TEXAS A&M UNIVERSITY
Engineering

Learning from Demonstrations: Applications to Autonomous UAV Landing and Minecraft

Student: Prabhasa Kalkur
Advisor: Dr. Dileep Kalathil

Oct 05, 2020

What is imitation learning?

Learning to imitate from expert behavior

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

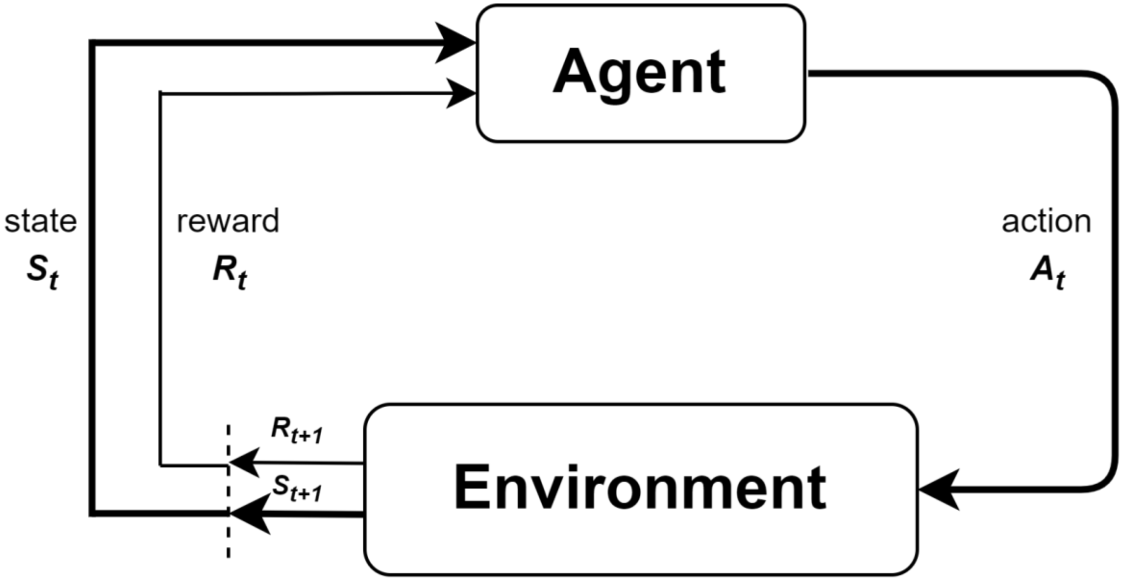
What is the presentation about?

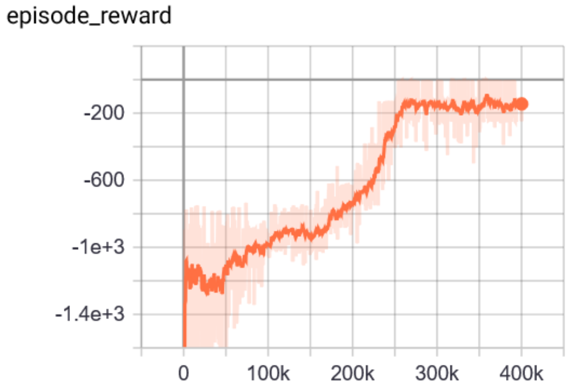
TEXAS A&M UNIVERSITY
Engineering

- Motivate the need for **sample-efficient** methods for learning complex behavior
- Pick **Imitation Learning** (IL) algorithms to learn desired behavior
- 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

- $s, s' \in S, a \in A$. Consider tuple $[\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{P}(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a}), \boldsymbol{R}(\boldsymbol{s}, \boldsymbol{a}), \gamma, H]$, define a policy (model) $\pi : S \to A$
  - **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^*

Sections

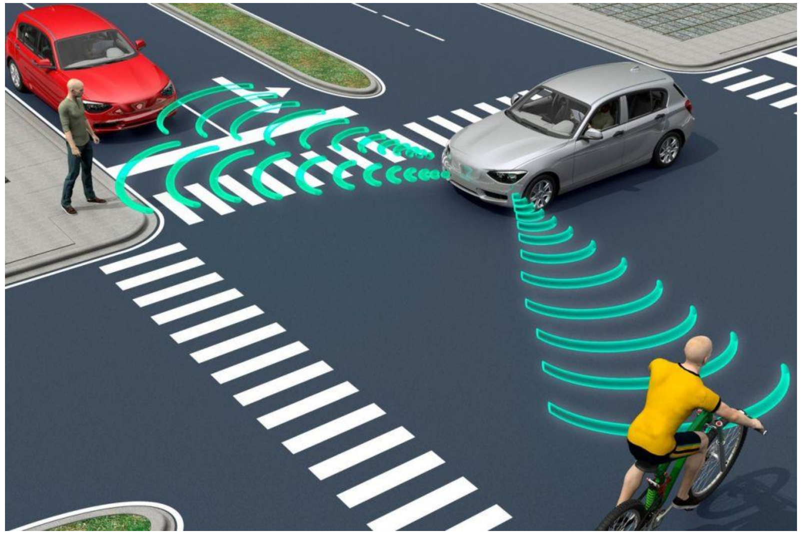
1. Introduction to Imitation Learning
2. Application 1: Autonomous UAV Landing
3. Application 2: Minecraft
4. Conclusions and Future Work

Why study imitation learning?

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

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:

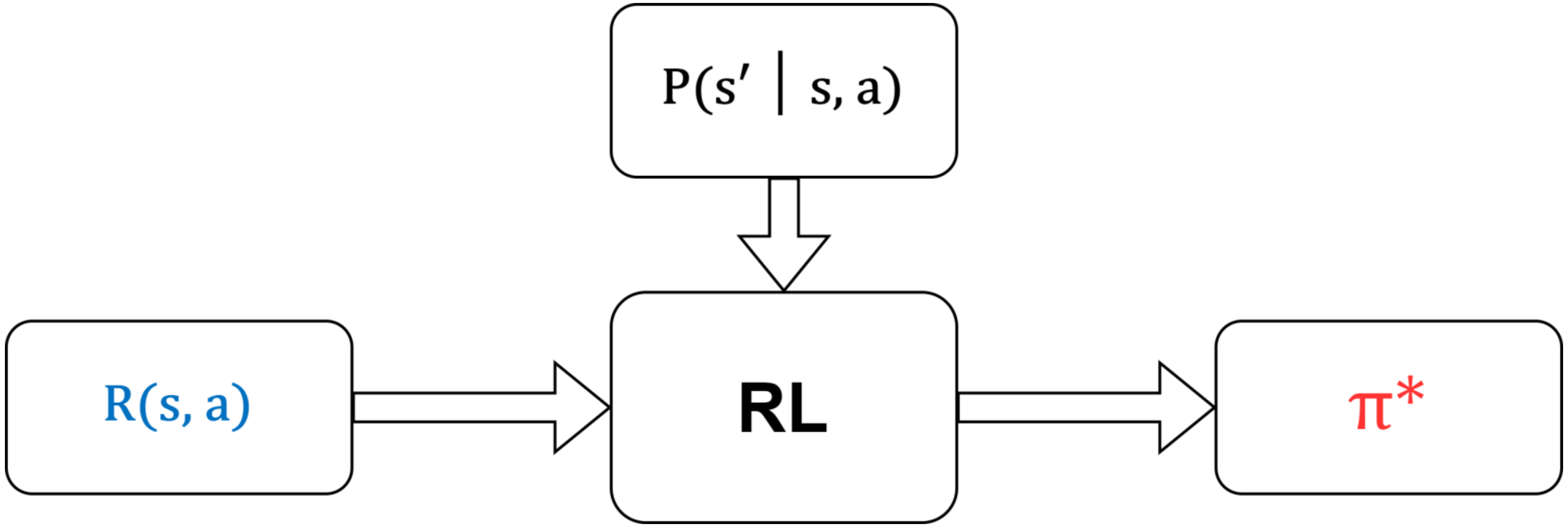
   – **NeurIPS 2019:** Game of Drones

   – **NeurIPS 2019 & 2020:** MineRL Challenge

4. How humans and animals fundamentally learn behavior

RL algorithms

- $s, s' \in S, a \in A$. 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 $\sum_{t=0}^{\infty} \gamma^t R_t$
  - **Metric:** (i) Reward convergence, (ii) Policy evaluation (testing)

