Multi-Robot Planning for Coordinated Off-road Maneuvers

Cora A. Dimmig[1,2], Adam Goertz[1], Adam Polevoy[1,2], Mark Gonzales[2], Bryanna Yeh[1], Kevin C. Wolfe[1], Bradley Woosley[3], John Rogers[3], and Joseph Moore[1,2]

***Abstract*—Achieving unified multi-robot coordination and motion planning in complex, off-road or rugged urban environments is a challenging problem. In this paper, we present a hierarchical approach for minimizing visibility and maximizing safety with a multi-robot team navigating through a hazardous environment. In particular, our approach first segments the environment based on the visibility of the robot team from an adversarial observer's perspective, creates a topological graph, and computes an optimal multi-robot plan on that graph using mixed-integer programming (MIP). This visibility information also informs the lower layers of the autonomy stack to achieve dynamically feasible multi-robot planning across the hierarchy. We then demonstrate our approach in simulation in an off-road environment with multiple vehicles and on hardware in a rugged urban environment with a single autonomous Clearpath Warthog. We also discuss future plans for hardware demonstrations of multi-vehicle operations in off-road environments and the incorporation of learning to compensate for additional environmental complexities.**

I. Introduction

Intelligent planning for collaborative multi-robot teams in complex environments remains a fundamental research problem. Even when restricted to discrete actions and states, the multi-robot Markov decision process can quickly become computationally intractable [1]. This situation is further exacerbated when robot teams must reason about continuous-time dynamics, dynamic obstacles, and complex off-road conditions. In this paper, we discuss our approach, Stratified Topological Autonomy with Learned Contexts (STALC), which seeks to develop a unified framework for coordinated motion planning with multi-robot teams. In particular, our approach seeks to solve the problem of minimizing detection and risk when traversing through a potentially hazardous environment. To do this, we develop a hierarchical planning approach that consists of three primary levels– a high-level multi-robot graph planner, a mid-level planner for designing low-visibility paths between nodes, and a low-level planner for generating dynamically feasible, collision free paths. Our approach also investigates the use of supervised learning to construct local visibility maps to improve the fidelity of low-visibility navigation for the local planner in complex terrains. To evaluate the effectiveness of our approach, we test our planner in a simulated off-road environment with multiple vehicles and in hardware in a rugged urban environment with a single ground vehicle. In simulation, we show that our planning approach can effectively navigate a team of robots through a complex off-road forested environment. In hardware, we show that we can achieve unified minimum-visibility planning in an environment with a single vehicle. We also discuss our plans for hardware testing in off-road environments with multiple vehicles. Fig. 1 depicts both our off-road simulation experiments and our hardware experiments in an urban environment with a Clearpath Warthog.

[1]Johns Hopkins University Applied Physics Laboratory, Laurel, MD 20723, USA. Email: `Cora.Dimmig@jhuapl.edu`, `Joseph.Moore@jhuapl.edu`

[2]Department of Mechanical Engineering, Johns Hopkins University, Baltimore, MD 21218, USA.

[3]DEVCOM Army Research Laboratory, Adelphi, MD 20783, USA.

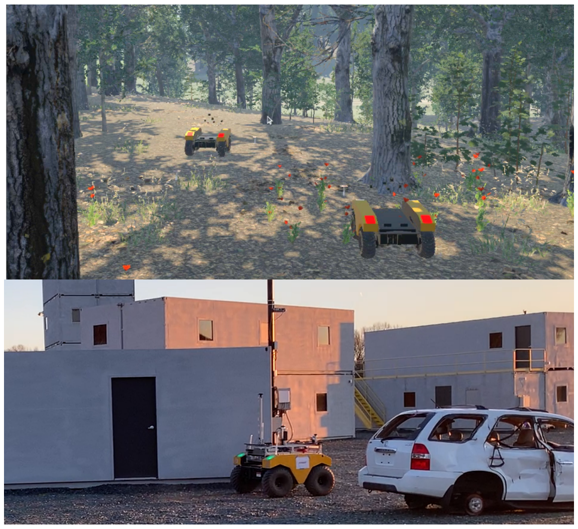


Fig. 1. (Top) Multiple Clearpath Warthogs executing coordinated maneuvers in an off-road environment. (Bottom) Clearpath Warthog navigating an urban environment while reducing visibility for a reconnaissance mission.

