&M UNIVERSITY
Engineering



reward convergence





policy evaluation

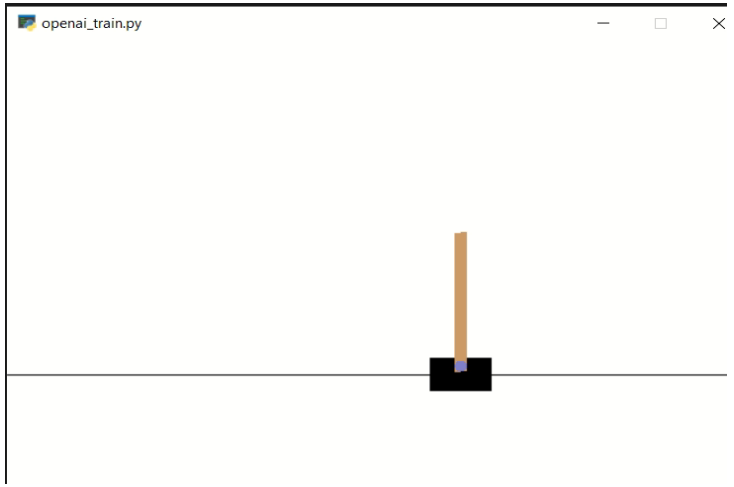
GAIL learns to achieve true cost **AND** imitate expert
GAIL score (mean, var) consistent over # demos – **sample-efficient**
BC improves over # demos, but only for optimal experts

Imitating suboptimal experts

TEXAS A&M UNIVERSITY
Engineering

Data	Optimal	Solved	Score	Mean length	Success
Expert	No	475	**(402, 134)**	402	**63**/100
GAIL policy	No	475	**(410, 178)**	411	**59**/100

- GAIL on suboptimal CartPole-v1 expert
- **Suboptimal experts can be imitated!**



Observations – GAIL on control tasks

TEXAS A&M UNIVERSITY
Engineering

- GAIL imitates Gym, MuJoCo control tasks. **Sample-efficient imitation**
- GAIL can imitate suboptimal experts
- BC policy improves with demos, but only for optimal experts
- GAIL very sensitive to choice of hyperparameter
- GAIL experiments hard to reproduce

Some questions…

TEXAS A&M UNIVERSITY
Engineering

1. How does imitation accuracy scale with dimensionality, demo data? GAIL sample-efficient (low-dim)
2. How 'smooth' are the learned policies compared to the expert policy? **Demo-dependent**
3. Can behaviors with sparse rewards be learned? At what cost?
4. Can GAIL imitate suboptimal experts? At what cost? **BC cannot.** GAIL can, with the right HPs
5. Can GAIL generalize?

Answered for a **low-dimensional, densely-rewarded, finite-horizon** control task. Let's try harder!

Sections

TEXAS A&M UNIVERSITY
Engineering

1. Need for sample-efficiency
2. Introduction to Imitation Learning
3. Application 1: Autonomous UAV Landing
4. Application 2: Minecraft
5. Conclusions and Future Work

AirSim: Autonomous UAV Navigation and Landing

APPLICATION 1

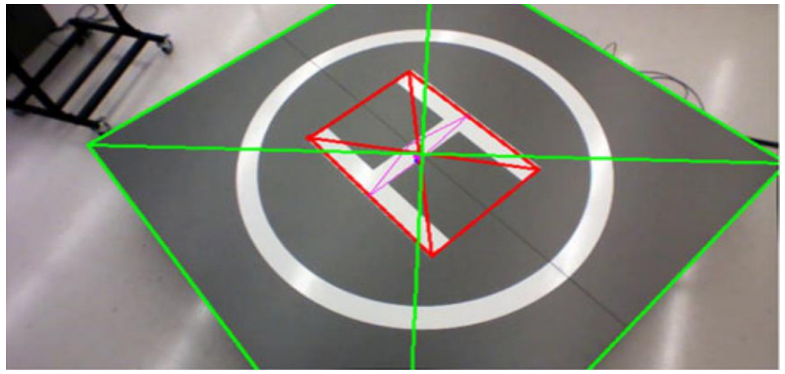
Landing on ships

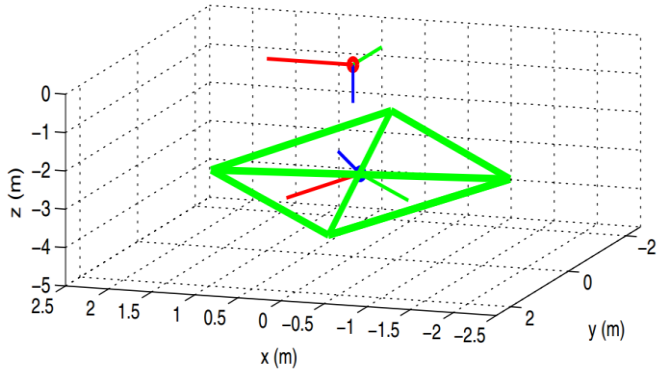
TEXAS A&M UNIVERSITY
Engineering

Landing Zone	Ground	Ship
Space	Large	Limited
Motion	None	6 DOF
Visual References	More	Less
Alternate L/D Places	Many	Less
Weather	Affected	Extremely affected

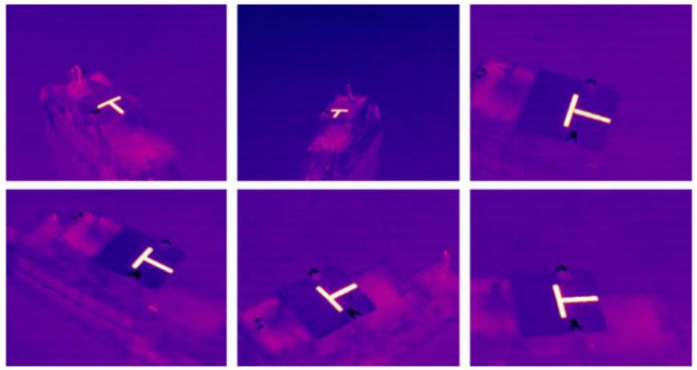
Slide credits: Bochan

Common Approaches

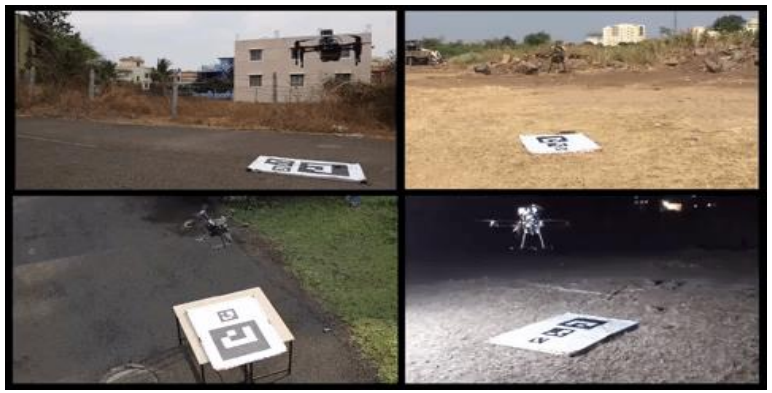




ALL of them are looking at landing spot



Computer vision-based for autonomous landing
by G.Xu, Pattern Recogn.Lett., 2009



Flytdock by flytbase company, June 8, 2018

Slide credits: Bochan

How does a pilot approach the ship?

TEXAS A&M UNIVERSITY
Engineering

Slide credits: Bochan

CONTRIBUTIONS

Landing a UAV on a ship without looking at landing spot (simulation)

Refer to a visual cue for positioning, just like a pilot

Bring pilot’s intuition and flying skills using imitation learning (GAIL)

Environment simulator

TEXAS A&M UNIVERSITY
Engineering

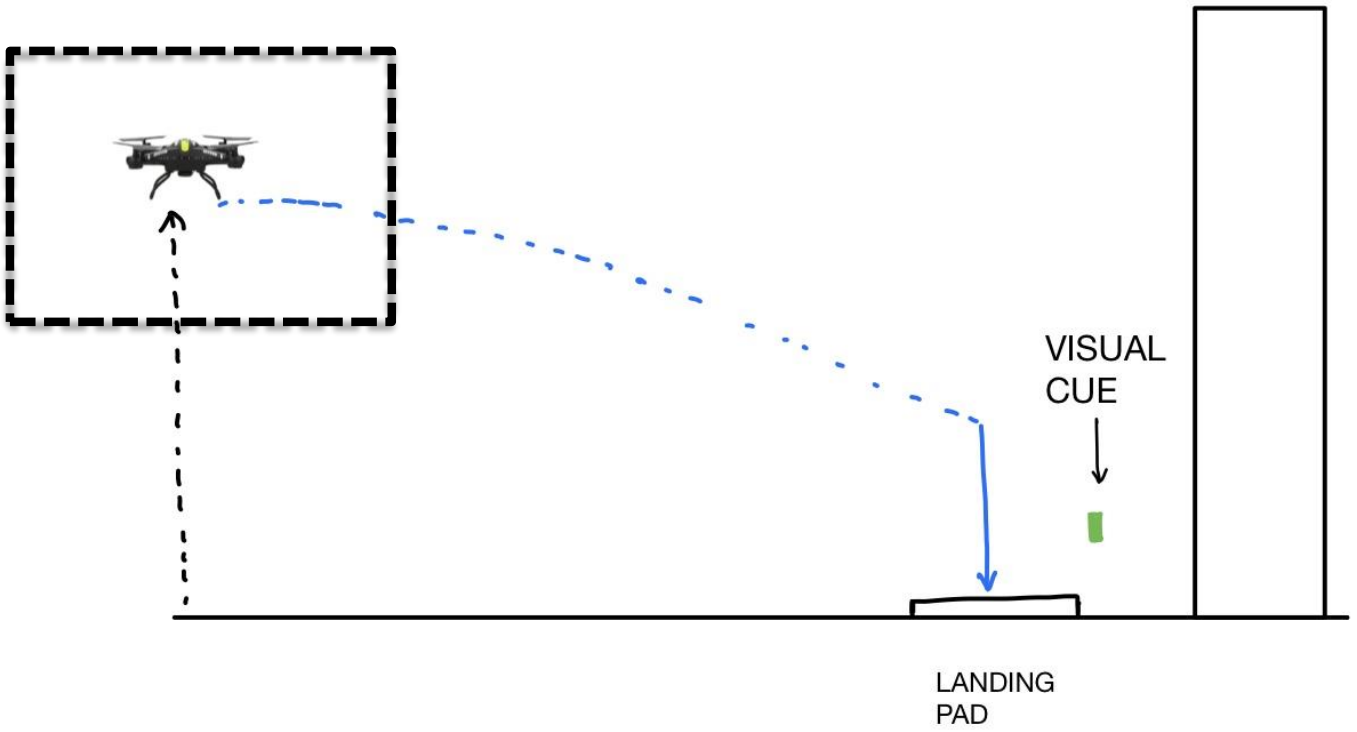
- Need a high-fidelity simulator environment for Unmanned Aerial Vehicles (UAVs)

- **Microsoft AirSim 2.0**
  - "An open source, cross platform simulator built on Unreal Engine"
  - Can integrate a flight controller for collecting demonstrations
  - Community support (NeurIPS 2019)

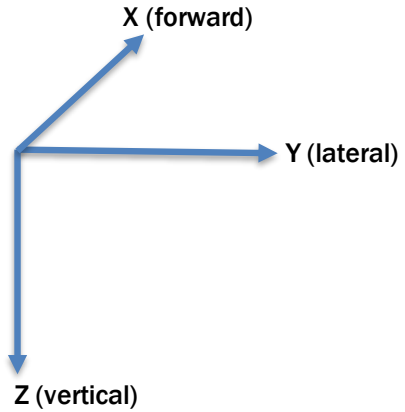
- Designed a custom ship deck
  - Landing pad, visual cue
  - Drone from AirSim

Model Concept

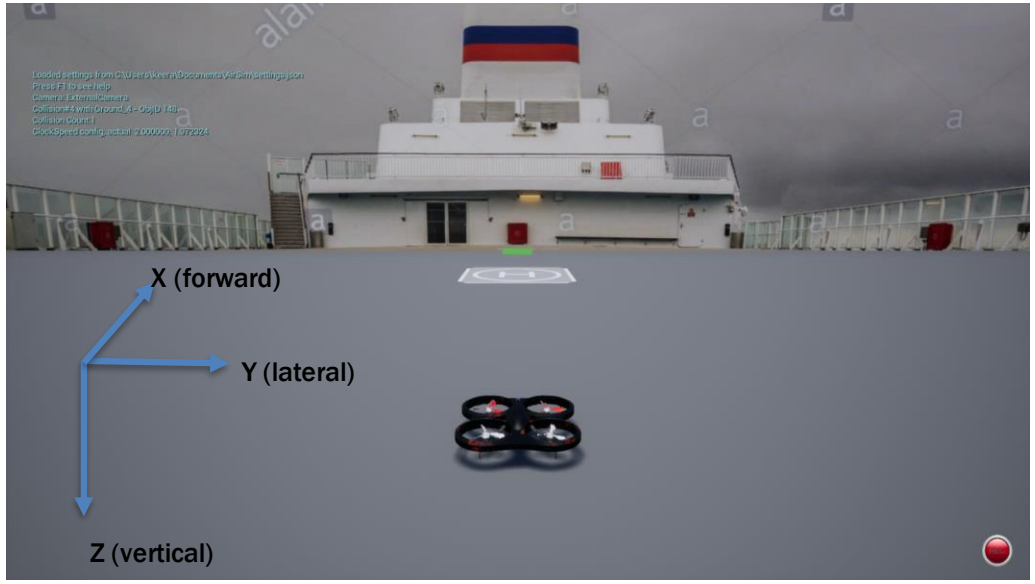
TEXAS A&M UNIVERSITY
Engineering

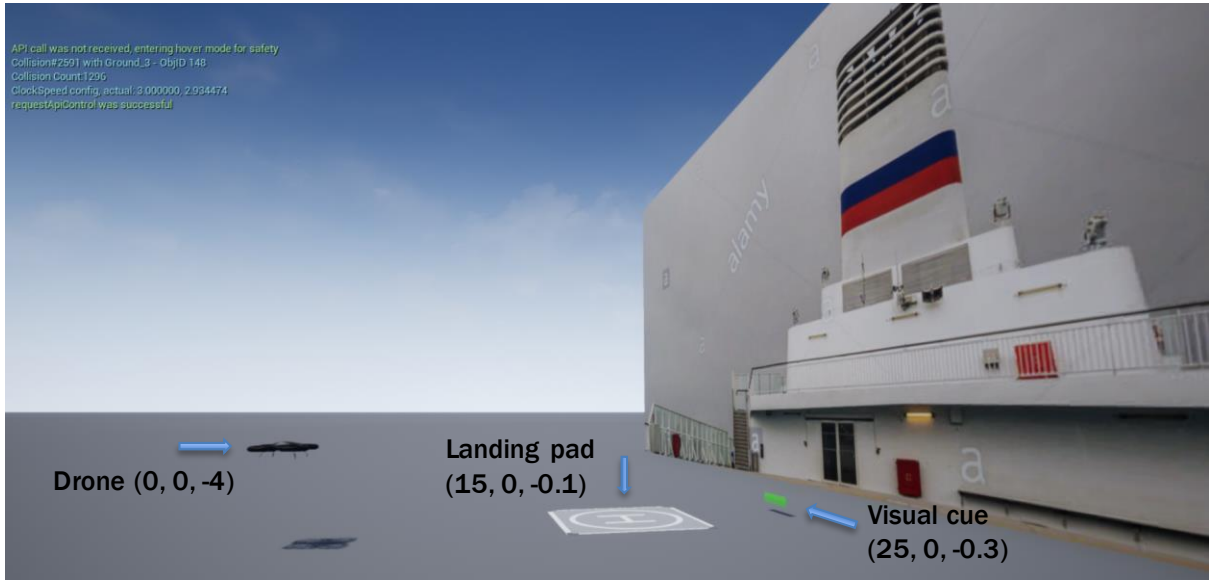


Component & Dimensions	Distance with respect to origin (world coordinate system)
Drone (1mx1m)	At origin
Landing Pad (Centre) (4m X 4m)	X= 15 m, Y= 0, Z= 0
Visual Cue (Centre) (1m X 0.2m)	X= 25 m, Y= 0, Z= 0.3 m



