Asymmetric Communication Policies in Multi-Agent Reinforcement Learning for Tethered Robots

Members:

Chintan Shah, Kimia Ramezani, Kelvin Mock

Carleton University

Project Background

What and Why is this project?

Problem Statement

- Effective communication uses partial observability.
- Communication constraints are challenging.
- Real-world applications often face 1-way or unreliable transmissions.
- Learning an effective communication protocol is challenging.
- Existing solutions often rely on:
  - Centralized training
  - Simple sender-receiver games

Motivation and Research Questions

Project Directions

- Effective asymmetric communication
- Decentralized policies
- Learned communication protocols
- Structured representations
- Sample-efficient training

Research Questions

- How do agents learn the environment to achieve goals effectively?
- How is achieving the goal optimized via asymmetric communications?
- How can agents overcome the difficulty of 1-way communications?
- How can agents maintain safe actions (e.g., no collisions, within tether)?

Carleton University

Previous study on Multi-Agent Communication

Reference	Solution/Method	Weakness	Connection Type
Blumenkamp .et al (2020)	**Graph Neural Networks (GNNs)** with a differentiable communication channel for self-interested agents to learn manipulative strategies.	Cooperative agents are vulnerable to exploitation if not allowed to adapt.	Two-way (bidirectional)
Meng et al. (2024)	Peer-to-peer communication via **MLPs** for personalized message sending/receiving.	Limited scalability in large-scale systems.	Two-way (bidirectional)
Foerster et al. (2016)	Differentiable communication channels for **centralized training** with shared models or gradients.	Suboptimal in practice, violates independence assumptions, and scales poorly with agent count.	Two-way (bidirectional)

Project Goals

- **Hypothesis**: Good communication optimizes task completion in MARL.

- To implement and benchmark different communication strategies.
  - Baseline: MAPPO + LSTM + Contrastive Loss + Computing Gradients
  - An Advanced Model: DIAL
- To define a strategy for all types of agents: Leader and Follower(s).
- To develop an end-to-end simulation with at least 1 working algorithm.
- To evaluate (and compare) metrics from different benchmarks.
- To show how they succeed and how they fail.

Literature Review

How do we validate our idea/approach?

Main sources for algorithm

Reference	Solution/Method	Connection Type
Lowe et al. (2017), Zhang et al (2024)	Centralized Training with **Decentralized Execution** (CTDE) is promising for cooperation.	Two-way (bidirectional)
Lin et al. (2021), Zhang et al (2024)	**Encoding-decoding** observations as communication; agents learn to reconstruct their observations.	Two-way (bidirectional)
Lin et al. (2021), Lo et al (2024)	Communication Alignment Contrastive Learning (CACL): **Contrastive learning** to align sent/received messages.	Two-way (bidirectional)

Communication Characteristics

- Fully Cooperative
- Partial Observability
- One-way communication
- Protocol Based
- NO restricted communication during learning
- Discrete message

Carleton
University

MAPPO

$$L_a^{CLIPP} = E_t[\min(r_t^a(\theta)\, A_t^a, clip\,(r_t^a(\theta), 1-\epsilon, 1+\epsilon)A_t^a)]$$

$$r_t^a(\theta) = \frac{\pi_\theta^a(u_t^a|o_t^a)}{\pi_{\theta_{old}}^a(u_t^a|o_t^a)}$$

$$L_{entropy} = -E[\pi(a|s)log\pi(a|s)]$$

Agent action: a

Time step :t

Individual observation: o_t^a

Joint action: $u^t = (u_t^1, \dots, u_t^N)$

Global reward: r_t

State: s

Advantage estimated via a critic network = A(s,a)

- Global information sharing in partial observation
- Centralized Coordination
- Stable Policy Updates

Carleton University

Challenges with MAPPO

Encoder-Decoder

- Problems with Relying on Past Data
- Capturing temporal dependencies

Unsupervised learning

- Ambiguous supervision
- Homogeneous policies

LSTM

- Remembers long-term dependencies (important in partial observability).
- Effective in capturing dynamics of spatial-temporal environments
- Improving exploration

- Continuous communication message vector: $m_t = [v_1, \ldots, v_n]$
- Decoded message : m'
- Follower local observation : o_t^f

- Leader message : $h_t^{enc} = LSTM_{enc}(h_{t-1}^{enc}, m_t)$
- Follower input: $h_t^{dec} = LSTM_{dec}(h_{t-1}^{dec}, [o_t^f, m_t])$