II. Related Work

To address the challenges associated with multi-robot planning in complex environments, researchers have often focused on solving the coordination and motion-planning problems independently [2], [3], without giving significant attention to the interactions between the two.

A. Multi-Robot Coordination

For many years, researchers have considered the problem of multi-robot coordination or task allocation. Oftentimes, solutions formulate the problem space as a set of discrete actions and states [4]–[8]. This includes Conflict Based Search (CBS) methods [9], auction-based methods [10] and game theoretic approaches [11]. However, many of these approaches consider agents independently, ultimately ignoring interactions and cooperation between vehicles. Vehicle interactions can cause the decision space of the problem to grow exponentially, which leads many researchers to consider decomposing the problem into smaller local problems, such as in this game theoretic approach [12]. Machine Learning (ML) approaches, often using Graph Neural Networks [13], [14], emerge as a way to solve the entire problem end-to-end while considering agent interactions [15] and coordination [16], [17]. However, ML approaches often suffer from poor generalization to environments that were not included in the training distribution and are categorically unable to provide optimality guarantees.

Mixed-Integer Programming (MIP) enables solving for end-to-end optimal solutions and has been studied across a variety of fields including recharging or timed delivery [18], [19], scheduling and task allocation [20], and communication provisioning [21]. However, MIP is NP-complete, leading many approaches to require simplifying heuristics [22] and/or offline computation for optimal solutions [23]. In this work, we overcome many of the computational challenges through a compact multi-agent planning formulation, and we present results rapidly solving end-to-end for optimal solutions.

B. Multi-Robot Motion Planning

Many researchers have also explored multi-robot motion planning in the context of obstacle avoidance or formation control. For instance, [24] uses MIP and minimum snap trajectories to design collision-free robot trajectories using a GPU. [25] generates a modified version of A* to handle inter-robot constraints. In [26], [27], the authors explore methods for multi-vehicle formation control in the presence of obstacles.

C. Unified Approaches

Over the years, fewer researchers have attempted to provide a unified approach for coordination and control (e.g., [28]–[31]). In many cases, even the examples that attempt to address the area of simultaneous multi-robot coordination and motion planning operate on incredibly simplified and structured environments. In this paper, we propose a unified approach for multi-robot coordination and motion panning where our high-level topological graph encodes information about the underlying metric spaces.

III. Problem Formulation

Our objective is solve for a plan for a multi-robot team that minimizes robot team visibility and maximizes safety in an adversarial environment. More specifically, we aim to avoid line-of-sight observations by a static adversary in such a way that we proceed tactically from one cover region to another. In addition to minimizing overall visibility, we introduce three foundational robot coordination constructs that we define as methods to reduce the risk of traversing a path:

- **Overwatch**: One team at a protected vantage point oversees the movement of a traversing team to mitigate risk.
- **Formations**: In areas of high vulnerability, moving in a formation increases awareness of the team.
- **Teaming**: Moving as a team in general can provide a higher level of protection to the robots.

We relegate team coordination to planning on a high-level graph, while minimizing visibility and maintaining formation is managed by mid-range and low-level planners, respectively.

IV. Technical Approach

Our technical approach consists of a three-level planning hierarchy including a top-level graph-based planner capable of coordinating multi-robot teams while traversing a hazardous environment. Our mid-level planner leverages a visibility map and A* to plan minimally visible paths between the graph's nodes. At the lowest-level, we utilize MPPI [32] to generate local plans, avoid obstacles, and follow the mid-range plan. All three levels share visibility and terrain maps to maintain consistent planning across the hierarchy.

A. Visibility Map

To enable low-visibility planning for the multi-robot team, we construct a visibility map which is used by all three layers of the hierarchy. We begin by assuming prior knowledge of a digital elevation model (DEM) of the environment together with a probability distribution, $\mathcal{D}$, over $\mathbb{R}^3$ representing the expected observer position. We make no assumptions about the form of $\mathcal{D}$ except that we are able to sample from it.

