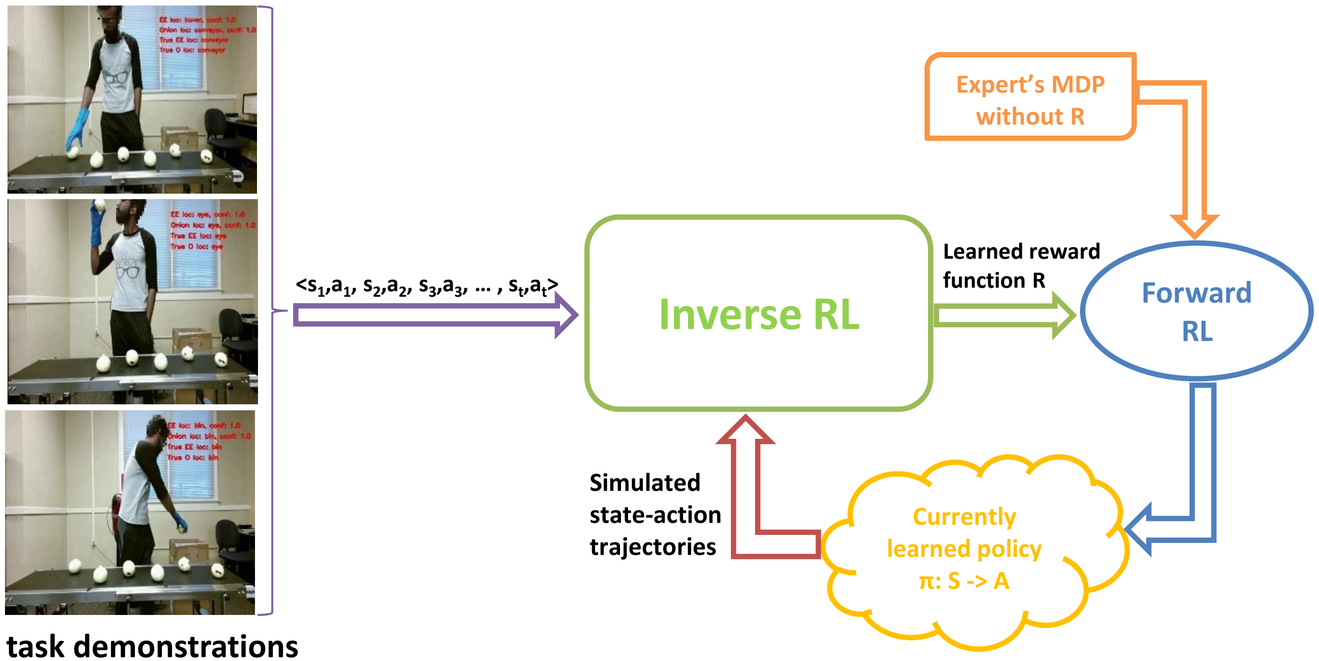
Dec-AIRL: Decentralized adversarial IRL for human-robot teaming

Prasanth Suresh, Yikang Gui, Prashant Doshi

THINC Lab, School of Computing, UGA, Athens, GA - 30605

Inverse Reinforcement Learning

- Popular Machine Learning technique
- Using expert demonstrations learner learns a reward function modelling the expert’s preferences


$<s_1,a_1, s_2,a_2, s_3,a_3, \dots , s_t,a_t>$
Inverse RL
Expert’s MDP without R
Learned reward function R
Forward RL
Currently learned policy π: S -> A
Simulated state-action trajectories

Expert task demonstrations converted to state-action pairs

Human-robot teaming

- Cobot collaboratively working with a human in a shared workspace
- Learning goals -> Task learning + avoiding adverse interactions (such as trying to pick same object as the human simultaneously)
- Complex behavior non-trivial to program
- Learning via online RL entails reward engineering
- Physical robot could try potentially dangerous actions during learning
- Can learn task and collaboration through human-human team demonstrations instead

