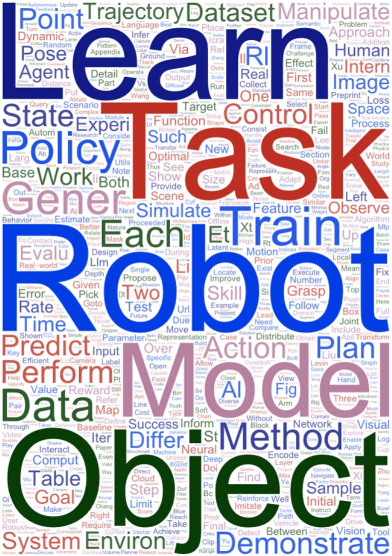


Robot Learning

Multi-Robot Learning

Wolfgang Hönig
Technical University of Berlin
Summer 2024

Motivation: Multi-Robot Systems

- Multiple robots (typically in a team) with a common goal
- Typical promises:
  - Achieve goal faster
  - Achieve goal more robustly
  - Higher flexibility (esp. heterogeneous systems)
  - Cheaper (?)

Motivation: Multi-Robot Systems

- Successful (industrial) solutions
  - Warehouse logistics (Amazon Robotics, former Kiva systems)

  - Aerial Drone shows (Intel, Verity Studios)

Motivation: Multi-Robot System Challenges

- Controls: additional constraint for inter-robot collision avoidance
- Decision Making: information sharing, task assignment, curse-of-dimensionality for centralized approaches, safety/robustness for decentralized systems
- Perception: sensing team members, sensor fusion

Outline

- **Handling Dynamic Neighbors**
  - LSTMs
  - CNNs
  - DeepSets
  - Graph Neural Networks
- Multi-Agent Reinforcement Learning (MARL)
- Discussion / Open Challenges

Dynamic Neighbors

- Team of robots has time-varying neighbors/observations/communication links
- Often need to learn with time-varying input dimensionality
  - Example: (Distributed) collision avoidance maps observation of neighboring robots to actions $f(\mathcal{Y}) \to u$
- Learned functions need to be permutation-invariant and support dynamic domain cardinality

LSTMs [3]

2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

Motion Planning Among Dynamic, Decision-Making Agents with Deep Reinforcement Learning

Michael Everett‡, Yu Fan Chen†, and Jonathan P. How‡

- Key idea: Feed observations of neighbors into an LSTM (closest neighbor last)

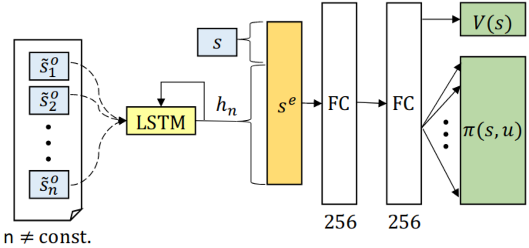


Fig. 3: Network Architecture. Observable states of nearby agents, $\tilde{\mathbf{s}}_i^o$, are fed sequentially into the LSTM, as unrolled in Fig. 2. The final

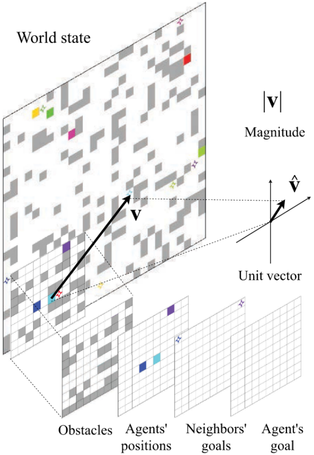
CNNs [14]

2378 IEEE ROBOTICS AND AUTOMATION LETTERS, VOL. 4, NO. 3, JULY 2019

PRIMAL: Pathfinding via Reinforcement and Imitation Multi-Agent Learning

Guillaume Sartoretti, Justin Kerr, Yunfei Shi, Glenn Wagner, T. K. Satish Kumar, Sven Koenig, and Howie Choset

- Key idea: Encode neighbor information as a picture
- Videos: `https://goo.gl/T627XD`



Deep Sets [21]

- Any continuous, permutation-invariant function $f(\mathcal{X})$ can be approximated:

Learns aggregation of hidden state

Learns representation of each element

$$f(\mathcal{X}) \approx \rho\left(\sum_{x \in \mathcal{X}} \phi(x)\right)$$

superposition in hidden state

- Improvement over Convolutional NN (CNN): continuous space, efficiency
- Example:

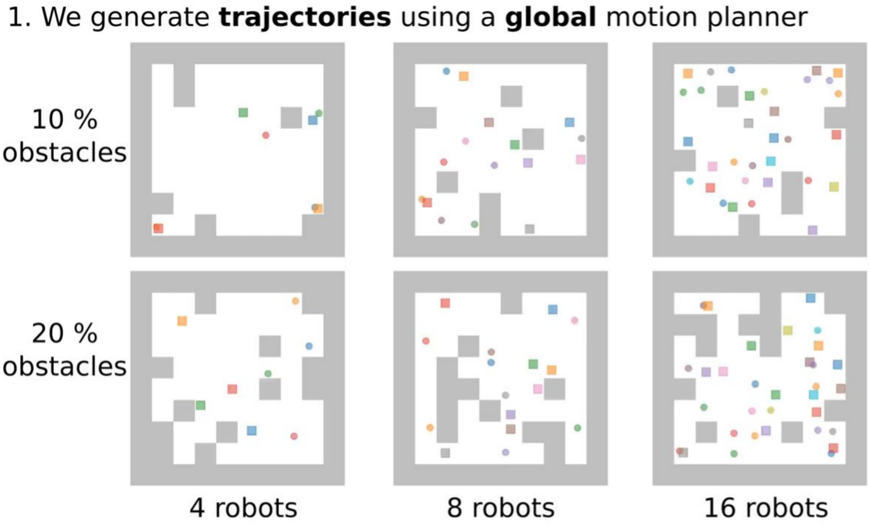
 +  = 9

 +  +  = 20

Case Study: GLAS [13]

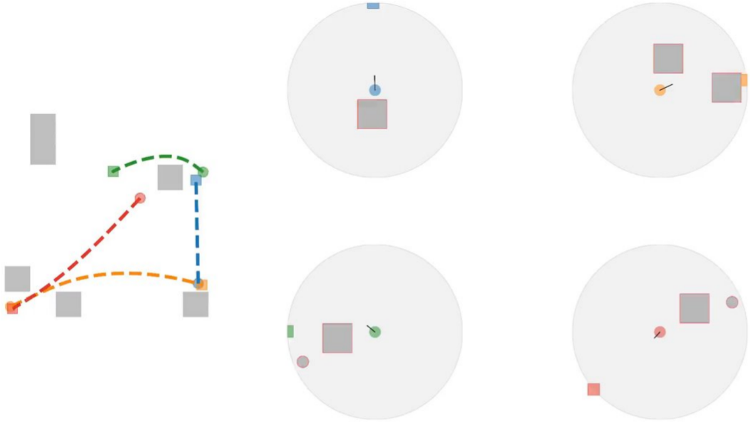
- Goal: imitate (slow) centralized controller using only local observations: $\pi : y \mapsto u$
- Data: Example trajectories by solving many multi-robot motion planning instances with a centralized planner
- Approach: Behavior Cloning + Privileged Teacher

Case Study: GLAS [13]



Case Study: GLAS [13]

2. We extract **local observations** and **actions**



Case Study: GLAS [13]

- Train (5 small feedforward networks trained jointly)

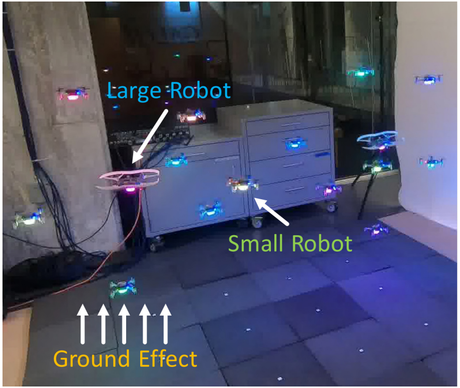
glas/architecture_simple.pdf

Case Study: GLAS [13]

- How would one train this in practice in pyTorch? [variable number of neighbors vs. batching]

Case Study: Neural-Swarm2 [15]

- Goal: predict aerodynamic interaction [unmodeled physics, as a function of neighbors' positions]



- Data: Real flight tests (synchronized trajectories with poses of robots and measured accelerations and motor commands)
- Approach: Behavior Cloning

