Hierarchical Uncertainty-Aware Multi-agent Collaborative Planning System

Yasmin Veys[1*], Harel Biggie[1*], Martina Stadler Kurtz[1*], Samuel Prentice[1*], Jacob Arkin[1], Travis Manderson[1], Akila Saravanan[1], Carlos Nieto-Granda[2], Long Quang[2], Ethan Stump[2], Chuchu Fan[1], Luca Carlone[1], Jonathan How[1], and Nicholas Roy[1]

[1] Massachusetts Institute of Technology, Cambridge, MA, USA,
[2] Army Research Laboratory, USA

Abstract. Teams of robots deployed in large outdoor environments tasked with complex missions, such as surveying or search and rescue, need to collaborate in order to efficiently execute the missions. We present a hierarchical planning system capable of minimizing the time it takes for team members to reach their goals, using route graphs derived from prior overhead views that contain metric-semantic information. The team is tasked via natural language, and we show that our collaborative planner outperforms a non-collaborative baseline across 22 field trials.

Keywords: multi-agent planning, language-grounding, stochastic graphs

1 Introduction

Teams of robots can be used to execute complex missions, such as search and rescue, surveying, and reconnaissance. To succeed in such time-critical missions, it is important that the mission goal is expressible in natural language and that the team collaborates efficiently to minimize the overall execution time, or *team makespan*. Prior information, such as satellite imagery, can be leveraged to estimate terrain and identify areas where the robots may be able to traverse. However, the traversability of some regions cannot be observed from overhead images and is, therefore, uncertain. Consider, for example, the scenario shown in Fig. 1 where a robot team is tasked with navigating through a large outdoor environment. The satellite image shows that there is a bridge over the river where an agent could cross. However, the bridge may be blocked, as seen from the ground observation. Although the bridge's traversability is most relevant to agent A, it is more efficient for agent B to inspect it and relay that information to the team, rather than having agent A travel to the bridge and potentially backtrack to its starting location. In this work, we aim to address both the theoretical and practical challenges of efficient multi-agent decision-making under uncertainty, while providing an interface to define the mission in natural language.

[*] These authors contributed equally.

Acknowledgments: This work was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-17-2-0181.

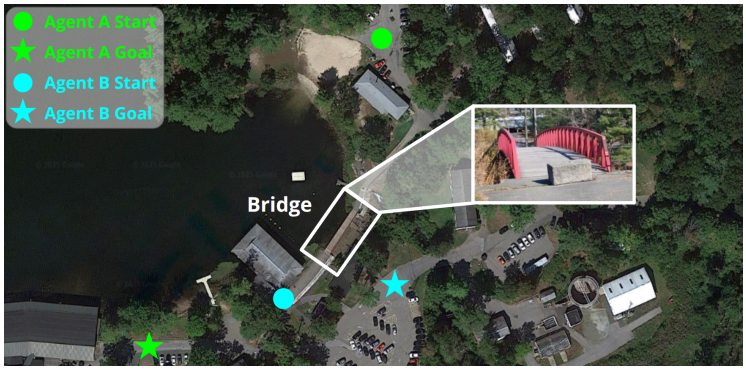


Fig. 1: Agent A needs to cross the bridge, but the traversability is unobservable from an overhead image. Agent B can provide the observation to reduce team makespan.

To generate effective multi-agent plans in the face of real-world environmental uncertainty, we must answer three critical questions: (1) *what* is useful to explore given the team objectives, (2) *when* does the team need the explored information to improve team performance (i.e., mission completion time), and (3) *who* should explore. Answering these questions requires solving a Partially Observable Markov Decision Process (POMDP) with a large action and observation space, long time horizon, and delayed rewards, which is unlikely to be computationally tractable in real-world environments. To improve tractability, we design stochastic graphs that sparsely represent the planning problem, and we ground the mission goals in the graphs via natural language instructions. We also develop efficient algorithms to generate plans that minimize team makespan.