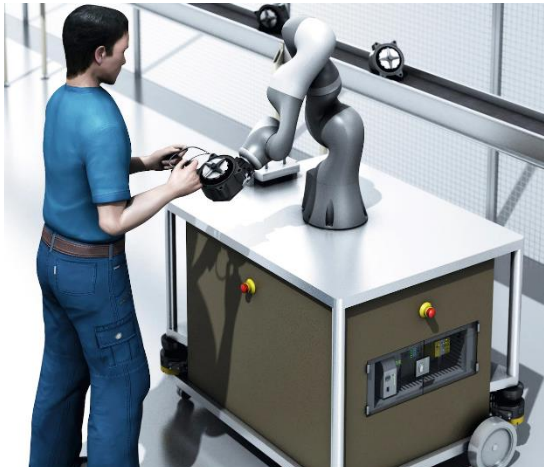


Figure: An example of human-robot collaboration in a shared workspace. Credit: Sick sensor inc.

Figure: Human-human team demonstrations of the onion-sorting task.

Related work

- Existing state-of-the-art multi-agent IRL techniques:
  - MA-AIRL (Yu, Lantao, et al. , ICML 2019) :
    - π: S -> A_i where A_i is the action space of agent i
    - Underlying model: Markov game
    - Self-interested agents
- Existing state-of-the-art multi-agent IL techniques:
  - Co-GAIL (Wang, Chen, et al. , CoRL 2022):
    - π: S -> A where S, A are the global state-action space
    - Underlying model: Multi-agent MDP
    - Fully centralized control

Decentralized AIRL: Overview

Formulates task as a decentralized MDP

Decentralized AIRL: Overview

Formulates task as a decentralized MDP

Models agents with local full-observability

Decentralized AIRL: Overview

Formulates task as a decentralized MDP

Models agents with local full-observability

Single reward function for the system

Decentralized AIRL: Overview

Formulates task as a decentralized MDP

Models agents with local full-observability

Single reward function for the system

Centralized Training Decentralized Execution (CTDE) using Dec-PPO

Decentralized AIRL: Overview

Formulates task as a decentralized MDP

Models agents with local full-observability

Single reward function for the system

Centralized Training Decentralized Execution (CTDE) using Dec-PPO

Learns a vector of policies (one for each agent)