Case Study: Neural-Swarm2 [15]: Heterogeneous Deep Sets

Learns aggregation for type $\mathcal{I}(i)$

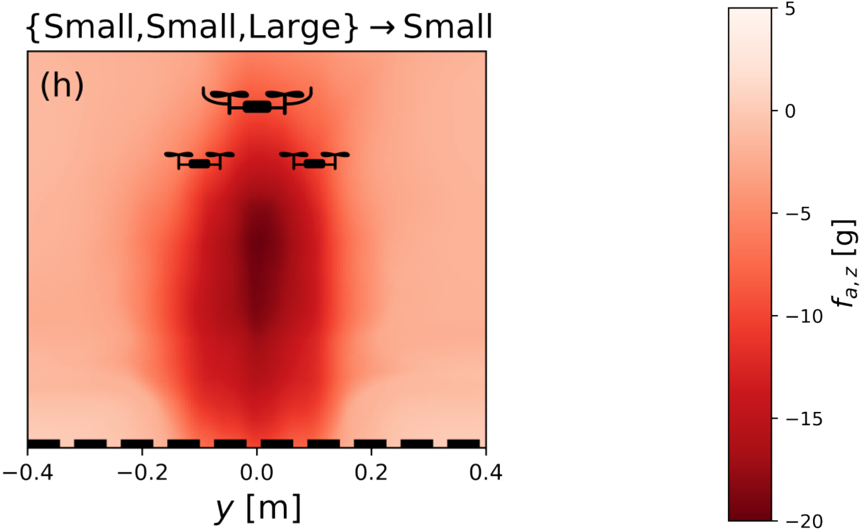
Learns representation from type $\mathcal{I}(j)$ neighbor

$$\hat{\mathbf{f}}_a^{(i)} \approx \boldsymbol{\rho}_{\mathcal{I}(i)}\left(\sum_{k=1}^{K}\sum_{\mathbf{x}^{(ij)}\in\mathbf{r}^{(i)}_{\mathrm{type}_k}} \boldsymbol{\phi}_{\mathcal{I}(j)}\left(\mathbf{x}^{(ij)}\right)\right)$$

superposition in

neuralswarm2/fig1a.pdf

Case Study: Neural-Swarm2 [15]

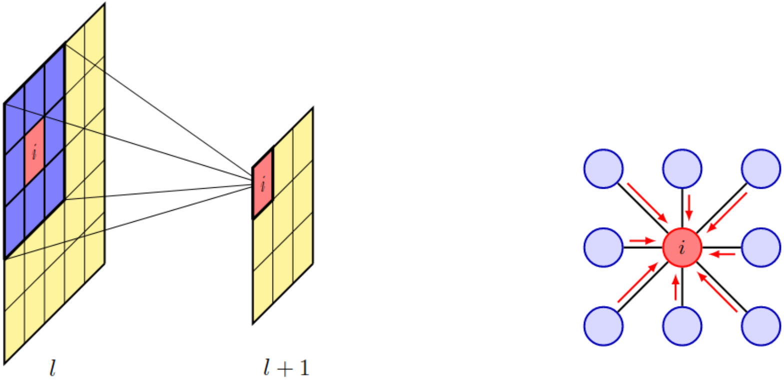


Case Study: Neural-Swarm2 [15]

`https://youtu.be/Y02juH6BDxo`

Graph Neural Networks (GNNs)

- Inspiration: CNNs as graph



(a) (b)

C. M. Bishop and H. Bishop. *Deep Learning: Foundations and Concepts.*
Springer International Publishing, Cham, 2024.
doi:10.1007/978-3-031-45468-4

Graph Neural Networks (GNNs)

- Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
- Basic case: learn features for each node $n \in \mathcal{V}$
- Use L layers with D-dimensional vector $h_n^{(l)}$

Graph Neural Networks (GNNs)

Algorithm 13.1: Simple message-passing neural network

Input: Undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
Initial node embeddings $\{\mathbf{h}_n^{(0)} = \mathbf{x}_n\}$
$\mathrm{Aggregate}(\cdot)$ function
$\mathrm{Update}(\cdot, \cdot)$ function