Reconstruction Loss:

$$L_{recon} = MSE(m, m')$$

Contrastive learning

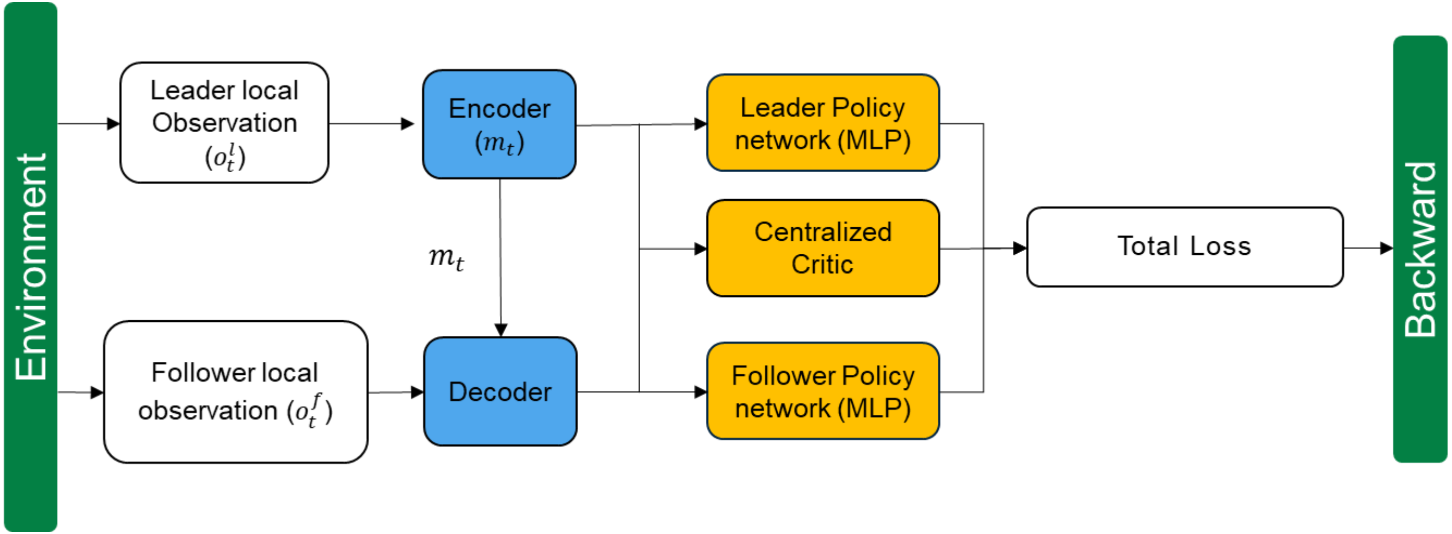
- Learn discriminative and task-relevant communication
- Learning Communication and enhance co-operation
- Reducing Message Redundancy
- Break policy symmetry and promote specialization or role-aware behavior.

  - cosine similarity function: Sim
  - Temperature: τ
  - Negative pairs: $\left(m_t, o'^{f}_{t'}\right)$

$$L_{contrast} = log \frac{\exp(\frac{sim\left(m_t, o_t^f\right)}{\tau})}{\Sigma_j \exp(\frac{sim\left(m_t, o'^{f}_{t'}\right)}{\tau})}$$

Algorithm

$$Total\ Loss = L_{pg} + \alpha L_{contrastive} + \beta L_{recon} + \gamma L_{entropy}$$



Experiment

How are we modeling the strategies of both agents?

How are they communicating?

Carleton University

Environment Setup

- Gymnasium – Atari
- Randomized Grid Map
- Form: RGB Array, String, or GUI
- Components:
  - Free cells
  - Soft/Hard Obstacles
  - Agent – Leader vs Follower
  - Targets
  - Tether
- Each State: communicate → act

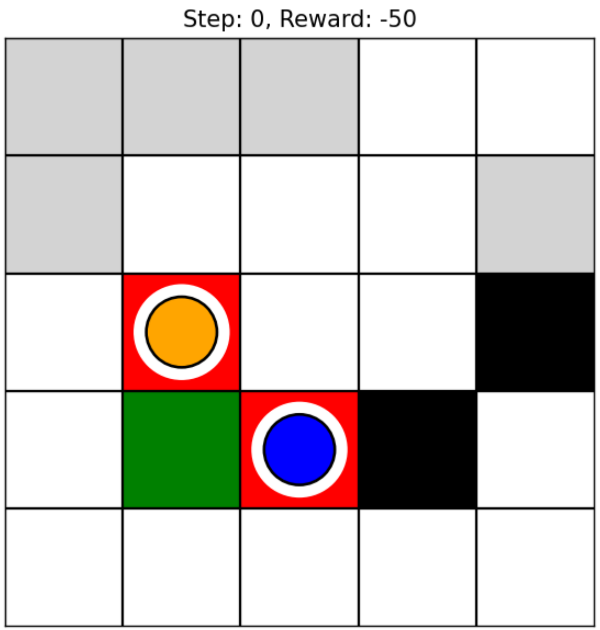


Figure 1: Visual Representation of the Grid

Environment Setup

Action Space

- UP
- DOWN
- LEFT
- RIGHT
- DIAGONALS (x4)
- STAY

(Velocity = 1 block)