Decentralized AIRL: Overview

Formulates task as a decentralized MDP

Models agents with local full-observability

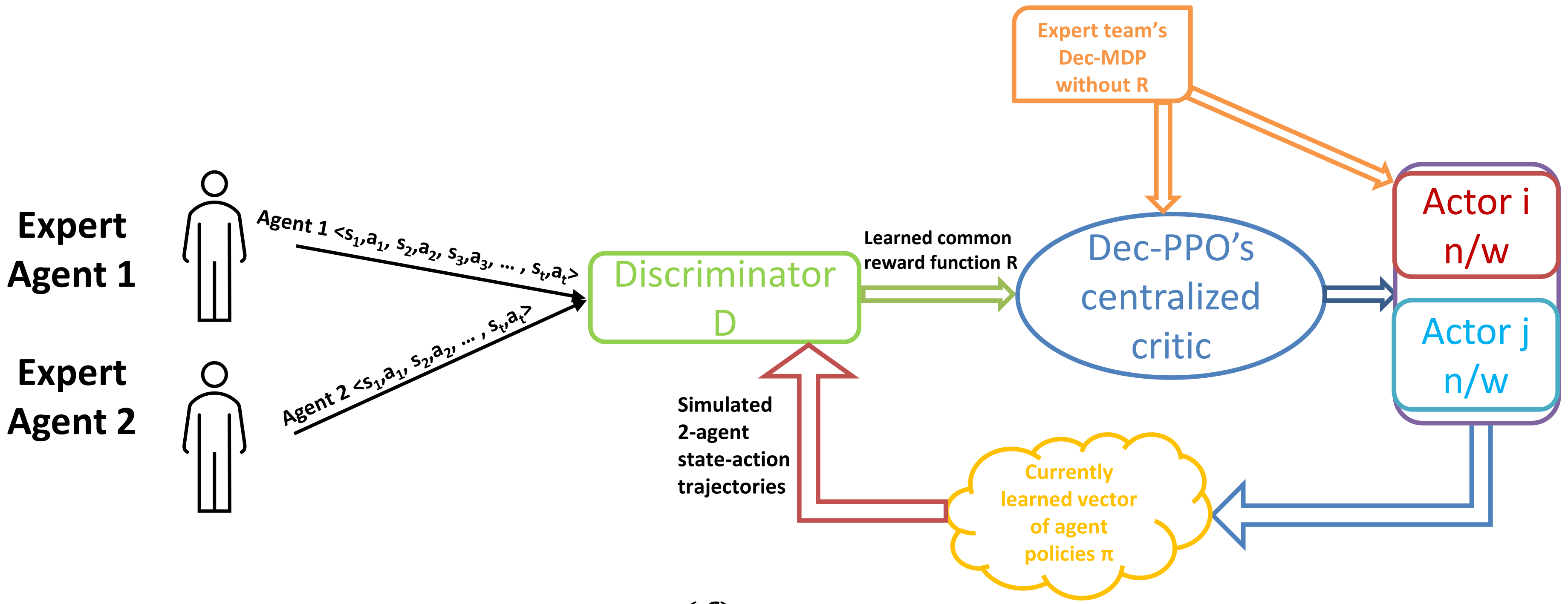
Single reward function for the system

Centralized Training Decentralized Execution (CTDE) using Dec-PPO

Learns a vector of policies (one for each agent)

Agent policy maps agent state to agent action

Decentralized Adversarial IRL


Expert team's Dec-MDP without R
Expert Agent 1
Expert Agent 2
Agent 1 <s1,a1, s2,a2, s3,a3, … , st,at>
Agent 2 <s1,a1, s2,a2, … , st,at>
Discriminator D
Learned common reward function R
Dec-PPO's centralized critic
Actor i n/w
Actor j n/w
Simulated 2-agent state-action trajectories
Currently learned vector of agent policies π

Discriminator $D = \dfrac{\exp(f)}{(\exp(f) + \pi)}$; where f – Advantage function

π – Learned joint policy

$R \leftarrow \log D - \log(1 - D) = f - \log \pi$; where R is the entropy regularized common reward

Decentralized PPO: Overview

- Single agent PPO – Actor-critic based RL method
- Dec-PPO - multi-agent generalization of PPO
- One centralized critic for many decentralized actors

Train critic network using following loss fn:

$$L_t^{VF}(\omega) = \left(V_\omega^\pi(s^t) - \hat{V}^{targ}(s^t)\right)^2$$ <- Critic value function loss

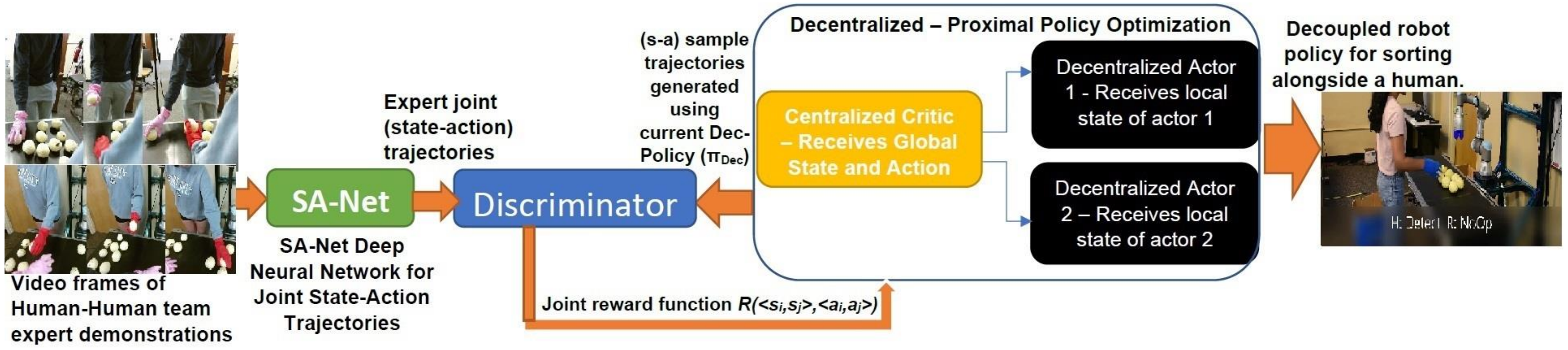
Train actor networks using following loss fn:

$$L_i^{CLIP}(\omega) = E_{(s_i^t,a_i^t)\sim\pi_{\omega,i}}[min(\lambda_i A^{\pi_{\omega,i}}, clip(\lambda_i, 1-\epsilon, 1+\epsilon)A^{\pi_{\omega,i}})]$$

$$L_j^{CLIP}(\omega) = E_{(s_j^t,a_j^t)\sim\pi_{\omega,j}}\left[min\left(\lambda_j A^{\pi_{\omega,j}}, clip\left(\lambda_j, 1-\epsilon, 1+\epsilon\right)A^{\pi_{\omega,j}}\right)\right]$$ <- Policy loss of actor networks

$$\lambda_i^t = \frac{\pi_{\omega,i}(a_i^t \mid s_i^t)}{\pi_{\omega,i}^{old}(a_i^t \mid s_i^t)} ; \lambda_j^t = \frac{\pi_{\omega,j}(a_j^t \mid s_j^t)}{\pi_{\omega,j}^{old}(a_j^t \mid s_j^t)}$$ <- Actors' importance sampling ratios

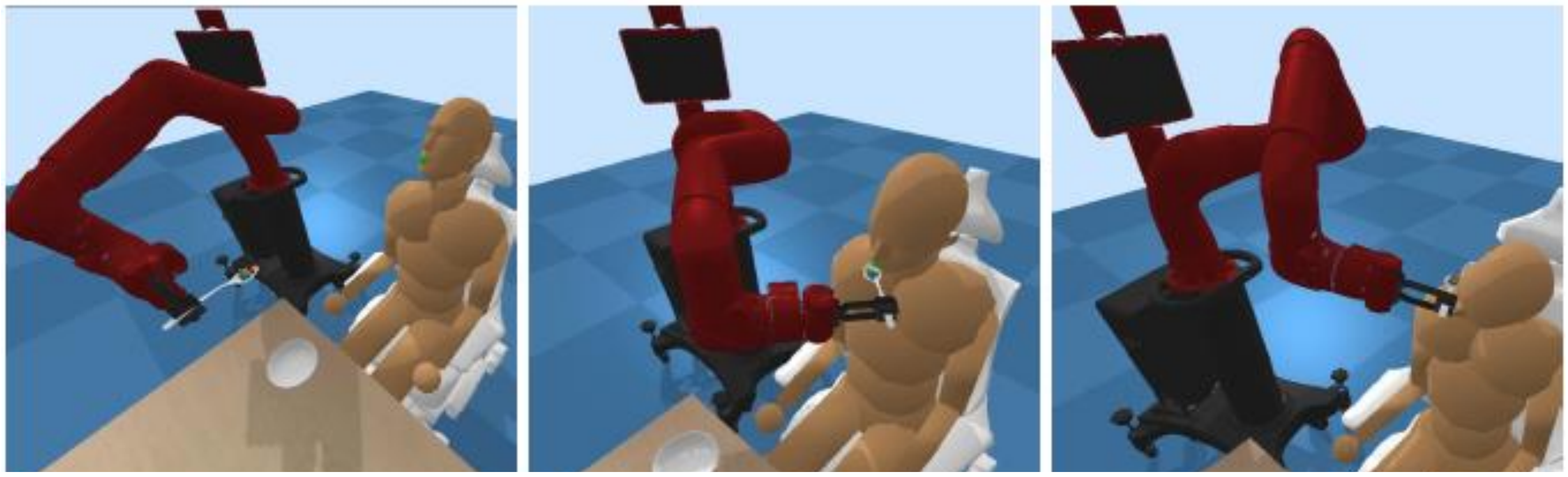
End-to-end pipeline



Video frames of Human-Human team expert demonstrations
SA-Net
SA-Net Deep Neural Network for Joint State-Action Trajectories
Expert joint (state-action) trajectories
Discriminator
(s-a) sample trajectories generated using current Dec-Policy (π_{Dec})
Joint reward function $R(<s_i,s_j>,<a_i,a_j>)$
Decentralized – Proximal Policy Optimization
Centralized Critic – Receives Global State and Action
Decentralized Actor 1 - Receives local state of actor 1
Decentralized Actor 2 – Receives local state of actor 2
Decoupled robot policy for sorting alongside a human.
H: Detect R: NoOp