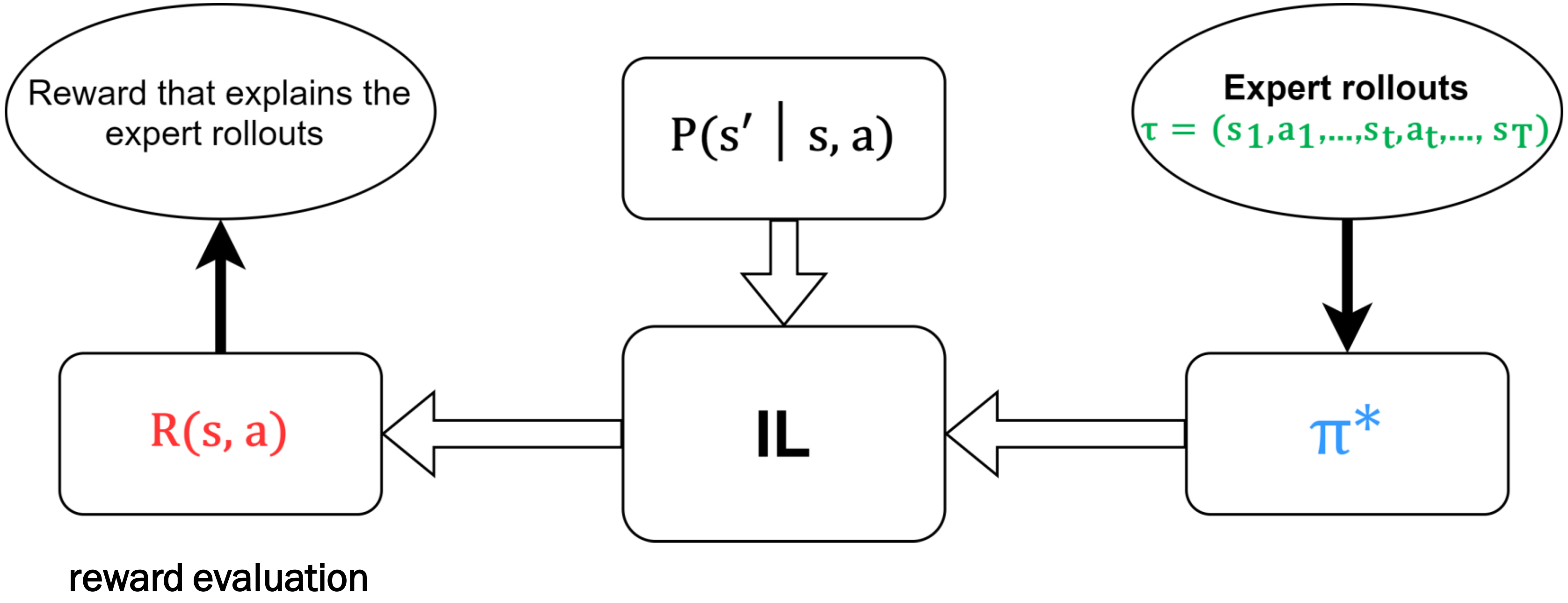


reward convergence

policy evaluation

IL algorithms

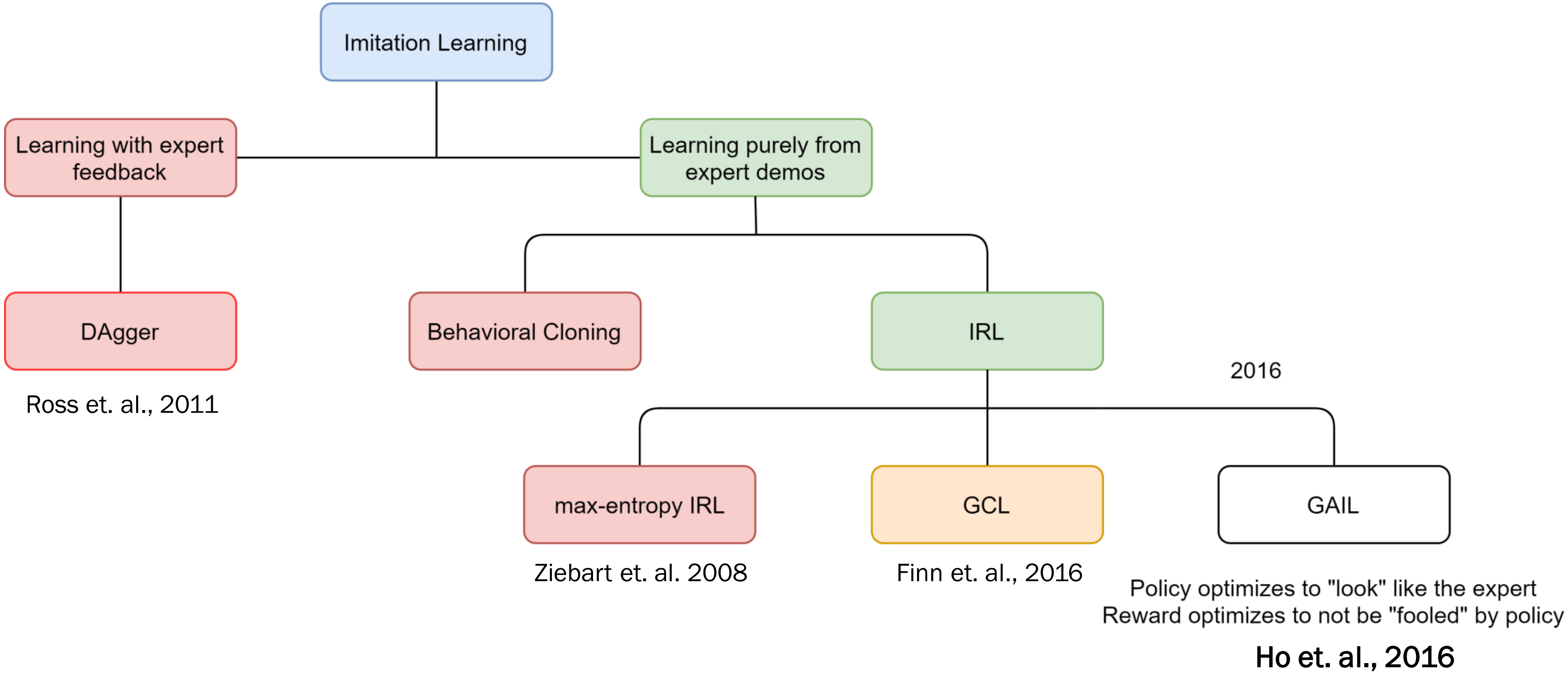
- $s, s' \in S, a \in A$. For MDP $[\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{P}(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a}), \boldsymbol{R}(\boldsymbol{s}, \boldsymbol{a}), \gamma]$, define a policy $\boldsymbol{\pi} : S \rightarrow A$
  - **Goal:** given $\boldsymbol{\tau} = (\boldsymbol{s}_0, \boldsymbol{a}_0, \boldsymbol{s}_1, \boldsymbol{a}_1, \ldots, \boldsymbol{s}_t, \boldsymbol{a}_t, \ldots, \boldsymbol{s}_T)$ generated from a $\boldsymbol{\pi}^*$, extract its $\boldsymbol{R}(\boldsymbol{s}, \boldsymbol{a})$
  - **Metric:** Reward evaluation (?)



Flowchart credits: Sapana

Imitation Learning approaches

TEXAS A&M UNIVERSITY
Engineering



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

Some questions…

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

Train RL -> rollout **expert** -> Train IL -> **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

- **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

Sections

1. Introduction to Imitation Learning
2. Application 1: Autonomous UAV Landing
3. Application 2: Minecraft
4. Conclusions and Future Work

AirSim: Autonomous UAV Navigation and Landing

APPLICATION 1

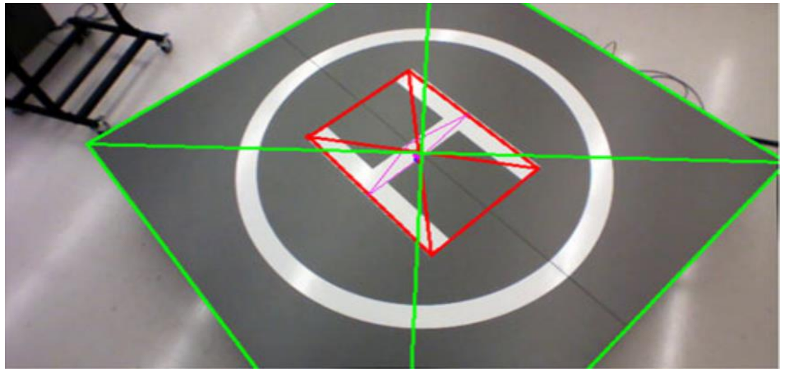
Landing on ships

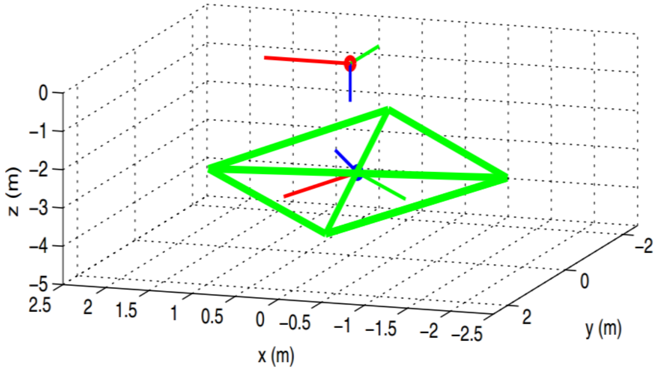
Landing Zone	Ground	Ship
Space	Large	Limited
Motion	None	6 DOF
Visual References	More	Less
Alternate L/D Places	Many	Less
Weather	Affected	Extremely affected

Slide credits: Bochan

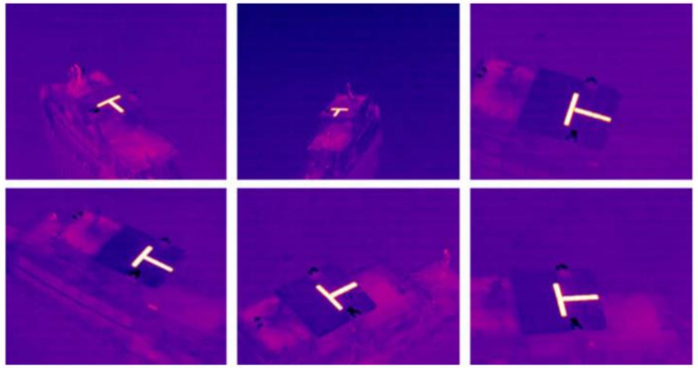
Common Approaches

TEXAS A&M UNIVERSITY
Engineering

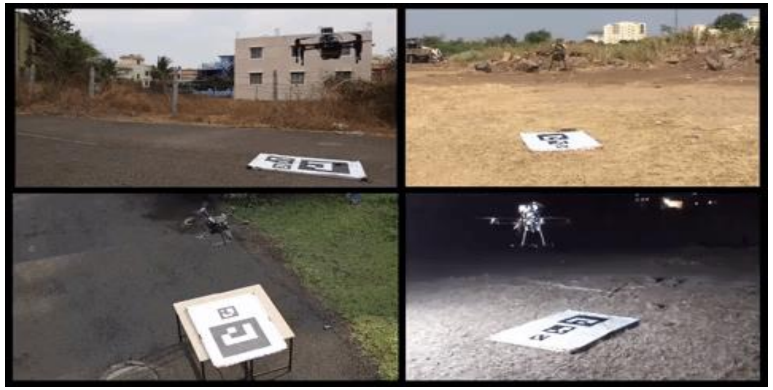




ALL of them are looking at landing spot



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?

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)

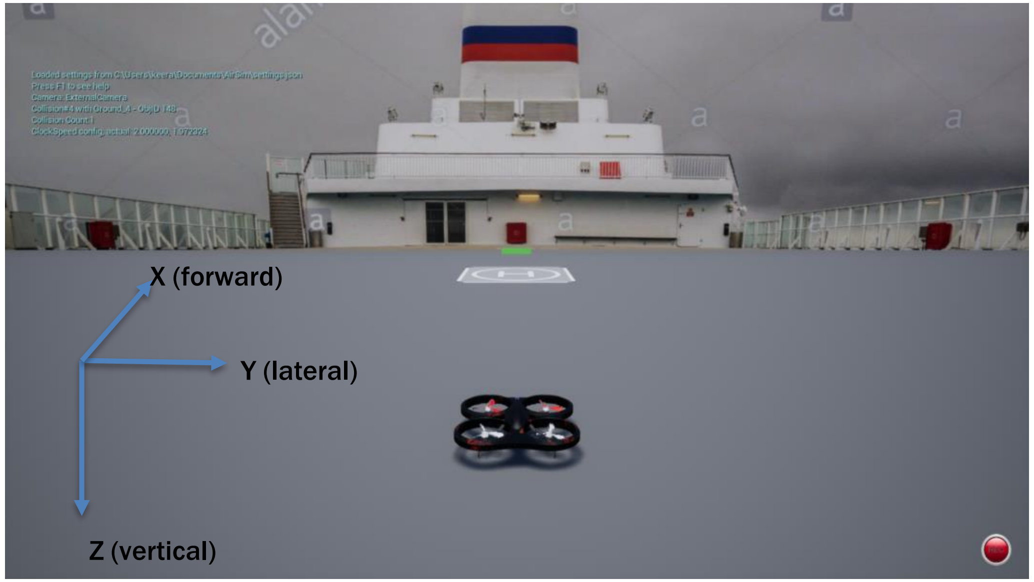
Environment simulator

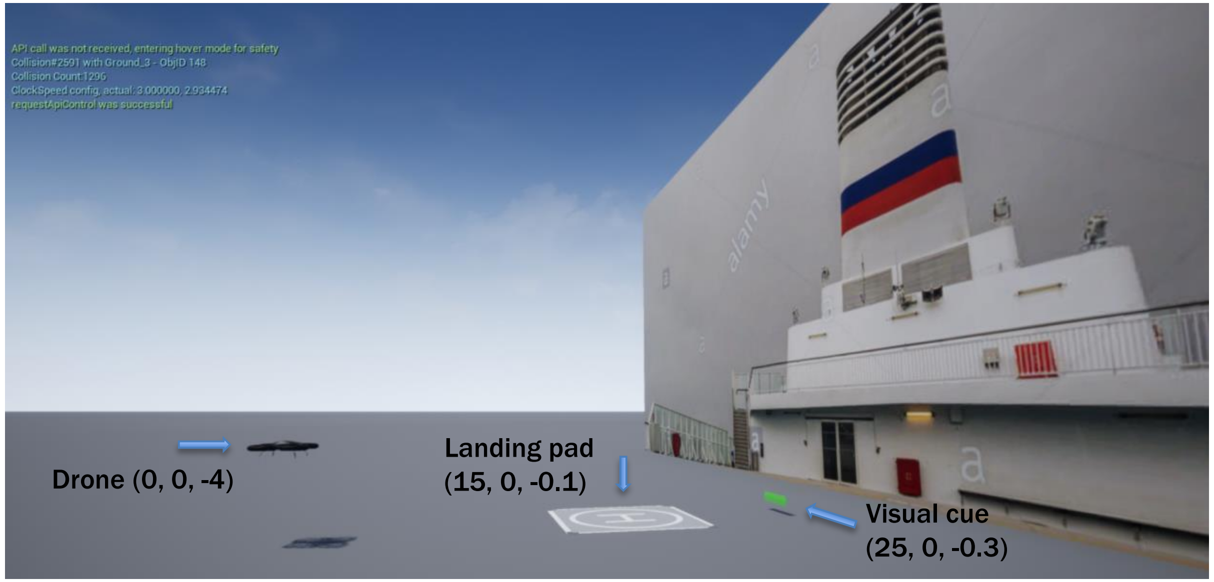
- Need a high-fidelity simulator environment for Unmanned Aerial Vehicles (UAVs)