Reward Structure

- Soft Obstacle = -10
- Hard Obstacle = -50 *(Reset)*
- Out-of-Bound = -50 *(Reset)*
- Out-of-Tether = -50 × count *(Reset)*
- Reaching Target = +50
- Each Step = -1
- STAY = -3

Algorithm Modeling

Hyper-parameter Tuning

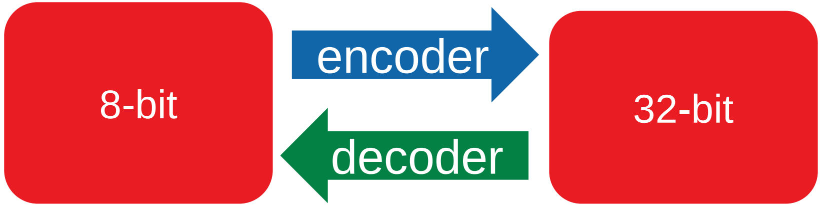
- Randomized Grid Search – with *n = 2* sets of random parameters
  - **Learning Rates**
  - **Contrastive Weights**
  - **Reconstruction Weights**
  - **Entropy Weights**
- Static Parameters
  - **Episodes** = 50
  - **Max Steps per Episode** = 100 or till Episode Reset (i.e., Target is Reached)
- Best Set: {'lr': 0.007425, 'contrastive_weight': 0.22, 'reconstruction_weight': 0.41, 'entropy_weight': 0.037}

Algorithm Modelling

Encoder

Layer Type	Output Shape	Param #
InputLayer	(None, 8)	0
Reshape	(None, 1, 8)	0
LSTM	(None, 1, 64)	18,688
LSTM	(None, 32)	12,416

- Total params: 31,104 (121.50 KB)
- Trainable params: 31,104 (121.50 KB)
- Non-trainable params: 0 (0.00 B)



Decoder

Layer Type	Output Shape	Param #
InputLayer	(None, 32)	0
RepeatVector	(None, 1, 32)	0
LSTM	(None, 1, 64)	24,832
LSTM	(None, 64)	33,024
Dense	(None, 8)	520

- Total params: 58,376 (228.03 KB)
- Trainable params: 58,376 (228.03 KB)
- Non-trainable params: 0

Carleton University

Algorithm Modelling

Leader’s Message

- Distance to the nearest obstacle: int or float
- Whether the path is clear or blocked: 0/1 int
- Leader can observe the follower or not: 0/1 int
- Leader's distance to follower: float
- Leader's suggested action (delta) in x direction: int
- Leader's suggested action (delta) in y direction: int
- Leader's current move (delta) in x direction: int
- Leader's current move (delta) in y direction : int

Algorithm Modelling

Policy Network – Leader vs Follower

- All Parameters are trainable
- Make an Action based on:
    - Partial Observation: 2 blocks
    - Leader's Message

Training with MAPPO – the baseline

1. Advantage: $R + \gamma V(s') - V(s))$
2. A2C Policy's Gradient Loss
3. Contrastive Loss – alignment
4. Entropy Bonus
5. Message Reconstruction Loss

Layer Type	Output Shape	Param #
InputLayer	(None, 8) OR (None, 2, 8)	0
Reshape	(None, 1, 8)	0
Dense	(None, 1, 64)	576
Dense	(None, 1, 64)	4,160
Dense	(None, 1, 9)	585
Reshape	(None, 9)	0

Algorithm Modelling

Critic Network

- Centralized Training
- Parameters Sharing scheme
- Returns a scalar: the Advantage Value
- Does not affect decentralized execution

× Slow Training Time

Layer	Output Shape	Param #
InputLayer	(None, 8)	0
Dense	(None, 64)	576
Dense	(None, 64)	4160
Dense	(None, 1)	65

Total params: 4,801 (18.75 KB)
Trainable params: 4,801 (18.75 KB)
Non-trainable params: 0 (0.00 B)

An Enhancement – DIAL (Differentiable Inter-Agent Learning)

Communication Efficiency

- Reduced number of models in use.
- Make uses of a GRU Layer.
- Decentralized design with parameter sharing scheme

Activation: Gumbel Softmax Layer

- Avoids precision (NaN) issues while computing losses during training
  - Logits are too small or too large → Clipped before softmax
  - Probabilities become 0 → Regularized the noise
  - Exploding Gradients → Lowered learning rate
- Can be trained with low resources

DIAL – Leader’s Model Architecture

Layer	Output	Param #	Connect
InputLayer	(None, 5, 5)	0	-
Reshape	(None, 25, 1)	0	InputLayer
GRU	(None, 8)	264	Reshape
Dense	(None, 64)	576	GRU
Dense: Action Logits	(None, 9)	585	Dense
Gumbel Softmax	(None, 9)	0	Dense: Action Logits
Message	(None, 8)	520	Dense
Dense	(None, 1)	65	Dense