AirSim environment: Front & Side view





The AirSim-v0 environment

TEXAS A&M UNIVERSITY
Engineering

Parameters	Details
State space (cts, dim = 6)	Drone position, velocity (X, Y, Z). **Goal:** 4x4 square around [15, 0, -0.1] Position: X [0, 17], Y [-2, 2], Z [-5, 0] – negative Z upwards Velocity: X [-1, 3], Y [-1, 1], Z [-4, 4]
Action space (cts, dim = 3)	[Pitch (rad), Roll (rad), Throttle (0, 1)]. Yaw zero. Negative pitch down
Termination / Horizon	Timeout (finite/infinite), out of bounds, below visual cue, crash, land

- Want to able to **classify expert demos** as optimal/suboptimal. Assign a simple proxy reward
- Higher reward for getting closer to landing pad, penalty for termination without reaching goal

The AirSim-v0 environment

TEXAS A&M UNIVERSITY
Engineering

Parameters	Details
State space (cts, dim = 6)	Drone position, velocity (X, Y, Z). **Goal:** 4x4 square around [15, 0, -0.1] Position: X [0, 17], Y [-2, 2], Z [-5, 0] – negative Z upwards Velocity: X [-1, 3], Y [-1, 1], Z [-4, 4]
Action space (cts, dim = 3)	[Pitch (rad), Roll (rad), Throttle]. Yaw set to zero. Negative pitch down
Proxy reward	1. Increase reward as it gets close to landing pad (1/x) – **order 10^{-2}** 2. Large positive reward if it lands inside the landing pad (+1000) 3. Other conditions (visual cue, bounding box, timeout) -10 Solved if landed (1000). **Sparse**
Termination	Timeout, out of bounds, below visual cue, crash, land. **Short horizon**
Solved	Mean reward of 1000 for 100 consecutive episodes of training

Generating human expert data

TEXAS A&M UNIVERSITY
Engineering

- Xbox controller:
  - Extremely sensitive
  - Cannot make custom calibration

- **“Taranis x9d” flight controller:**
  - Smoother data logging
  - Disabled yaw from the controller

- **Collected 120 demonstrations of landing UAV**
  - Started at random positions inside the box
  - Maneuver: different heights and at varying speeds
  - Collected (state, action, reward) pairs using AirSim APIs


Pause
00:00:06
Select Area
Audio
Record Pointer
Camera
Anaconda Prompt (Miniconda3) - py...

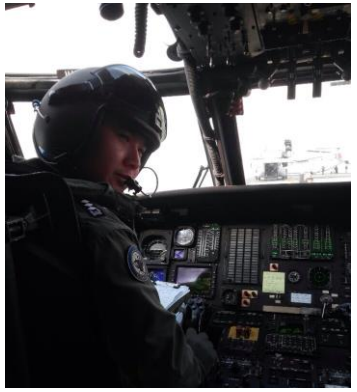
releaseApiControl was successful
Joystick (T,R,P,Y,Buttons): 0.000000, 0.000000, 0.120000, 0.000000, 0000000000000000
SDK Mode: Angle
Vehicle is already armed
Collision Count: 1
requestApiControl was successful

Anaconda Prompt (Miniconda3) - py...

```
-0.007070789113640785 -0.0663386657834053 0.1200
0000476837158 0.0 0.5954999923706055 0 0.0 0.6341
36438369751 -0.0275372676551342 -0.00759696960449
21875 0.023222249001264572 -0.024524075910449028
-5.1333308874745 85e-06 -1.6557676792144775 0.1081
6861689090729 8.055675425566733e-05 -0.0026625116
729832735 -0.06487688531364562 2.8654918560913466
e-05
```

Pause 00:00:11 Select Area Audio Record Pointer

Expert Trajectories: humans



Optimal experts

Suboptimal experts

Human expert demos – stats

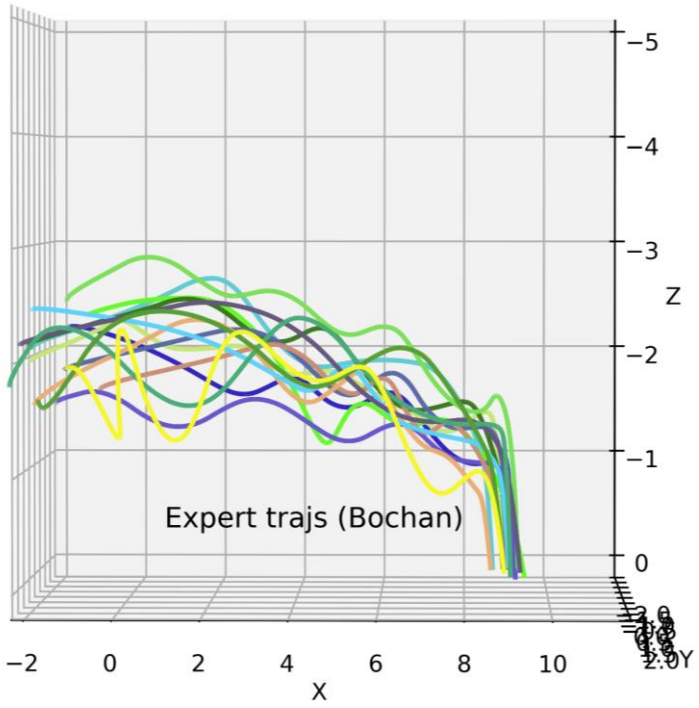
TEXAS A&M UNIVERSITY
Engineering

Expert	Optimal	Solved	Expert score (mean, std)	Expert length
Optimal	Yes	120/120	(1141, 27)	362
Suboptimal	No	132/140	(1116, **284**)	307

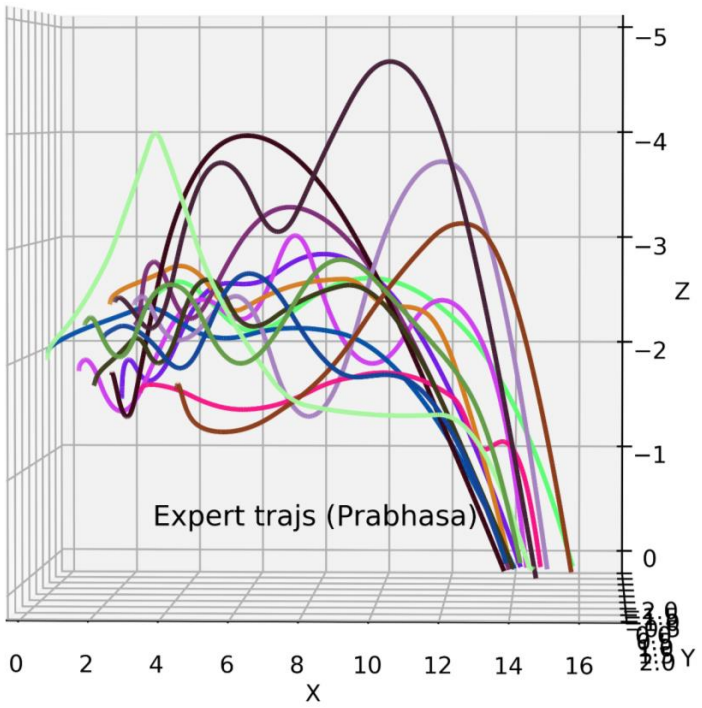
- **True reward (for proxy function):** 1000
- **Task:** Train GAIL on expert samples [**5, 10, 20, 50, 120**] to learn behavior (optimal/suboptimal)

Expert Trajectories: humans

TEXAS A&M UNIVERSITY
Engineering



Optimal



Suboptimal

GAIL on optimal human expert

[5, 10, 20, 50, 120] experts, **large horizon**

Observations

TEXAS A&M UNIVERSITY
Engineering

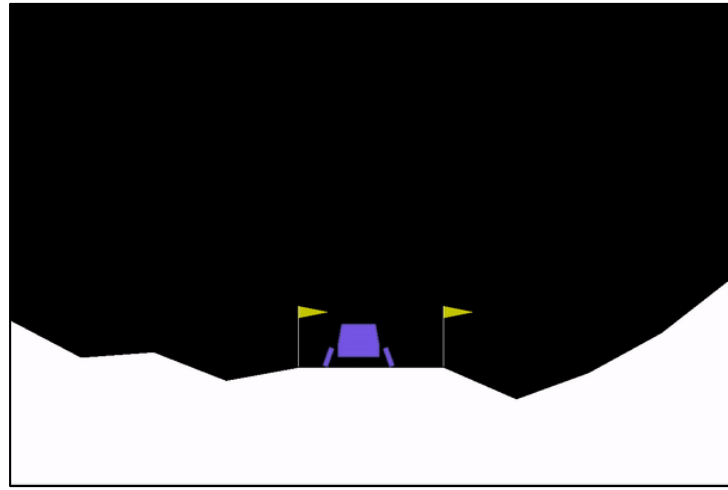
- GAIL did not learn to navigate and land with just 5, 10 demonstrations
- With >20 demos, GAIL learned to navigate but did not learn to land
- Instead, it **hovered** around the landing pad. Note that the horizon is not fixed

- **Explanation:** cumulative reward for hovering long periods ≈ reward for landing
- We will look at another control task that has similar properties as AirSim-v0

Case study: LunarLanderContinuous-v2

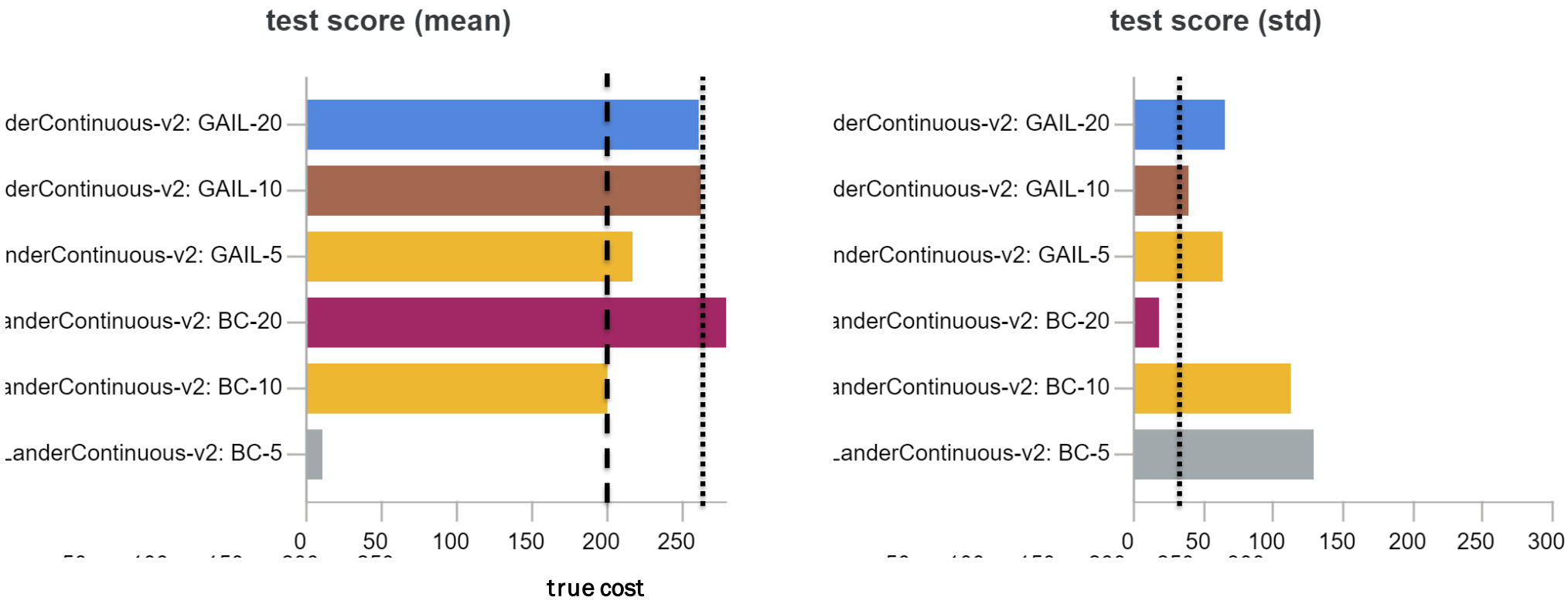
TEXAS A&M UNIVERSITY
Engineering

Properties	Description
State space (cts, dim = 6) and (disc, dim = 2)	Lander position (2), velocity (2), angle & angular speed (2), leg contact (2) Position: X [-1, 1], Y [0, 1.4]. Goal: b/w flags at [-0.2, 0.2] Velocity: typically [-1, 1], can be higher. Angle [-0.4, 0.4] in radians
Action space (cts, dim = 2)	Engine throttle [main engine, left-right engines] Main engine: [-1, 0] off, [0, 1] throttle from 50% to 100% power Left-right: [-1, -0.5] fire left engine, [0.5, 1] fire right engine, [-0.5, 0.5] off
Reward	1. Lander crashes or comes to rest, receiving additional -100 or +100 2. Each leg ground contact is +10 3. Firing main engine is **-0.3** points each frame 4. Solved is 200 points, **semi-sparse**
Termination / Horizon	Land (+100) or crash (-100), **no termination (horizon not fixed)**
Solved	Mean reward **200** over 100 consecutive episodes
Expert Trajectories used	[5, 10, 20], reward (284, 22), episode length 182



LunarLanderContinuous-v2: GAIL and BC

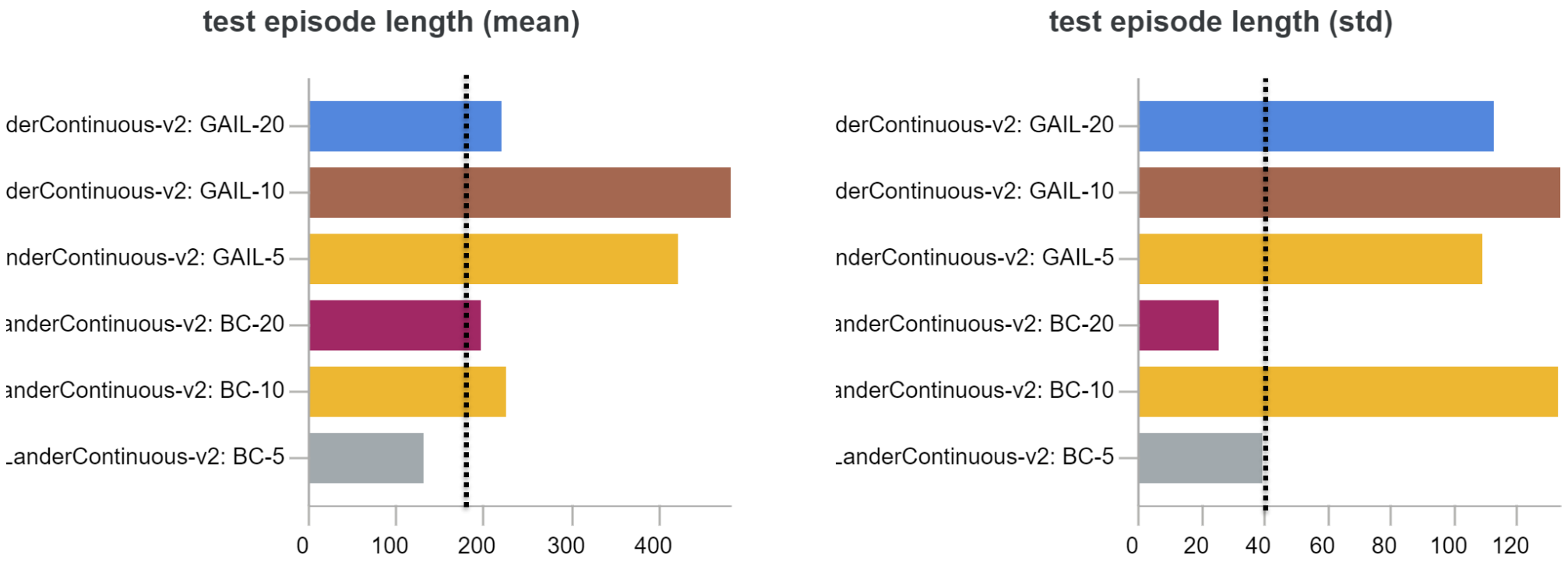
TEXAS A&M UNIVERSITY
Engineering



policy evaluation

LunarLanderContinuous-v2: GAIL and BC

TEXAS A&M UNIVERSITY
Engineering



Episode length of learned policy

GAIL learns to exploit **long horizon** of task to improve score!

