MCD411

B.Tech Project

Cooperative Multi-Agent Reinforcement Learning for UAVs

submitted on: 22nd November 2021

Presenters

Rachit Jain
2018ME10032

Sadanand Modak
2018ME10039

Supervisor

Prof. Arnob Ghosh
Department of Mechanical Engineering

Co-Supervisor

Prof. Shaurya Shriyam
Department of Mechanical Engineering

PHASE 1

PROBLEM DISCUSSION

INTRODUCTION | LITERATURE REVIEW | METHODOLOGY

speaker

Sadanand Modak

INTRODUCTION

- Tasks in unknown environments, possibly dangerous

- Wildfire monitoring, search and rescue missions, and target-tracking, searching, or attacking

- Complexity of tasks in real-world

- Cooperative multi-UAV systems far more efficient
- Exploration and mapping of unmapped environments

- Model of environment not known

- Planning by Dynamic Programming not applicable
- Reinforcement Learning for optimal control of UAVs
- Learning from interaction, trial-and-error learning, no explicit supervisor

- Single-agent RL exhaustively researched

- State and action spaces large: Function approximators (DNN)
- MADRL is the state-of-the-art

This work aims at exploring MADRL based multi-UAV systems.

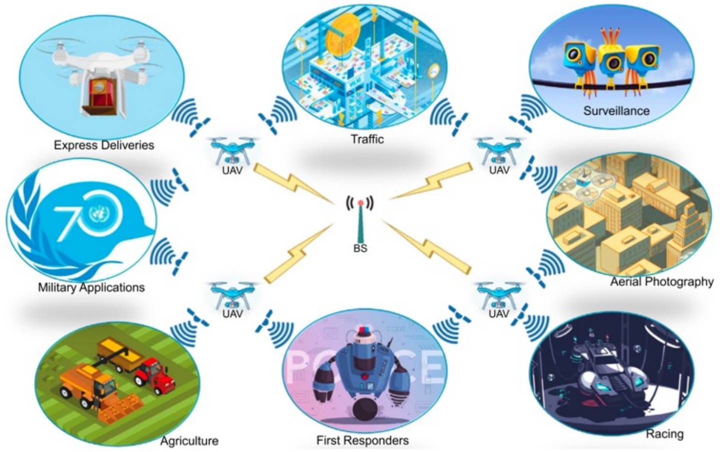


Fig: Applications of UAV Clusters [16]

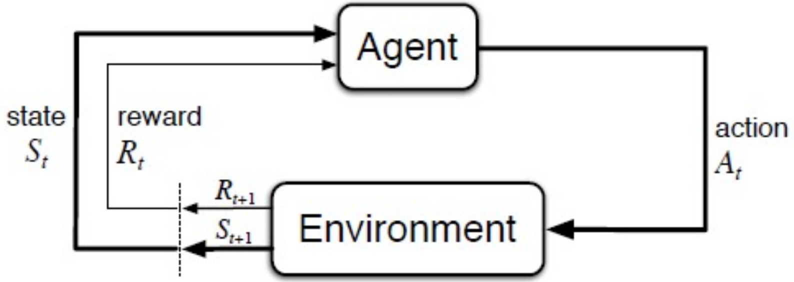


Fig: Basic RL Architecture [1]

NOTE: UAV stands for Unmanned Aerial Vehicles

LITERATURE REVIEW

Categories of Algorithms:

1. <u>*Policy-based Learning:*</u> the agent directly optimizes on the parameter vector ϴ of the DNN and learns the policy π. *Ex: Monte-Carlo Policy Gradient uses MC target*
2. <u>*Value-based Learning:*</u> learns the Q-function directly using the Bellman Optimality equation, then GPI for control. *Ex: DQN*
3. <u>*Actor-Critic Methods:*</u> learn both Policy and Q-function by maintaining two separate DNNs. *Ex: DDPG*

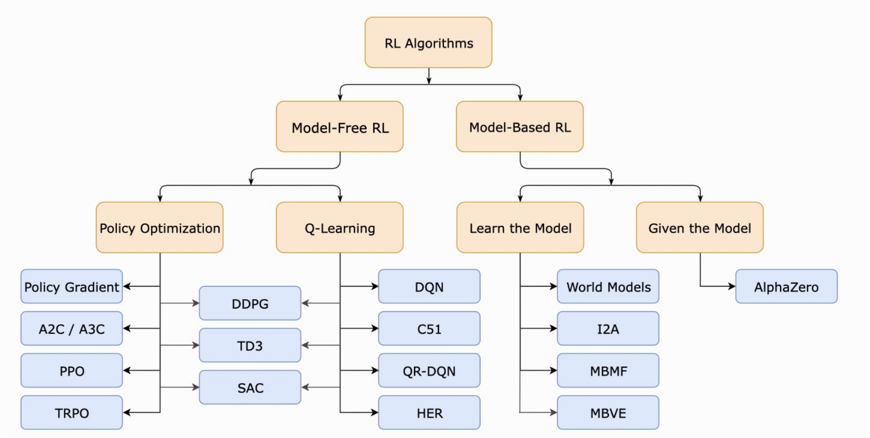


Fig: Taxonomy of DRL [13]

DDPG (Deep Deterministic Policy Gradient) [25]

- Actor-critic algorithm which has Q-learning (critic) and Policy Gradient algorithm (actor); off-policy algorithm
- Q-network: optimizes on mean-squared Bellman error

$$L(\phi, \mathcal{D}) = \underset{(s,a,r,s',d)\sim\mathcal{D}}{\mathrm{E}}\left[\left(Q_\phi(s,a) - \left(r + \gamma(1-d)\max_{a'} Q_\phi(s',a')\right)\right)^2\right]$$

- Policy-network: optimizes with respect to policy parameters ϴ to select greedy action (*argmax*(Q))
- Ensuring stability via Experience Replays and Fixed-Q targets