For a given DEM and observer distribution, we first segment the environment into regions of low and high visibility from the perspective of the observer. We accomplish this by adapting the notion of a viewshed [33] to the context of a distribution over observer positions. The viewshed of an area is a binary mask marking the regions of a DEM that are visible from a given observer position. Our goal is to estimate the probability that each point in the environment can be seen by the observer.

Taking N samples $d \sim \mathcal{D}$ from the adversary distribution, we compute a viewshed $\mathcal{V}_d$ for each sample. Discretizing the environment into a 2D grid, for each point (x, y) in the environment, the visibility probability is the expectation over the viewsheds from the sampled adversary positions. We refer to the map containing the probability of being seen at each point as the *visibility map*, and write $\mathcal{P}_{vis}$.

$$\mathbb{E}_{d \in \mathcal{D}}[\mathcal{V}_d(x, y)] \approx \frac{\sum_{i=0}^{N} \mathcal{V}_d(x, y)}{N} = \mathcal{P}_{vis}(x, y) \qquad (1)$$

B. Environment Segmentation

In order to construct a topological graph for planning, we use the visibility map to compute a set of nodes, V, and edges, E, together with a score for the visibility of each edge We also compute an overwatch score for each node, edge pair (v_i, e_j), denoting the overwatch that a team of robots positioned at node i can provide for a team of robots traversing edge j.

1) Node Selection: To identify the regions of low visibility (which we call *cover regions*) from the visibility map, we threshold the map to extract regions below a pre-defined visibility cutoff (nominally 0.1) and perform a connected-components search to find discrete regions whose size exceeds a threshold value. The nodes of the topological graph are placed at the centroid of each cover region.

2) Graph Pruning & Edge Weight Calculation: Next, we compute the set of graph edges and their costs to traverse. Initially, we consider the topological graph to be fully-connected. In order to reduce the number of edges considered in the optimization, we prune the graph by eliminating redundant edges. A redundant edge is any edge e connecting node v_i to node v_j where the optimal path from v_i to v_j passes through a cover region associated with another node $v_k \notin \{v_i, v_j\}$.

When traversing a path, $S = \{s_1, s_2, ..., s_n\}$, $s_i = [x_i, y_i]^T$, through the environment, we consider the total probability of non-detection, p_{nd}, as the product of the probabilities of non-detection at each cell (assuming independence).

$$p_{nd} = \prod_{s \in S} (1 - \mathcal{P}_{vis}(s)) \tag{2}$$

Taking the negative log of this quantity, we find that we can compute negative log non-detection probability by summing the contributions of each cell along the path.

$$-\log(p_{nd}) = \sum_{s \in S} -\log(1 - \mathcal{P}_{vis}(s)) \tag{3}$$

As the probability of detection approaches one, the cost to traverse a cell approaches infinity. In order to avoid infinite values in the optimization problem, we bound the maximum probability of detection at each cell to $1-\epsilon$ (for a small positive ϵ). We call the map containing the negative log non-detection probabilities at each cell $\mathcal{N}_{vis}$.

To identify redundant edges and compute the edge weights for the graph, we use an optimal path planning algorithm to compute a path between each pair of nodes in the graph. In this work we use a standard A* planner where the heuristic cost, $h(s)$, is the Euclidean distance from the current position to the goal, $g(s)$ denotes the cost to follow the optimal path from the start to s, and the successor cost, $c(s, s')$, is the cost to go from a cell in the map to a neighbor, with terms incorporating both the distance and the detection probability.

$$c(s, s') = ||s - s'||(1 + \lambda_p \mathcal{N}_{vis}(s')) \tag{4}$$
$$g(s') = g(s) + c(s, s') \tag{5}$$

In (4), λ_p is a scaling factor weighting the contribution of the visibility cost. Since the true cost-to-go is equal to the heuristic cost plus a non-negative visibility term, the heuristic $h(s)$ is trivially admissible.