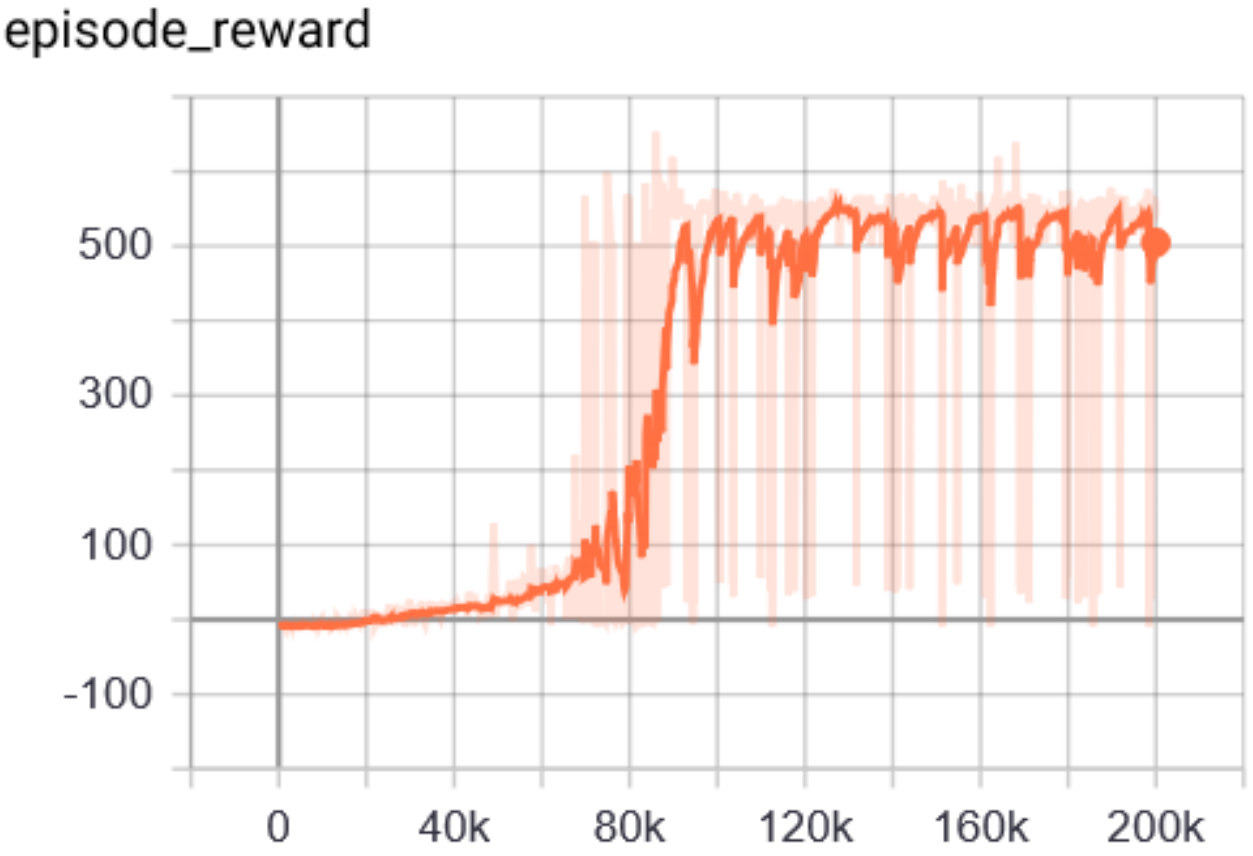
GAIL on suboptimal human expert

[20, 50, 120] experts, **finite horizon (400 steps, same as expert)**

AirSim-v0 (IL)
Suboptimal expert

GAIL hyperparameters

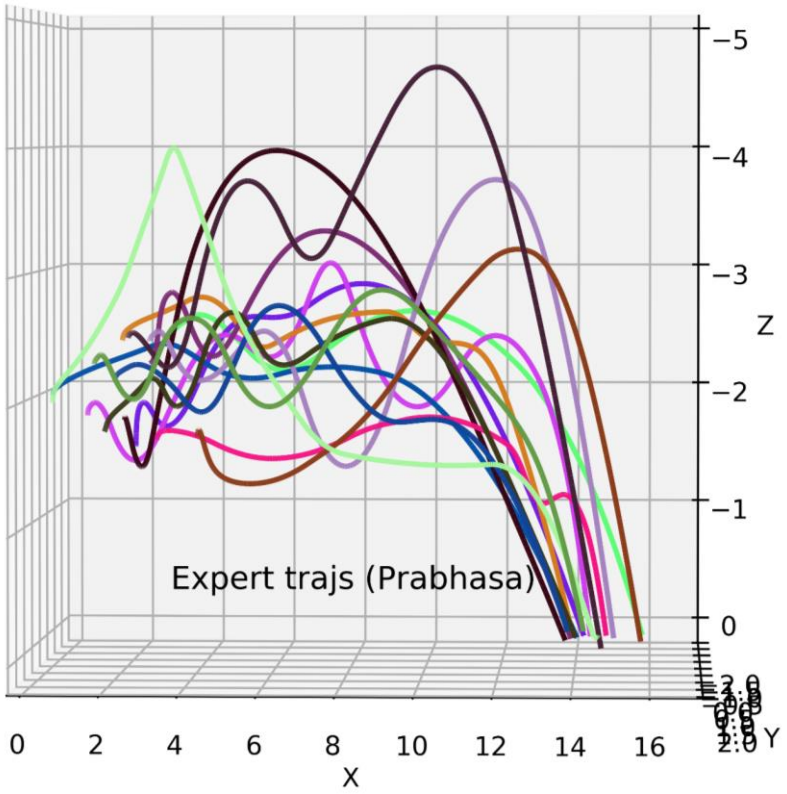
- n_timesteps: 2e5
- policy: 'MlpPolicy'
  [128, 128]
- gamma: 0.99
- learning_rate: 3e-4
- timesteps_per_batch: 256
- buffer_size: 1e6



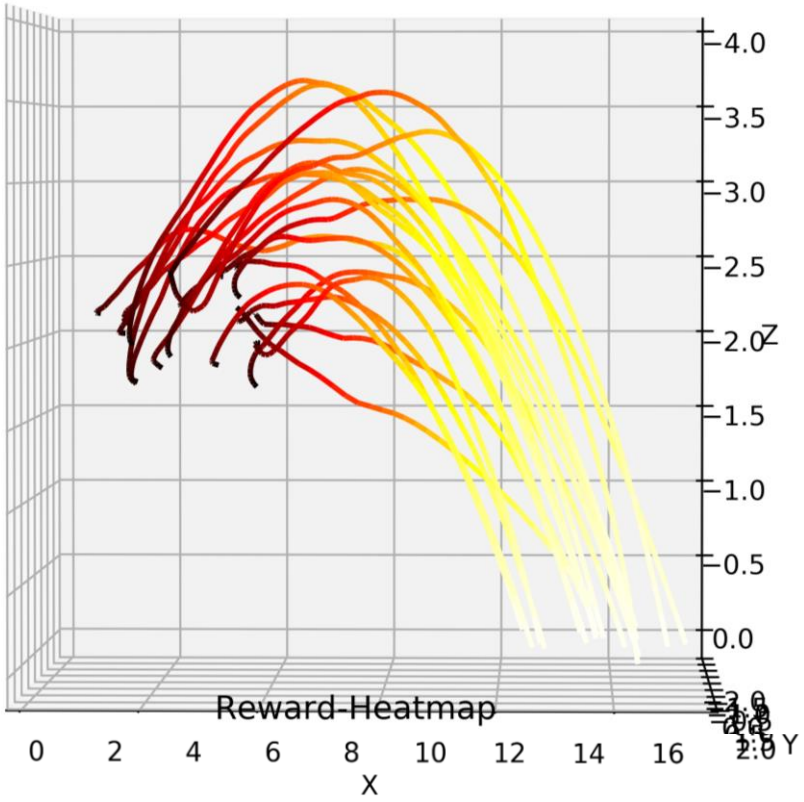
reward convergence

GAIL on suboptimal human expert

TEXAS A&M UNIVERSITY
Engineering



Suboptimal expert



GAIL-learned policy

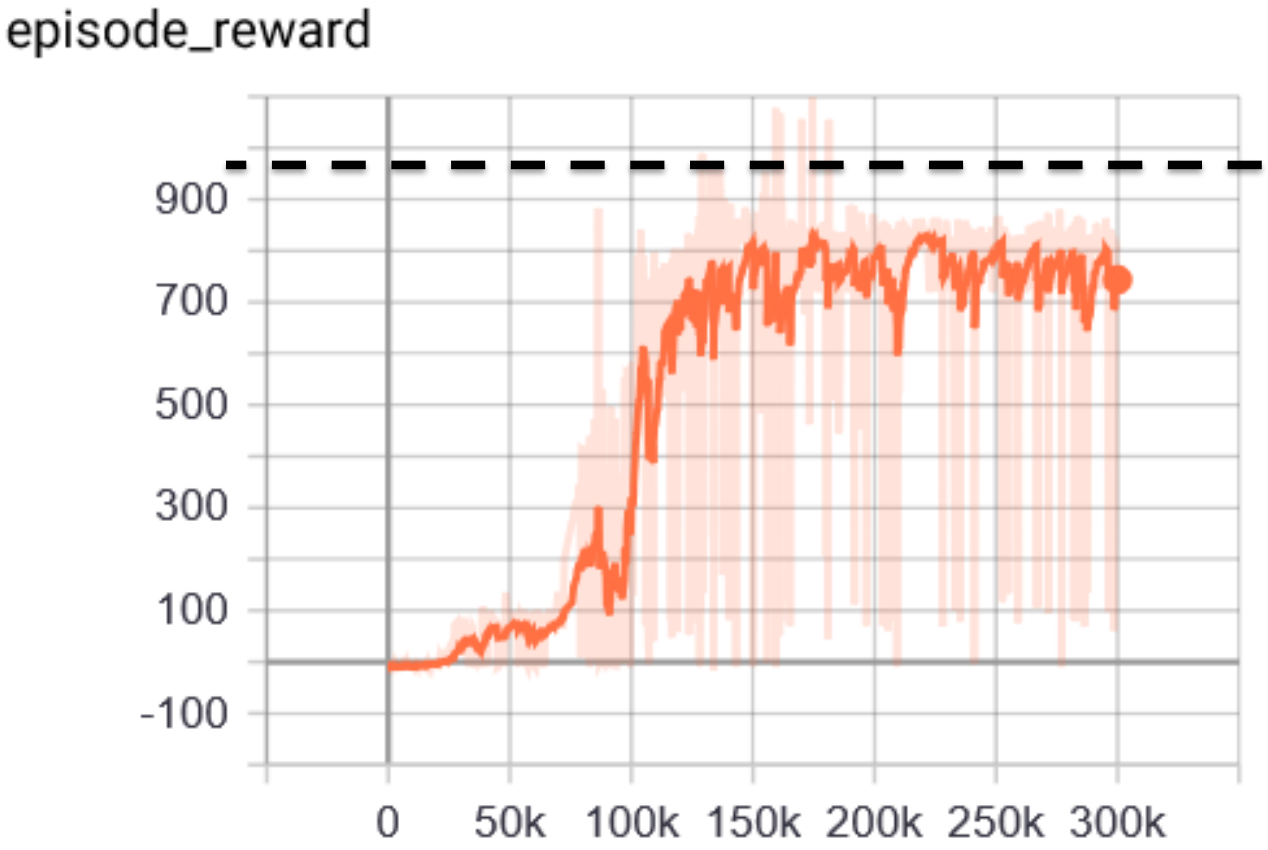
GAIL on optimal human expert

[20, 50, 120] experts, **finite horizon (400 steps, same as expert)**

AirSim-v0 (IL)
Optimal expert

GAIL hyperparameters

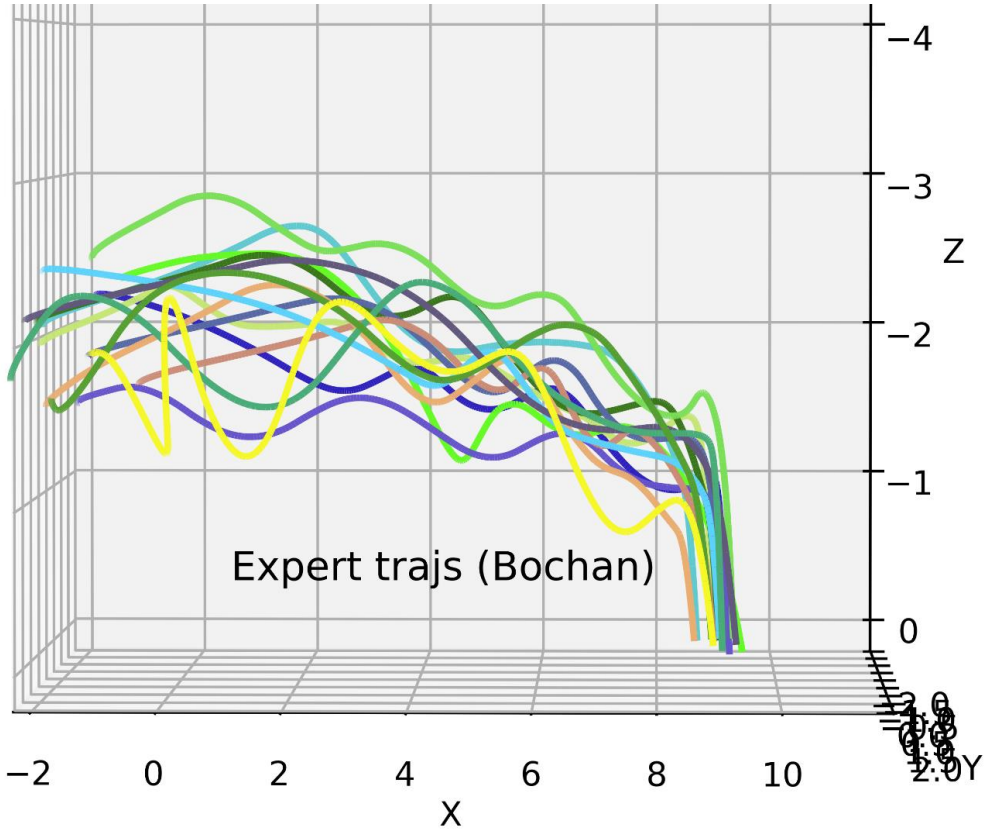
- n_timesteps: 3e5
- policy: 'MlpPolicy'
  [128, 128]
- gamma: 0.99
- learning_rate: 3e-4
- timesteps_per_batch: 256
- buffer_size: 1e6



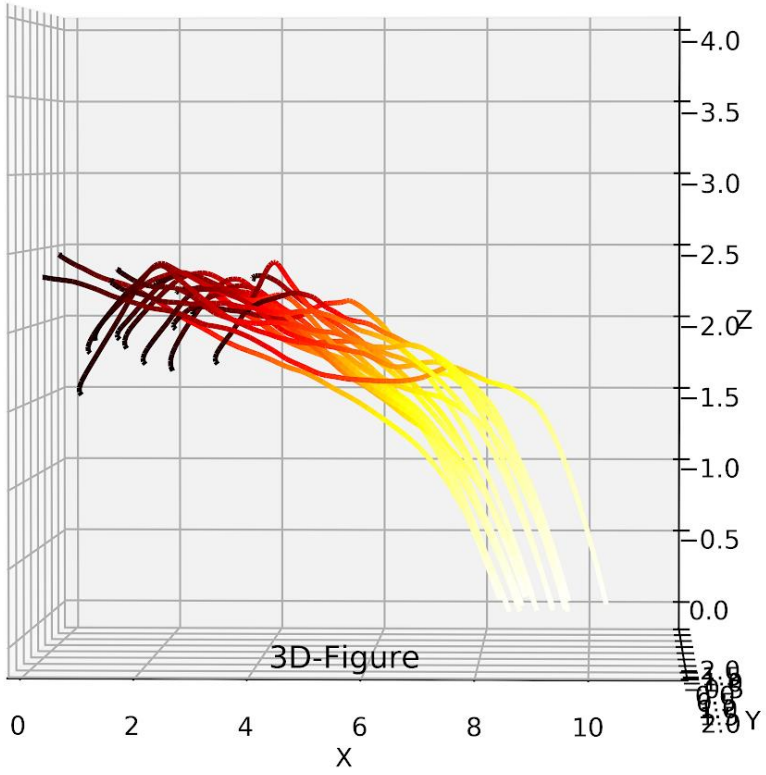
reward convergence

GAIL on optimal human expert

TEXAS A&M UNIVERSITY
Engineering



Optimal expert



GAIL-learned policy

Conclusions

TEXAS A&M UNIVERSITY
Engineering

- GAIL can imitate **navigation** (point A to point B)
- GAIL can learn suboptimal **landings.** HP-dependent
- Explanation: landings may be too ‘non-smooth’ for GAIL to learn
- Rendering of learned policy: front-view

- Can we perhaps **construct a proxy reward** that conveys smoother landings?
- Can RL algorithms learn smoother landings from this proxy?