- **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

AirSim environment: Front & Side view





The AirSim-v0 environment

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

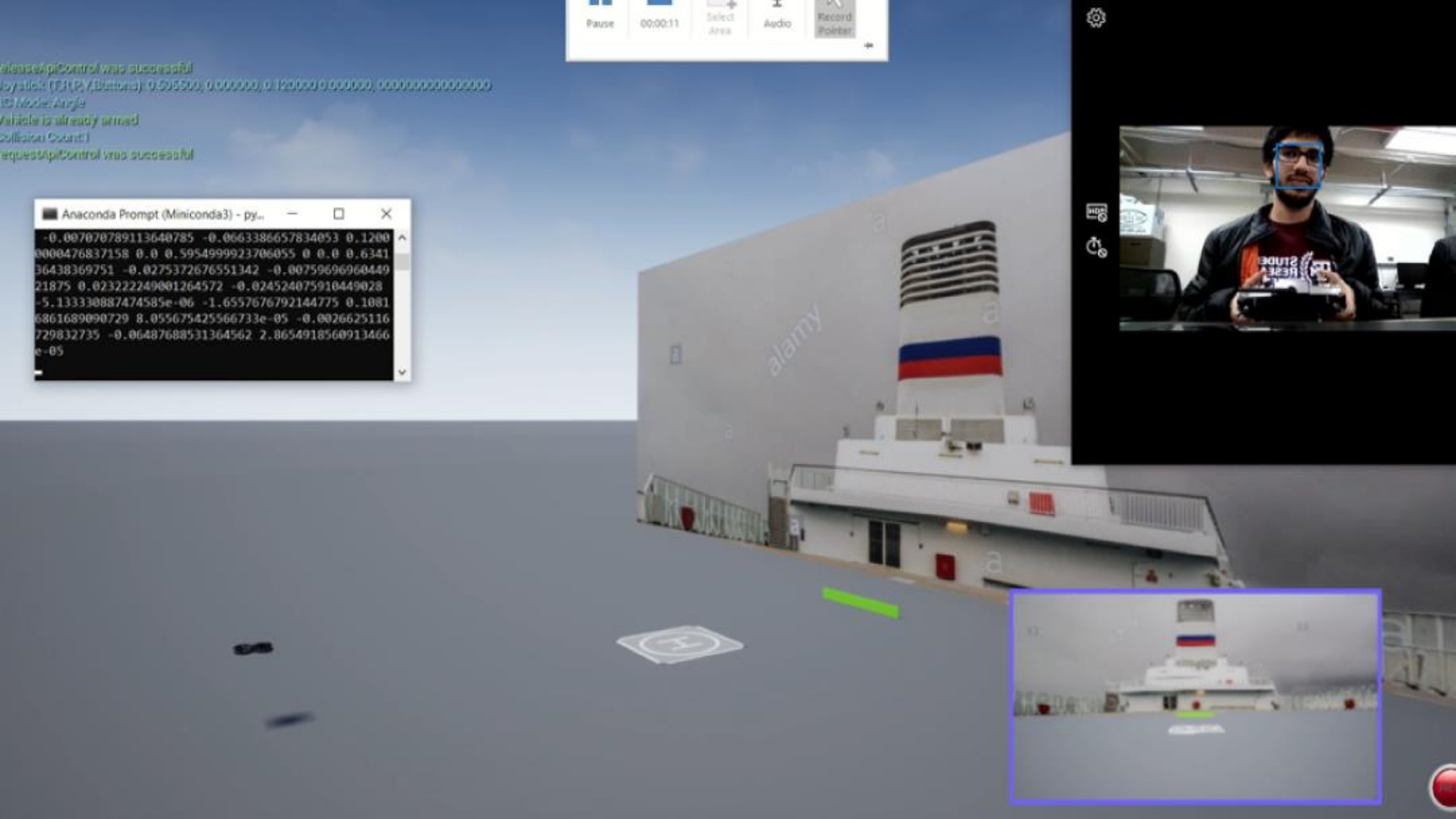
Generating human expert data

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
00:00:11
Audio
Record Pointer
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
-5.133330887474585e-06 -1.6557676792144775 0.1081
6861689090729 8.055675425566733e-05 -0.0026625116
729832735 -0.06487688531364562 2.8654918560913466
e-05
alamy

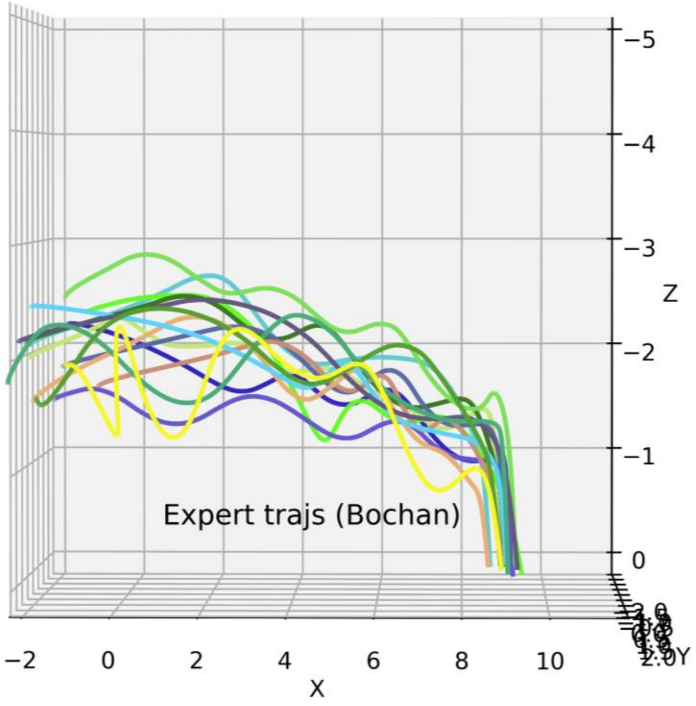
Human expert demos – stats

TEXAS A&M UNIVERSITY Engineering

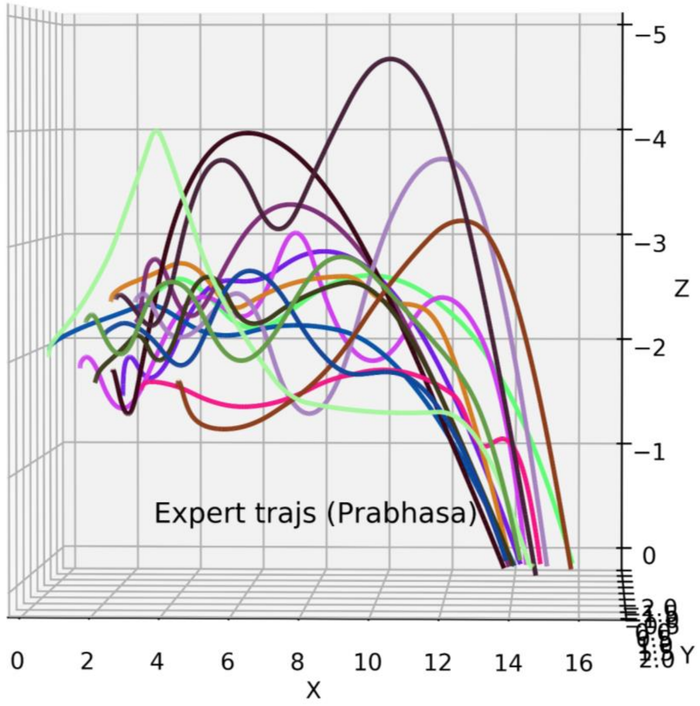
Expert	Optimal	Solved	Expert score (mean, std)	Expert length
Optimal	Yes	120/120	(1141, 27)	362
Suboptimal	No	132/140	(1116, **284**)	307

- **True reward (for proxy function):** 1000
- **Task:** Train GAIL on expert samples [**5, 10, 20**, **50**, **120**] to learn behavior (optimal/suboptimal)
- **Video of expert data collection:** https://youtu.be/e1noOIhzhQ4