After computing a path between each pair of nodes, we eliminate those paths (and their corresponding edges in the graph) that pass through a cover region which is not their start or goal. Finally, we compute the weight for each edge that remained after the pruning step described above by considering the optimal path S_e.

$$w_e = \sum_{s \in S_e} \mathcal{N}_{vis}(s) \tag{6}$$

3) Overwatch: An overwatch opportunity represents the ability of a team of robots positioned in a cover region to provide support to another team as they traverse an edge in the graph. For each node, edge pair, (v_i, e_j), we compute an overwatch score, which quantifies the visibility of edge e_j from node v_i. The process of computing this overwatch score is very similar to the visibility map calculation; however, instead of sampling from the adversary distribution, we instead sample uniformly from the cover region associated with v_i to produce the visibility map for an observer located at node v_i, which we call the *overwatch map*, $\mathcal{O}_{v_i}$. We then apply equations (3), (6) to $\mathcal{O}_{v_i}$ to compute the overwatch score for each edge in the graph.

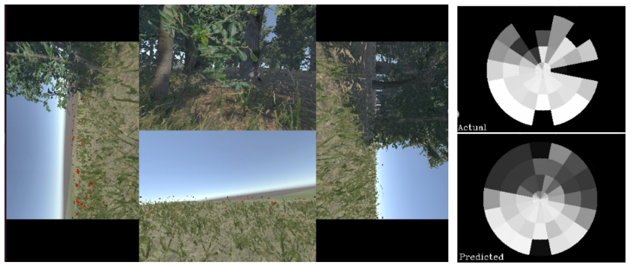


Fig. 2. (Left) View from four cameras oriented along the body-frame axes on an autonomous ground vehicle. (Right) Output of the learned visibility map (bottom) compared against the ground truth (top).

C. Graph Planner

We encode the dynamic topological graph structure in a MIP problem by tracking the number of agents at each location (node or edge) and constraining the flow between nodes and edges to enforce movement on the graph. We define cost functions for for the cost of traversing (based on the edge weights on the graph), the cost reduction from overwatch, and a cost for time. Overall, we minimize these cost functions in order to reduce visibility and the time to complete a particular objective. Our cost functions depend the positions of the robots relative to their teammates and reflect the dynamic element of the topological graphs, i.e. the cost of traversing decreases when the conditions for overwatch, moving in formation, or other general teaming are met. Further implementation details of this approach are presented in [34].

D. Mid-Level Planner

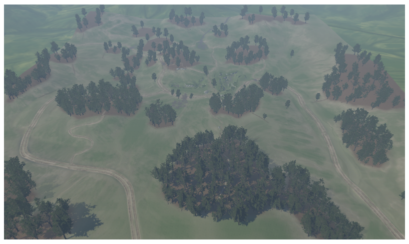
When navigating between two nodes in the graph, we utilize a mid-level planner to generate a minimally visible path to follow. Our mid-level planner queries the graph pruner for the precomputed A* path between the two nodes. The path is truncated based on the robot's current position, then passed to the low level planner.

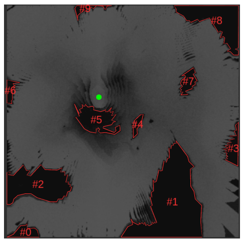
E. Low Level Planner

To avoid obstacles, which may not exist in the global map, and to generate dynamically feasible trajectories, we leverage a local planner. In particular, we use MPPI [32], and design a set of cost-functions to enable following of the minimal-visibility A* paths as well as maintaining formations between vehicles. We not only specify the position in the formation for each vehicle, but we also share receding-horizon trajectories among the team to enable predictive collision avoidance.

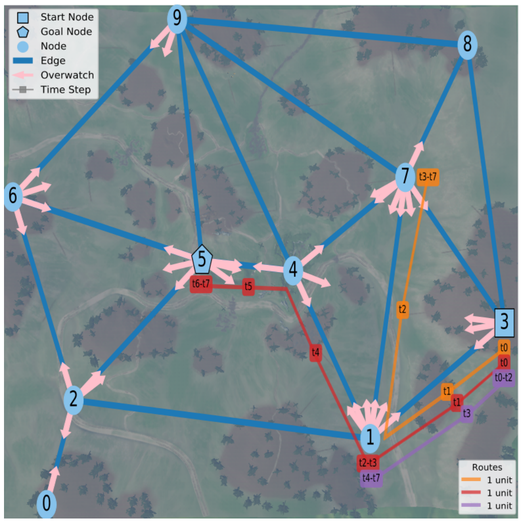
F. Learned Visibility Map

While the A* paths generated from the DEM provide a good approximate path for minimizing visibility, teams will necessarily need to traverse visible areas when traveling between cover regions, or to deviate from the planned path while avoiding previously unknown obstacles or non-traversible terrain. In order to minimize visibility during these periods, we are also developing a learned perception model using image data from the onboard cameras to predict a local visibility map for each robot. This enables the incorporation of information about new sources of cover, such as buildings or vegetation,