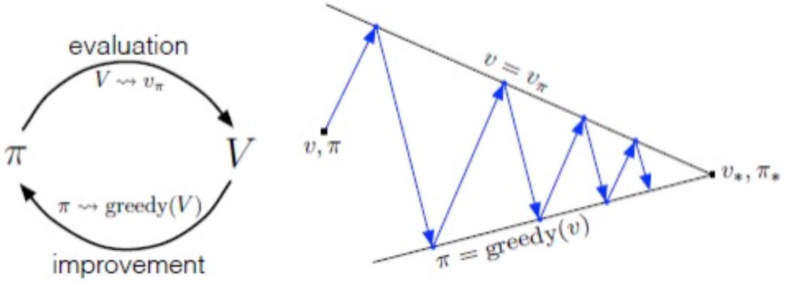


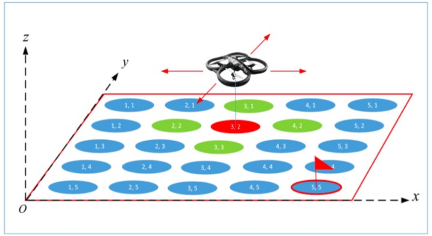
Fig: Generalized Policy Iteration (GPI) [1]

Rachit Jain & Sadanand Modak

LITERATURE REVIEW

Autonomous UAV Navigation Using Reinforcement Learning [15]

- PID controller for changing parameters of UAV flight for stability
- Q-Learning for learning the Q-value function of the state space
- Simplified 2D representation of state space; discretized space; constant altitude assumption
- Single-agent learning for a simplified UAV setup



Cooperative and Distributed Reinforcement Learning of Drones for Field Coverage [18]

- Used centralized-execution and training with Q-learning approach
- Considered the joint state space and joint action space as a whole in the MDP; therefore, very large sized spaces
- Learn cooperatively to provide a full coverage of an unknown field of interest
- Did not use DL approach; game-theoretic view was adopted

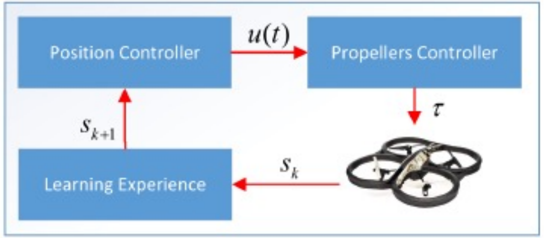


Fig: RL setup for UAV in discretized 2D space [15]

IA2C: Independent Advantage Actor-Critic [35]

- First-of-its kind approach for independent decentralized training and decentralized execution in MARL
- It uses global reward sharing between agents, ie, all agents get access to the global reward (sum of all agents' rewards)
- Each agent only gets to see the partial global state, ie, only the states of its neighbouring agents (consensus-based) where the neighbouring agents are defined based on the communication graph

PROBLEM STATEMENT & METHODOLOGY

The main objectives are:

- Study the fundamentals of Reinforcement Learning and the state-of-the-art research in MADRL
- Execution and Simulation of Centralised algorithms (DDPG, MADDPG)
- Exploring Decentralized Training with Decentralized Execution for Networked Agents with Consensus Update using MADRL algorithm (IA2C)
- Execution and Simulation of IA2C in two different environment cooperative environments
- Investigating the effects of noise (a realistic phenomenon) in communication channels on global average episodic rewards that indicate the convergence characteristics of the algorithm
- Extending communication to all agents in the system
- Implemented a ‘delayed IA2C’ algorithmic setup

The aim of this work, therefore, is to understand the multi-agent reinforcement learning (MARL) problem in the cooperative scenarios and then do a generalized approach study to do algorithmic novelties and numerous experimental simulations to validate the results which could later be applied to a variety of use-cases, along with trying to mimic practical scenarios.

GENERAL MDP FORMULATION

State Space (S)	Agent Locations
	Target Locations
Observations (O)	Location of agents
	Target location as seen by the agent
Actions (A)	Moving to a connected node
Rewards (R)	Negative Reward for each time-step passed
	Negative Reward for collision with other agents
	Positive Reward if the target is achieved
Probability (P)	The probability that the agent transitions from one location to another depending upon states and actions of all agents

Fig: General MDP Formulation for Multi-Agent RL Problem for UAVs

Application: An intersection negotiation task with traffic from all directions

1. S → Discrete space of joint locations of all cars (fully observed, hence S = O)
2. A → Continuous action space for travel direction and discrete displacement along the chosen direction
3. R → Loss for collision; Gain for distance travelled; Gain for reaching correct lane

Application: Target Tracking problem for UAVs

1. S → Discrete space of joint locations of all UAVs and Targets in 2D
2. O → Locations of other UAVs and Targets within field of view
3. A → Discrete action space (left, right, forward, backward, stay)
4. R → Loss for collision; Gain for target within FoV

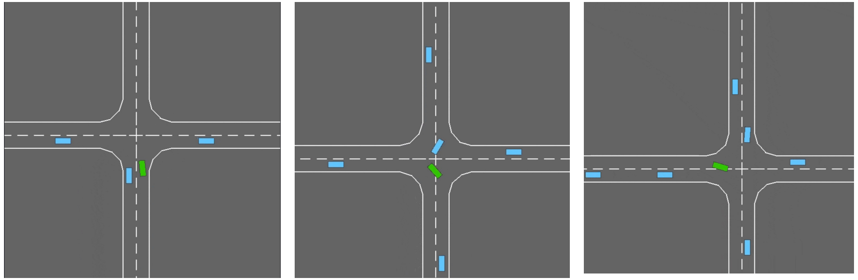


Fig: Intersection negotiation task representation

CACC MDP FORMULATION

State Space (S)	State given by a 5-tuple: (a) v_state: fractional v_star velocity of that agent, where v_star is the target velocity (defined as 15 m/s in configuration of this environment) (b) vdiff_state: difference of velocities of current and its leading agent (c) vhdiff_state: difference of velocities of current and the max speed allowed (defined as 30 m/s in configuration of this environment) (d) h_state: fractional headway wrt h_star of that agent, where h_star is the target headway (defined as 20 m in configuration of this environment) (e) u_state: fractional acceleration wrt to maximum allowed acceleration (defined as 2.5 m/s^2 in configuration of this environment)
Observations (O)	State (5-tuple) of the agent itself and the states of neighbouring agents, i.e., an array of 5-tuples
Actions (A)	2-tuple (α, β) that represents the human driver behaviour 4 discrete actions possible 1. α is headway gain and can take either value 0 or 0.5 2. β is relative velocity gain and can take either value 0 and 0.5
Rewards (R)	Large negative reward (-1000) for collision
	Scaled down by factor of 0.1, negative reward proportional to square of the acceleration
	Negative reward proportional to square of the difference between current and desired velocity
	Scaled up by a factor of 5, negative reward proportional to square of the difference between the current headway and the normal headway if current headway is less than normal headway, else 0
State Transition Probabilities (P)	Given s and action a, the next state s' is deterministic and is found based on some formulaes for the OVM (Optimal Velocity Model) controller