Expert Trajectories: humans



Optimal



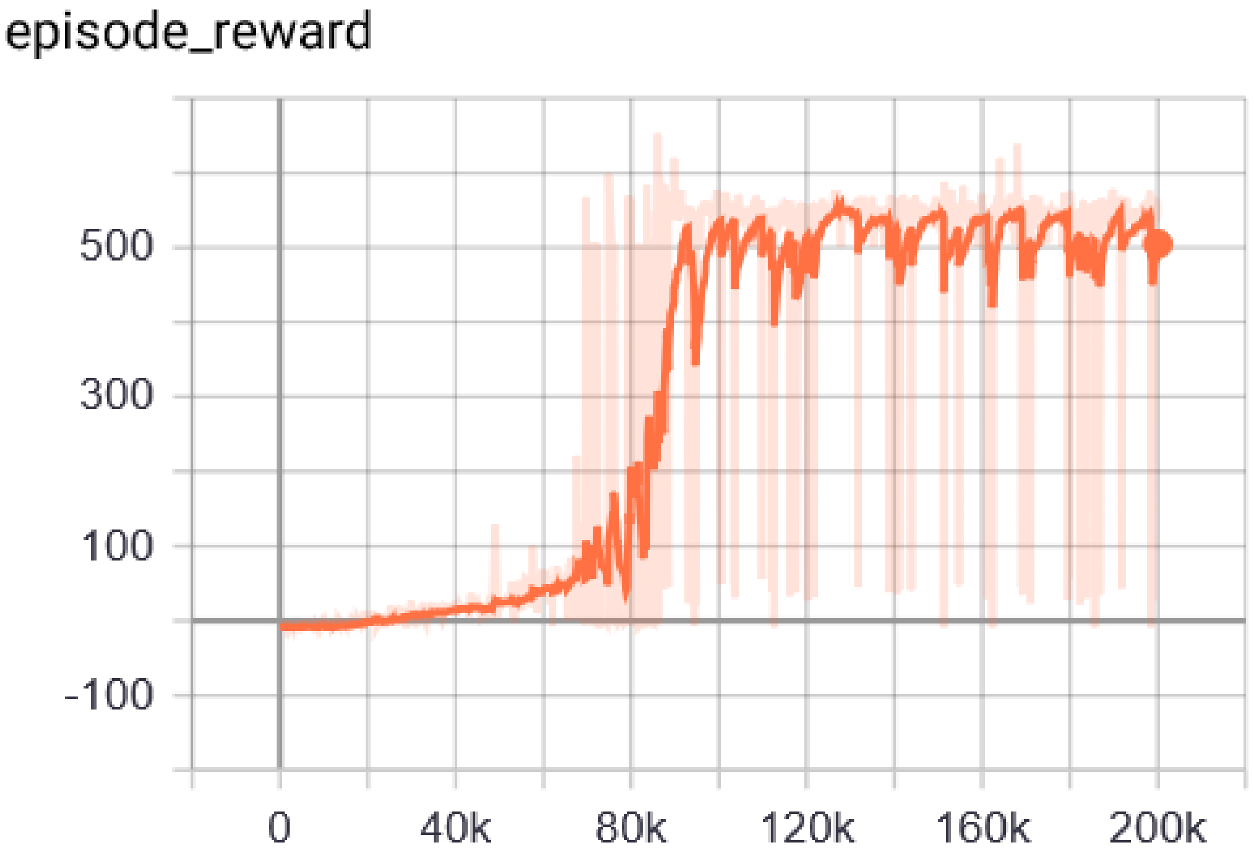
Suboptimal

GAIL on suboptimal human expert

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

Learned model: https://youtu.be/lUDpZna4uhk

AirSim-v0 (IL)
Suboptimal expert

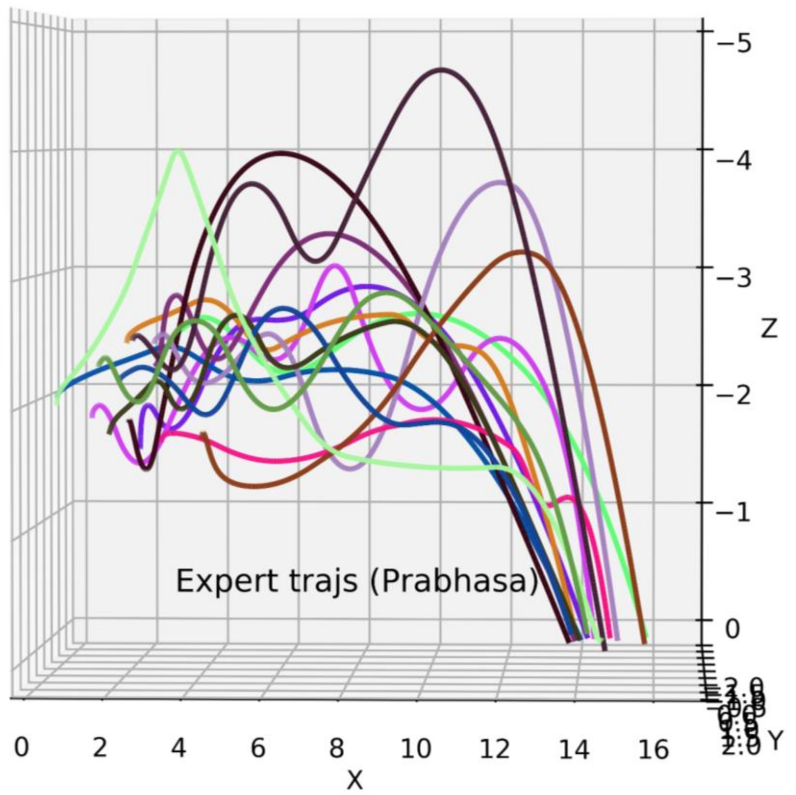


reward convergence

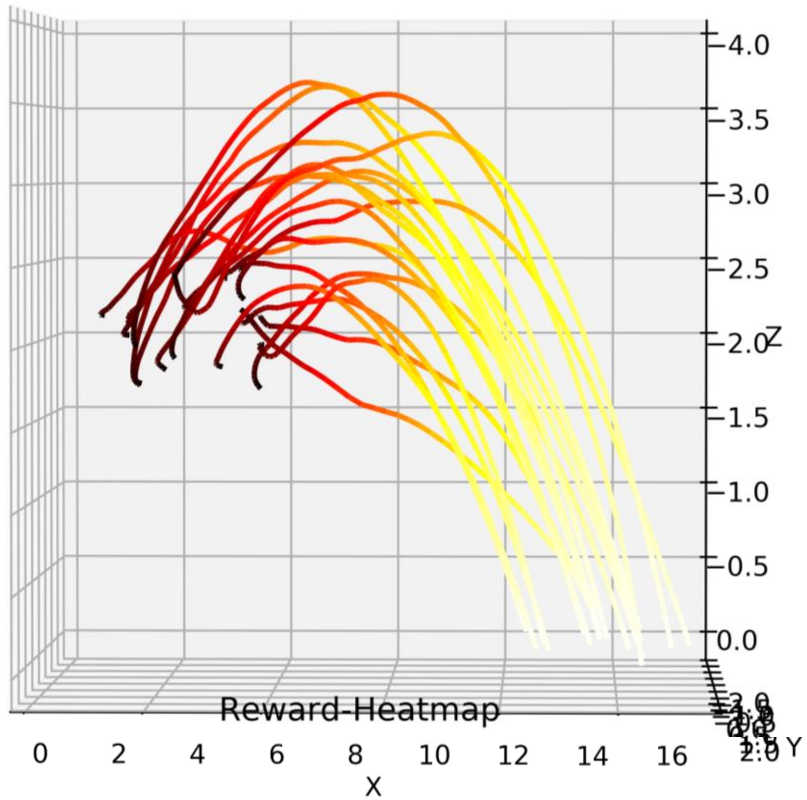
GAIL hyperparameters

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

GAIL on suboptimal human expert



Suboptimal expert



GAIL-learned policy

GAIL on optimal human expert

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

Learned model: https://youtu.be/3ilW7LzqI2Y

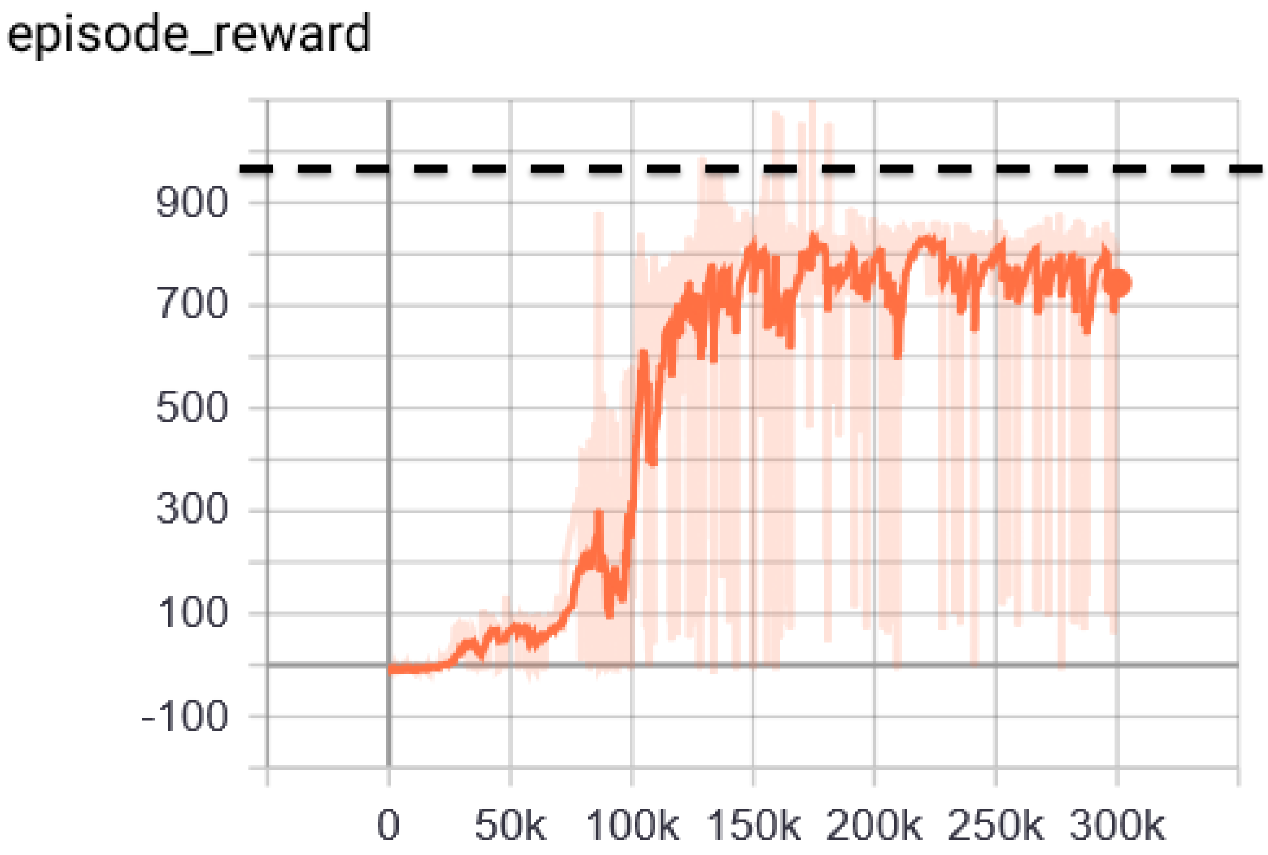
TEXAS A&M UNIVERSITY Engineering

AirSim-v0 (IL)