Output: Final node embeddings $\{\mathbf{h}_n^{(L)}\}$

```
// Iterative message-passing
```

for $l \in \{0, \dots, L-1\}$ **do**

$$\mathbf{z}_n^{(l)} \leftarrow \mathrm{Aggregate}\left(\left\{\mathbf{h}_m^{(l)} : m \in \mathcal{N}(n)\right\}\right)$$

$$\mathbf{h}_n^{(l+1)} \leftarrow \mathrm{Update}\left(\mathbf{h}_n^{(l)}, \mathbf{z}_n^{(l)}\right)$$

end for

return $\{\mathbf{h}_n^{(L)}\}$

Graph Neural Networks (GNNs)

- Examples for Aggregate/Update:
  - Aggregate($\{h_m^{(l)} : m \in \mathcal{N}(n)\}$) = $MLP_\rho \left(\sum_{m \in \mathcal{N}(n)} MLP_\phi(h_m^{(l)}) \right)$
  - Update($h_n^{(l)}, z_n^{(l)}$) = $f(W_{self} h_n^{(l)} + W_{neigh} z_n^{(l)} + b)$
- Extensions to have input/output features per edge and graph [See e.g., [1]]
- Training "as usual" (on whole graphs)
- In practice: PyG `https://www.pyg.org/` or DGL `https://www.dgl.ai/`

Case Study: Learning to Communicate for Multi-Robot Path Finding [8]

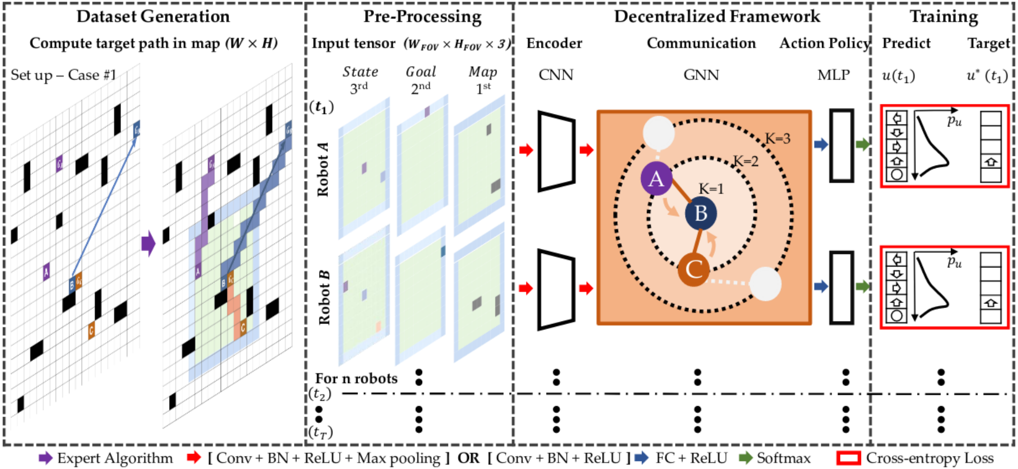
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Graph Neural Networks for Decentralized Multi-Robot Path Planning

Qingbiao Li[1], Fernando Gama[2], Alejandro Ribeiro[2], Amanda Prorok[1]

- Goal: Learn how to communicate to imitate a centralized Multi-Agent Path Finding expert
- Data: Trajectories computed by a centralized expert
- Approach: IL w/ DAgger

Case Study: Learning to Communicate for Multi-Robot Path Finding [8]



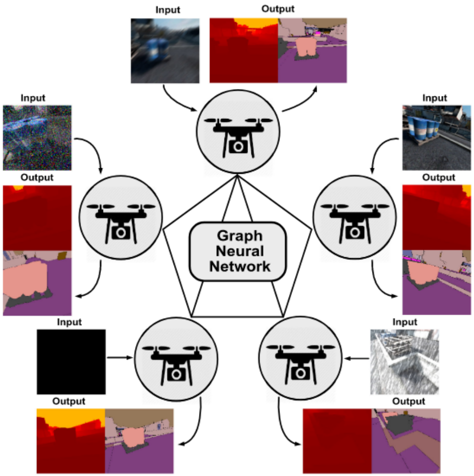
Case Study: Multi-Robot Perception [23]

IEEE ROBOTICS AND AUTOMATION LETTERS, VOL. 7, NO. 2, APRIL 2022 2289

Multi-Robot Collaborative Perception With Graph Neural Networks

Yang Zhou, *Graduate Student Member, IEEE*, Jiuhong Xiao, Yue Zhou, and Giuseppe Loianno, *Member, IEEE*