Fig: MDP of Cooperative Adaptive Cruise Control Environment (CACC)

GANTT CHART

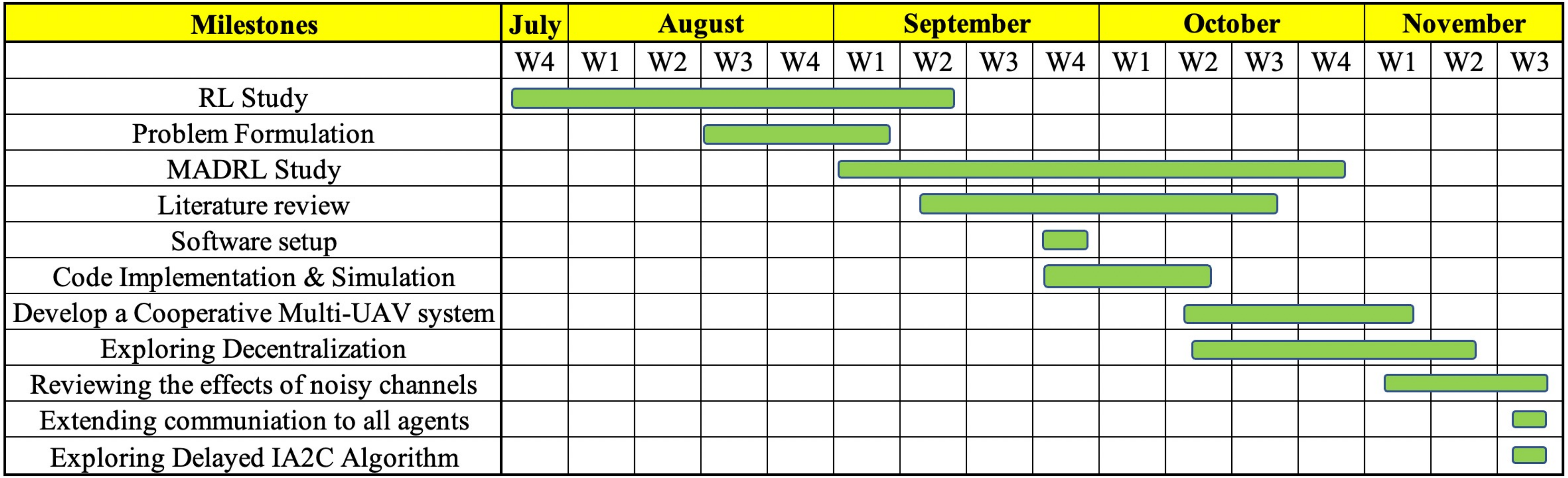


Fig: Gantt Chart describing objective timeline

PHASE 2

WORK PROGRESS

THEORY | EXPERIMENTAL SETUP | RESULTS

speaker

Rachit Jain

MOTIVATION

Centralized vs Decentralized

- Communication with Agents
- Practical Installation Cost
- Computational Resources

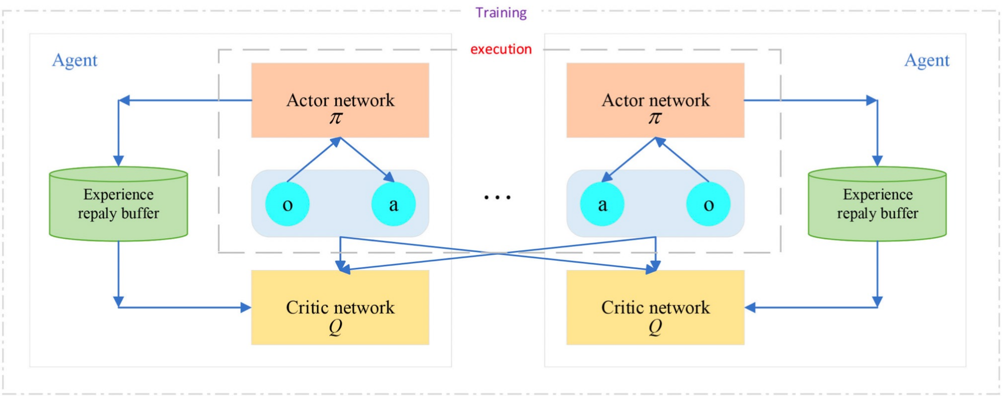


Fig: Centralized Training with Decentralized Execution [19]

Privacy over Communication

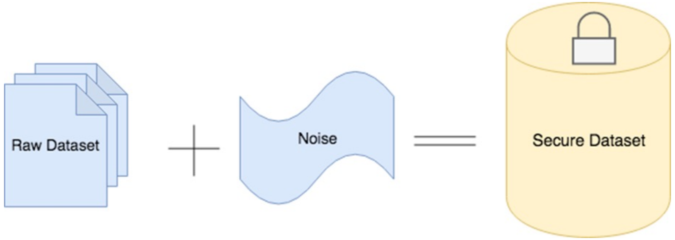


Fig: Differential privacy via noise addition [33]

- Noisy Communication
- Deceive other agents
- False signals but changed learnability

THEORY & ALGORITHMS

Multi-Agent Deep Deterministic Policy Gradient (MADDPG)

Decentralized Agents with Centralized Critic

Applicable to Mixed Scenarios

No specific structure on communication b/w agents

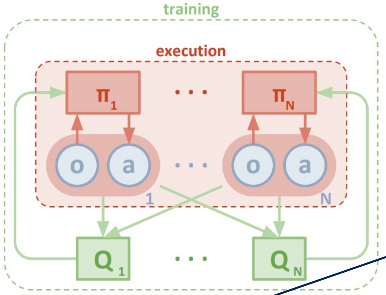


Fig: Representation of MADDPG [2]

Multi-Agent Soft Actor-Critic (MASAC)

Flexibility of DL + Decision Making of RL algos

Faster on learning vs traditional

Based on Maximum Entropy

Faster Convergence

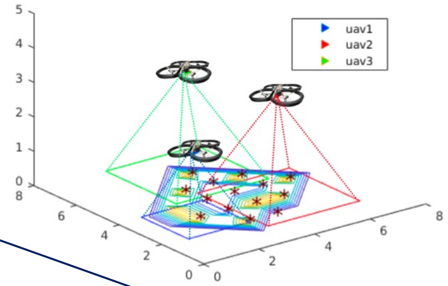


Fig: 3D rep of UAV Team covering field [18]