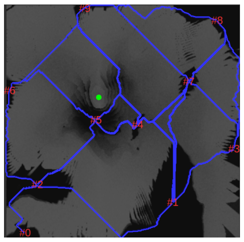

(a) Meadow environment [35]

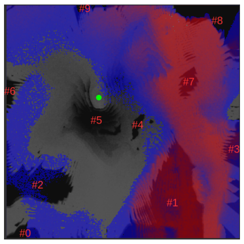


(b) Visibility map



(f) Robot team path solution


(c) Pruned A* paths


(d) Node 1 overwatch visibility

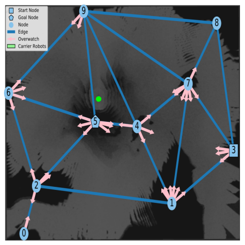
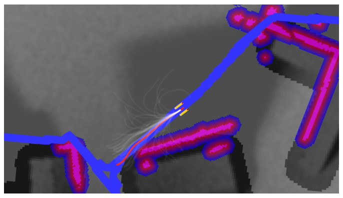
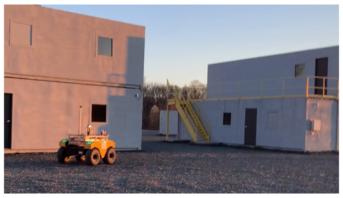

(e) Dynamic topological graph

Fig. 3. Sequential steps in our method for a simulated meadows environment, as seen in (a). In (b), we generate a visibility map to segment the regions of cover (outlined in red). The observer's hilltop location is indicated by the green covariance ellipse. We then generate A* paths between these regions of cover and prune the edges between nodes based on these paths, as seen in blue in (c). For each node we generate visibility maps for overwatch, (d) shows an example for node 1. Using this information, we form our dynamic topological map in (e). Overwatch opportunities are indicated from the overwatch nodes to edges that can be observed with pink arrows. We then use MIP to solve for paths for each robot in the team through this graph, shown in (f). A robot team of 3 starts at node 3 with the objective of at least one agent reaching node 5 while minimizing visibility through overwatch and teaming.

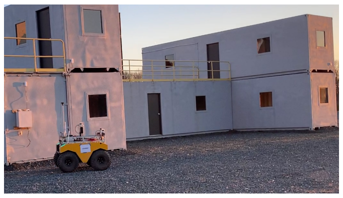

(a) A* Path with LIDAR detections

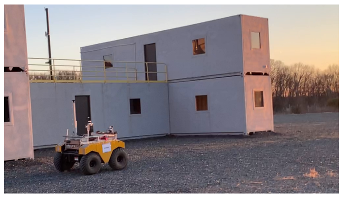

(b) Step 1

(c) Step 2

(d) Step 3

(e) Step 4


(f) Step 5


(g) Step 6

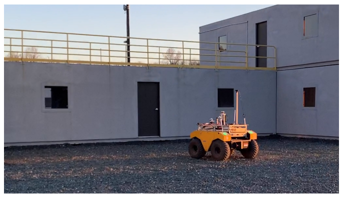

(h) Step 7

Fig. 4. Clearpath Warthog following A* paths to minimize visibility to potential observers while navigating between regions of cover.

into the local planner when such features do not appear in the DEM. We train the local visibility map predictor using virtual camera views from a simulation environment. Virtual cameras are placed in concentric rings facing inwards toward the robot, and the visibility of the robot in each camera is logged during a series of trajectories. We then use the images from four onboard cameras to learn a polar grid map where each cell represents a fixed range of angles and distances from the robot's local coordinate frame and its value holds the estimated visibility from that cell, as seen in Fig. 2.