- Goal: Learn what to communicate for depth estimation or segmentation
- Data: Labeled Data mostly from simulator; some from real flights
- Approach: Behavior Cloning
- Video: `https://youtu.be/2bdhLI3dqo0`



GNN Applications

- Flocking (in simulation) [17, 7, 5]
- Navigation (simulation + RL) [19]
- Graph Control Barrier Function (simulation + IL w/ DAgger) [22]
- Learning to Communicate Variations [9, 5]

Outline

- Handling Dynamic Neighbors
  - LSTMs
  - CNNs
  - DeepSets
  - Graph Neural Networks
- **Multi-Agent Reinforcement Learning (MARL)**
- Discussion / Open Challenges

MARL Definition

- Single Robot: MDP $(\mathcal{S}, \mathcal{A}, P, R, P_0, \gamma)$ with state space $\mathcal{S}$, action space $\mathcal{A}$, transition probabilities $P(s_{t+1} \| s_t, a_t)$, reward fct $r_t = R(s_t, a_t)$, initial state distribution $P_0(s_0)$, and discounting factor $\gamma \in [0, 1]$.

MARL Definition

- Single Robot: MDP $(\mathcal{S}, \mathcal{A}, P, R, P_0, \gamma)$ with state space $\mathcal{S}$, action space $\mathcal{A}$, transition probabilities $P(s_{t+1} \| s_t, a_t)$, reward fct $r_t = R(s_t, a_t)$, initial state distribution $P_0(s_0)$, and discounting factor $\gamma \in [0, 1]$.
- Multi-Robot: Markov game $(N, \mathcal{S}, \mathcal{A}, P, R, P_0, \gamma)$ with N robots, $\mathcal{S}$ *joint* state space, $\mathcal{A} = A_1 \times A_2 \times \ldots \times A_N$ *joint* action space, reward fct $r_1, \ldots, r_N = R(s, a)$
- Goal: Find policy (or policies) that maximize expected reward

Rewards

- **Fully cooperative**: $r_1 = r_2 = \ldots = r_N$ [No credit assignment; difficult to train]
- **Competitive**: zero-sum games ($\sum_i r_i = 0$), prey-predator games (cooperative per team; competitive per game)
- **Mixed Cooperative-Competitive**: (local) reward shaping, to achieve a common goal

Learning

- **Centralized** model as stacked robot (centralized training & inference)
- **Independent Learning** each robot learns own policy (decentralized training & inference)
- **Centralized Training Decentralized Execution (CTDE)**

Challenges

- Non-Stationarity: if policy of other agents can't be observed, the Markov assumption is violated (e.g., distributed Q-Learning)
- Scalability: in standard policy gradient algorithms, the probability of estimating the policy gradient correctly might decrease exponentially with the number of agents
  [Concrete example: appendix of [10]]

Approaches

- Centralized critic, e.g., Multi-Agent deep deterministic policy gradient (MADDPG, [10])
- Factorized value functions, e.g., Value Decomposition Networks (VDN, [16])
- Communication Learning

Practical Considerations

- VMAS (Vectorized Multi-Agent Simulator for Collective Robot Learning) `https://github.com/proroklab/VectorizedMultiAgentSimulator` [Simple 2D physics engine build in pyTorch]
- MARLlib `https://github.com/Replicable-MARL/MARLlib`
- More Details/Overview about MARL:

Y. Wang, M. Damani, P. Wang, Y. Cao, and G. Sartoretti. Distributed reinforcement learning for robot teams: A review. *Current Robotics Reports*, 3(4):239–257, Dec. 2022.
doi:10.1007/s43154-022-00091-8

J. Orr and A. Dutta. Multi-agent deep reinforcement learning for multi-robot applications: A survey. *Sensors*, 23(7):3625, Jan. 2023.
doi:10.3390/s23073625

Case Study: Distributed Collision Avoidance (Ground) [4]