Optimal expert

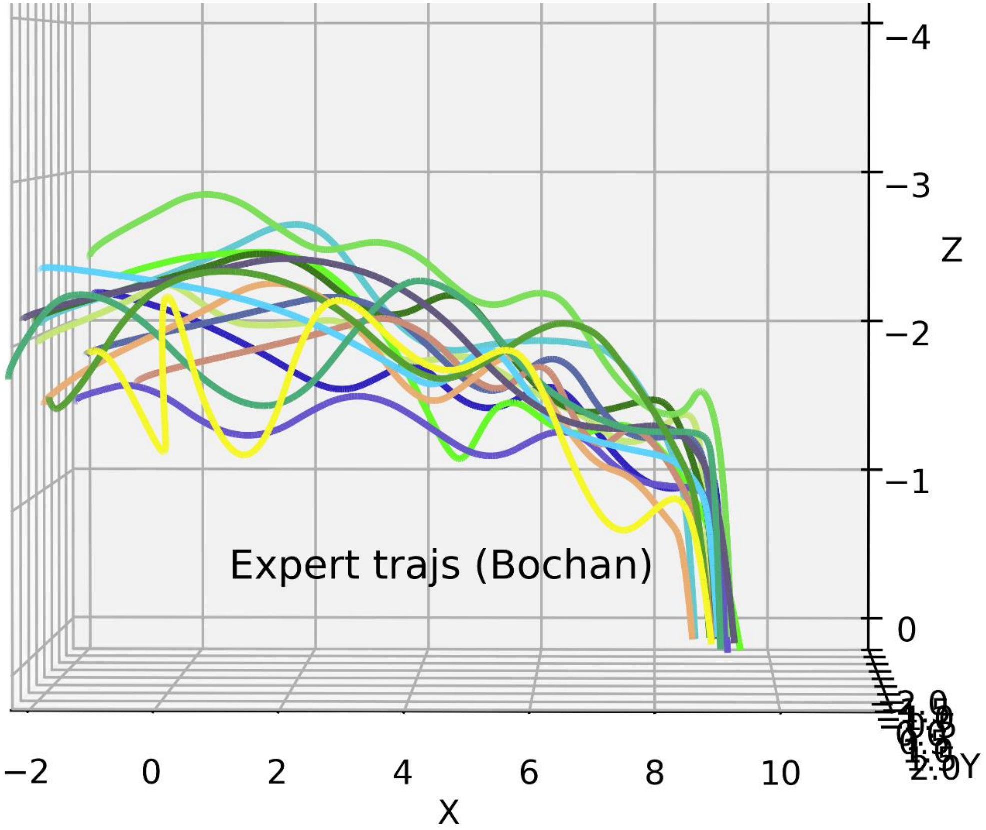
GAIL hyperparameters

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

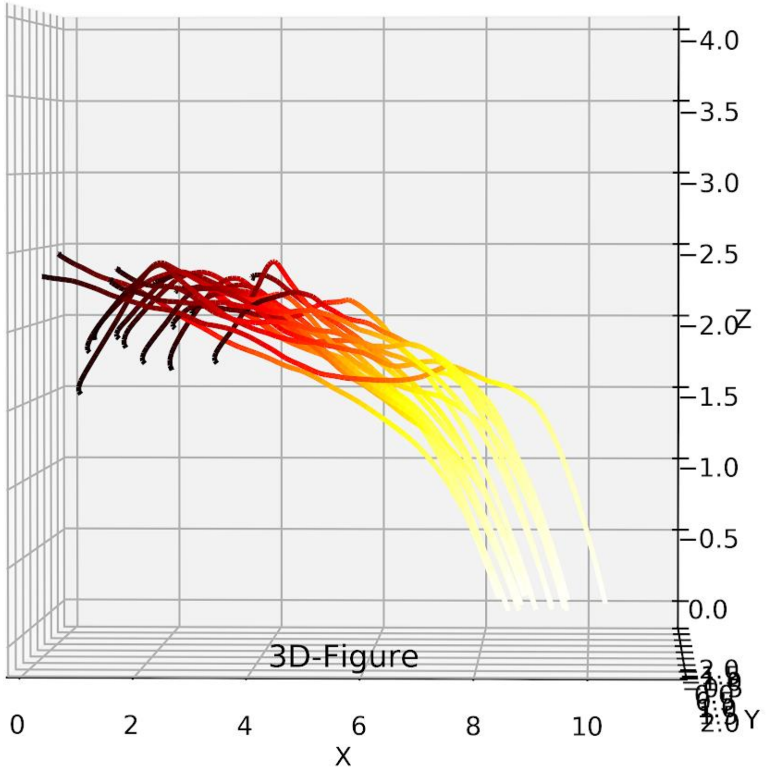


reward convergence

GAIL on optimal human expert



Optimal expert



GAIL-learned policy

Conclusions

TEXAS A&M UNIVERSITY Engineering

- GAIL can imitate **navigation** (point A to point B)
- GAIL can learn optimal **landings.** HP-dependent
- Explanation: landings may be too 'non-smooth' for GAIL to learn

- 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?

Learned models:

Simple - https://youtu.be/qJJOOOWfYcI

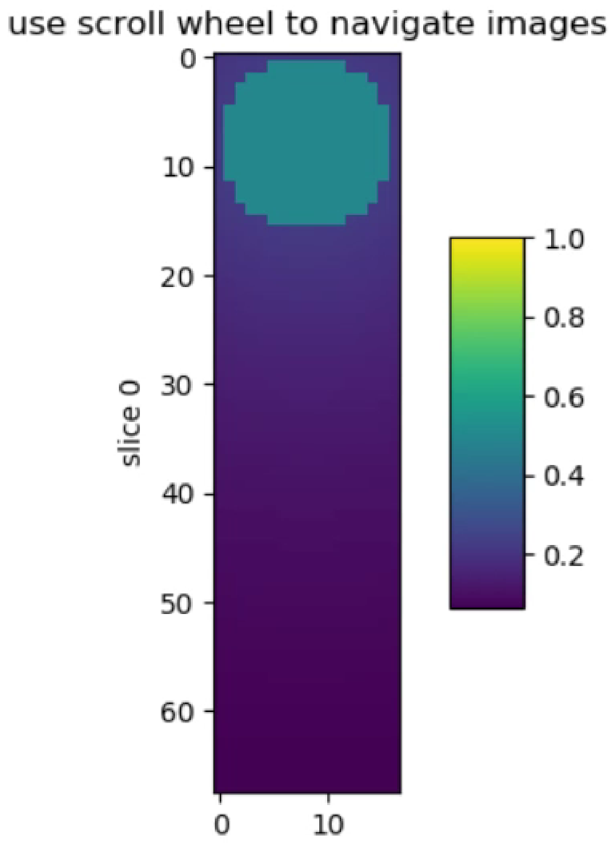
Complex - https://youtu.be/cFpFTDo-V7k

Proxy reward function design

TEXAS A&M UNIVERSITY Engineering

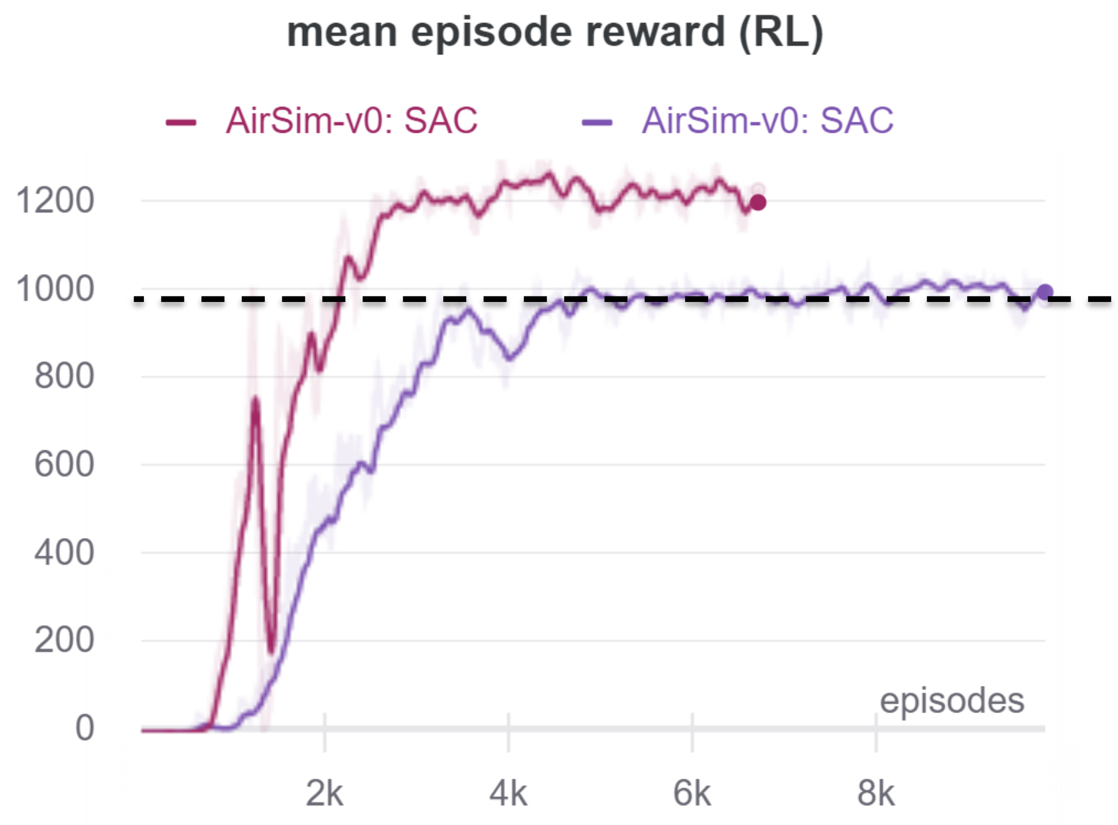
1. **Simple reward (sparse):**
   – Increase reward as it gets close to landing pad (1/x)
   – Large positive reward if it lands inside the landing pad (+1000)
   – Other conditions: -10 (visual cue, out of bounds, crash, timeout)

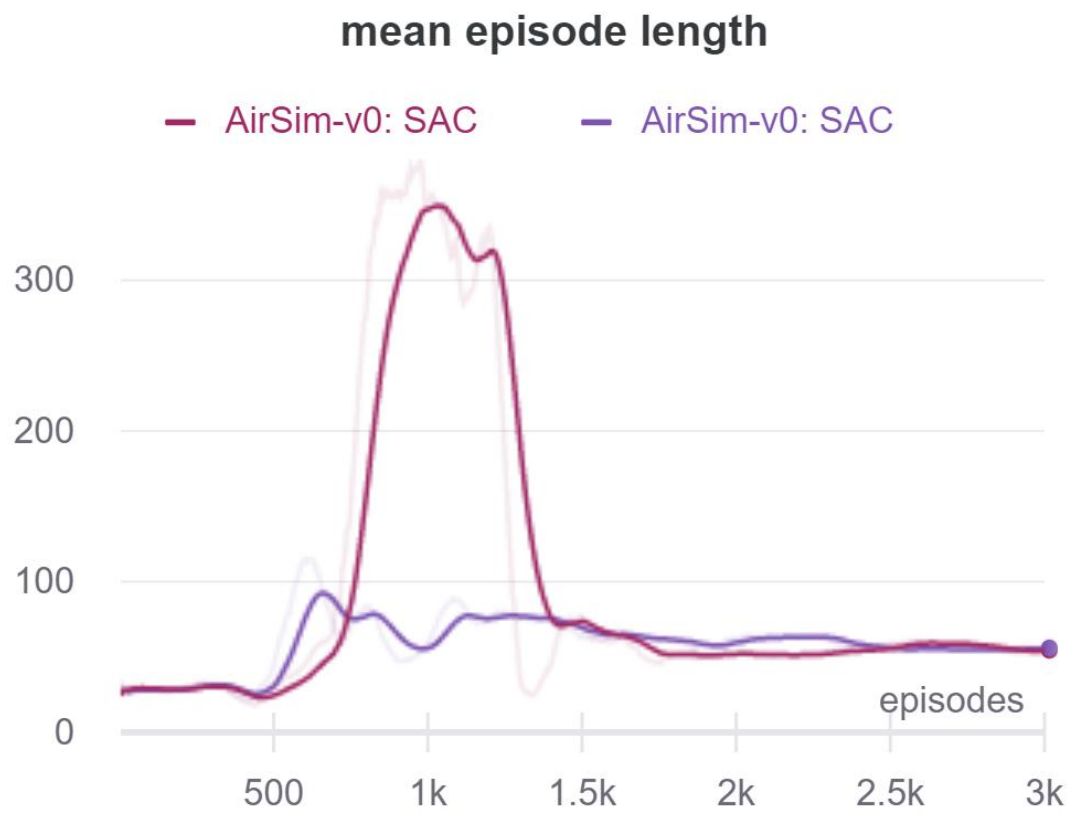
2. **Complex reward (sparse):**
   – Increase reward as it gets close to landing pad (1/x)
   – **Scale** goal reward according to drone heading, speed (1250-750)
   – Other conditions: -10 (visual cue, out of bounds, crash, timeout)