Total params: 2,010 (7.85 KB)
Trainable params: 2,010 (7.85 KB)
Non-trainable params: 0 (0.00 B)

Carleton University

DIAL – Follower’s Model Architecture

Layer	Output	Param	Connect
InputLayer 1	(None, 5, 5)	0	-
InputLayer 2	(None, 8)	0	-
Reshape 1	(None, 25, 1)	0	InputLayer 1
Reshape 2	(None, 8, 1)	0	InputLayer 2
Concentrate	(None, 33, 1)	0	Reshape 1 Reshape 2
GRU	(None, 64)	12864	Concentrate
Dense	(None, 64)	4160	GRU
Dense: Action Logits	(None, 9)	585	Dense

Total params: 17,609 (68.79 KB)
Trainable params: 17,609 (68.79 KB)
Non-trainable params: 0 (0.00 B)

Results

Why is communication so significant?

How is our model performing in an execution?

Performance

- Training Time (including Tuning): 83963.22 seconds (~23 hours)
- Average Time to train 50 episodes: 41,981.61 (~11.66 hours)

Side Concerns

- Too much memory consumption
- CPU: ~14GB in total
- GPU: Max Step Time exceeds 12.5 ms

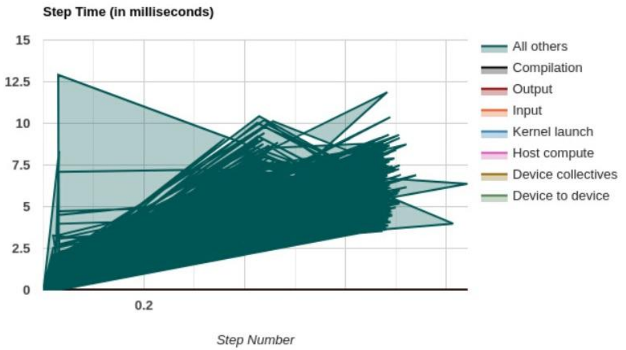


Figure 2: GPU Usage Graph in a Training process

Evaluation Metrics

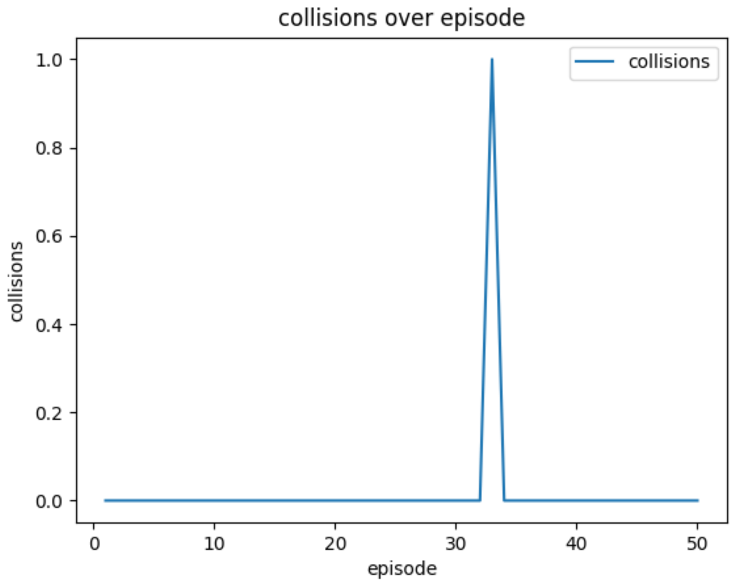


Figure 3: Performance of Collision vs Episode

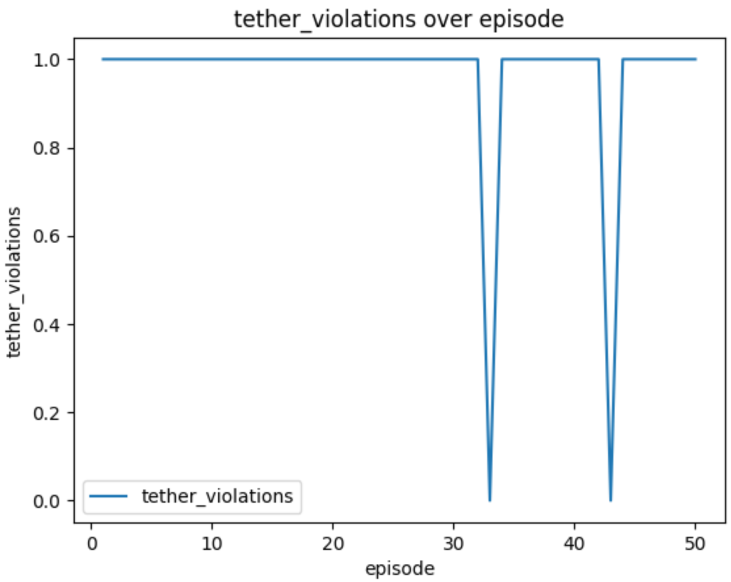


Figure 4: Performance of Tether Violations vs Episode

Evaluation Metrics