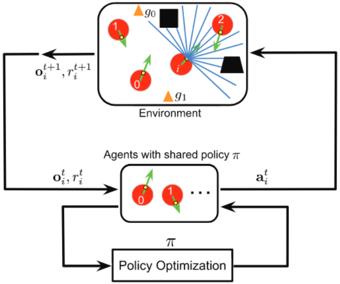
Article

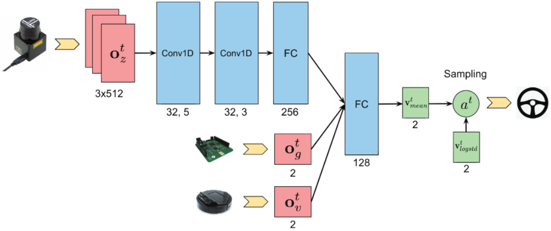
Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios

The International Journal of
Robotics Research
2020, Vol. 39(7) 856–892

DOI: 10.1177/0278364920916531
journals.sagepub.com/home/ijr
SAGE

Tingxiang Fan[1*], Pinxin Long[2*], Wenxi Liu[3] and Jia Pan[1]





Case Study: Distributed Collision Avoidance (Ground) [4]

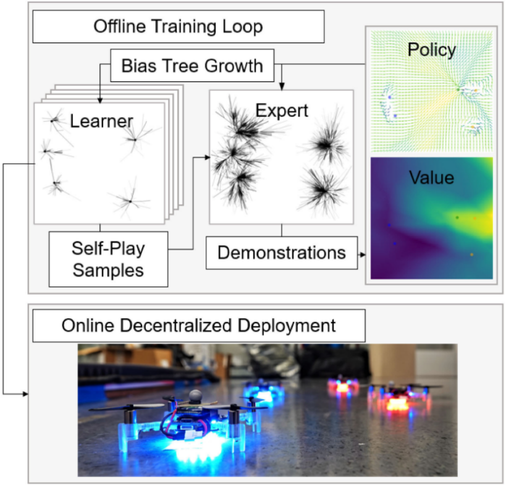
- Goal: find decentralized policy: $\pi : y, g \mapsto u$
- Data: Collected in simulation during RL (input LIDAR, relative goal, velocity; output: action)
- Approach: PPO (centralized learning, decentralized execution; shared policy)
- Video: `https://sites.google.com/view/hybridmrca`

Case Study: Distributed Collision Avoidance (UAVs) [6]

- Goal: find decentralized policy: $\pi : y, g \mapsto u$
- Data: Collected in simulation during RL (input state, nearby obstacles, nearby neighbors; output: thrust per rotor)
- Approach: IPPO (centralized learning, decentralized execution; shared policy)
- Video: `https://sites.google.com/view/obst-avoid-swarm-rl`

Case Study: Neural Tree Expansion [12]

- Goal: find decentralized policies for multi-team games (e.g., reach-target avoid)

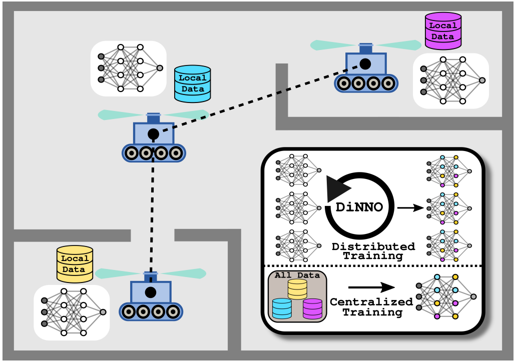


- Data: Collected with a neural-biased "expert" (large Monte-Carlo Tree Search)
- Approach: MCTS + IL + DAgger (essentially: AlphaZero in continuous state spaces)
- Video: `https://youtu.be/mklbTfWl7DE`

Outline

- Handling Dynamic Neighbors
  - LSTMs
  - CNNs
  - DeepSets
  - Graph Neural Networks
- Multi-Agent Reinforcement Learning (MARL)
- **Discussion / Open Challenges**

DiNNO: Distributed Neural Network Optimization [20]