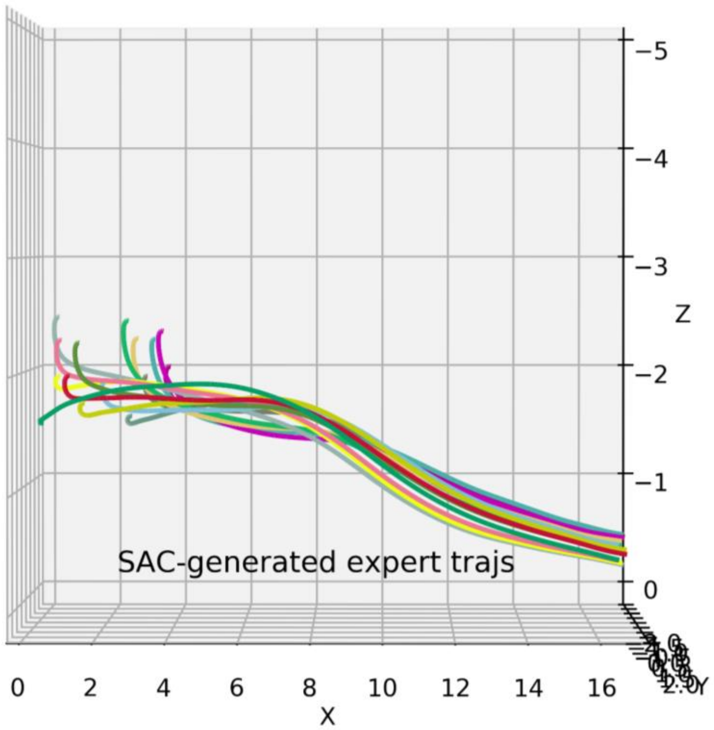
Reward Heatmap

SAC on AirSim-v0: Proxy rewards

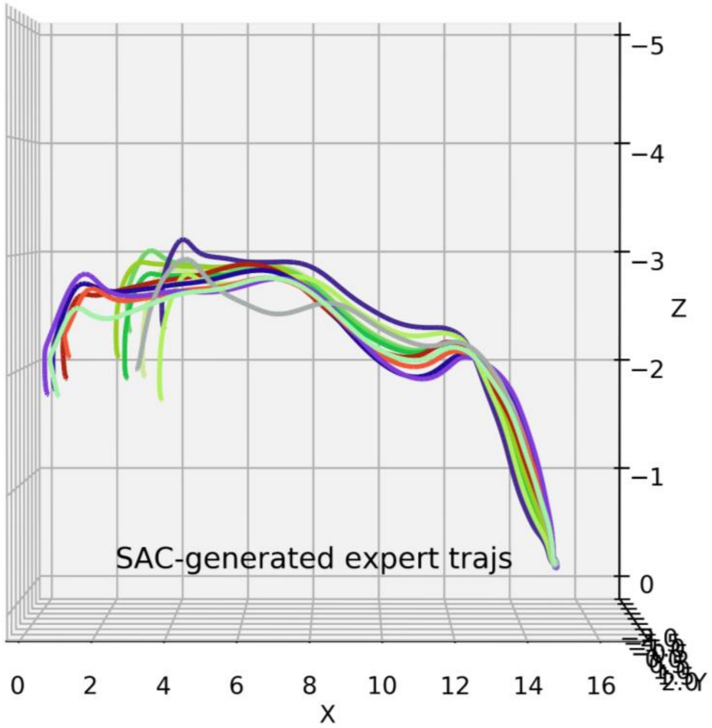




RL-generated expert demos



Proxy: simple

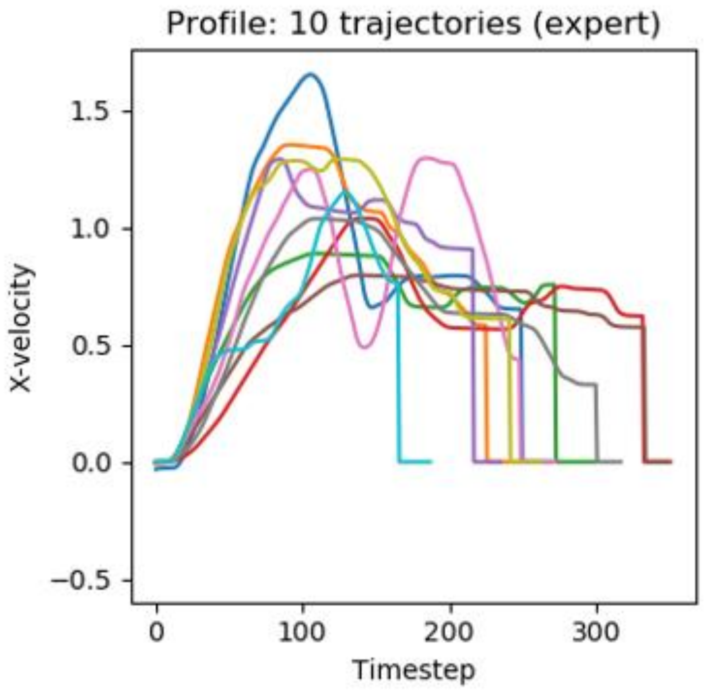


Proxy: complex

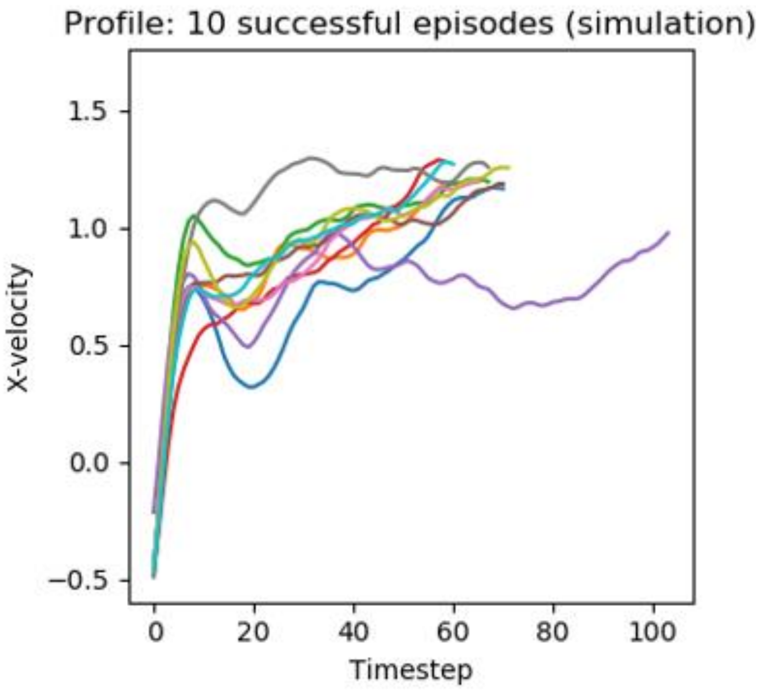
LANDING: HUMANS vs GAIL vs RL

Change in speed

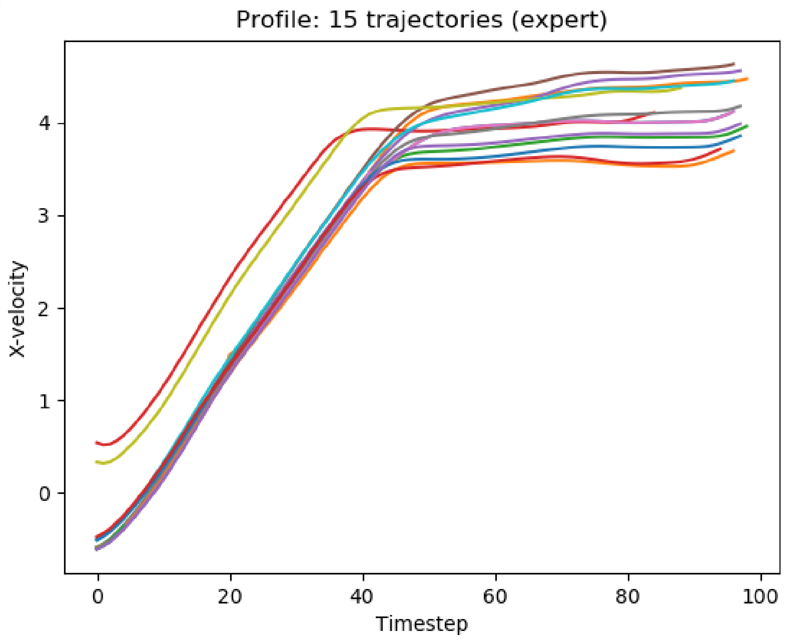
Forward speed



Human expert

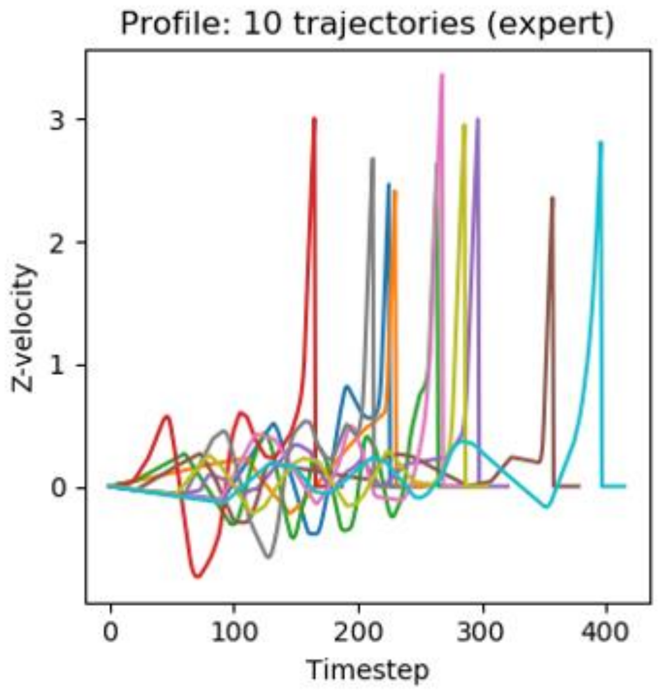


GAIL-learned policy

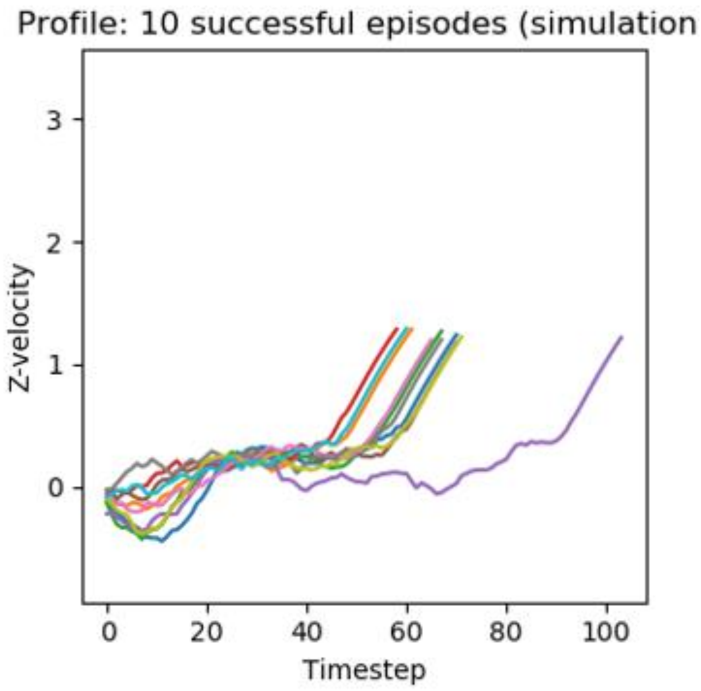


RL expert

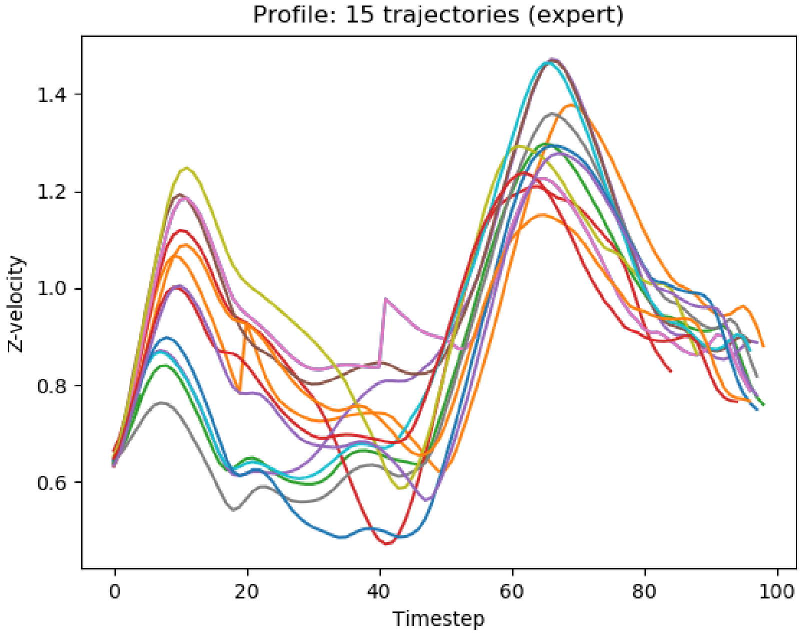
Descent speed



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…

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

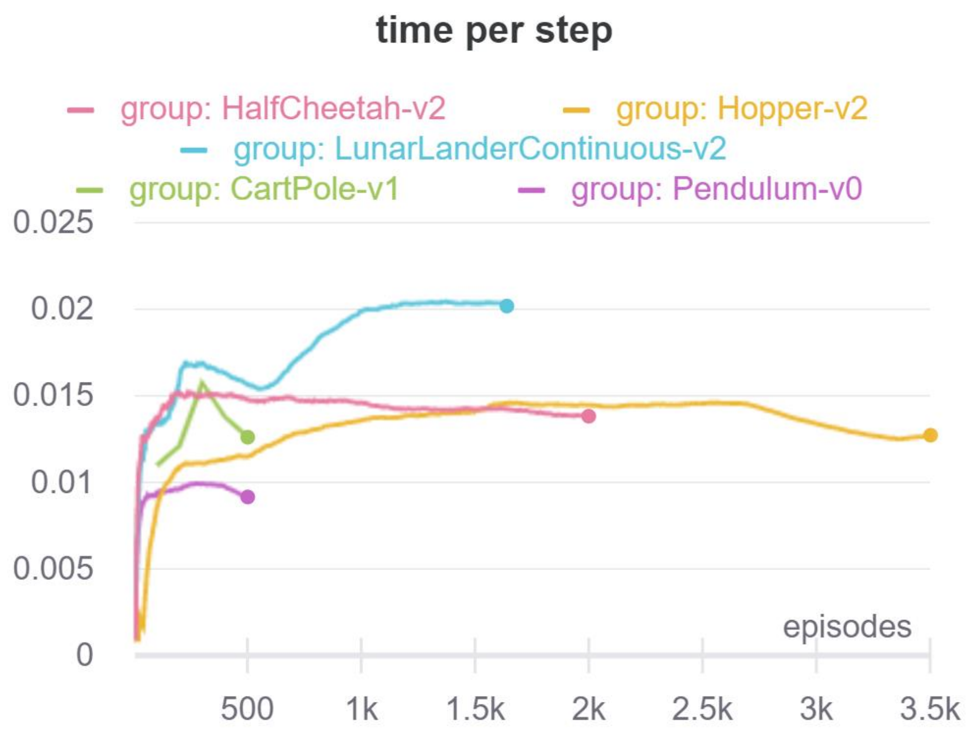
Summary

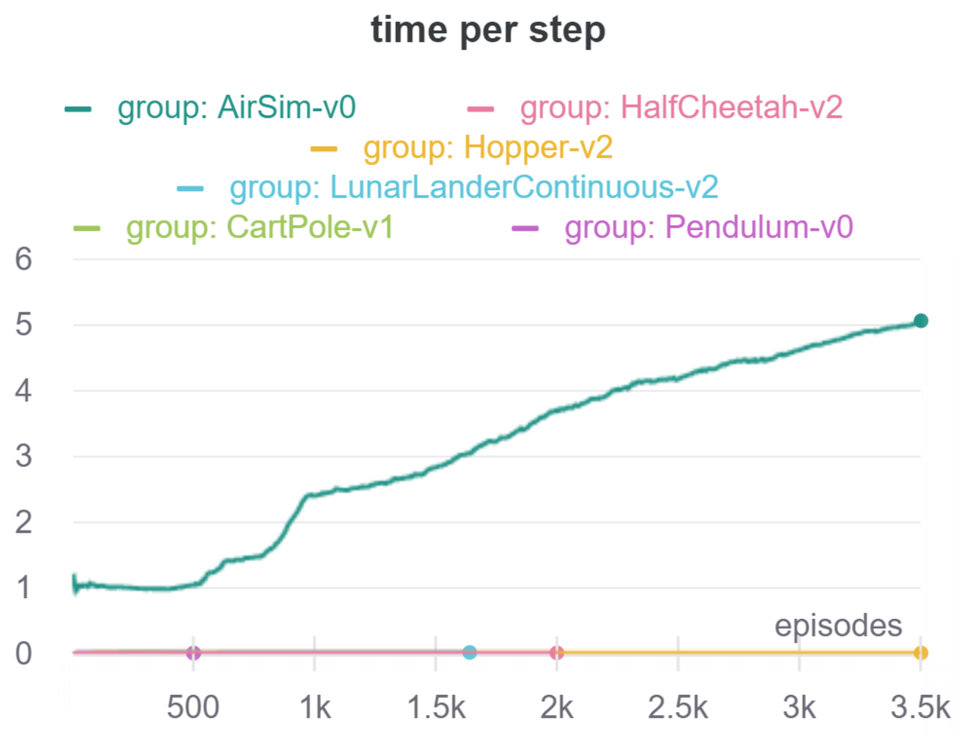
TEXAS A&M UNIVERSITY Engineering

- 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

Environment samples are expensive!