However, our stochastic route graph abstracts away many properties that are needed to represent real-world plan execution, including unmodeled uncertainty, timing delays, and robot navigation failures. To enable the team of agents to execute the abstract plans in the real world, we design a hierarchical planning system that addresses a number of these practical challenges. Our bi-level planning approach uses a higher-level abstract planner to produce collaborative, multi-agent plans and a lower-level agent planner that generates trajectories to carry out the actions from the abstract planner.

We evaluate the ability of our system to ground natural language to planning goals across 100 language instructions with varying degrees of linguistic difficulty. Furthermore, we showcase a real-world deployment spanning 22 field trials of our collaborative, multi-agent planning approach and show it achieves statistically significant faster mission completion times, even while accounting for an increase in planning time.

2 Approach

Consider a team (Λ) of N agents with heterogeneous capabilities, where each agent is tasked with navigating to a 2D metric goal (x, y). A user can provide a

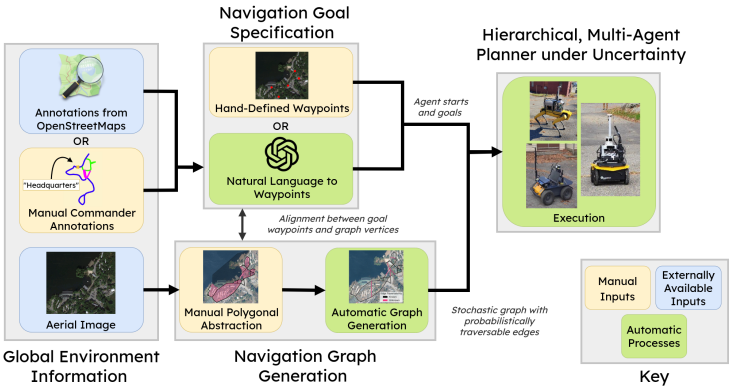


Fig. 2: System diagram of goal generation, graph generation, and planning.

metric goal or semantic goal through a natural language instruction I. During execution, the team's objective is to execute a plan that achieves the mission goals while minimizing overall team makespan. As agents progress towards their navigation goals, they communicate observations about the environment with one another via a centralized base station. Our architecture (Fig. 2) consists of a natural language goal generator, a probabilistic graph generator, and a hierarchical collaborative planner, which enables centralized planning under uncertainty.

2.1 Planning Model Construction

We first describe a method of constructing a planning model, given a prior map of the environment and the natural language instruction I. We assume that I encodes the goals of the mission and that the prior map contains coarse information about semantic regions in the environment.

Constructing Stochastic Route Graphs: To enable long-horizon reasoning about traversability uncertainty and to leverage planners [1] that produce uncertainty-aware, multi-agent navigation policies, we represent the environment as a stochastic graph, where each edge represents a possible path through the environment, annotated with the probabilities that the paths are actually traversable by the different robots in the team. The graph is constructed using the approach in [2] from an overhead image that is decomposed into regions with similar traversability properties (e.g., forests, fields, and roads). The regions that are known to be traversable, such as roads, make up known freespace.

Given a decomposition of the overhead map into semantic regions, we construct a graph G whose vertices $v \in V$ are associated with metric locations (x, y) and whose edges $e \in E$ represent paths between those locations. Each edge is assigned a cost c and a traversability probability τ based on the semantic class of the region it intersects. We assume that the ground truth traversability of an edge is static and can only be observed by an agent who has navigated to a vertex adjacent to that edge. To construct a G that sparsely represents quality

paths in the environment, we must reason about the environmental uncertainty when choosing which edges to include. We begin by constructing a graph that lies entirely in known freespace and then iteratively augment G by adding probabilistic "shortcut" edges that are likely to improve expected plan costs. The resulting G is given as input to the planner described in Section 2.2.