Independent Advantage Actor-Critic (IA2C)

Decentralized training and Decentralized (consensus-based) execution

Global reward sharing among all agents

Only neighbouring states shared

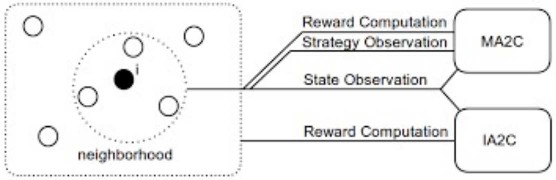


Fig: Comparison of the two MARL approaches MA2C and IA2C [33]

Rachit Jain & Sadanand Modak

EXPERIMENTS

Centralized Training & Decentralized Execution

Cooperative Navigation

- N agents reaching L landmarks cooperatively
- Relative dynamic position of others
- Rewards based on proximity to landmark
- Heavy collision penalty

Predator-Prey Environment

- N slower agents chase the faster adversary
- Random update of locations that can't be breached by the agents while moving
- Heavy reward on successfully catching prey

Physical Deception

- N agents cooperate to deceive the adversary from going to location
- Reward based on successful deception
- Penalty on cooperating agents being together

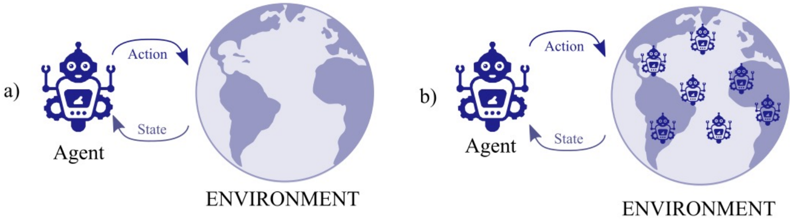
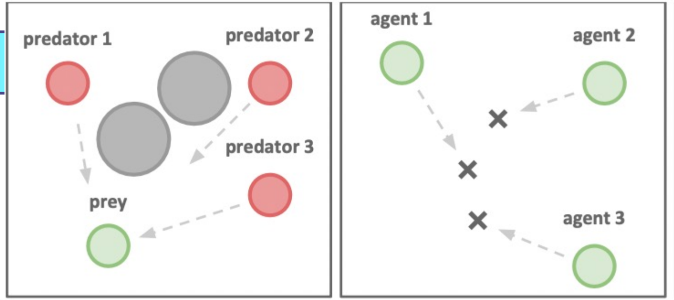


Fig: Multi Agent Representation [19]



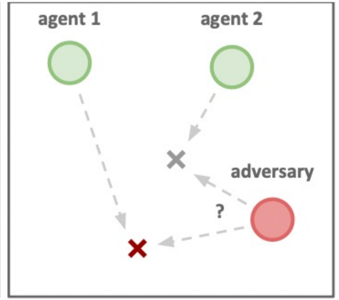


Fig: The scenarios for Predator-Prey, Cooperative Navigation and Physical Deception [19]

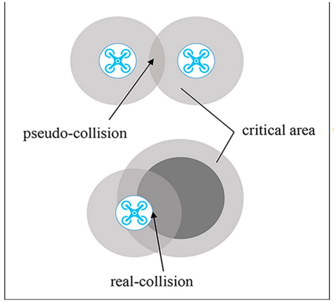


Fig: Critical area, pseudo-collision and real-collision [19]

EXPERIMENTS

CACC Slowdown Scenario

- N Agents moving at fast speeds and need to 'slowdown' to prevent collision but keeping good speeds and distance

Decentralized Training & Decentralized Execution

CACC Catchup Scenario

- N Agents moving at slow speeds and need to 'catchup' to follow others with optimal speeds and distance

Common Objectives

- Vehicle Following but at varied distance
- Reward on higher the velocity & less distance
- Huge collision penalty

- Horizontal & Vertical moves in the grid lanes
- Beware collision on the traffic signal network intersection.

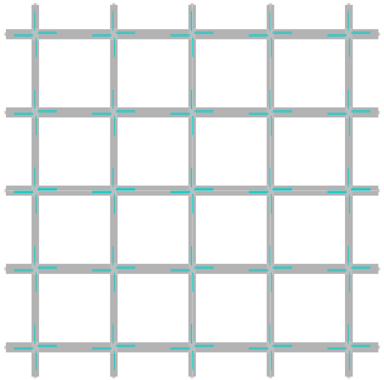


Fig: Environment for Traffic Signal Network [28]

Further Experiments

'Delayed' IA2C

Communication occurs at a small delay to mimic practical scenario more closely

Extending communication

Communication is extended amongst all agents to see centralized training in independent setup

EXPERIMENTAL SETUP

Centralized Training & Decentralized Execution

Objective: Cooperatively reach specified locations, deceive adversary or catch enemy

N UAVs with adversary (if needed)

100 steps in each episode for each UAV

20,000 Number of Episodes (general)

Decentralized Training & Decentralized Execution

Objective: Cooperatively maintain high speeds and decent distance without collision

4 UAVs moving horizontally & vertically

500,000 Number of Steps

60 seconds for each training episode

Average Episodic Rewards (AER) were recorded after certain set of episodes for visualisation

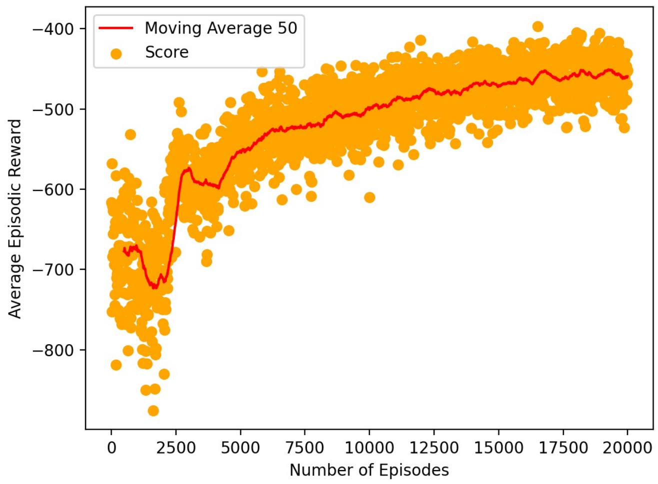
RESULTS & DISCUSSION

Cooperative Navigation

- 3 Cooperative UAVs (purple)
- 3 Target Locations (black)
- 20000 episodes
- More 2 hours to simulate
- MADDPG algorithm for each of the cooperative agents
- Rendered simulation to gain more accurate understanding of how agents are interacting with the environment

Insights

- Gradual process of increase in net reward as the number of episodes pass by.
- The scores still increasing with episodes
- The agents go closer to their respective landmarks as the number of training episodes passes increase.