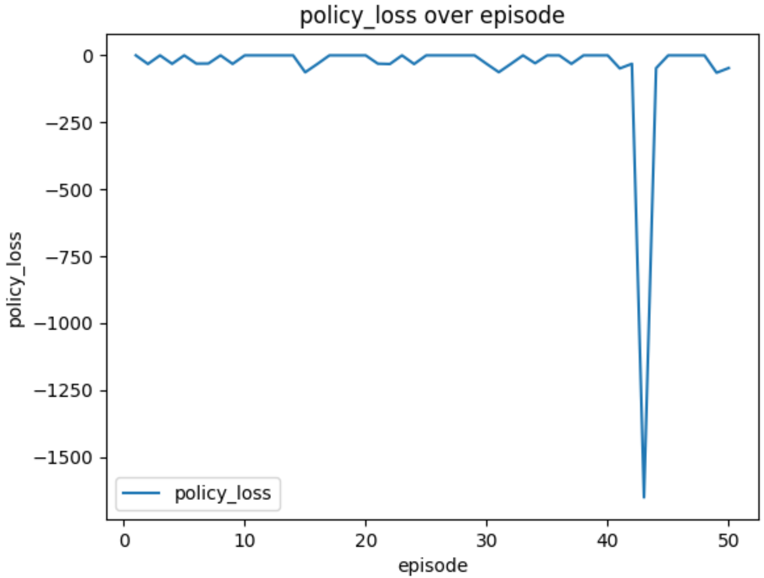


Figure 5: Performance of Policy Loss vs Episode

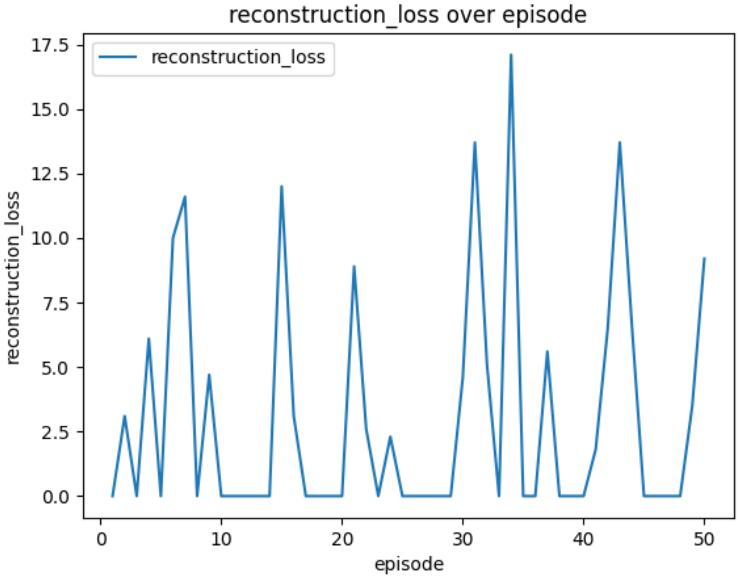


Figure 6: Performance of Reconstruction Loss vs Episode

Evaluation Metrics

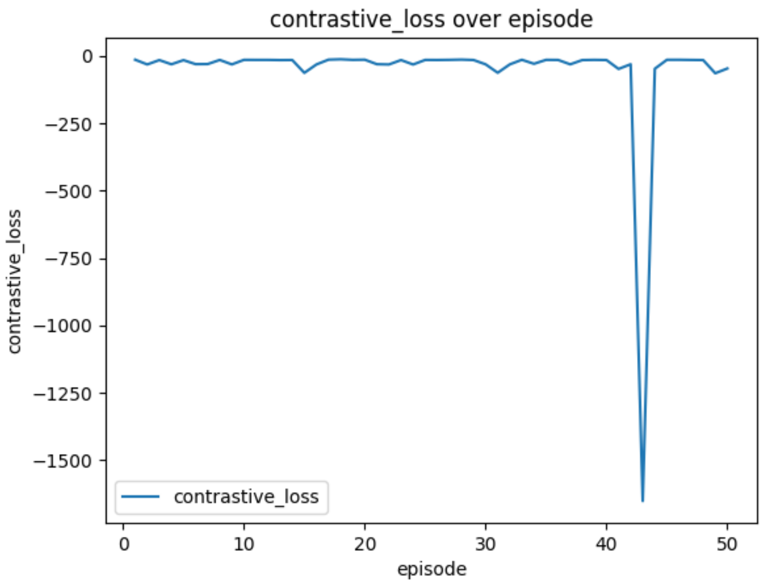


Figure 7: Performance of Contrastive Loss vs Episode

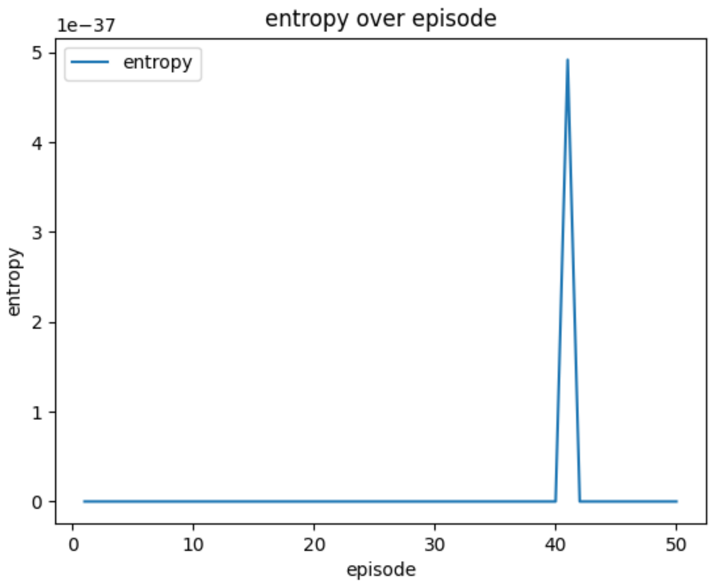


Figure 8: Performance of Entropy vs Episode

Carleton University

Evaluation Metrics

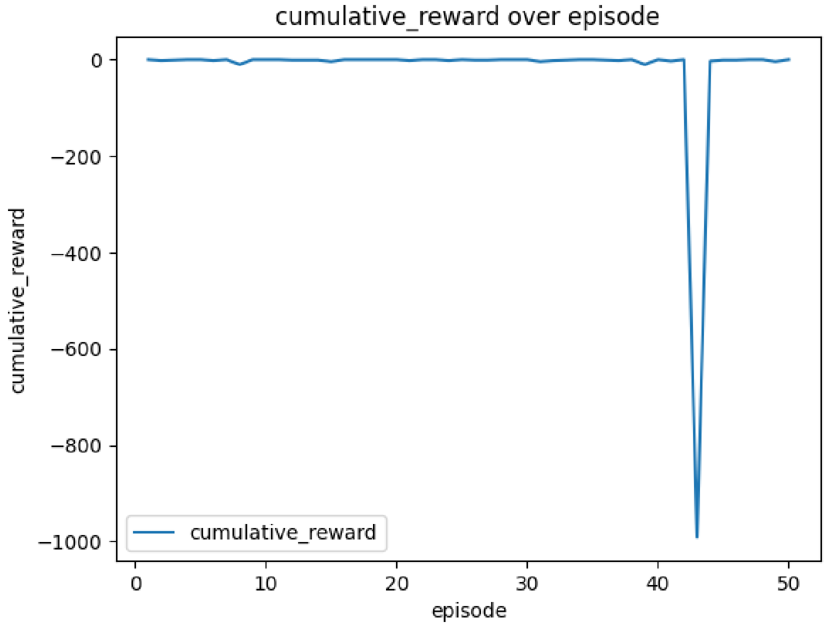


Figure 9: Performance of Cumulative Reward vs Episode

Evaluation Metrics

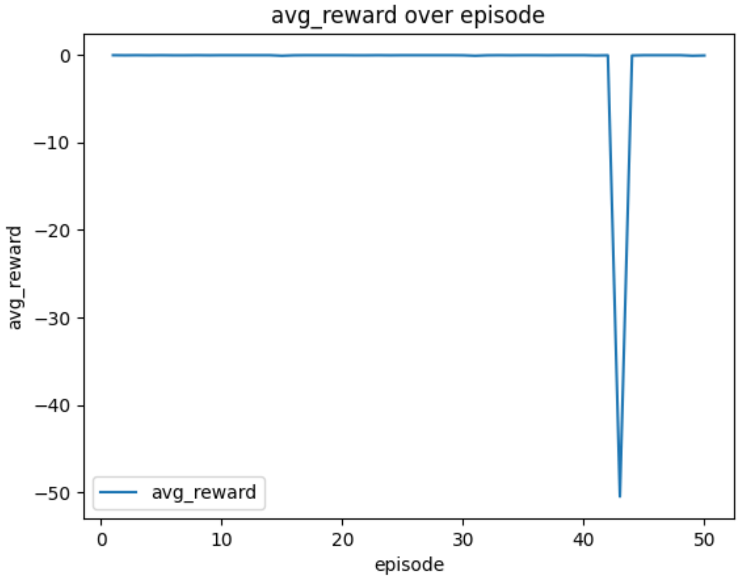


Figure 10: Performance of Episode vs Average Reward

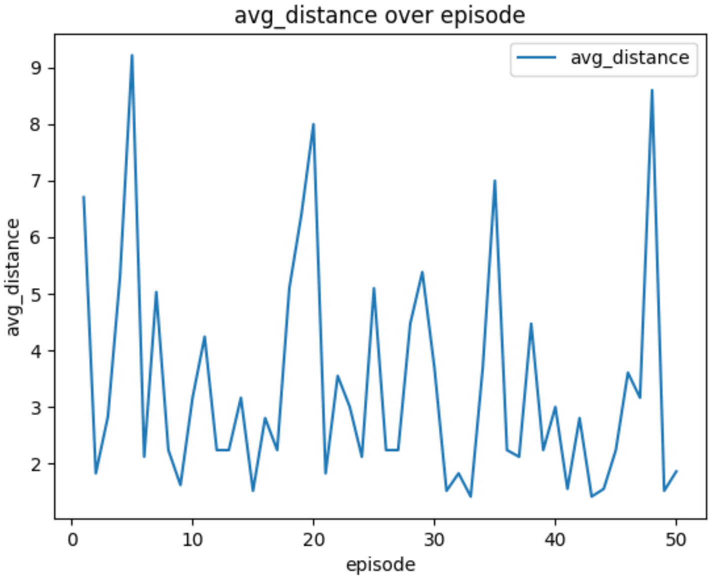


Figure 11: Performance of Average Distance travelled vs Episode

Carleton University

Evaluation Metrics

The DIAL Method

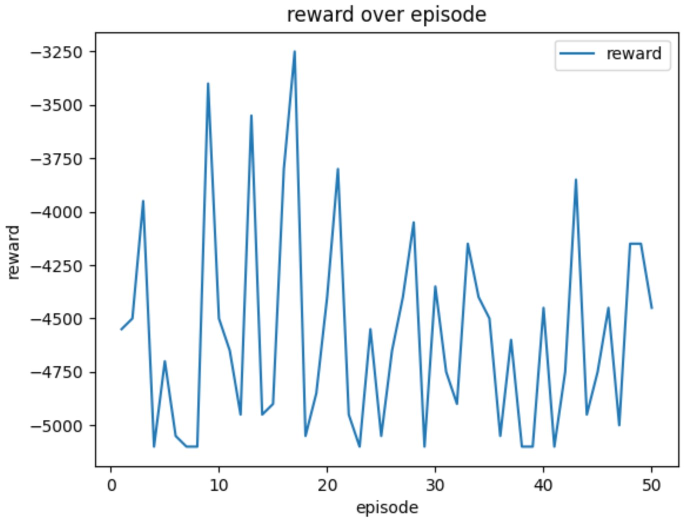


Figure 12: Performance of Episodic Reward in DIAL Method

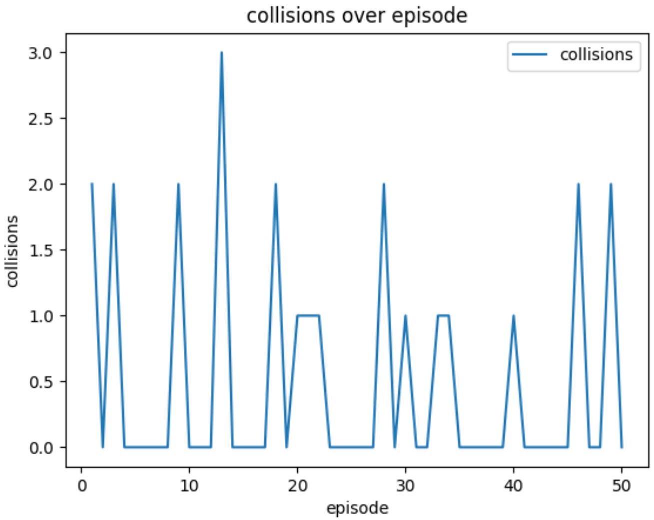


Figure 13: Occurrences of Episodic Collisions in DIAL Method

Carleton University

Evaluation Metrics

The DIAL Method

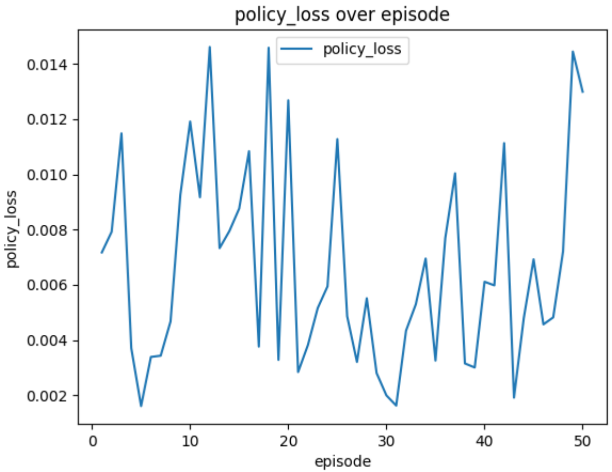


Figure 14: Episodic Policy Loss in DIAL Method

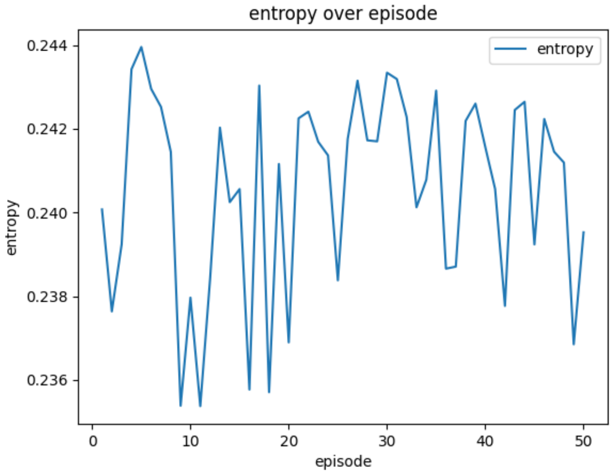