Translating Natural Language Instructions to Metric Goals: To infer the goals in the graph from language, we define a metric-semantic world model $\mathcal{W}_s$ consisting of annotated landmarks with which a language model ($\text{LM}(\cdot)$) implicitly maps phrases ℓ from the instruction I to metric waypoints. Additionally, we assume access to agent ids, $r = [r_\lambda : \forall \lambda \in \Lambda]$, that are used to associate individual goals with specific agents in a team. Given the world model ($\mathcal{W}_s$), instruction (I), and agent ids (r), the language model produces a paired list (g) of agent ids and metric goals: $\text{LM}(\mathcal{W}_s, I, r) \rightarrow [(r_0, (x_0, y_0)), \ldots, (r_N, (x_N, y_N))] = g$.

In practice, we prompt a pre-trained large language model (LLM) [3], [4] to produce g, populating the context with examples, the world model $\mathcal{W}_s$, the agent identifiers r, and the instruction I. These goal pairs are then passed to our collaborative planner described in Section 2.2, which assigns each of the goals to the closest v on G.

2.2 Hierarchical Collaborative Planning

Given the goals g grounded from natural language and the navigation graph G, the planner must produce a plan for each agent such that the overall team makespan is minimized. During online execution, it may be advantageous for agents to sense and share edge traversability information with teammates. However, determining which edges are useful to sense, when the team needs the sensed information, and who should sense is challenging.

Collaborative Multi-agent Planning Under Uncertainty: In [1], we formulate the collaborative planning under uncertainty problem as a POMDP and design multi-agent macro-actions and a value function approximation-based macro-action pruning strategy to increase problem tractability. Reasoning over macro-actions, which are sequences of primitive actions, instead of the primitive actions themselves, improves tractability by reducing planning depth. We consider three macro-actions that capture intuitively useful agent behaviors in a single planning step: (1) navigating via one or more known traversable edges to the goal, (2) navigating to an unknown edge to sense its traversability, and (3) waiting in place for information to be sensed by a teammate.

While reasoning over macro-actions reduces the depth of collaborative multi-agent planning, we still need to consider a large number of macro-actions at each planning step. Many macro-actions (e.g, those that sense edges that are not on low-cost paths to the goal) are unlikely to improve team planning performance. To quickly prune macro-actions that are unlikely to produce high-quality collaborative plans from the plan space, we use macro-action-based value functions. For additional details about the approach, please refer to [1].

Real-World Hierarchical Planning System: We demonstrate in [1] that we can reduce the makespan of the team plan in simulation, but our models

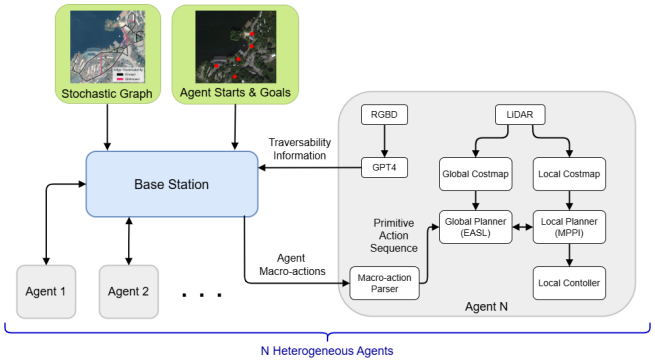


Fig. 3: Hierarchical Multi-Agent Planner

abstract away properties critical to real-world plan execution, including unmodeled uncertainty, timing delays, and robot failures. For real-world execution of these abstract plans, we design a hierarchical planning system, shown in Fig. 3, that reintroduces real-world constraints.

The planner described in the previous section runs at a centralized base station and coordinates communication between the N agents during online execution. At each planning step, the centralized planner assigns a macro-action to each agent, which is executed using the agent's onboard bilevel planner, consisting of a higher-level macro-action planner and a lower-level primitive-action planner. The higher-level planner sequences primitive actions, monitors their outcomes, and has some ability to adjust primitive action goals based on the results. At the lower level, the agent plans and executes trajectories to navigate along an edge, sense the traversability of an edge, or wait in place to receive sensed information. Execution in the real world relies on an autonomy stack that includes GPS-based state estimation, GPS-enabled local mapping, a VLM-based observation function, and a controller capable of executing motor commands.