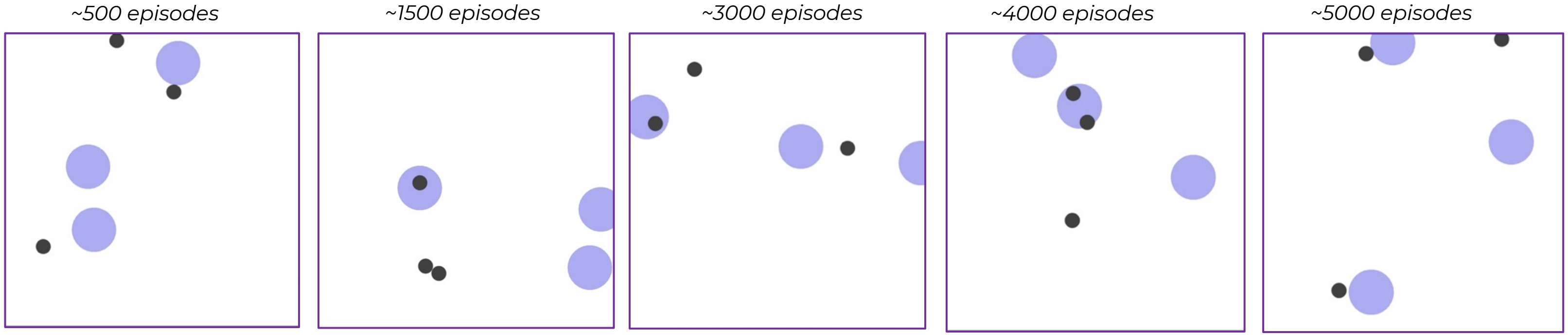


Fig: Simulation for Cooperative Navigation environment for distinct episodes

RESULTS & DISCUSSION

Predator-Prey Movement

- 3 collaborative UAVs (pink)
- 1 adversary UAV (green)
- 2 non-breachable landmarks (black)
- Mixed environment
- Cooperative agents learning with MADDPG
- Adversary trained on DDPG
- Run for 10,000 episodes
- Less collisions and closer the target,

- Less collisions and closer the target, more the reward

Insights

- Quicker learnability
- The agents go closer to the prey without touching the landmarks with episodes
- Some episodes with fairly high rewards at the end

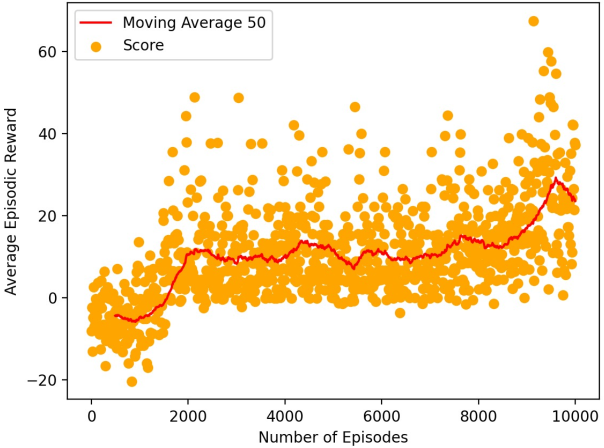


Fig: Training on Predator Prey

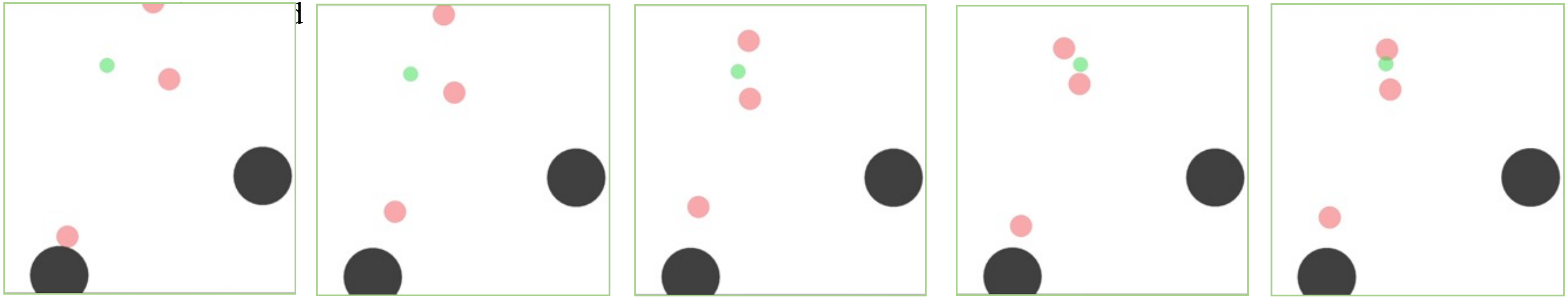


Fig: Training on Predator-Prey environment for 10000 episodes for a particular episode depicting how the cooperative agents learn to attack the prey without touching landmarks

RESULTS & DISCUSSION

Physical Deception

- 2 cooperative UAVs
- 1 adversary UAV
- Mixed environment
- MADDPG algorithm for all
- 10,000 episodes run; extremely heavy on computation
- Agents penalised for colliding with each other while rewarded based on the proximity to the landmarks.
- Quick Learnability due to the application of MADDPG

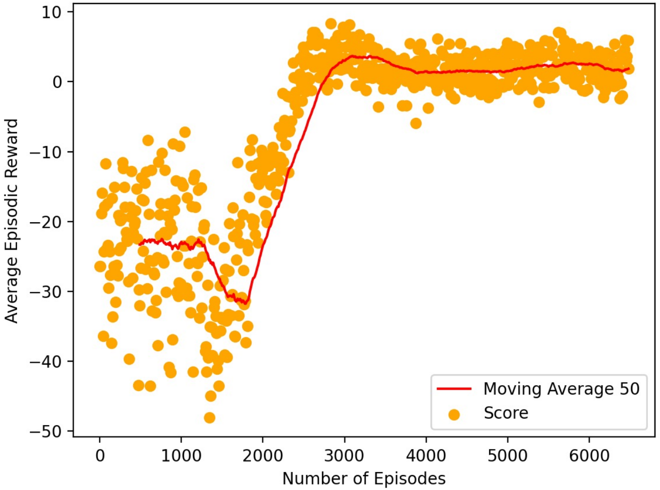
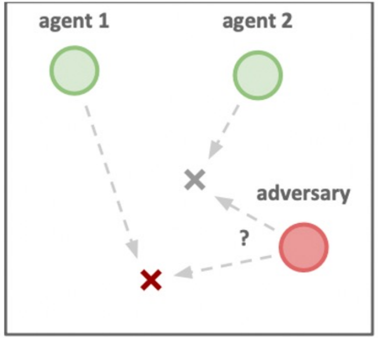