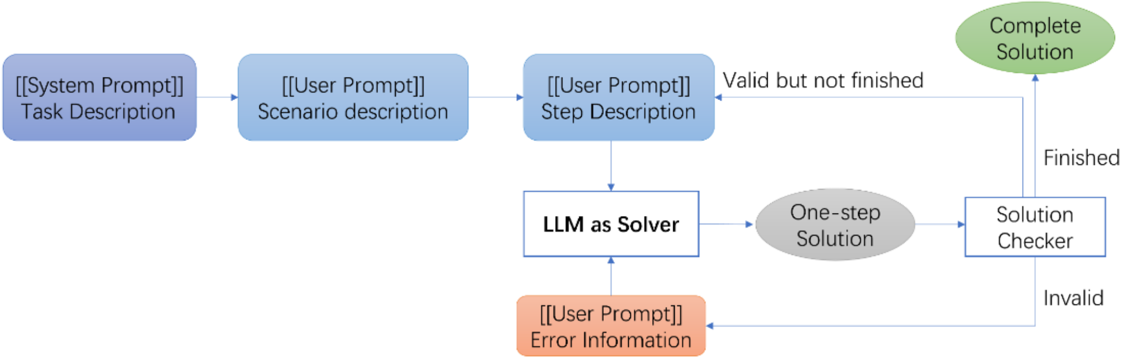
- Collect data locally, local augmented Lagrangian update, share resulting weights via consensus
- Works for IL and RL
- Web: `https://msl.stanford.edu/projects/dist_nn_train`

LLMs and Multi-Robots [2]

Why Solving Multi-agent Path Finding with Large Language Models has not Succeeded Yet

Weizhe Chen [1] **Sven Koenig** [1] **Bistra Dilkina** [1]

- (Arxiv, Jan. 2024)



LLMs and Multi-Robots [2]

Agent 1 is currently in (0,2), and wants to go to (3,1).
Agent 2 is currently in (1,3), and wants to go to (2,0).
The map is as follows, where '@' denotes a cell with an obstacle that an agent cannot pass, and '.' denotes an empty cell that an agent can pass.
The bottom-left cell is (0,0) and the bottom-right cell is (31,0):

....
...@
....
.@..

In the next step:
Agent 1 can move ['stay at (0, 2)', 'right to (1, 2)', 'up to (0, 3)', 'down to (0, 1)'].
Agent 2 can move ['stay at (1, 3)', 'left to (0, 3)', 'right to (2, 3)', 'down to (1, 2)'].

Open Challenges

- Deployment to real robots (especially RL)
- Safety (esp. partially unknown dynamics, perception)
- Interpretability (of communication)

Conclusion

- Multi-Robot brings new challenges
  - Large state space (or violation of Markov assumption)
  - Dynamic number of neighbors
  - Reasoning about communication
- Deep Sets: permutation invariant architecture that is easy to train and computationally efficient [useful for $\pi : x, \mathcal{N} \mapsto u$]
- GNN: Generalization of deep sets [useful for learning communication]
- Learning a decentralized policy from a centralized expert works well (IL + DAgger)
- Deployment to real robot teams remains challenging

[1] C. M. Bishop and H. Bishop.
Deep Learning: Foundations and Concepts.
Springer International Publishing, Cham, 2024.
doi:10.1007/978-3-031-45468-4.

[2] W. Chen, S. Koenig, and B. Dilkina.
Why solving multi-agent path finding with large language model has not succeeded yet, Feb. 2024.
arXiv:2401.03630, doi:10.48550/arXiv.2401.03630.

[3] M. Everett, Y. F. Chen, and J. P. How.
Motion planning among dynamic, decision-making agents with deep reinforcement learning.
In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3052–3059, Madrid, Oct. 2018. IEEE.
doi:10.1109/IROS.2018.8593871.

[4] T. Fan, P. Long, W. Liu, and J. Pan.
Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios.
The International Journal of Robotics Research, 39(7):856–892, June 2020.
doi:10.1177/0278364920916531.

[5] F. Gama, Q. Li, E. Tolstaya, A. Prorok, and A. Ribeiro.
Synthesizing decentralized controllers with graph neural networks and imitation learning.
IEEE Transactions on Signal Processing, 70:1932–1946, 2022.
doi:10.1109/TSP.2022.3166401.

[6] Z. Huang, Z. Yang, R. Krupani, B. Şenbaşlar, S. Batra, and G. S. Sukhatme.
Collision avoidance and navigation for a quadrotor swarm using end-to-end deep reinforcement learning, May 2024.
arXiv:2309.13285, doi:10.48550/arXiv.2309.13285.

[7] R. Kortvelesy and A. Prorok.
Modgnn: Expert policy approximation in multi-agent systems with a modular graph neural network architecture.
In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9161–9167, Xi'an, China, May 2021. IEEE.

doi:10.1109/ICRA48506.2021.9561386.