EXPERT REWARD DESIGN

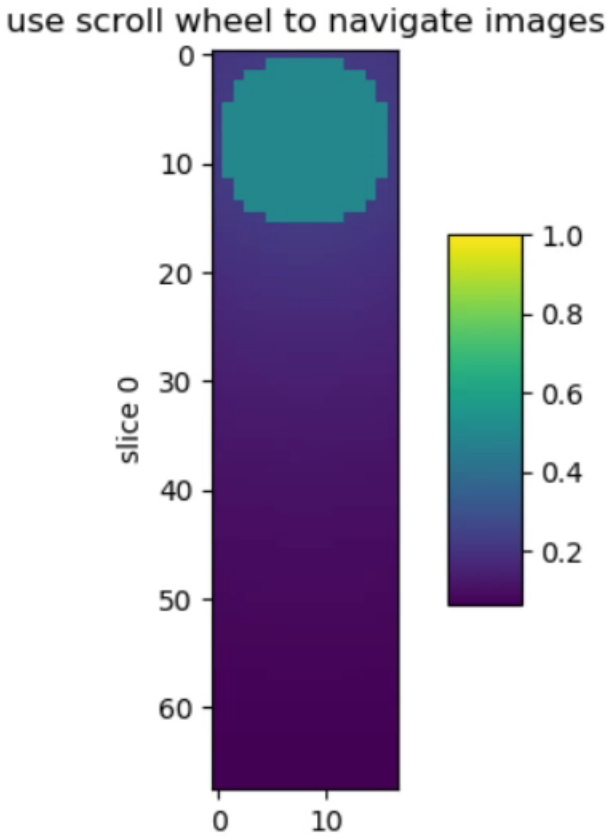
Can RL learn a ‘smoother’ landing than human expert?

Proxy reward function design

TEXAS A&M UNIVERSITY
Engineering

1. **Simple reward (sparse):**
   - Increase reward as it gets close to landing pad (1/x)
   - Large positive reward if it lands inside the landing pad (+1000)
   - Other conditions: -10 (visual cue, out of bounds, crash, timeout)

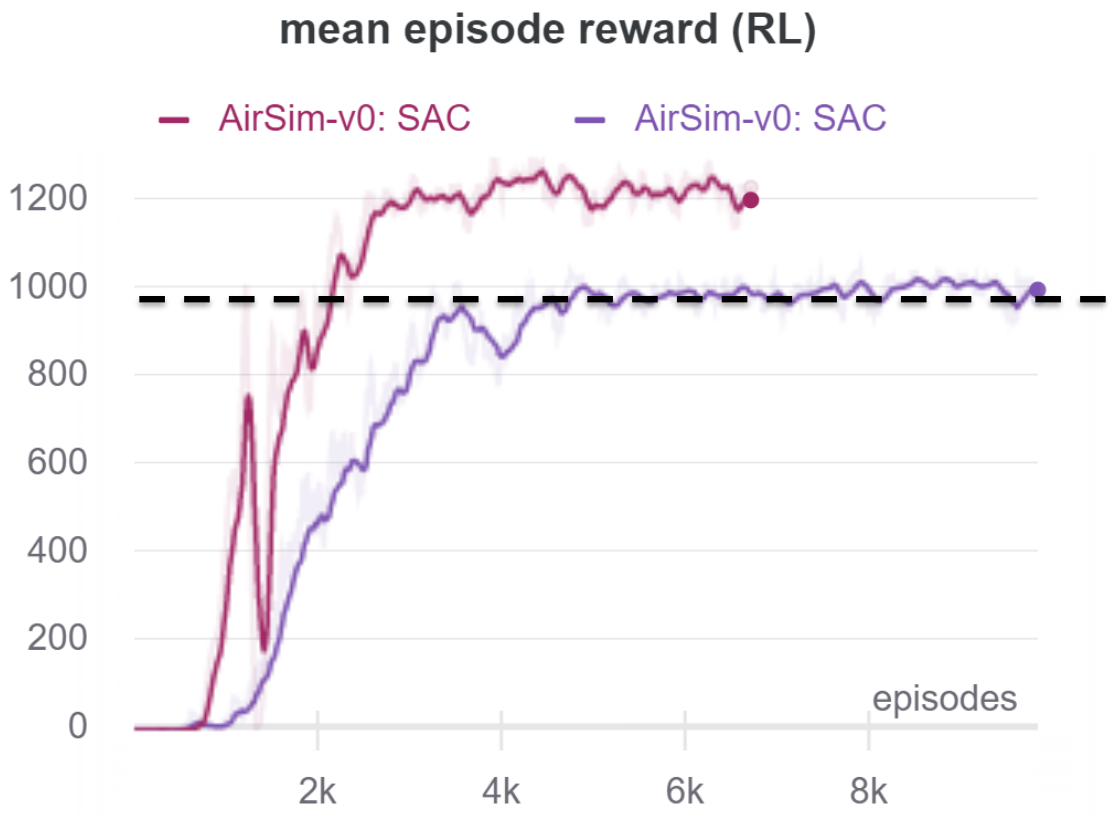
2. **Complex reward (sparse):**
   - Increase reward as it gets close to landing pad (1/x)
   - **Scale** goal reward according to drone heading, speed (1250-750)
   - Other conditions: -10 (visual cue, out of bounds, crash, timeout)