Fig: Training on Physical Deception environment for 6500 episodes



Insights

- Quick learnability
- Initially the rewards have quite high variance since agents still learning whether to go to target or to deceive!
- They learn to constantly get better episodic rewards

RESULTS & DISCUSSION

CACC Slow-down Scenario

Without Noise

- Increasing trajectory while oscillating AER
- Slower learning

With Light Noise

- Rate of increase lesser
- Better AER initially

With Heavy Noise

- Quick Learnability though oscillating behaviour

Possibly, the noise makes the agents realise that their neighbours are closer than they actually are and hence the addition of noise leads to better learnability in terms of better AER as episodes go by

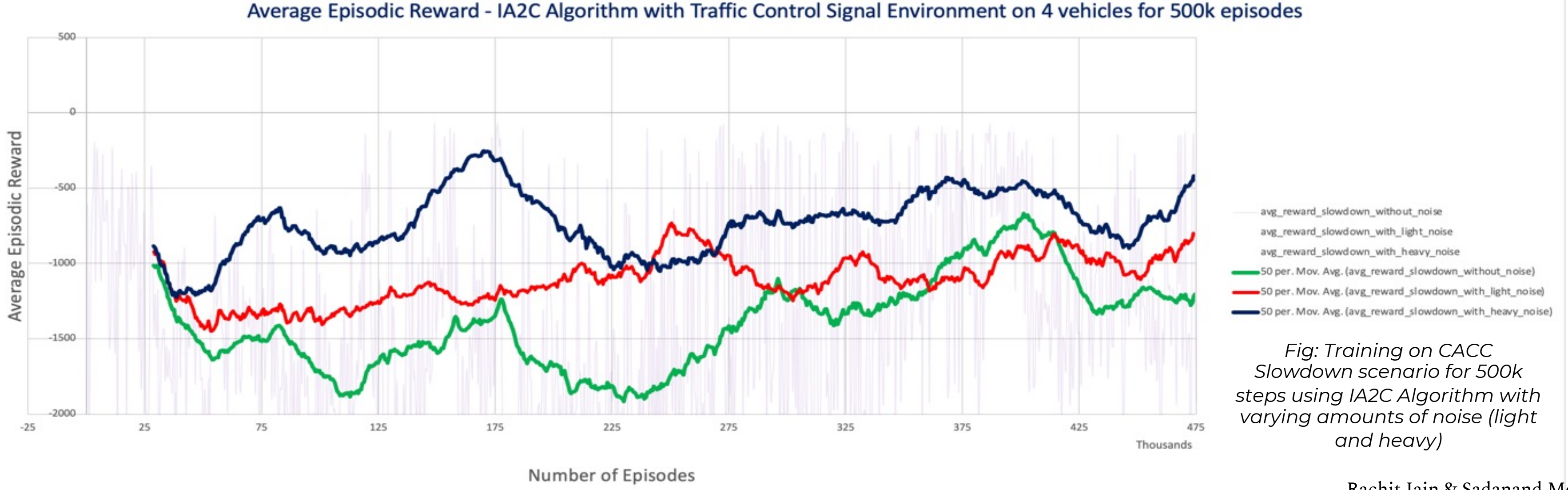


Fig: Training on CACC Slowdown scenario for 500k steps using IA2C Algorithm with varying amounts of noise (light and heavy)

RESULTS & DISCUSSION

CACC Catch-up Scenario

Without Noise

- Relatively steady AER variations
- Higher values than earlier

With Light Noise

- Higher oscillations
- Improved AER over episodes

With Heavy Noise

- Starts to show better learnability at the end of simulation

Relatively steady AER and addition of noise makes less significant efforts towards improving learnability; slow speeds initially prevent significant initial collisions and thus relatively higher overall rewards

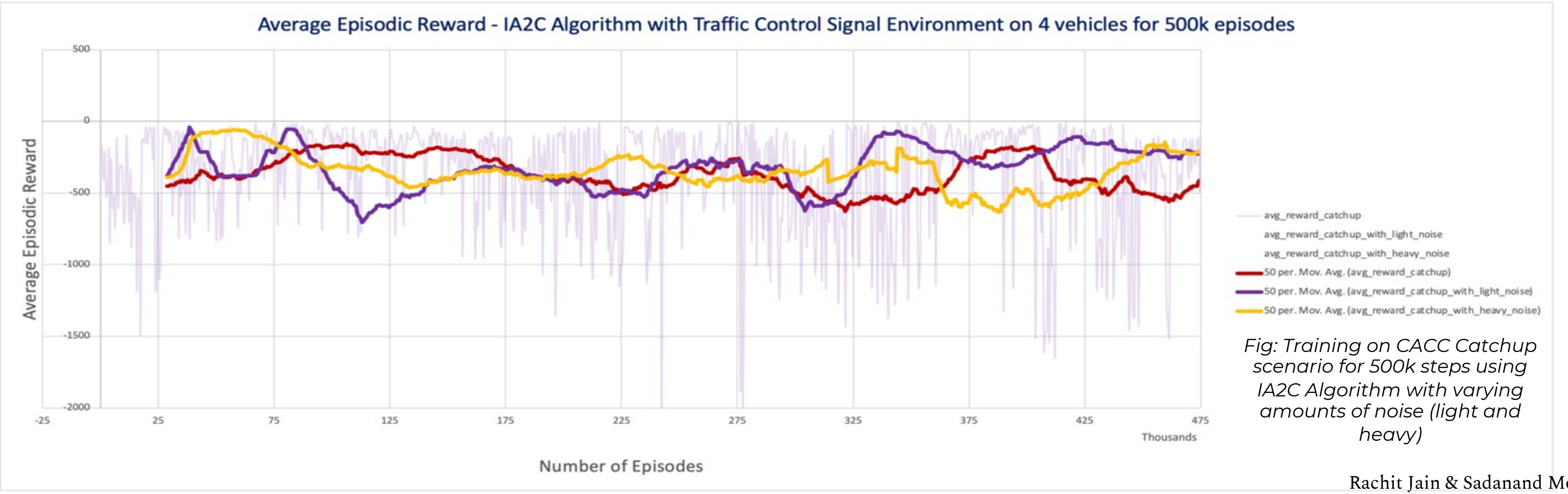


Fig: Training on CACC Catchup scenario for 500k steps using IA2C Algorithm with varying amounts of noise (light and heavy)