Experiment 1 – Simulated patient assistance

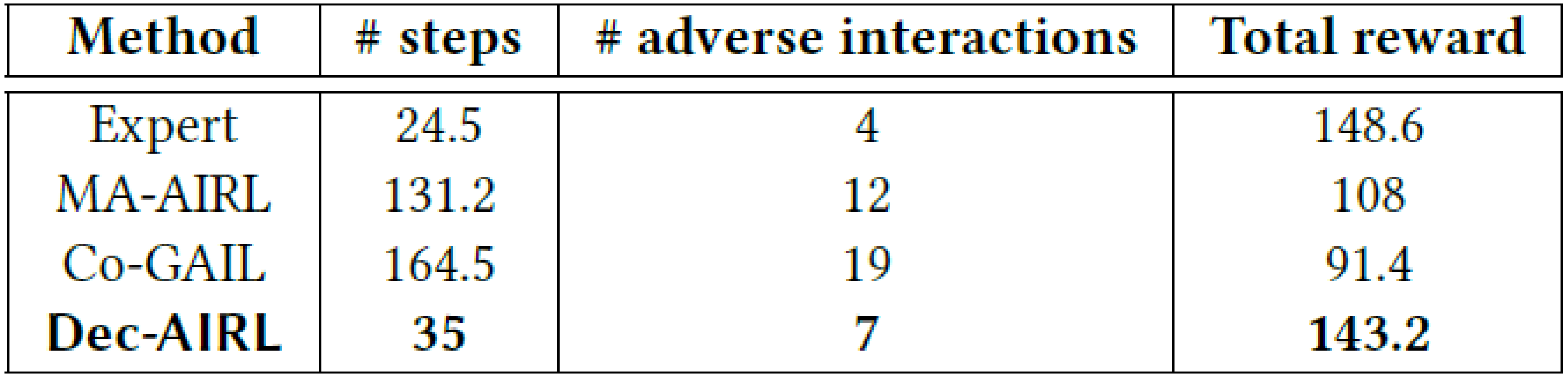
- Cobot needs to feed human successfully
- Avoid collisions while reaching moving human



(a) Cobot approaching human with food. (b) Cobot successfully feeding the human. (c) Cobot hitting the human – adverse interaction.

- Attributes like human joint angles are not perfectly observable by the cobot

Simulated patient assistance: Results



Method	# steps	# adverse interactions	Total reward
Expert	24.5	4	148.6
MA-AIRL	131.2	12	108
Co-GAIL	164.5	19	91.4
Dec-AIRL	**35**	**7**	**143.2**

- Baseline policies encounter more adverse interactions, drop some food and gain less reward
- Dec-AIRL takes a longer path than expert, but successfully feeds the human