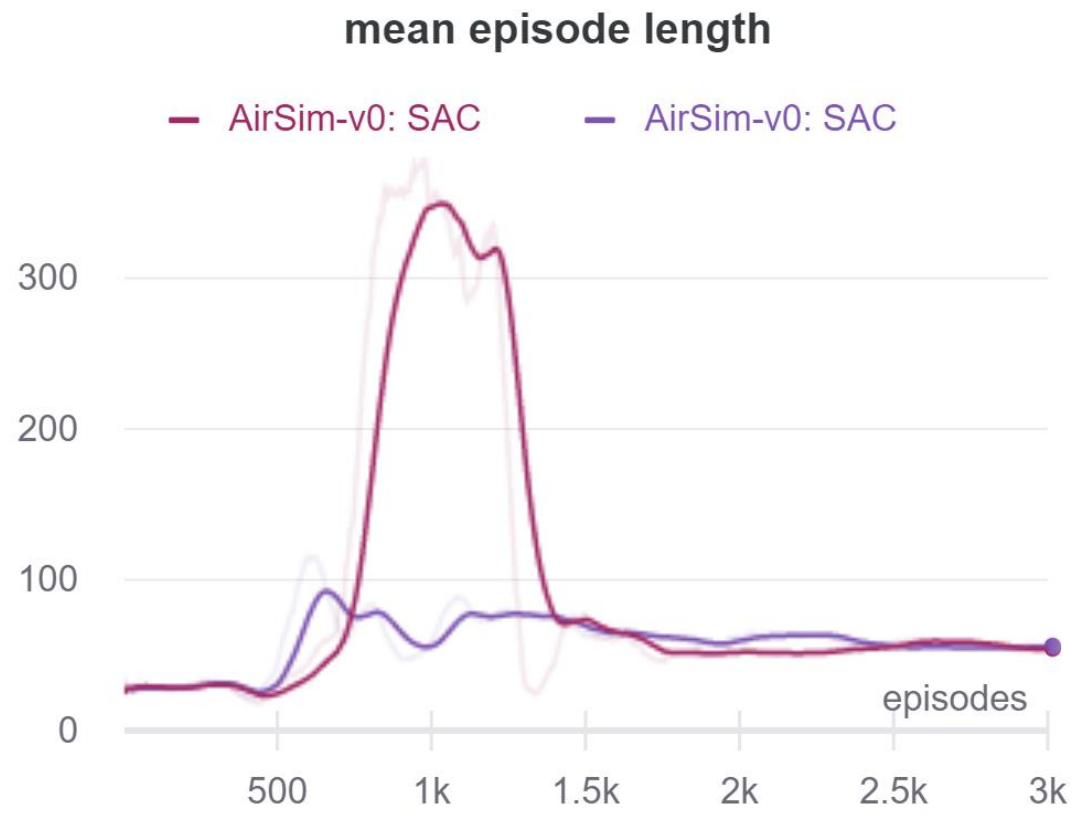


Reward Heatmap

SAC on AirSim-v0: Proxy rewards

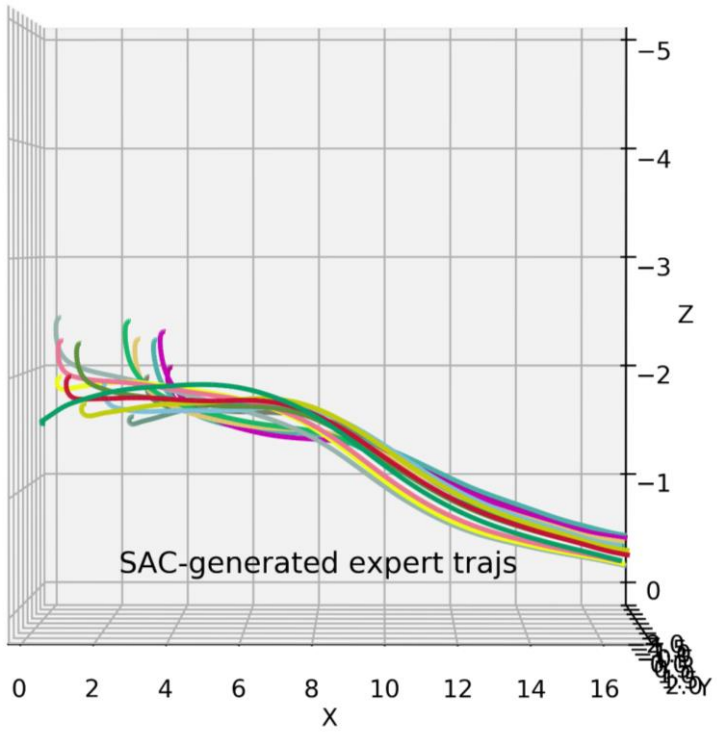
TEXAS A&M UNIVERSITY
Engineering



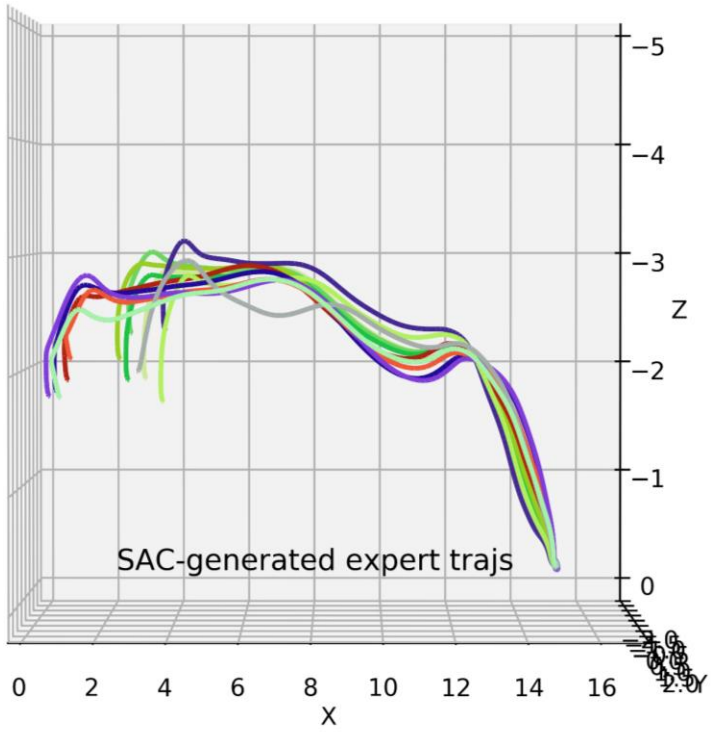


RL-generated expert demos

TEXAS A&M UNIVERSITY
Engineering



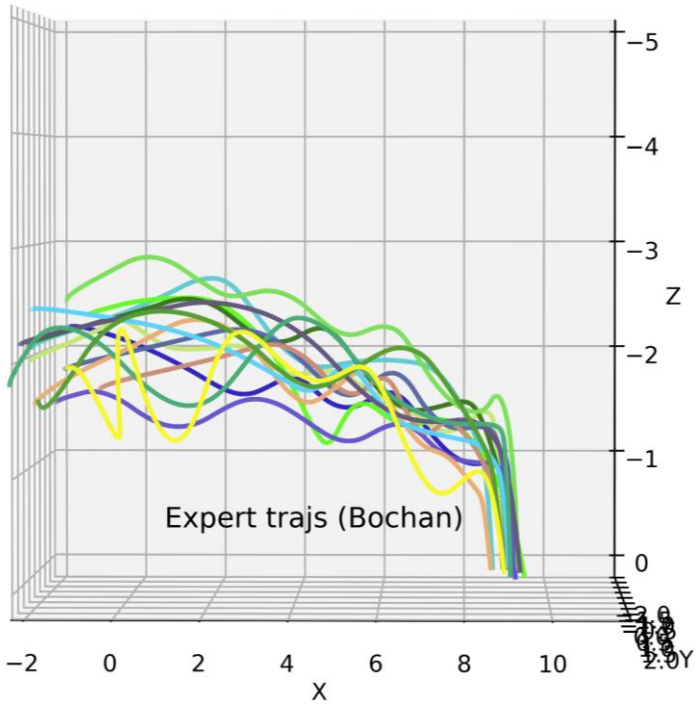
Proxy: simple

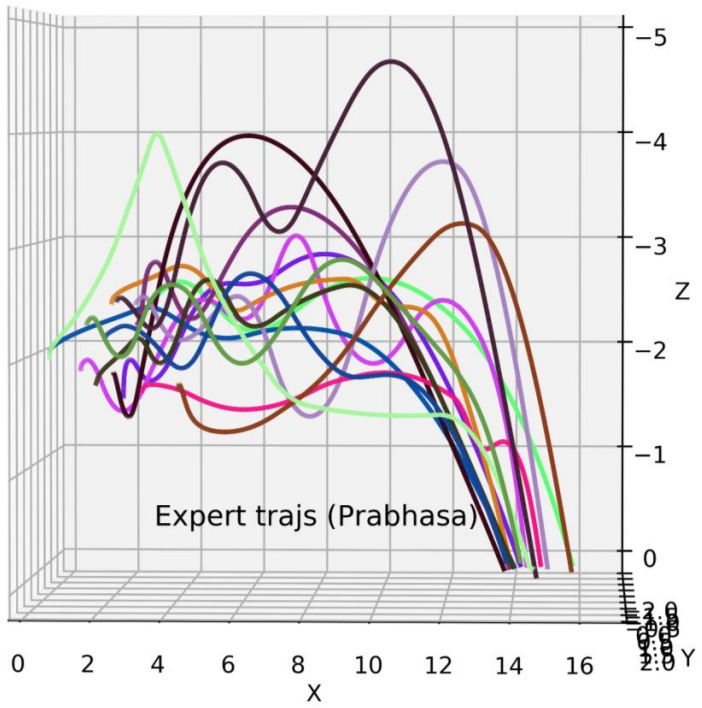


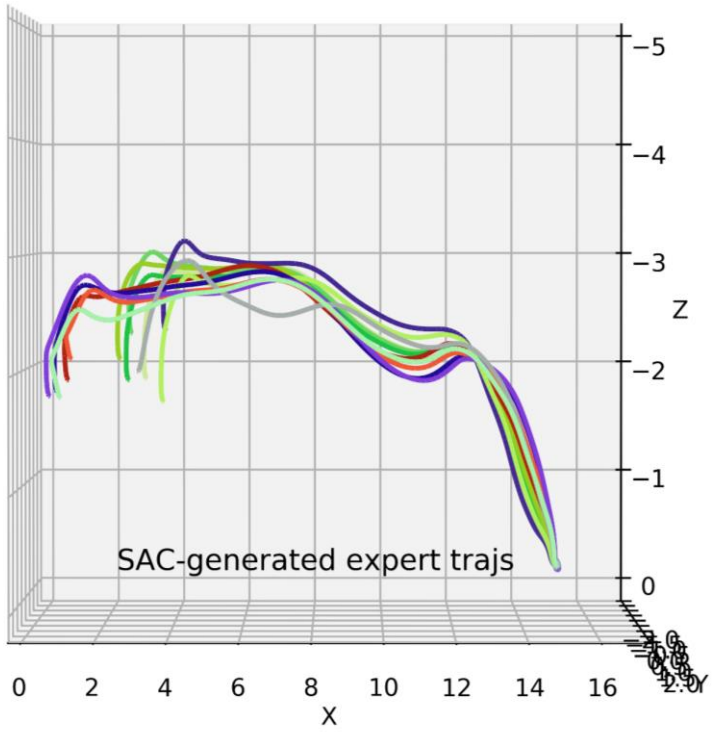
Proxy: complex

Expert demos: humans vs RL

TEXAS A&M UNIVERSITY
Engineering







Optimal

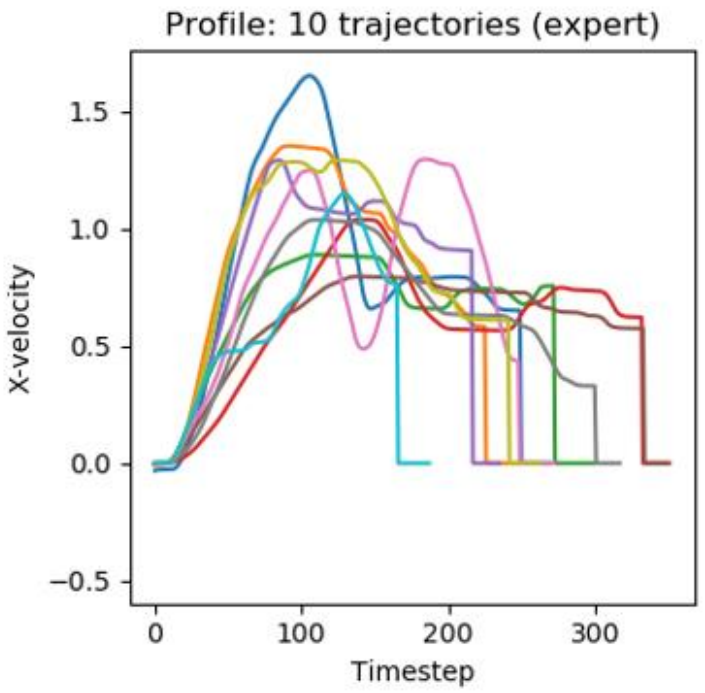
Suboptimal

Proxy: complex

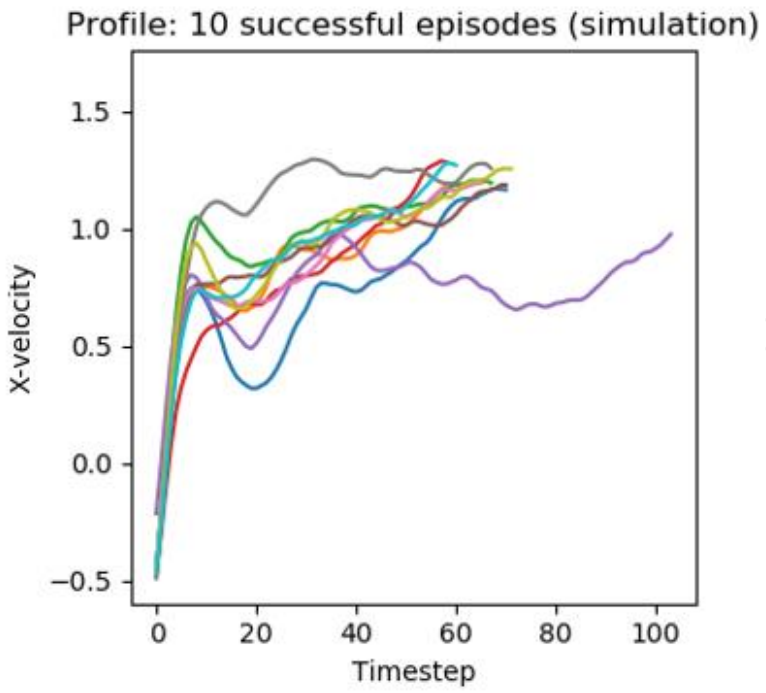
LANDING: HUMANS vs GAIL vs RL

Change in speed

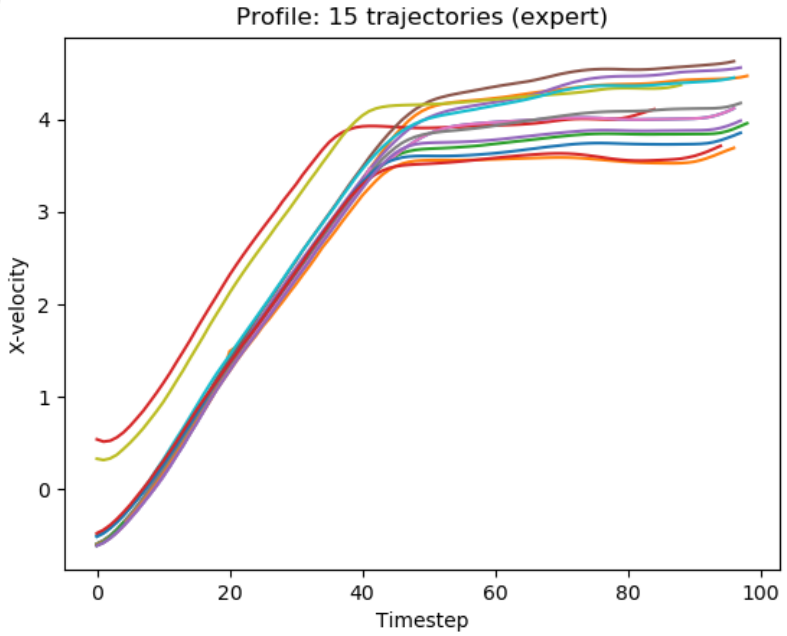
Forward speed



Human expert

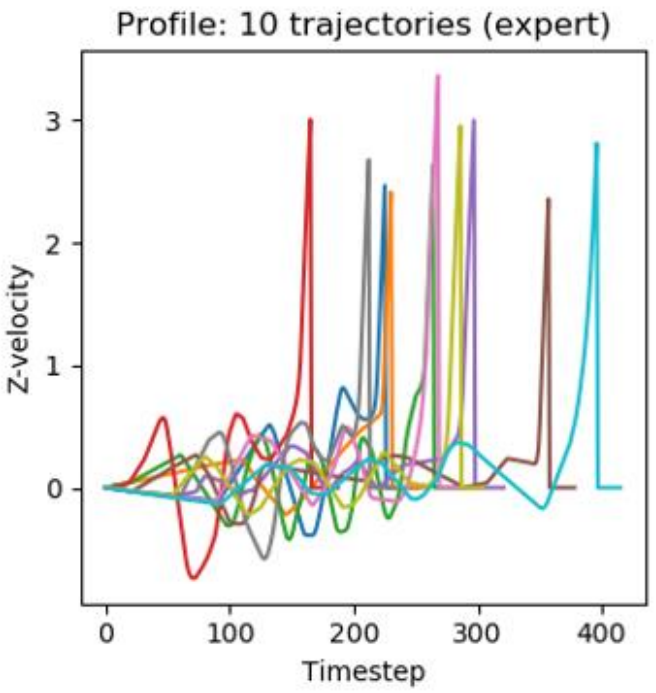


GAIL-learned policy

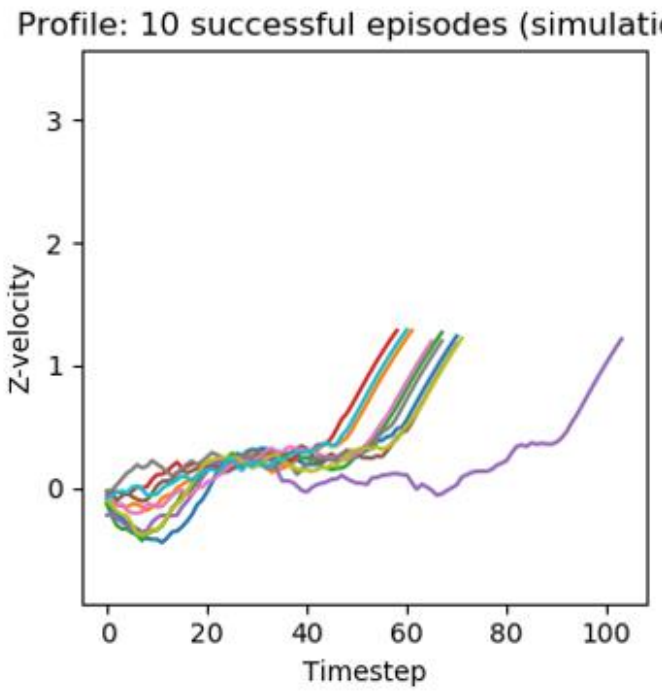


RL expert

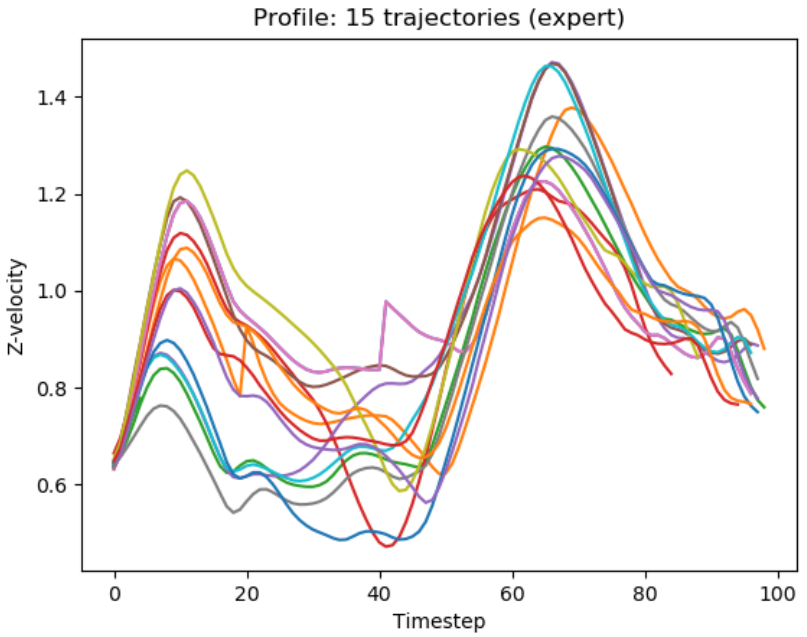
Descent speed



Human expert



GAIL-learned policy



RL expert

Summary: humans vs GAIL vs RL

TEXAS A&M UNIVERSITY
Engineering

Expert	Maneuver type demonstrated/learned	Optimal	Landed	Score (mean, std)	Expert length
Human: Optimal	**Hard landing (pilot)**	Yes	120/120	(1141, 27)	362
Human: Suboptimal	**Large variance**	No	132/140	(1116, 284)	307
GAIL: optimal (20 demos)	**Navigation, *Landing**	*Yes	84/100	(1048, 292)	80
GAIL: suboptimal (20 demos)	**Navigation only**	*No	12/100	(684, 580)	80
SAC: Simple	**Shortest-path**	Yes	99/100	(1106, 112)	99
SAC: Complex	**Smooth landing**	Yes	97/100	(1265, 225)	94

*Smooth landings crucial for perfect imitation with GAIL

Some questions…

TEXAS A&M UNIVERSITY
Engineering

1. How does imitation accuracy scale with dimensionality, demo data? Sample-efficient
2. How 'smooth' are the learned policies compared to the expert policy? smooth if expert is smooth
3. Can sparse rewards be learned? At what cost? **Yes, needs >20 demos,** tuned HPs
4. Can GAIL imitate suboptimal experts? navigation easy, landing difficult. Tuned HPs
5. Can GAIL generalize? Tried different bounding boxes for optimal expert, GAIL policy. Need tuned HPs

GAIL: Pros and Cons

TEXAS A&M UNIVERSITY
Engineering

- Pros
  - Can handle unknown dynamics
  - Can scale to large neural network reward functions
  - Can perform well on real-world tasks (with an efficient policy optimizer)
- Cons
  - Adversarial optimization (GANs) hard to train!
  - Requires smooth experts for imitation
  - First person demonstrations typically used (no “teaching” as such)

Chelsea Finn, RL Bootcamp, 2016

Summary

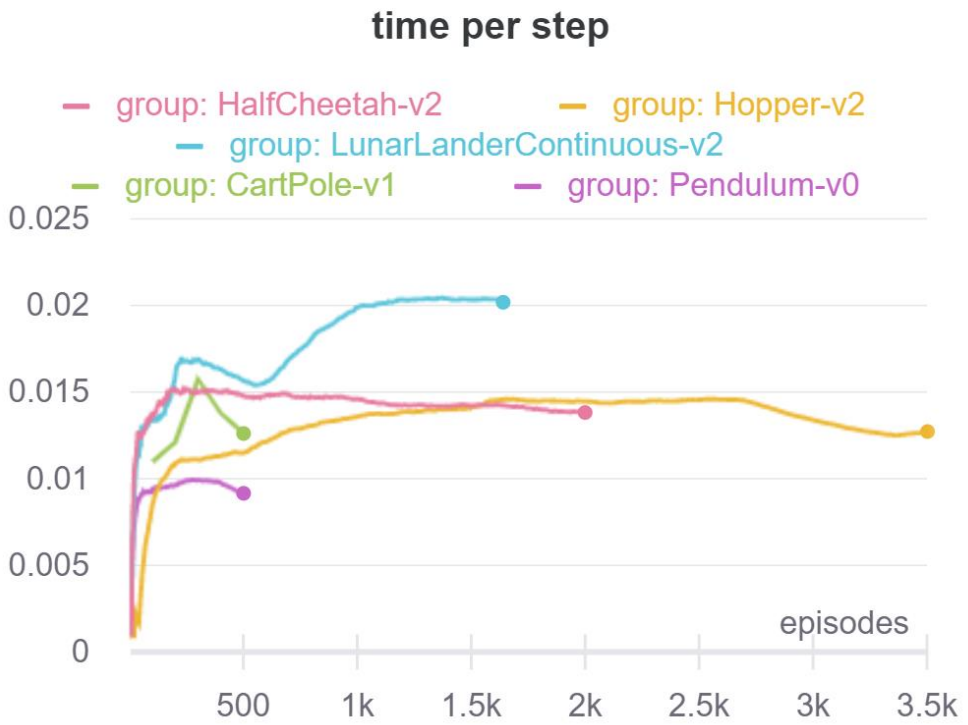
TEXAS A&M UNIVERSITY
Engineering

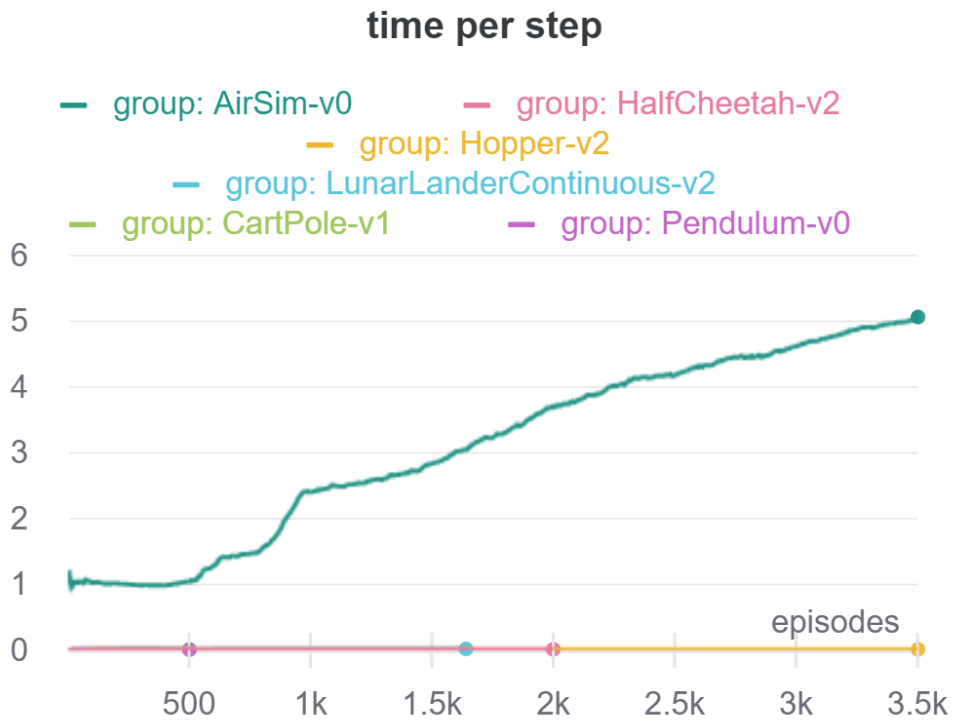
- GAIL **can imitate** AirSim-v0 human experts. Navigation easy, landing not-so-easy
- RL on proxy rewards can generate **smoother landings** – necessary for GAIL
  - reward ϵ space of cost functions explored
- **Expensive training time** limits number of experiments you can run
- Lack of **tuned HPs** affects imitation accuracy

ADDRESSING TIME BOTTLENECKS

Environment samples are expensive!