RESULTS & DISCUSSION

Extending communication to all agents

- Increasing trends for both curves
- Adding communication from all agents improves learnability as expected

Communication from all agents improves learnability

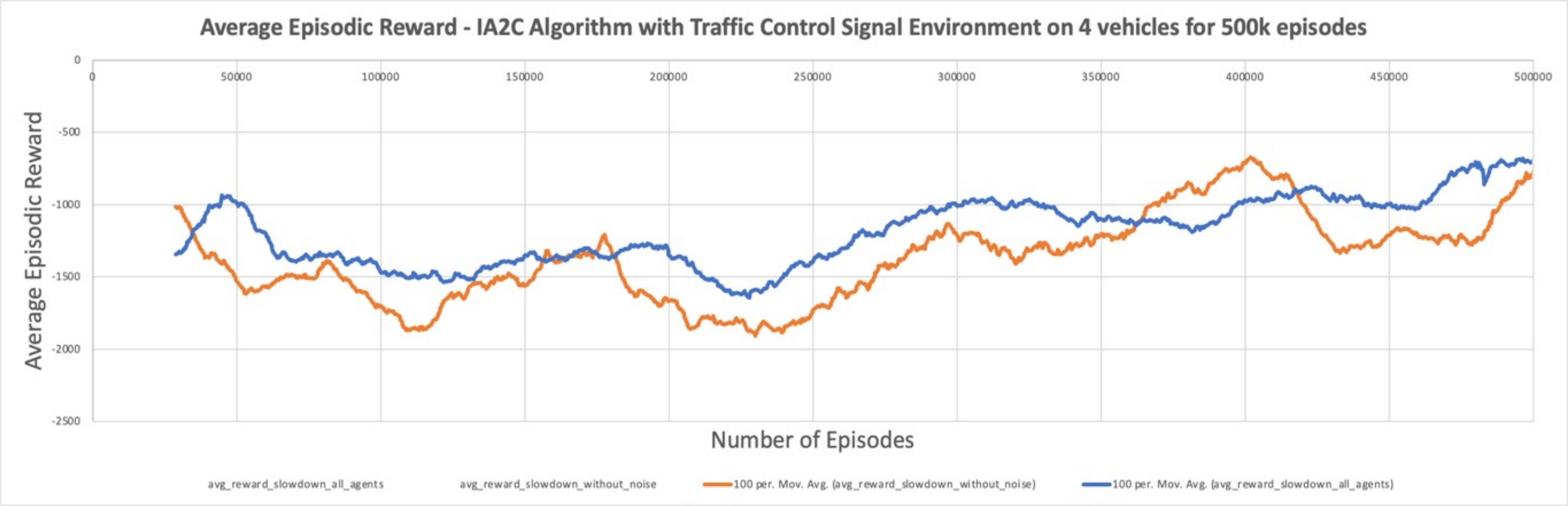


Fig: Visualisation for Training on CACC Slowdown for 500k episodes using IA2C Algorithm with 4 vehicles upon extending the idea of communication from all neighbours

RESULTS & DISCUSSION

'Delayed' IA2C Algorithmic Setup

- Algorithmic addition to mimic delay in communication amongst agents
- Improved results due to less number of vehicles interacting in the environment

A delayed communication would lead to poor results in case of more number of agents in the same environment

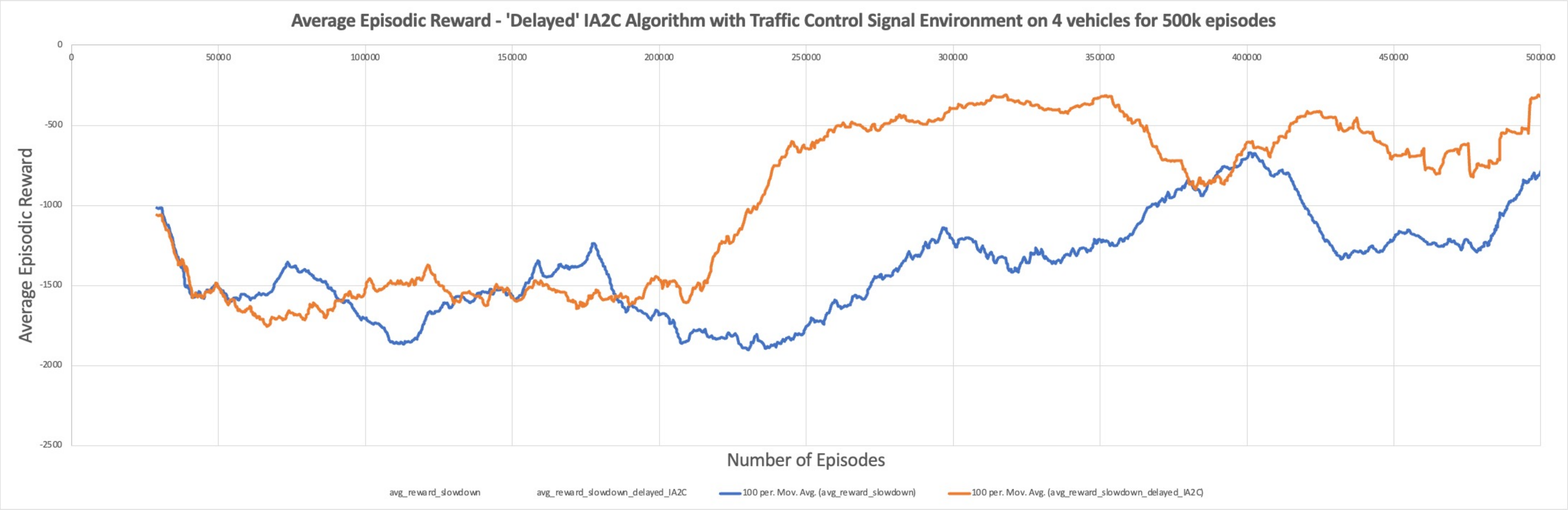


Fig: Visualisation for Training on CACC Slowdown for 500k steps using 'Delayed' IA2C approach with 4 vehicles