Figure 15: Episodic Entropy in DIAL Method

Carleton University

Evaluation Metrics

- Quality Assured: Unit Test covers **overall 69.8%** of our codes

Modules	Coverage
MARL-based learning functions	79%
Model’s Prediction Results	31%
Map Generation functions	98%
Agent Class	42%
Environment’s Codes	99%

Carleton University

Conclusions & Future Work

How do we approach our research questions?

What’s next?

Conclusions

Centralized Training – Parameter Sharing

- Agents learn the environment via a centralized actor-critic network.
- Agents optimize communications (and actions) via backpropagation.

Asymmetric Communication

- Agents succeed in a 1-way communication via the encoder and decoder.

Making an Action

- Agents maintain safe actions from huge penalties in training and environment reset in execution.

Future Work

Publishing

- Benchmarking algorithms could be experimented and discussed.
- Existing ones could be optimized in performance: Memory Usages.

Scaling Up

- Different grid sizes, more agents / obstacles / available targets
- Initializing grids with agents not necessarily already within tether
- Introducing the effect of noises during communications

Carleton University

References

- Blumenkamp, J., & Prorok, A. (2020). *The emergence of adversarial communication in multi-agent reinforcement learning*. In *Proceedings of the 4th Conference on Robot Learning (CoRL 2020)*, Cambridge, MA, USA.
- Jakob N. Foerster, Yannis M. Assael, Nando de Freitas, Shimon Whiteson. "Learning to Communicate with Deep Multi-Agent Reinforcement Learning". In Proceedings of the NeurIPS. 2016. https://arxiv.org/pdf/1605.06676, pp: 2137-2145.
- Lo, Y. L., Sengupta, B., Foerster, J., & Noukhovitch, M. (2024). Learning multi-agent communication with contrastive learning. In Proceedings of the International Conference on Learning Representations (ICLR) 2024.
- Meng, X., & Tan, Y. (2024). PMAC: Personalized multi-agent communication. In Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence (AAAI-24). Association for the Advancement of Artificial Intelligence.
- Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. Advances in neural information processing systems, 30, 2017.

Carleton University

References

- Toru Lin, Jacob Huh, Christopher Stauffer, Ser Nam Lim, and Phillip Isola. Learning to ground multiagent communication with autoencoders. Advances in Neural Information Processing Systems, 34,2021.