TEXAS A&M UNIVERSITY
Engineering



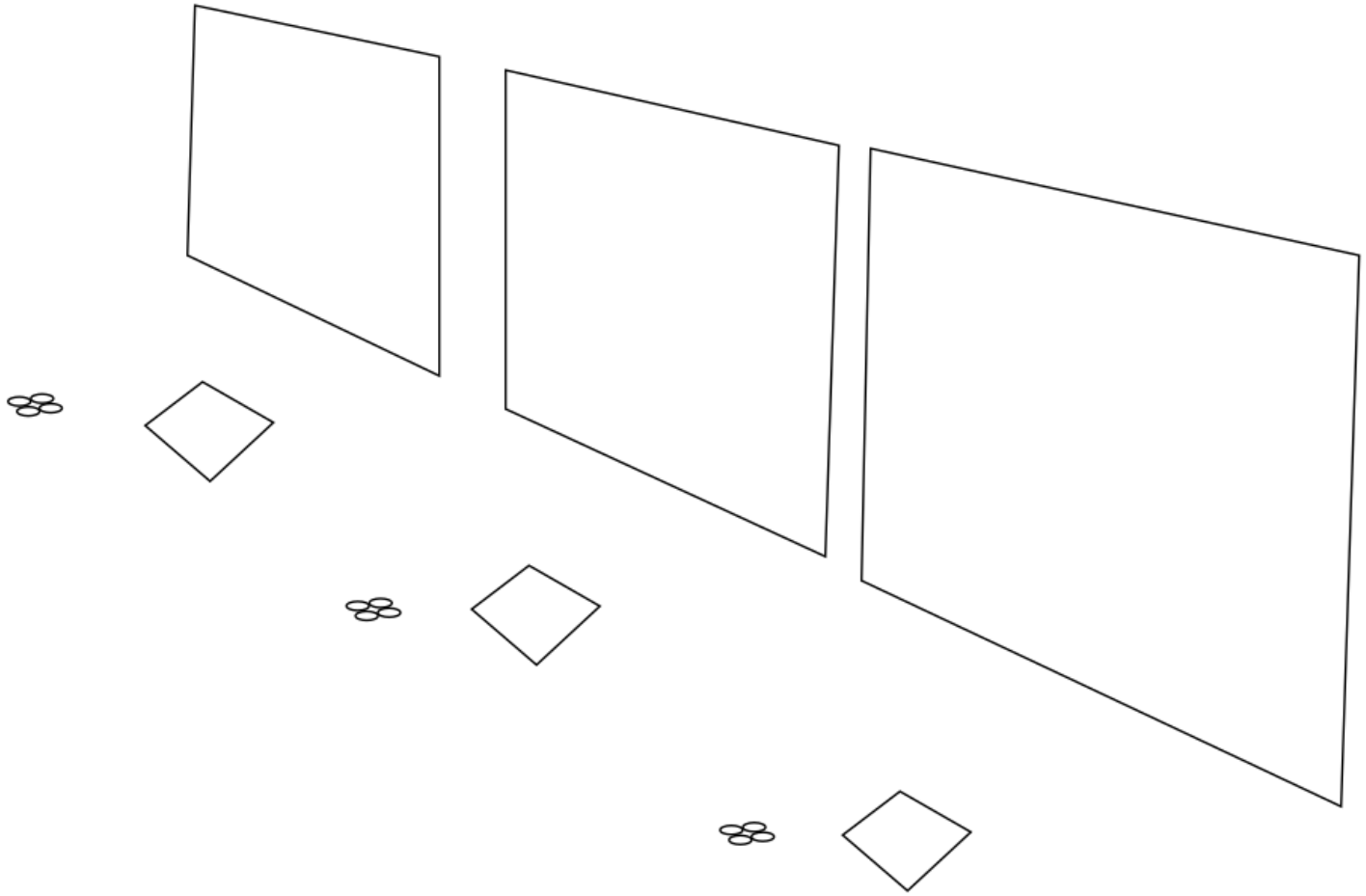


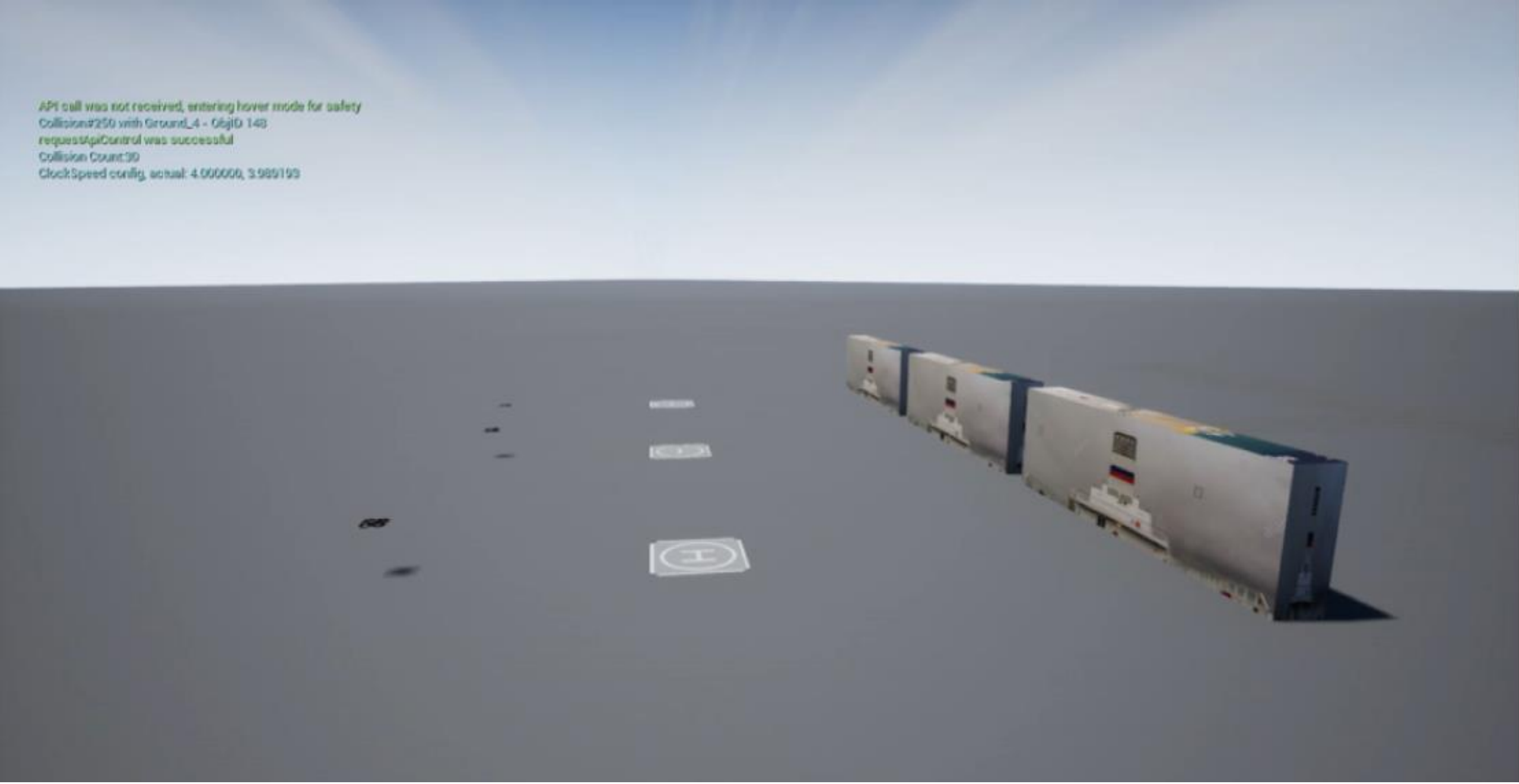
Some numbers to crunch on...

TEXAS A&M UNIVERSITY
Engineering

Property	CartPole-v1	Hopper-v2	AirSim-v0
Dimension (state, action)	(4, 2)	(11, 3)	(6, 3)
Timesteps (GAIL)	3e5	1e6	1e6
Episode length (max)	500	1000	400 (human), 100 (RL)
Training time	**20 minutes**	**2 hours**	**36 hours** (at 4x)
Time/env interaction	4 ms	7.2 ms	129.6 ms
Time/episode	2 s	7.2 s	17.1 s (human), 4.3 s (RL)
Clock speed	Processor (4.8GHz)	Processor (4.8GHz)	4 x **real time**
Cumulative mean reward	Converged	Converged	Did not converge

Setting up multiple experiments




API call was not received, entering hover mode for safety
requestApiControl was successful
Collision Count:39

Sections

TEXAS A&M UNIVERSITY
Engineering

1. Need for sample-efficiency
2. Introduction to Imitation Learning
3. Application 1: Autonomous UAV Landing
4. Application 2: Minecraft
5. Conclusions and Future Work

MineRL: Chopping trees and mining a Diamond in Minecraft

APPLICATION 2

MineRL Competition: NeurIPS 2020

TEXAS A&M UNIVERSITY
Engineering

- Lack of large-scale imitation learning datasets
- **MineRL:** a large-scale dataset of seven different tasks on Minecraft (60 mil pairs)

Why Minecraft:

- Open-world env, sparse rewards, many innate task hierarchies and sub-goals
- 90 million monthly active users, easy to collect a large-scale dataset
- Env simulator available: Microsoft Malmo

MineRL Competition: Description

TEXAS A&M UNIVERSITY
Engineering

- Competition on sample-efficient reinforcement learning using human priors
- Address two crucial challenges in RL. Solving hierarchical environments with
  - Sparse rewards
  - Long time horizon
- Develop algorithms to mine a Diamond object in Minecraft using limited
  - Train time (4 days)
  - Compute (single GPU)
  - Samples from the environment simulator (8 million)

MineRL Competition: Solution approaches

TEXAS A&M UNIVERSITY
Engineering

- "…highlight a variety of research challenges, including open-world multi-agent interactions, long-term planning, vision, control, navigation, and explicit and implicit subtask hierarchies"

- Want to avoid massive datasets and hand-engineered features

- Complex, hierarchical, sparsely-rewarded task that demands use of:
  - Efficient exploration techniques
  - Training with human priors (e.g. fD algorithms) ☑
  - Reward shaping using IL techniques

MineRL Competition: Details

TEXAS A&M UNIVERSITY
Engineering

- Two competition tracks:
  - **Demonstrations and Environment:** MineRL dataset + 8M env interactions ☑
  - **Demonstrations Only:** MineRL dataset only

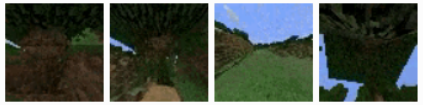
- What's new from 2019: Vectorized state, action space that **obfuscates** the agent's actions
  - Prevent participants from using domain knowledge
  - **State:** images + 1-D vector containing comprehensive set of features from the game
  - **Actions:** 1-D vector containing keyboard presses, mouse movements (pitch, yaw), player GUI interactions, and agglomerative actions such as item crafting

Visualizing the MineRL envs & dataset

Envs: MineRLTreeChopVectorObf-v0 and MineRLObtainDiamondVectorObf-v0

Competition Environments

MineRLTreechopVectorObf-v0



In treechop, the agent must collect 64 *minercaft:log*. This replicates a common scenario in Minecraft, as logs are necessary to craft a large amount of items in the game, and are a key resource in Minecraft.

The agent begins in a forest biome (near many trees) with an iron axe for cutting trees. The agent is given +1 reward for obtaining each unit of wood, and the episode terminates once the agent obtains 64 units.