[8] Q. Li, F. Gama, A. Ribeiro, and A. Prorok.
Graph neural networks for decentralized multi-robot path planning.
In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11785–11792, Las Vegas, NV, USA, Oct. 2020. IEEE.
doi:10.1109/IROS45743.2020.9341668.

[9] Q. Li, W. Lin, Z. Liu, and A. Prorok.
Message-aware graph attention networks for large-scale multi-robot path planning.
IEEE Robotics and Automation Letters, 6(3):5533–5540, July 2021.
doi:10.1109/LRA.2021.3077863.

[10] R. Lowe, YI. WU, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch.
Multi-agent actor-critic for mixed cooperative-competitive environments.
In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
URL:
https://proceedings.neurips.cc/paper_files/paper/2017/hash/68a9750337a418a86fe06c1991a1d64c-Abstract.html.

[11] J. Orr and A. Dutta.
Multi-agent deep reinforcement learning for multi-robot applications: A survey.
Sensors, 23(7):3625, Jan. 2023.
doi:10.3390/s23073625.

[12] B. Riviere, W. Honig, M. Anderson, and S.-J. Chung.
Neural tree expansion for multi-robot planning in non-cooperative environments.
IEEE Robotics and Automation Letters, 6(4):6868–6875, Oct. 2021.
doi:10.1109/LRA.2021.3096758.

[13] B. Riviere, W. Honig, Y. Yue, and S.-J. Chung.
Glas: Global-to-local safe autonomy synthesis for multi-robot motion planning with end-to-end learning.

IEEE Robotics and Automation Letters, 5(3):4249–4256, July 2020.
doi:10.1109/LRA.2020.2994035.

[14] G. Sartoretti, J. Kerr, Y. Shi, G. Wagner, T. K. S. Kumar, S. Koenig, and H. Choset.
Primal: Pathfinding via reinforcement and imitation multi-agent learning.
IEEE Robotics and Automation Letters, 4(3):2378–2385, July 2019.
doi:10.1109/LRA.2019.2903261.

[15] G. Shi, W. Honig, X. Shi, Y. Yue, and S.-J. Chung.
Neural-swarm2: Planning and control of heterogeneous multirotor swarms using learned interactions.
IEEE Transactions on Robotics, 38(2):1063–1079, Apr. 2022.
doi:10.1109/TRO.2021.3098436.

[16] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel.
Value-decomposition networks for cooperative multi-agent learning based on team reward.
2018.

[17] E. Tolstaya, F. Gama, J. Paulos, G. Pappas, V. Kumar, and A. Ribeiro.
Learning decentralized controllers for robot swarms with graph neural networks.
In *Proceedings of the Conference on Robot Learning*, pages 671–682. PMLR, May 2020.
URL: https://proceedings.mlr.press/v100/tolstaya20a.html.

[18] Y. Wang, M. Damani, P. Wang, Y. Cao, and G. Sartoretti.
Distributed reinforcement learning for robot teams: A review.
Current Robotics Reports, 3(4):239–257, Dec. 2022.
doi:10.1007/s43154-022-00091-8.

[19] C. Yu, H. Yu, and S. Gao.
Learning control admissibility models with graph neural networks for multi-agent navigation.
In *6th Annual Conference on Robot Learning*, Aug. 2022.

URL: https://openreview.net/forum?id=xC-68ANJeK_.

[20] J. Yu, J. A. Vincent, and M. Schwager.
Dinno: Distributed neural network optimization for multi-robot collaborative learning.
IEEE Robotics and Automation Letters, 7(2):1896–1903, Apr. 2022.
doi:10.1109/LRA.2022.3142402.

[21] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola.
Deep sets.
In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
URL: https://papers.nips.cc/paper_files/paper/2017/hash/f22e4747da1aa27e363d86d40ff442fe-Abstract.html.

[22] S. Zhang, K. Garg, and C. Fan.
Neural graph control barrier functions guided distributed collision-avoidance multi-agent control.
In *7th Annual Conference on Robot Learning*, Aug. 2023.
URL: https://openreview.net/forum?id=VscdYkKgwdH.

[23] Y. Zhou, J. Xiao, Y. Zhou, and G. Loianno.
Multi-robot collaborative perception with graph neural networks.
IEEE Robotics and Automation Letters, 7(2):2289–2296, Apr. 2022.
doi:10.1109/LRA.2022.3141661.