Simulated patient assistance: Results

- Baselines Co-GAIL and MA-AIRL condition cobot policy on global state

Co-GAIL policy

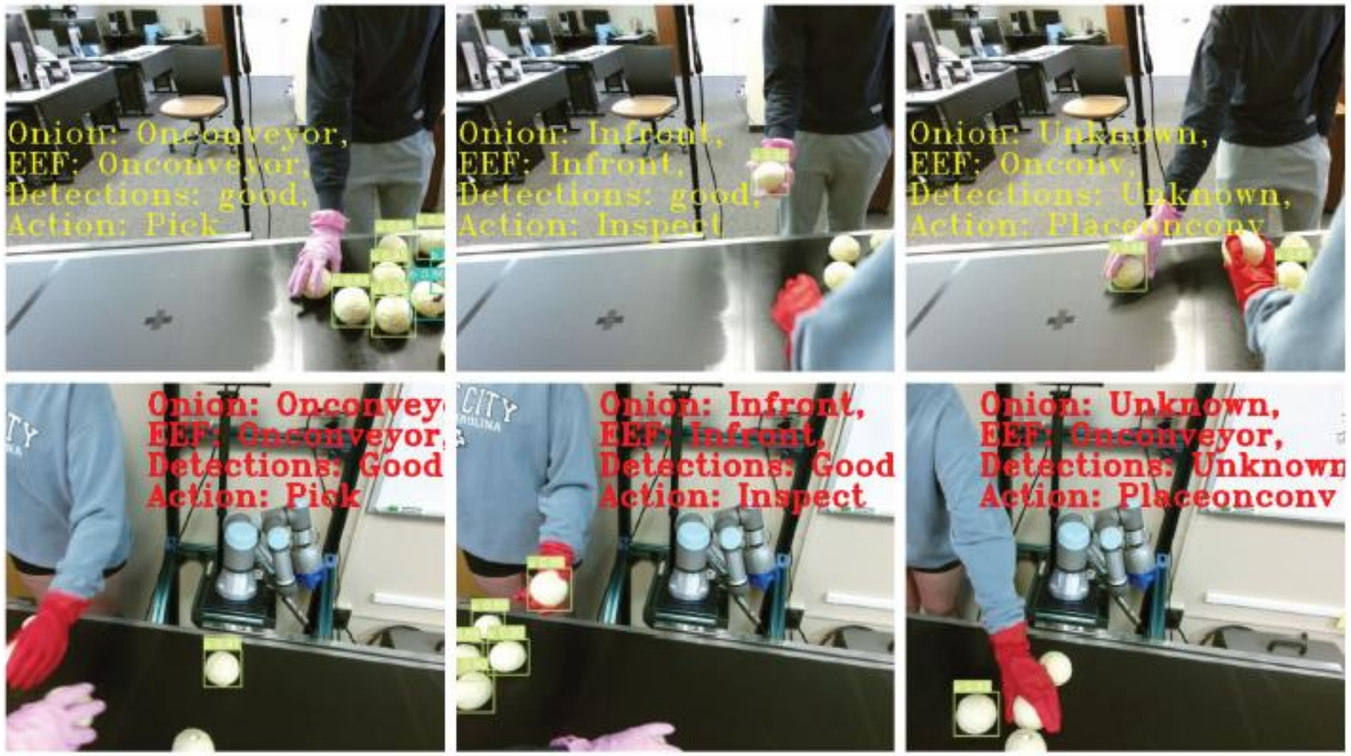
MA-AIRL policy

Dec-AIRL policy

- Dec-AIRL improves upon baselines by learning a decentralized policy

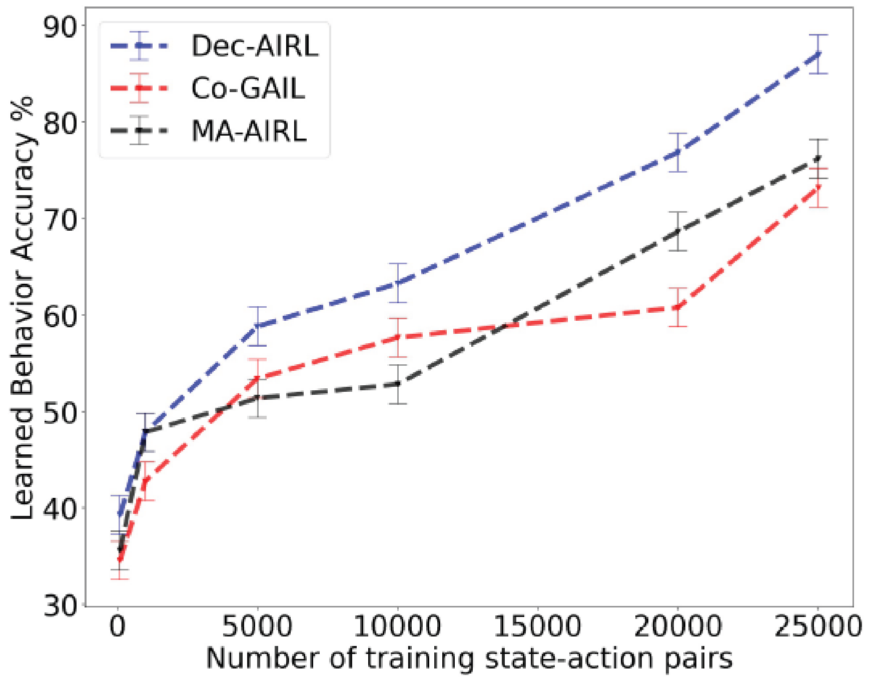
Experiment 2 – Human-cobot line sorting

- Collaboratively sort onions on a conveyor line
- Expert demos are by a human-human team
- Time synced frames sent to SA-Net to get joint trajectories