Observation Space

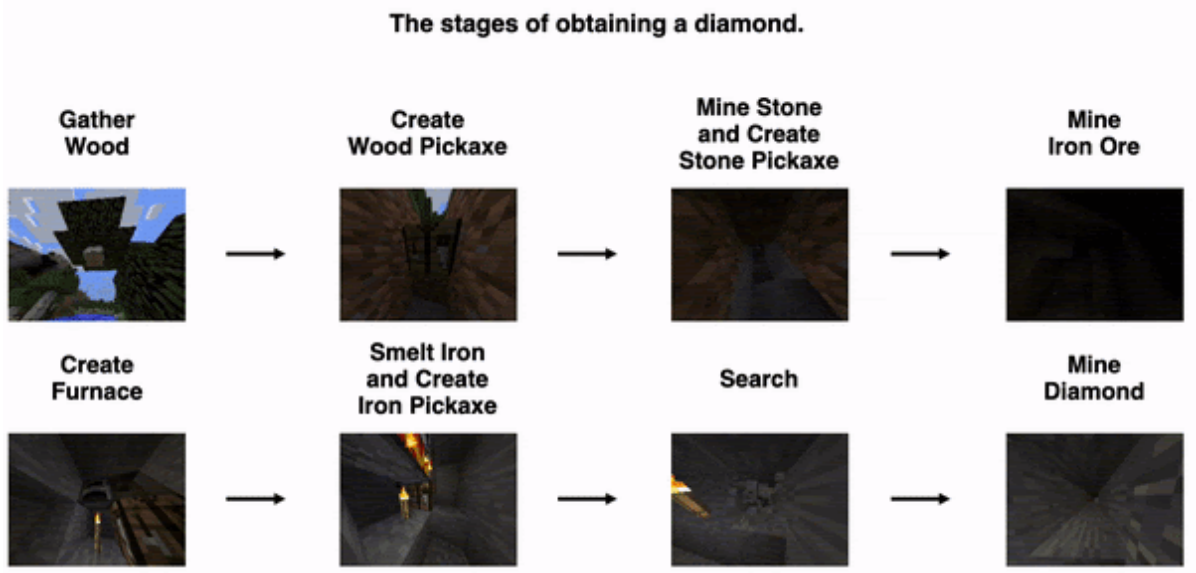
```
Dict({
    "pov": "Box(low=0, high=255, shape=(64, 64, 3))",
    "vector": "Box(low=-1.2000000476837158, high=1.2000000476837158, shape=(64,))"
})
```

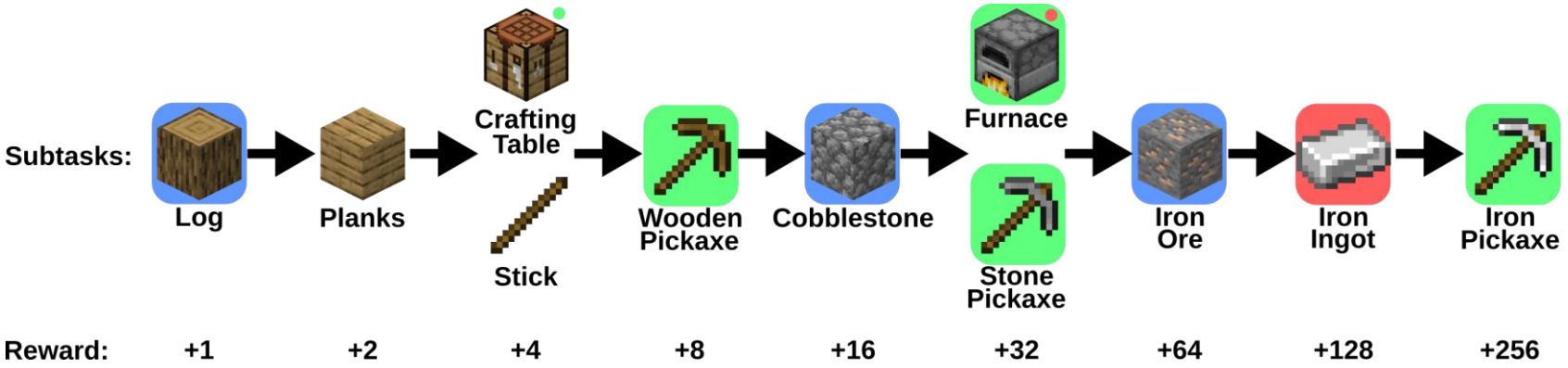
Action Space

```
Dict({
    "vector": "Box(low=-1.0499999523162842, high=1.0499999523162842, shape=(64,))"
})
```

```
"attack": "Discrete(2)",
"back": "Discrete(2)",
"camera": "Box(low=-180.0, high=180.0, shape=(2,))",
"forward": "Discrete(2)",
"jump": "Discrete(2)",
"left": "Discrete(2)",
"right": "Discrete(2)",
"sneak": "Discrete(2)",
"sprint": "Discrete(2)"
```

Obtain Diamond: Tasks and Rewards





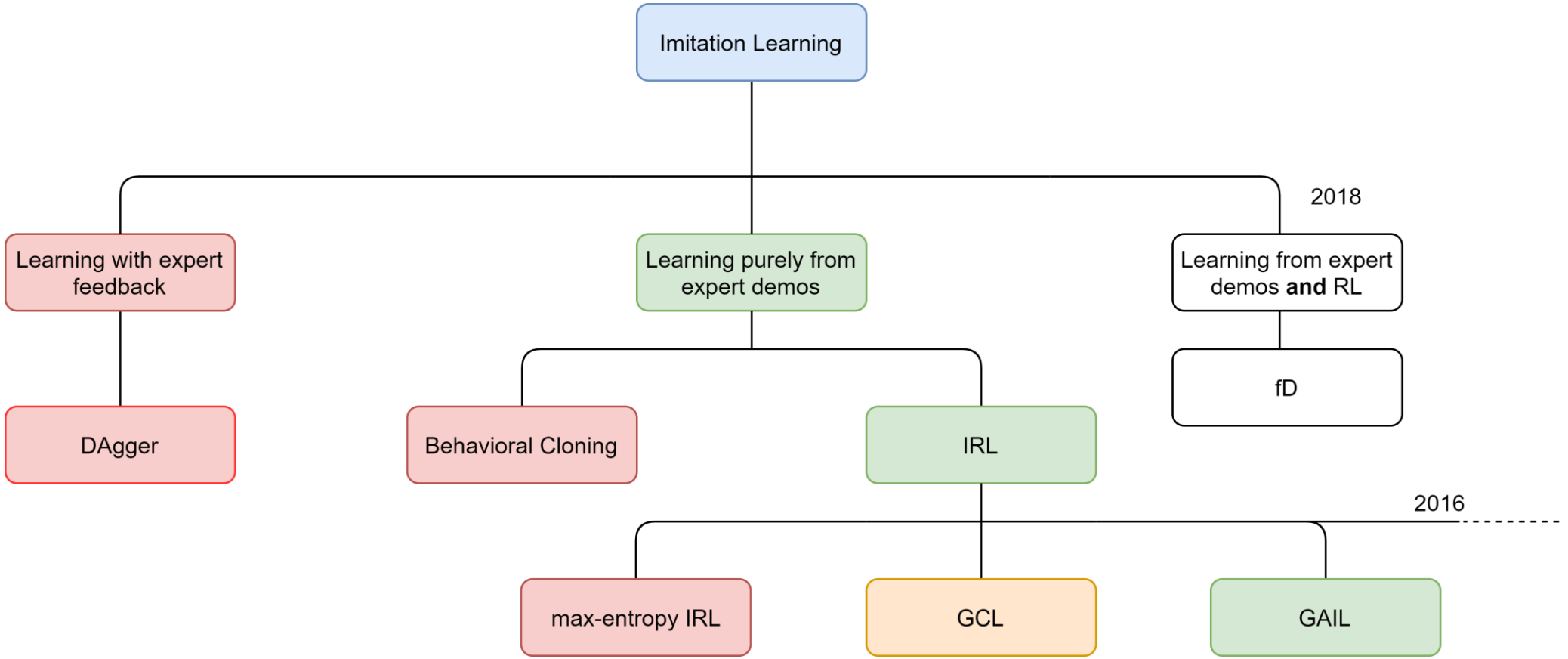
Observation Space

```
Dict({
    "equipped_items.mainhand.damage": "Box(low=-1, high=1562, shape=())",
    "equipped_items.mainhand.maxDamage": "Box(low=-1, high=1562, shape=())",
    "equipped_items.mainhand.type": "Enum(air,iron_axe,iron_pickaxe,none,other,stone_axe,stone_pickaxe,
    "inventory": {
            "coal": "Box(low=0, high=2304, shape=())",
            "cobblestone": "Box(low=0, high=2304, shape=())",
            "crafting_table": "Box(low=0, high=2304, shape=())",
            "dirt": "Box(low=0, high=2304, shape=())",
            "furnace": "Box(low=0, high=2304, shape=())",
            "iron_axe": "Box(low=0, high=2304, shape=())",
            "iron_ingot": "Box(low=0, high=2304, shape=())",
            "iron_ore": "Box(low=0, high=2304, shape=())",
            "iron_pickaxe": "Box(low=0, high=2304, shape=())",
            "log": "Box(low=0, high=2304, shape=())",
            "planks": "Box(low=0, high=2304, shape=())",
            "stick": "Box(low=0, high=2304, shape=())",
            "stone": "Box(low=0, high=2304, shape=())",
            "stone_axe": "Box(low=0, high=2304, shape=())",
            "stone_pickaxe": "Box(low=0, high=2304, shape=())",
            "torch": "Box(low=0, high=2304, shape=())",
            "wooden_axe": "Box(low=0, high=2304, shape=())",
            "wooden_pickaxe": "Box(low=0, high=2304, shape=())"
    },
    "pov": "Box(low=0, high=255, shape=(64, 64, 3))"
})
```

Action Space

```
t({
 "attack": "Discrete(2)",
 "back": "Discrete(2)",
 "camera": "Box(low=-180.0, high=180.0, shape=(2,))",
 "craft": "Enum(crafting_table,none,planks,stick,torch)",
 "equip": "Enum(air,iron_axe,iron_pickaxe,none,stone_axe,stone_pickaxe,wooden_axe,wooden_pickaxe)",
 "forward": "Discrete(2)",
 "jump": "Discrete(2)",
 "left": "Discrete(2)",
 "nearbyCraft": "Enum(furnace,iron_axe,iron_pickaxe,none,stone_axe,stone_pickaxe,wooden_axe,wooden_pickaxe
 "nearbySmelt": "Enum(coal,iron_ingot,none)",
 "place": "Enum(cobblestone,crafting_table,dirt,furnace,none,stone,torch)",
 "right": "Discrete(2)",
 "sneak": "Discrete(2)",
 "sprint": "Discrete(2)"
```

RL with human priors (RL + IL!)



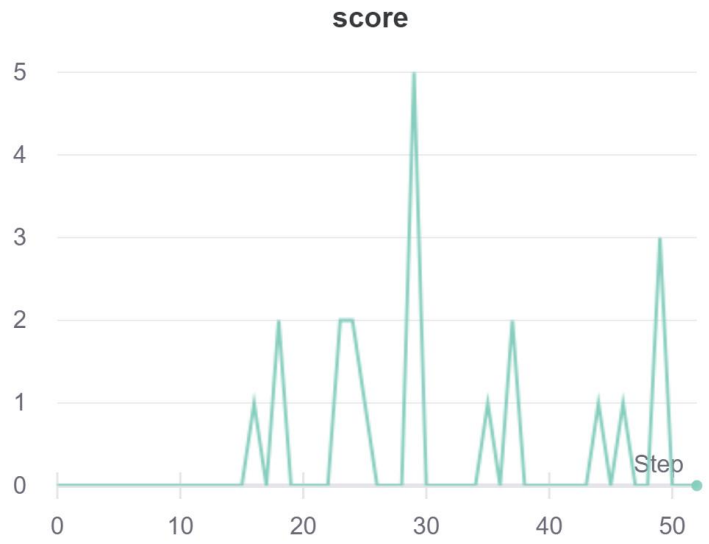
Tools

TEXAS A&M UNIVERSITY
Engineering

- **RL library:** Medipixel 0.10
- **Framework:** Pytorch 1.3.1
- **Hyperparameters (HPs):** Medipixel 0.10
- **Results (train score vs episodes, test score):** W&B 0.10

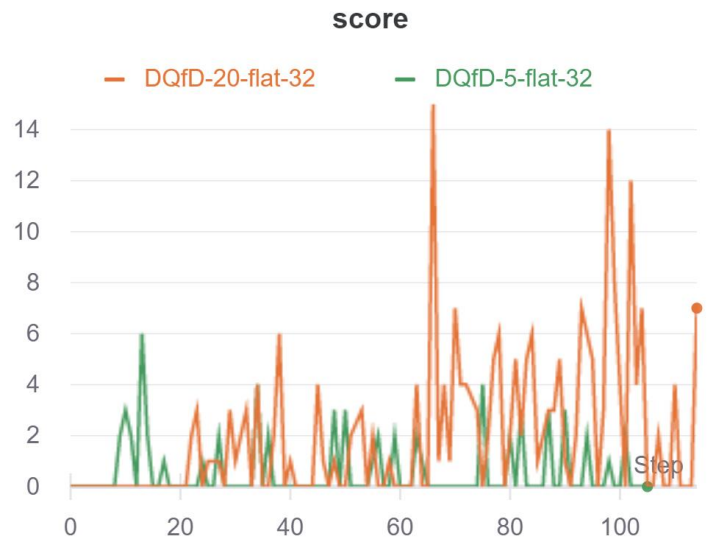
DQN (RL) vs DQfD (RL+IL)

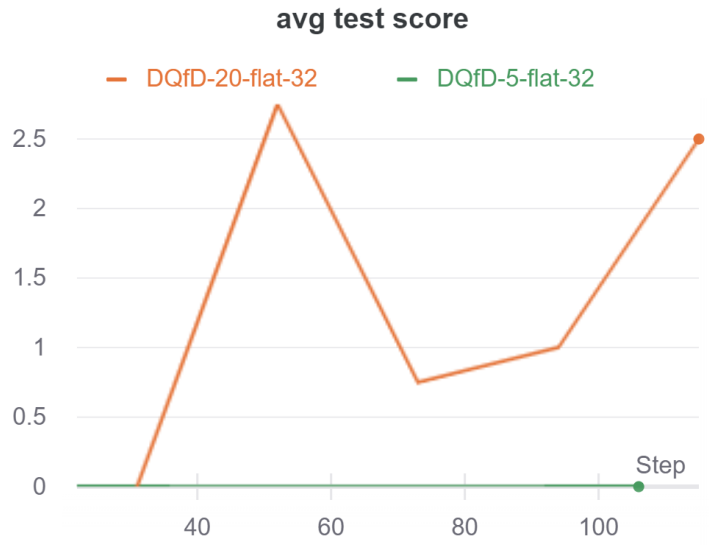
Env: MineRLTreeChopVectorObf-v0


score
5
4
3
2
1
0
Step
0
10
20
30
40
50


avg test score
0.4
0.3
0.2
0.1
0
Step
10
20
30
40
50

DQN


score
DQfD-20-flat-32
DQfD-5-flat-32
14
12
10
8
6
4
2
0
Step
0
20
40
60
80
100


avg test score
DQfD-20-flat-32
DQfD-5-flat-32
2.5
2
1.5
1
0.5
0
Step
40
60
80
100

DQfD

Submitted to MineRL Competition: NeurIPS 2020

TEXAS A&M UNIVERSITY
Engineering

85764	prabhasak	**submitted**	0.000	0.000	RL+ IL	Thu, 1 Oct 2020	View Code

Δ	#	Participants	Media	Reward	N/A	tags	Entries
●	01	NoActionWasted	-	**9.64**	**0.0**	**IL**	15
●	02	michal_opano...	-	**9.29**	**0.0**	**IL**	11
▲	03	CU-SF	-	**6.47**	**0.0**	**RL+ IL**	12
▲	04	HelloWorld	-	**6.01**	**0.0**	**RL+ IL**	7
●	05	NuclearWeapon	-	**4.34**	**0.0**	**RL+ IL**	7

Sections

TEXAS A&M UNIVERSITY
Engineering

1. Need for sample-efficiency
2. Introduction to Imitation Learning
3. Application 1: Autonomous UAV Landing
4. Application 2: Minecraft
5. Conclusions and Future Work

Tasks Studied

TEXAS A&M UNIVERSITY
Engineering

Task	*State dim	*Action dim	Reward quality	Termination, Horizon	Imitation successful?
Pendulum-v0	3C	1C	Dense	Fixed, small	**Yes**
LunarLanderCts-v2	4C + 2D	2D	Semi-sparse	Not fixed, large	**Yes**
Hopper-v2	11C	3C	Dense	Fixed, large	**Yes (better)**
AirSim-v0	6C	3C	Sparse	Fixed, small	****Yes**
MineRLTreechopVectorObf-v0	pov: 64x64x3 vector: 64C	64C (64D)	(extremely) sparse	Fixed, very large	**No**
MineRLObtainDiamondVectorObf	pov: 64x64x3 vector: 64C	64C (64D)	(extremely) sparse	Not fixed, very large	**No**

*C: continuous. D: discrete
**Suboptimal landings, >20 optimal demos

RL vs IL

TEXAS A&M UNIVERSITY
Engineering

Learning with a cost function vs imitating with demo data

Control Task	Reward	Task length (max)	Episodes / env interactions (RL)	Episodes / env interactions (IL)	Converged in episodes (RL)	Converged in episodes (IL)
Pendulum-v0	Dense	200	500 (1e5)	1500 (3e5)	200	800
CartPole-v1	Dense	500	500 (1e5)	2500 (1e6)	400	1400
LunarLanderCts-v2	Sparse	N/A	1300 (5e5)	1650 (1e6)	**800**	**1000**
Hopper-v2	Dense	1000	5300 (2e6)	3400 (2e6)	**3500**	**2500**
AirSim-v0	Sparse	400	10k (5e5)	16k (1e6)	**3000**	**7300**

CONCLUSIONS

TEXAS A&M UNIVERSITY
Engineering

- Need sample-efficient learning for complex, long-horizon tasks
- IL (GAIL) is a sample-efficient approach to learn from demonstrations
- IL can be used to imitate (even suboptimal) experts from sparsely-rewarded environments
  - Requires smooth experts and careful HP tuning for perfect imitation
- Application 1: Designed a novel method of autonomous UAV landing (simulation)
- Application 2: Discussed potential of IL + RL on a complex, sparse, long-horizon, hierarchical task

Future Extensions: AirSim

TEXAS A&M UNIVERSITY
Engineering

- Learning reward functions with smoothness properties (e.g. WAIL)
- Collecting human expert data with smoother maneuvers, for better imitation
- Complex maneuvers. E.g. side-entry (yaw), landing on a moving platform, wind
- Switch to a quadcopter for learning (Parrot Anafi with Gazebo)
- Multi-agent, transfer learning, and meta-learning methods to learn behaviors that can be generalized to unknown environments (e.g. point-to-point navigation)

Future Extensions: MineRL

TEXAS A&M UNIVERSITY
Engineering

- Use CNNs to learn representations from the image
- Employ hierarchical learning, multi-agent RL to learn implicit/explicit hierarchies in tasks
- Train on datasets of individual tasks in hierarchy, to bring in diversity among demonstrations
- Algorithmic contributions for sparsely-rewarded, hierarchical tasks with long-horizons
- LeNS Lab should participate in MineRL NeurIPS 2021!