CONCLUSION & FUTURE WORK

- Multiple experimental scenarios were **simulated for the application of centralised algorithms** and the training curve was obtained as expected, with the scores increasing.
- Significantly large training times even with DNNs as function approximators shows that such real-world problems with multiple agents are not at all well-suited to be carried out with **conventional RL algorithms** with lookup tables.
- The effects of **realistic noise additions** at three different levels (no, little, heavy) to the observations of each of the agents were investigated not only made the environment mimic the practical scenario a bit more closely, but this little addition helped the learnability of the algorithm in some scenarios.
- As an extension to it, "**delayed-IA2C**" was also implemented where a delay in the communication channels was modelled to mimic practical communication between agents. Furthermore, the idea of communication was extended to all agents to claim the improvement in results observed.
- Effect of constraints related to autonomous vehicles in the problem can further add onto this research.

These directions would be explored with work to be done towards further developing and investigating the field of Cooperative Multi-Agent Reinforcement Learning for UAVs.

REFERENCES

[1] Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. MIT press, 2018.
[2] Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments
[3] Contrasting Centralized and Decentralized Critics in Multi-Agent Reinforcement Learning
[4] Achiam, Joshua, et al. "Constrained policy optimization." International Conference on Machine Learning.
[5] Z. Wang, Y. Zhang, C. Yin and Z. Huang, "Multi-agent Deep Reinforcement Learning based on Maximum Entropy," 2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), 2021, pp. 1402-1406, doi: 10.110
[6] Lyapunov-Based Reinforcement Learning for Decentralized Multi-Agent Control; Qingrui Zhang, Hao Dong, and Wei Pan; Sept 2020
[7] Reducing Overestimation Bias in Multi-Agent Domains Using Double Centralized Critics; Johannes Ackermann, Volker Gabler, Takayuki Osa, Masashi Sugiyama; Dec 2019
[8] Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, Sergey Levine; Jan 2018
[9] Distributed Distributional Deterministic Policy Gradients; Gabriel Barth-Maron, Matthew W. Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva TB, Alistair Muldal, Nicolas Heess, Timothy Lillicrap; April
[10] QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning; Tabish Rashid, Mikayel Samvelyan, Christian S.Witt, Gregory Farquhar, Jakob Foerster, Shimon Whiteson; ICML 18
[11] A library of multi-agent reinforcement learning components and systems
[12] Multi-agent reinforcement learning: An overview; L. Bus ̧oniu, R. Babusˇka, and B. De Schutter; 2010
[13] Multi-Agent Particle Environments
[14] Multi-Agent Reinforcement Learning: A Review of Challenges and Applications; Lorenzo Canese †, Gian Carlo Cardarilli, Luca Di Nunzio †, Rocco Fazzolari, Daniele Giardino †, Marco Re † and Sergio Spanò; 2021
[15] Autonomous UAV Navigation Using Reinforcement Learning; Huy X. Pham, Hung M. La, David Feil-Seifer, Luan V. Nguyen; Jan 2018
[16] Application of reinforcement learning in UAV cluster task scheduling; Jun Yang a, Xinghui You a,∗, Gaoxiang Wu a,∗, Mohammad Mehedi Hassan b, Ahmad Almogren b, Joze Guna c; Jan 2019
[17] Reinforcement Learning for UAV Attitude Control; WILLIAM KOCH, RENATO MANCUSO, RICHARD WEST, and AZER BESTAVROS; 2019
[18] Cooperative and Distributed Reinforcement Learning of Drones for Field Coverage; Huy Xuan Pham, Hung Manh La, David Feil-Seifer, and Ara Nefian; Sept 2018
[19] Joint Optimization of Multi-UAV Target Assignment and Path Planning Based on Multi-Agent Reinforcement Learning; HAN QIE, DIANXI SHI, TIANLONG SHEN, XINHAI XU, YUAN LI AND LIUJING WANG
[20] Papers with Code
[21] TensorFlow 2 Implementation of Multi-Agent Reinforcement Learning Approaches
[22] Watkins, C.J.C.H., Dayan, P. Q-learning. Mach Learn 8, 279–292 (1992) ; [23] Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." arXiv preprint arXiv:1312.5602
[24] A2C / A3C (Asynchronous Advantage Actor-Critic): Mnih et al, 2016; [25] DDPG (Deep Deterministic Policy Gradient): Lillicrap et al, 2015
[26] Rashid, Tabish, et al. "Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning." International Conference on Machine Learning. PMLR, 2018.
[27] Consensus Algorithm Blockchain: https://toshitimes.com/why-consensus-algorithms-matter-for-developers/; [28] Networked Multi Agent Reinforcement Learning (NMARL) – GitHub Repository
[29] PolicyInferring: Lowe, Ryan, et al. "Multi-agent actor-critic for mixed cooperative-competitive environments." Advances in Neural Information Processing Systems, 2017.
[30] FingerPrint: Foerster, Jakob, et al. "Stabilising experience replay for deep multi-agent reinforcement learning." arXiv preprint arXiv:1702.08887, 2017.
[31] ConsensusUpdate: Zhang, Kaiqing, et al. "Fully decentralized multi-agent reinforcement learning with networked agents." arXiv preprint arXiv:1802.08757, 2018.
[32] Cooperative Adaptive Cruise Control: Definitions and Operating Concepts: https://journals.sagepub.com/doi/10.3141/2489-17
[33] https://medium.com/secure-and-private-ai-writing-challenge/differential-privacy-e5c7b933ef9e
[34] https://en.wikipedia.org/wiki/Laplace_distribution
[35] Multi-Agent Deep Reinforcement Learning for Large-scale Traffic Signal Control
[36] MULTI-AGENT REINFORCEMENT LEARNING FOR NETWORKED SYSTEM CONTROL

[33] https://medium.com/secure-and-private-ai-writing-challenge/differential-privacy-e5c7b933ef9e
[34] https://en.wikipedia.org/wiki/Laplace_distribution; [35] Multi-Agent Deep Reinforcement Learning for Large-scale Traffic Signal Control
[36] MULTI-AGENT REINFORCEMENT LEARNING FOR NETWORKED SYSTEM CONTROL