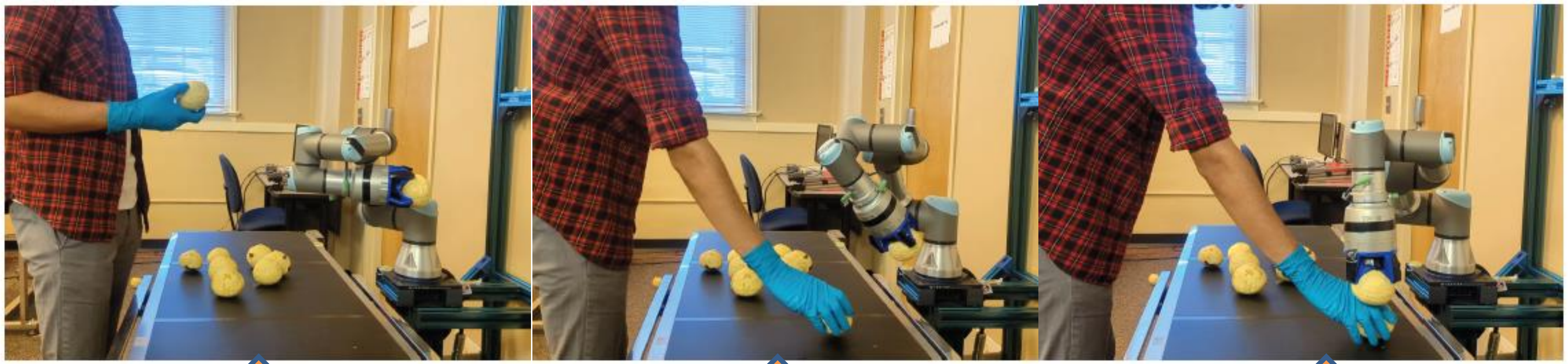
Human-cobot line sorting: Results

Method	(TP,FP,TN,FN)	Precision	Recall
Human-human	(4,1,5,0)	0.8	1.0
Human-cobot (MA-AIRL)	(3,2,3,2)	0.6	0.6
Human-cobot (Dec-AIRL)	**(3,2,4,1)**	**0.6**	**0.75**



Human-cobot line sorting – Baseline policy

- MA-AIRL policy encounters adverse interaction since human state estimate is noisy during execution