TEXAS A&M UNIVERSITY
Engineering

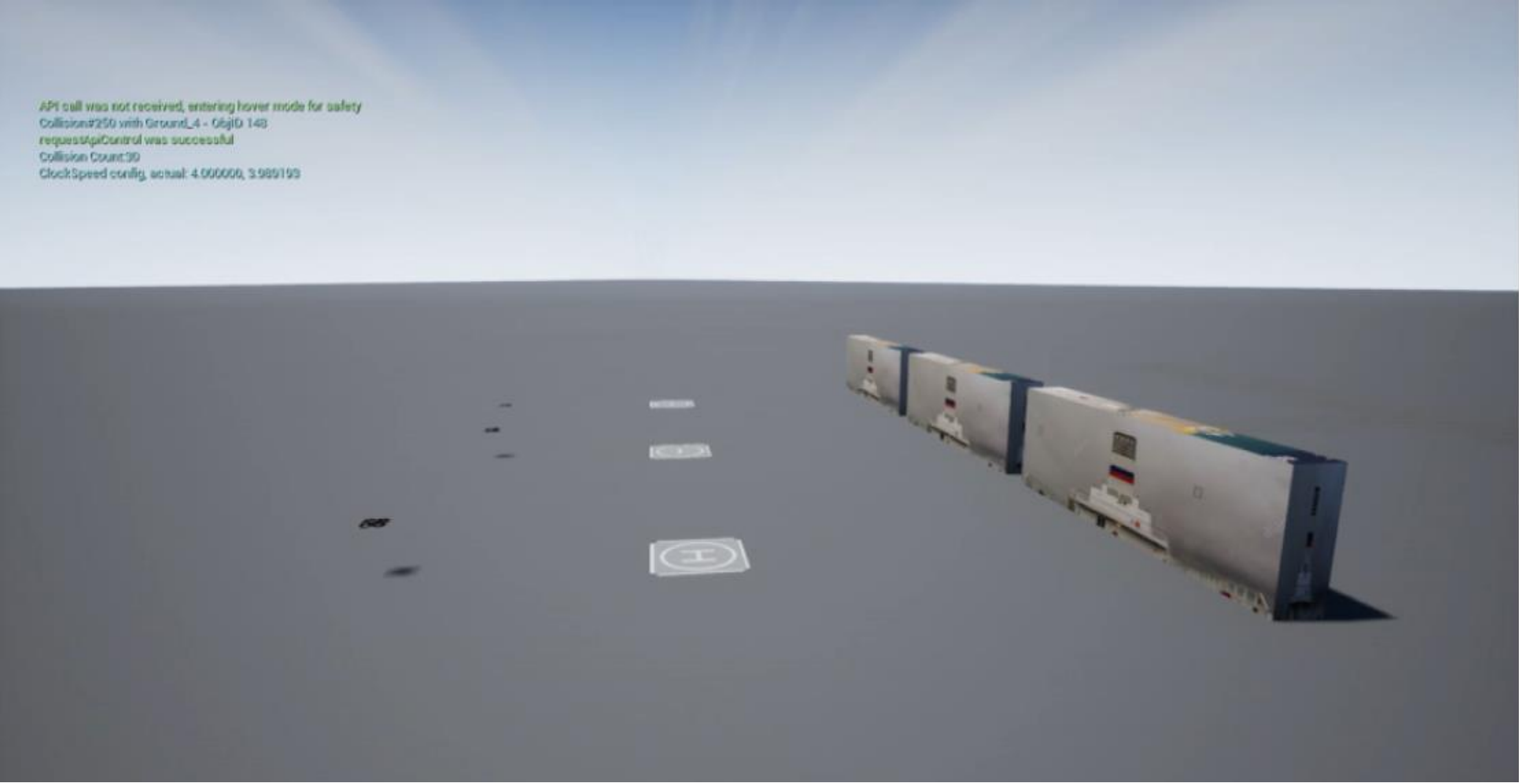




Some numbers to crunch on...

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


API call was not received, entering hover mode for safety
Collision#250 with Ground_4 - ObjID 148
requestApiControl was successful
Collision Count:30
ClockSpeed config, actual: 4.000000, 3.989193

Sections

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

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

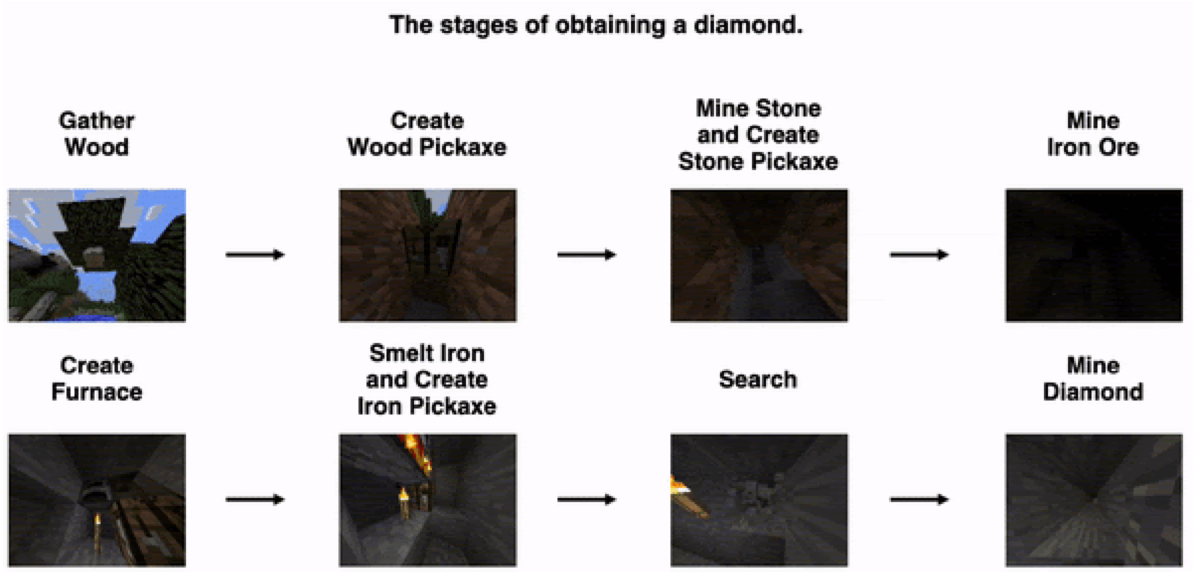
- What's new from 2019: Vectorized state, action space that **obfuscates** the agent's actions
  - Prevent participants from using domain knowledge
  - **State:** images + 1-D vector containing comprehensive set of features from the game
  - **Actions:** 1-D vector containing keyboard presses, mouse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