V. Results and Discussion

We evaluate our approach in both simulation and preliminary hardware experiments. For a large simulated environment ($\approx$ 500m $\times$ 500m), we are able to generate a topological graph from a visibility map and achieve coordinated maneuvers with a team of 3 ground vehicles (see Fig. 3). In preliminary hardware tests, we demonstrated minimum visibility planning with a single autonomous ground vehicle by coupling our high-level graph-based planner with the mid-level A* planner and low-level stochastic model predictive control approach. To do this, we constructed both a visibility map and a traversability map from an *a priori* mesh of the urban environment. Fig. 4 depicts a sequence of steps tracking an A* path in a hardware experiment. Moving forward, we plan to test multiple robots in an off-road environment by leveraging *a priori* terrain elevation data. As demonstrated in our simulation experiments, we hypothesize that our hierarchical planning approach will be able to effectively generate coordinated tactics using *a priori* data while compensating for changing environments with the lower-levels of the autonomy stack. Future research will include receding-horizon graph-based planning to update the visibility map online to account for changing conditions.

Acknowledgment

We gratefully acknowledge the support of the Army Research Laboratory under grant W911NF-22-2-0241.

References

[1] A. Torreño, E. Onaindia, A. Komenda, and M. Štolba, “Cooperative multi-agent planning: A survey,” *ACM Comput. Surv.*, vol. 50, no. 6, Nov 2017.

[2] Z. Yan, N. Jouandeau, and A. A. Cherif, “A survey and analysis of multi-robot coordination,” *International Journal of Advanced Robotic Systems*, vol. 10, no. 12, p. 399, 2013.

[3] J. K. Verma and V. Ranga, “Multi-robot coordination analysis, taxonomy, challenges and future scope,” *Journal of Intelligent & Robotic Systems*, vol. 102, pp. 1–36, 2021.

[4] K. M. Wurm, C. Dornhege, B. Nebel, W. Burgard, and C. Stachniss, “Coordinating heterogeneous teams of robots using temporal symbolic planning,” *Autonomous Robots*, vol. 34, no. 4, pp. 277–294, 2013.

[5] M. Koes, I. Nourbakhsh, and K. Sycara, “Heterogeneous multirobot coordination with spatial and temporal constraints,” in *AAAI*, vol. 5, 2005, pp. 1292–1297.

[6] B. A. Ferreira, T. Petrović, and S. Bogdan, “Distributed mission planning of complex tasks for heterogeneous multi-robot teams,” *arXiv preprint arXiv:2109.10106*, 2021.

[7] M. Limbu, Z. Hu, S. Oughourli, X. Wang, X. Xiao, and D. Shishika, “Team coordination on graphs with state-dependent edge costs,” in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 679–684.

[8] M. Limbu, Y. Zhou, G. Stein, X. Wang, D. Shishika, and X. Xiao, “Team coordination on graphs: Problem, analysis, and algorithms,” *arXiv preprint arXiv:2403.15946*, 2024.

[9] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, “Conflict-based search for optimal multi-agent pathfinding,” *Artificial Intelligence*, vol. 219, pp. 40–66, 2015.

[10] M. Lagoudakis, M. Berhault, S. Koenig, P. Keskinocak, and A. Kleywegt, “Simple auctions with performance guarantees for multi-robot task allocation,” in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 1, Sep 2004, pp. 698–705.

[11] S. Park, Y. D. Zhong, and N. E. Leonard, “Multi-robot task allocation games in dynamically changing environments,” in *2021 IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2021, pp. 8678–8684.

[12] D. Shishika and V. Kumar, “Local-game decomposition for multiplayer perimeter-defense problem,” in *2018 IEEE Conference on Decision and Control (CDC)*, Dec 2018, pp. 2093–2100.

[13] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, “Graph neural networks: A review of methods and applications,” *AI open*, vol. 1, pp. 57–81, 2020.

[14] 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)*, May 2021, pp. 9161–9167.