Acknowledgements

- **Advisor:** Prof. Dileep Kalathil
- **Sapana Chaudhary**, PhD, Texas A&M University
- **Kishan Panaganti Badrinath**, PhD, Texas A&M University
- **Deepankar Chanda**, Masters, Texas A&M University
- **Bochan Lee**, PhD, Texas A&M University
- **Vinicius Guimares Goecks**, Postdoctoral Researcher, U.S. Army CCDC Research Laboratory
- **LeNS Lab mates:** Desik, Archana, Trupthi, Aria, Kiyeob, Gargi, Manav
- All other friends I made here at A&M!

TEXAS A&M UNIVERSITY

Engineering

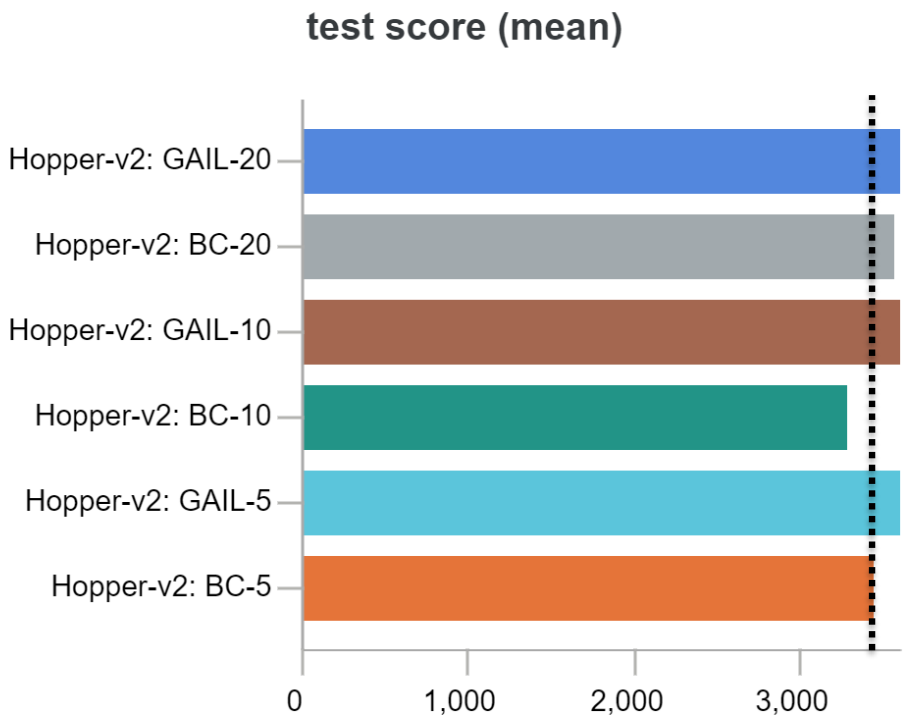
THANK YOU!

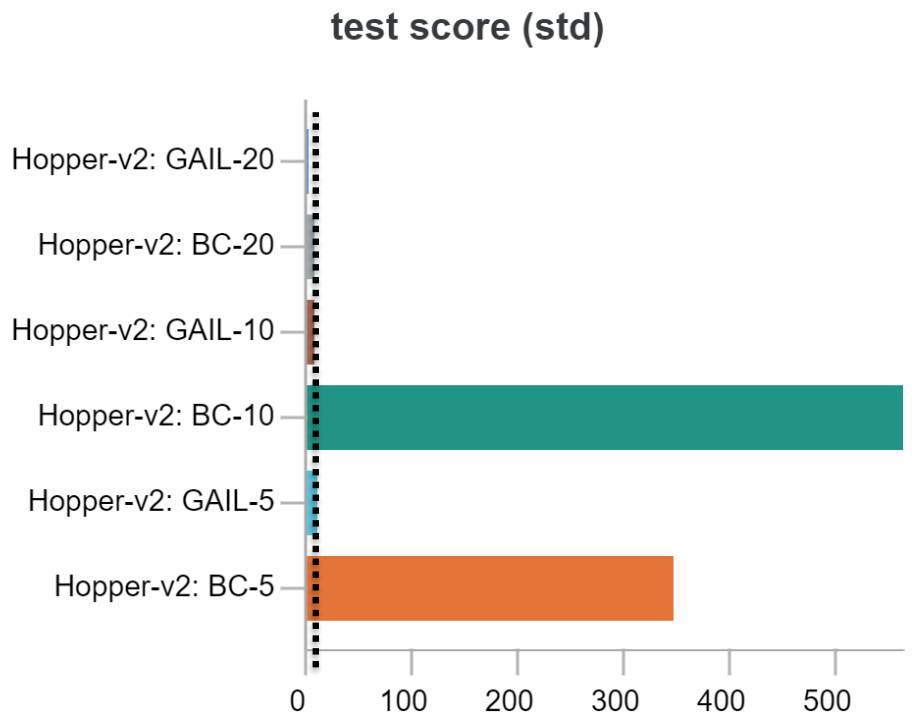
QUESTIONS?

APPENDIX

Hopper-v2: GAIL and BC (better than expert!)

TEXAS A&M UNIVERSITY
Engineering



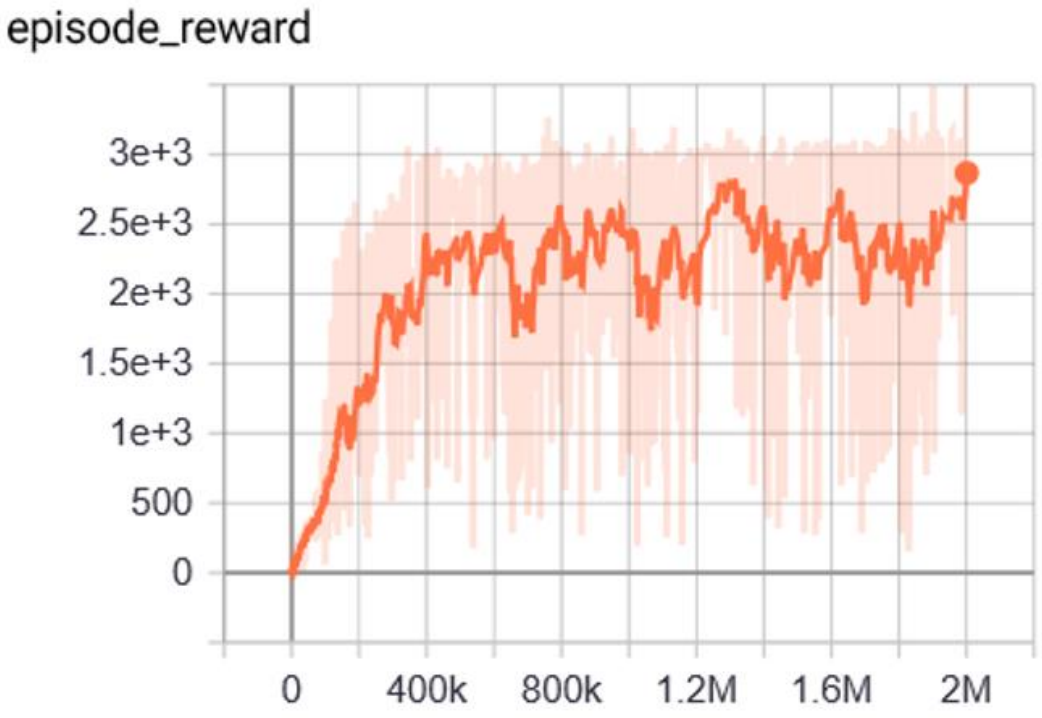


EFFECTS OF EXPERT DATA NORMALIZATION

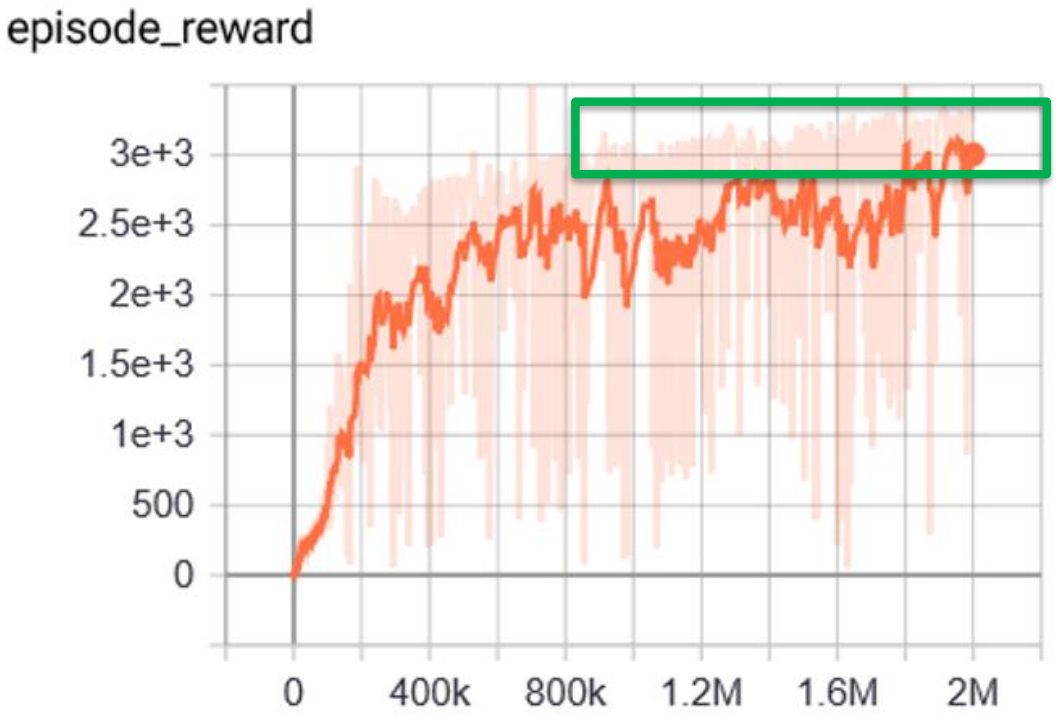
ON BOTH RL AND IL

Hopper-v2: TRPO without norm vs with norm

TEXAS A&M UNIVERSITY
Engineering



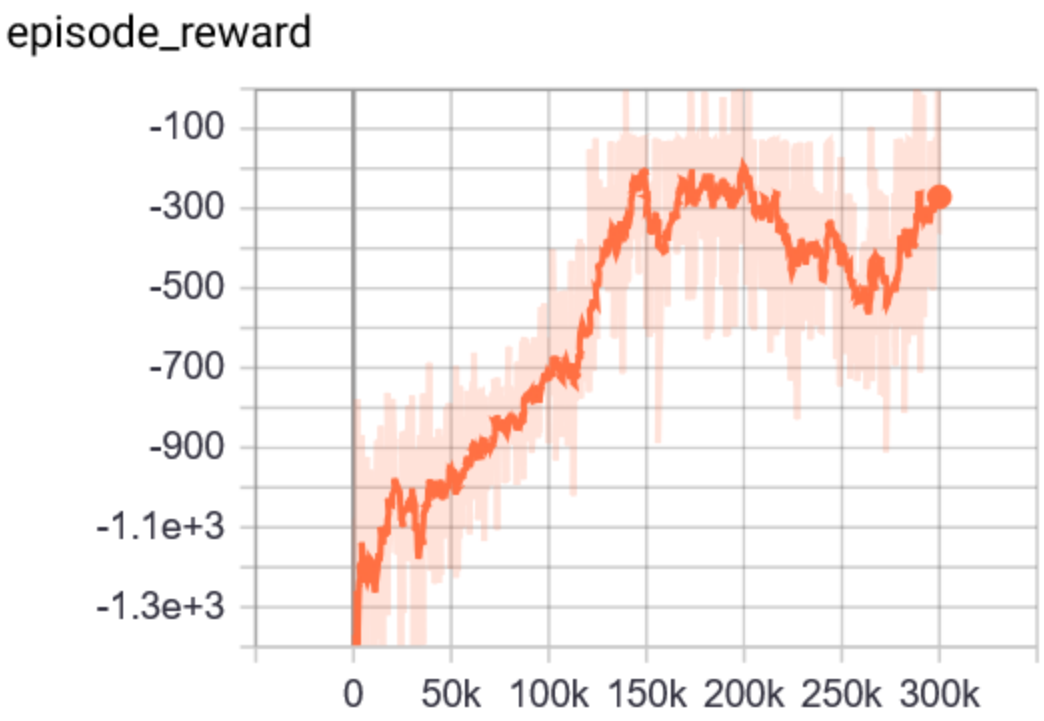
RUN 1, 2, 3



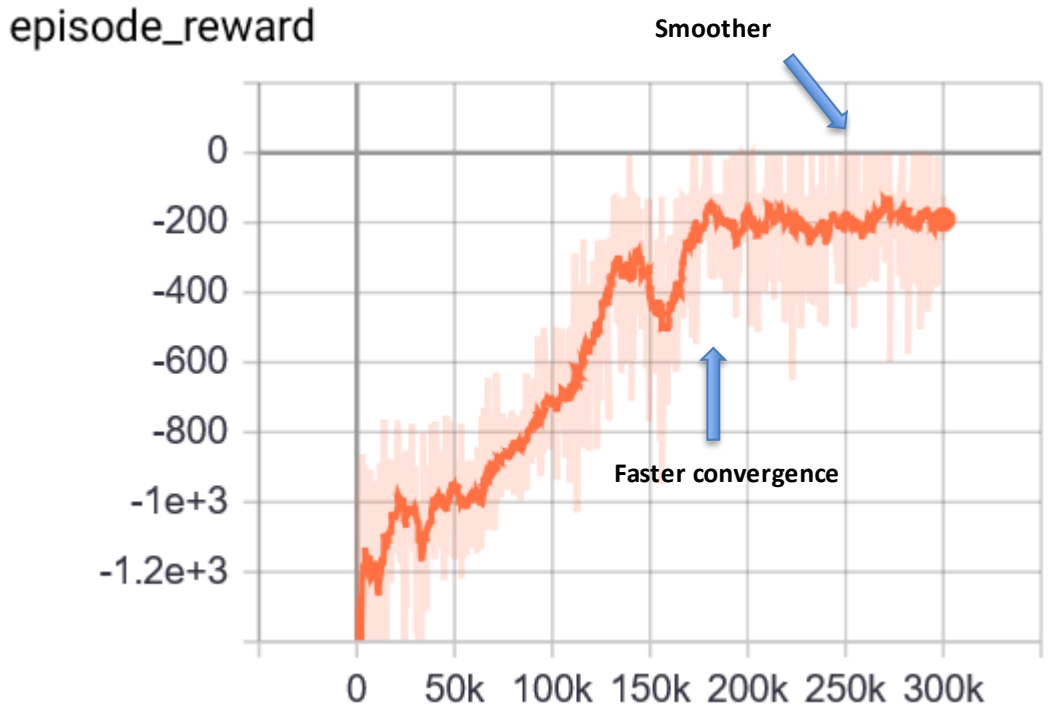
RUN 1, 2, 3

Pendulum-v0: GAIL without norm vs with norm

TEXAS A&M UNIVERSITY
Engineering



RUN 1



RUN 1