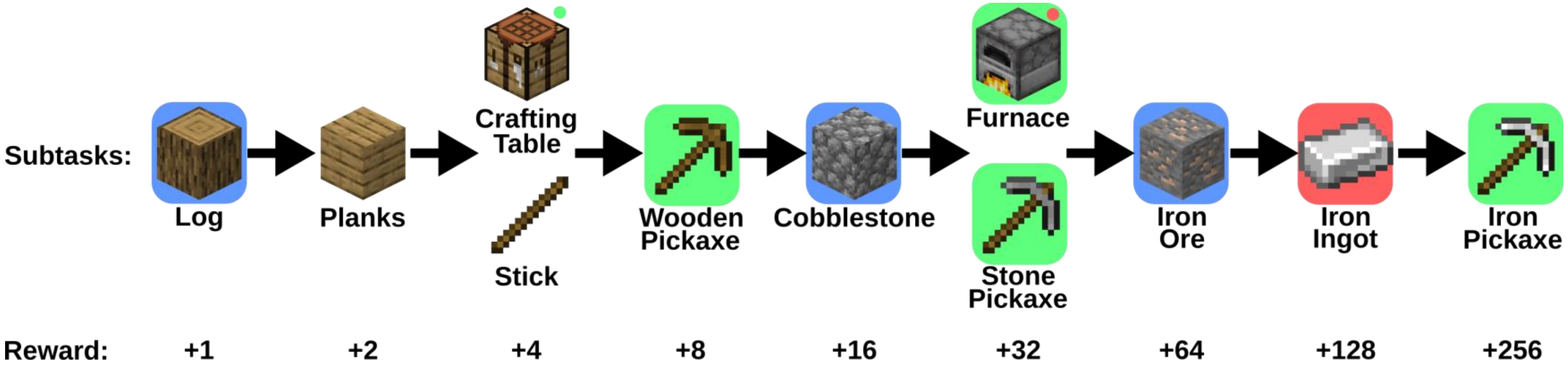
Visualizing the MineRL envs & dataset

MineRLTreeChopVectorObf-v0: https://youtu.be/q9DtmFJMc5I

MineRLObtainDiamondVectorObf-v0: https://youtu.be/mexGyw1PoT0

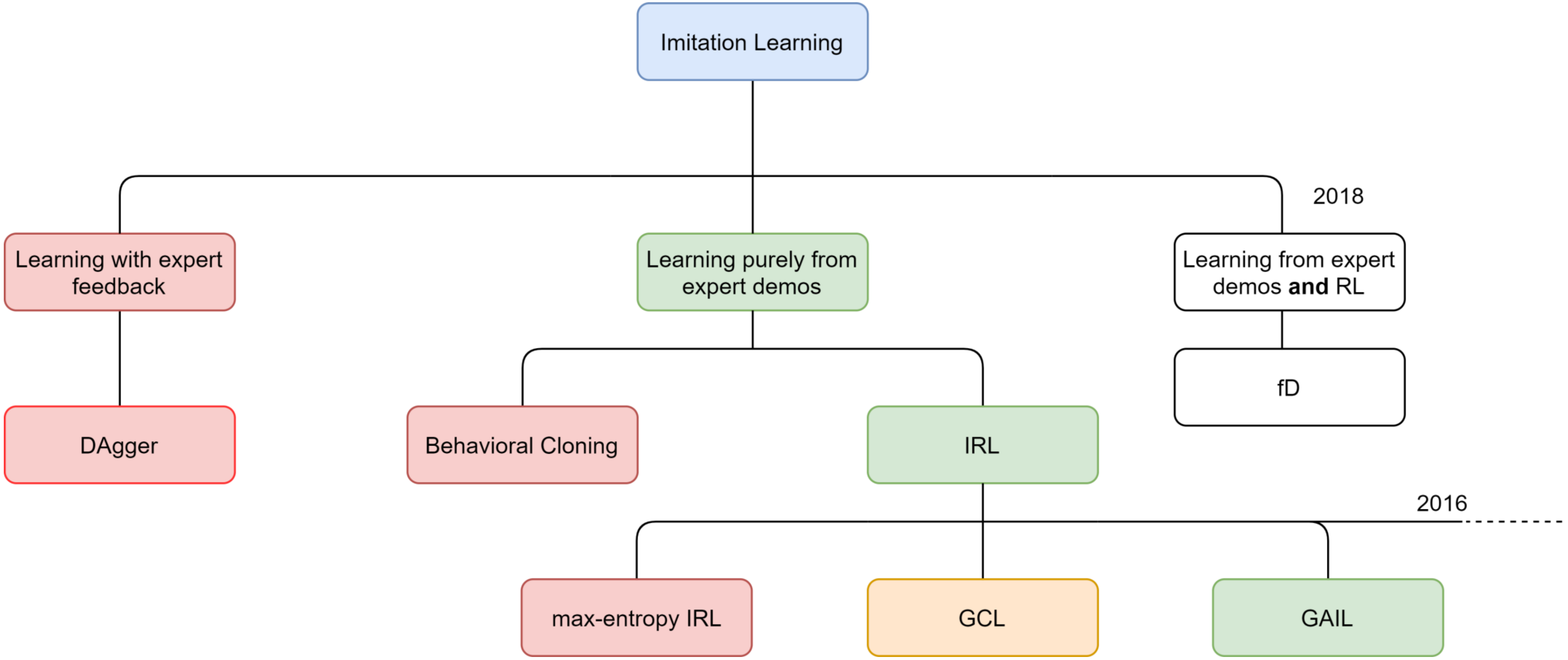
Obtain Diamond: Tasks and Rewards





RL with human priors (RL + IL!)

TEXAS A&M UNIVERSITY Engineering



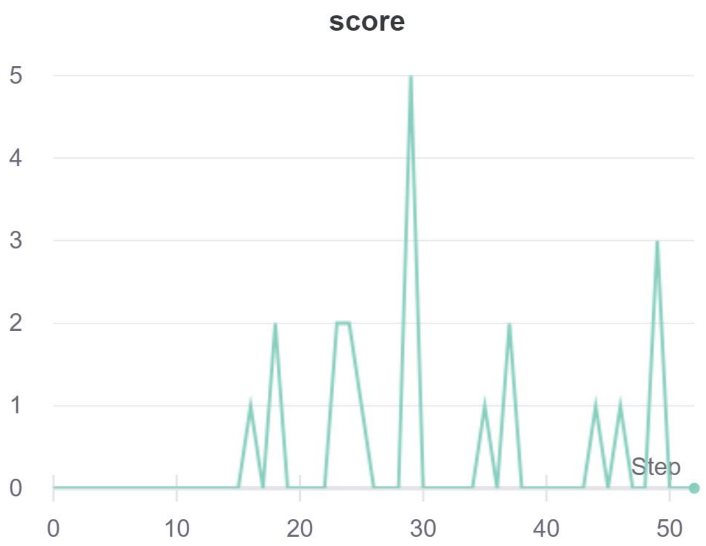
Tools

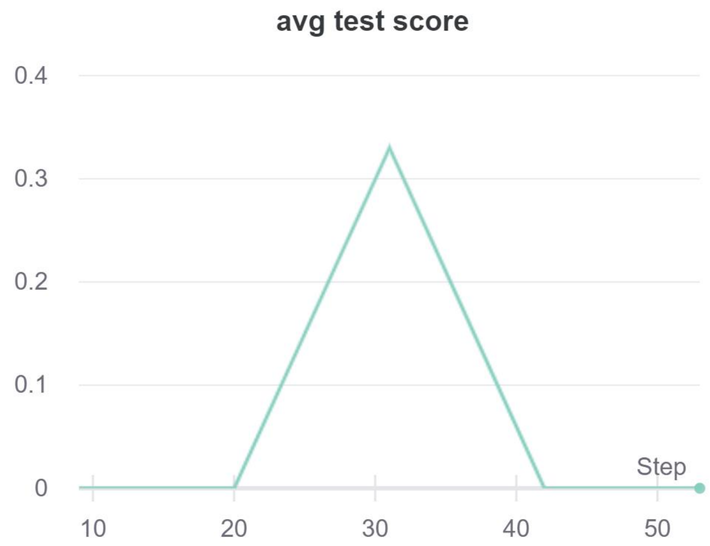
- **RL library:** Medipixel 0.10
- **Framework:** Pytorch 1.3.1
- **Hyperparameters (HPs):** Medipixel 0.10
- **Results (train score vs episodes, test score):** W&B 0.10

DQN (RL) vs DQfD (RL+IL)

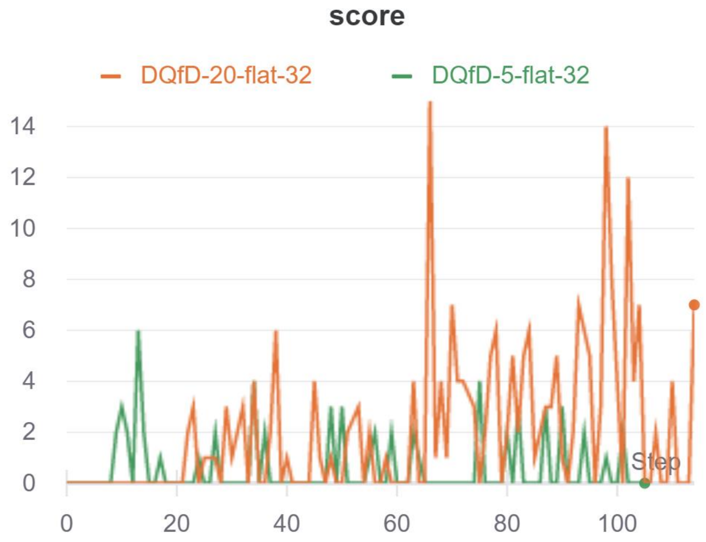
MineRLTreeChopVectorObf-v0: https://youtu.be/YDpVRyZndCg

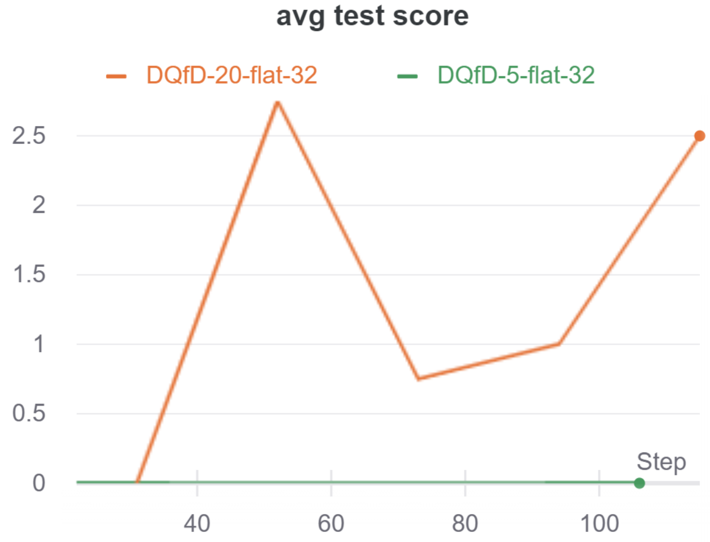
MineRLObtainDiamondVectorObf-v0: https://youtu.be/b-SGp7PKbxM


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

1. Introduction to Imitation Learning
2. Application 1: Autonomous UAV Landing
3. Application 2: Minecraft
4. Conclusions and Future Work

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

TEXAS A&M UNIVERSITY
Engineering

THANK YOU!