- Yat Long Lo, Biswa Sengupta, Jakob Foerster, Michael Noukhovitch. “LEARNING MULTI-AGENT COMMUNICATION WITH CONTRASTIVE LEARNING”. In: Proceedings of the ICLR 2024 Conference. 2024. https://arxiv.org/pdf/2307.01403.
- Zinovi Rabinovich. Additional Lecture on “Multi-Agent Reinforcement Learning Communications (Brief) Review”. Carleton University. 2025.
- Zhang, Y., Li, M., Zeng, X., Song, N., Zhang, J., Peng, B., & Zhang, P. (2024). Research on multi-agent cooperative tasks based on improved proximal policy optimization. In Proceedings of the 2024 12th China Conference on Command and Control.

Carleton University

Thank You

Carleton University

GitHub: https://github.com/kmock930/MARL-autonomous-vehicle

Carleton University

Appendix: GRU

- Remembers long-term dependencies (important in partial observability).
- Effective in capturing dynamics of spatial-temporal environments
- Improving exploration

- Continuous communication message vector: $m_t = [v_1, \dots, v_n]$
- Decoded message : m'
- Follower local observation : o_t^f

- Leader message : $h_t^{enc} = GRU_{enc}(h_{t-1}^{enc}, m_t)$
- Follower input: $h_t^{dec} = GRU_{dec}\left(h_{t-1}^{dec}, [o_t^f, m_t]\right)$

Reconstruction Loss:

$$L_{recon} = MSE(m, m')$$