[15] Y. Liu, W. Wang, Y. Hu, J. Hao, X. Chen, and Y. Gao, “Multi-agent game abstraction via graph attention neural network,” *AAAI Conf. on Artificial Intelligence*, vol. 34, no. 05, pp. 7211–7218, Apr 2020.

[16] L. Zhou and P. Tokekar, “Multi-robot coordination and planning in uncertain and adversarial environments,” *Current Robotics Reports*, vol. 2, pp. 147–157, 2021.

[17] M. Limbu, Z. Hu, X. Wang, D. Shishika, and X. Xiao, “Scaling team coordination on graphs with reinforcement learning,” *arXiv preprint arXiv:2403.05787*, 2024.

[18] N. Mathew, S. L. Smith, and S. L. Waslander, “A graph-based approach to multi-robot rendezvous for recharging in persistent tasks,” in *2013 IEEE Int. Conf. on Robotics and Auto.*, May 2013, pp. 3497–3502.

[19] N. Kamra and N. Ayanian, “A mixed integer programming model for timed deliveries in multirobot systems,” in *2015 IEEE Int. Conf. on Automation Science and Engineering (CASE)*, Aug 2015, pp. 612–617.

[20] M. Koes, I. Nourbakhsh, and K. Sycara, “Heterogeneous multirobot coordination with spatial and temporal constraints,” in *AAAI*, vol. 5, 2005, pp. 1292–1297.

[21] E. F. Flushing, L. M. Gambardella, and G. A. Di Caro, “Simultaneous task allocation, data routing, and transmission scheduling in mobile multi-robot teams,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep 2017, pp. 1861–1868.

[22] J. Yu and S. M. LaValle, “Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics,” *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1163–1177, Oct 2016.

[23] B. A. Asfora, J. Banfi, and M. Campbell, “Mixed-integer linear programming models for multi-robot non-adversarial search,” *IEEE Robotics and Automation Lett.*, vol. 5, no. 4, pp. 6805–6812, Oct 2020.

[24] M. Hamer, L. Widmer, and R. D’andrea, “Fast generation of collision-free trajectories for robot swarms using gpu acceleration,” *IEEE Access*, vol. 7, pp. 6679–6690, 2018.

[25] G. Wagner and H. Choset, “Subdimensional expansion for multirobot path planning,” *Artificial intelligence*, vol. 219, pp. 1–24, 2015.

[26] P. Ogren and N. E. Leonard, “Obstacle avoidance in formation,” in *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, vol. 2. IEEE, 2003, pp. 2492–2497.

[27] X. Liang, H. Wang, Y. H. Liu, W. Chen, and T. Liu, “Formation control of nonholonomic mobile robots without position and velocity measurements,” *IEEE Transactions on Robotics*, vol. 34, no. 2, pp. 434–446, 2018.

[28] B. Woosley, “Efficient simultaneous task and motion planning for multiple mobile robots using task reachability graphs,” Ph.D. dissertation, University of Nebraska at Omaha, 2015.

[29] S. H. Arul, A. J. Sathyamoorthy, S. Patel, M. Otte, H. Xu, M. C. Lin, and D. Manocha, “Lswarm: Efficient collision avoidance for large swarms with coverage constraints in complex urban scenes,” *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3940–3947, 2019.

[30] S. Kumar and S. Chakravorty, “Multi-agent generalized probabilistic roadmaps: Magprm,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 3747–3753.

[31] P. G. Stankiewicz, S. Jenkins, G. E. Mullins, K. C. Wolfe, M. S. Johannes, and J. L. Moore, “A motion planning approach for marsupial robotic systems,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–9.

[32] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, “Aggressive driving with model predictive path integral control,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1433–1440.

[33] J. Wang, G. J. Robinson, and K. White, “Generating viewsheds without using sightlines,” *Photogrammetric Engineering and Remote Sensing*, vol. 66, pp. 87–90, 2000.

[34] C. A. Dimmig, K. C. Wolfe, and J. Moore, “Multi-robot planning on dynamic topological graphs using mixed-integer programming,” *arXiv preprint arXiv:2303.11966*, 2023.

[35] Nature Manufacture, “Meadow - environment set.” [Online]. Available: https://naturemanufacture.com/meadow-environment-set/