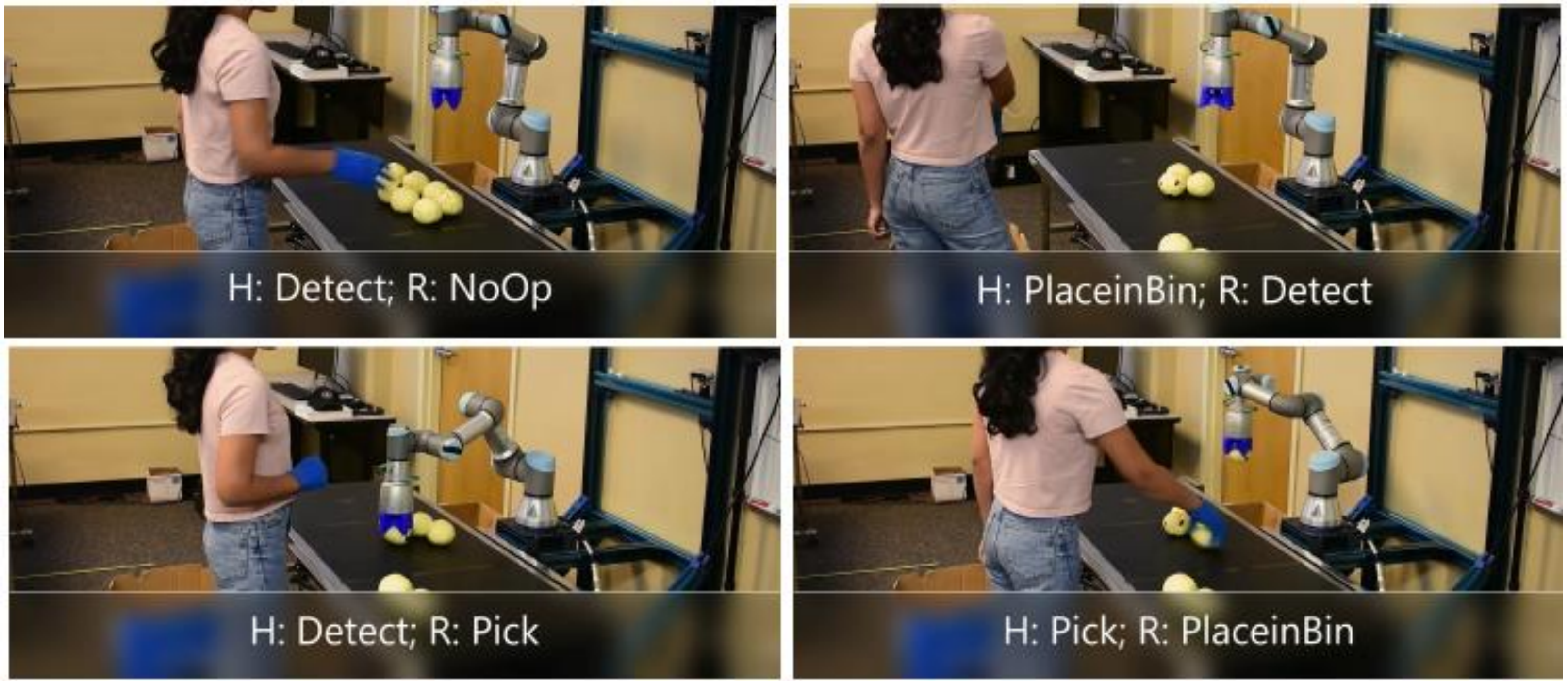
Cobot noisily perceiving human's onion prediction as bad

Cobot assuming human will discard their onion in the bad bin

Cobot simultaneously placing on conveyor causing an adverse interaction

Human-cobot line sorting – Our policy

- Dec-AIRL policy fares much better because the learned cobot policy is decentralized.



Conclusion

- Presented a novel IRL method using a decentralized MDP as the underlying model
- Showed generalizability across two domains, 1. Continuous, simulated, and 2. Discrete and realistic
- Presented Dec-PPO as part of Dec-AIRL that uses CTDE learning
- Learned to collaborate while avoiding adverse interactions

Future work

- In cases where local full-observability is infeasible, problem can be modelled as a Dec-POMDP
- Onion sorting domain can be generalized to continuous state-action setup to further test deployability
- Sample complexity analysis can be done to examine learning efficiency and convergence
- Sub-optimality and/or bounded-rationality of humans can be factored into learning
- After IRL convergence, online RL or human-in-the-loop techniques can be used to refine cobot’s policy if need be

Bibliography (partial)

- Saurabh Arora and Prashant Doshi. 2021. A survey of inverse reinforcement learning: Challenges, methods and progress. Artificial Intelligence 297 (2021).
- Daniel S. Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. 2002. The Complexity of Decentralized Control of Markov Decision Processes.
- Craig Boutilier. 1996. Planning, learning and coordination in multiagent decision processes.
- Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym.
- Zackory Erickson, Vamsee Gangaram, Ariel Kapusta, C. Karen Liu, and Charles C. Kemp. 2020. Assistive Gym: A Physics Simulation Framework for Assistive Robotics. IEEE International Conference on Robotics and Automation (ICRA) 1
- Chelsea Finn, Sergey Levine, and Pieter Abbeel. 2016. Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization. arXiv preprint arXiv:1603.00448
- Justin Fu, Katie Luo, and Sergey Levine. 2018. Learning Robust Rewards with Adverserial Inverse Reinforcement Learning. In International Conference on Learning Representations.