To navigate along a graph edge, the agent starts at the GPS coordinate that corresponds to its current vertex in the graph. Its navigation goal is set to the GPS coordinate of the target vertex, with some positional tolerance. We use EASL [5] to generate waypoints to the goal and then use MPPI [6] to generate motor commands to follow the EASL trajectory. The agent leverages its local occupancy map to avoid obstacles. We use GPT-4o [3] to sense the traversability of an edge, given an image taken from the agent's front-facing camera and a prompt that asks if the path is navigable. This VLM-based observation function is more effective than one that relies solely on the geometry of the environment, as it accounts for additional semantic cues that may indicate untraversability (e.g., rocks or tree roots). If it is beneficial for an agent to wait for sensed information, it pauses until the centralized planner provides that information.

While this approach improves local navigation, imperfect planning abstractions can still lead to failures across different levels of the planning hierarchy.

To prevent primitive waypoint-based navigation failures, the higher-level macro-action planner re-attempts four similar (augmented by a 0.5m in the x or y direction) primitive navigation actions upon failure, rather than directly marking the macro-action as failed and reporting the failure to the centralized base planner. To coordinate agents executing variable-duration macro-actions, such as traversing edges with different weights, we implement an interrupt-based strategy. When an agent completes a macro-action, it sends the result (success or failure, current state, and observations) to the centralized planner. The planner then issues interrupts to all other agents, prompting them to safely terminate their current macro-action. Once all agents have safely terminated, the centralized planner replans and sends new macro-actions.

3 Experiments

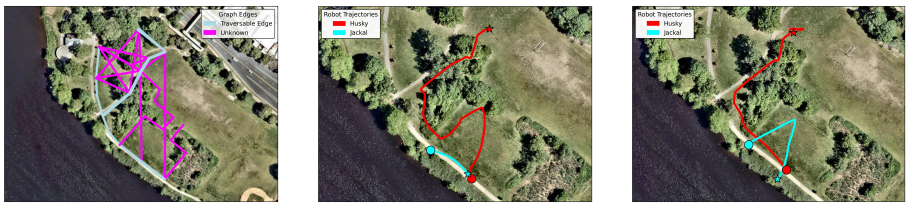
In this section, we describe the experiments to evaluate the proposed hierarchical planning system. We demonstrate the ability to use the algorithms presented in [1] and [2] to execute complex, large-scale, multi-agent missions in the real world. We provide results for language grounding experiments and three real-world planning scenarios at different testing sites, Magazine Beach (Fig. 4), Cambridge, MA (MB-1 and MB-2, 2-agent), and Camp Buckner (Fig. 5, West Point, NY (CB-1, 3-agent).

3.1 Heterogeneous Robot Team

We demonstrate our approach using a heterogeneous team of robots containing a Clearpath Jackal, a Clearpath Husky, and a Boston Dynamics Spot (added for CB-1). Of the three robots, the Spot is the fastest and can navigate over rough and steep terrain. The Jackal is the faster of the two wheeled robots, but can only navigate on flat surfaces, like paved roads. The Husky is the slowest of the three agents, and can reliably navigate on paved roads and some rough terrain. Each robot is equipped with an Oak-D or Intel RealSense RGBD camera and a Velodyne or Ouster lidar for mapping. Localization is performed using a combination of LiDAR SLAM and GPS. The robots communicate with the centralized base station, which runs the centralized multi-agent planner described in Section 2.2, via a network of Silvus mesh radios.

3.2 Collaborative Planner vs. Noncollaborative Baseline

In this section, we describe the three different scenarios, MB-1, MB-2, and CB-3, and discuss the performance of our collaborative multi-agent planner in comparison to a noncollaborative baseline. In the noncollaborative setting, the agents are still able to communicate sensed information with each other, but no agent will actively divert from its own optimal path to collect information for another agent. Agents only passively sense the uncertain edges along their path to the goal to maximize the chance that useful information is shared among the team.



(a) The generated route graph with the uncertain regions across forested areas shown in pink.

(b) The non-collab. trajectory where the Husky senses the forest edge for itself.

(c) The collab. trajectory where the Jackal senses the forest edge for the Husky, allowing it go directly.

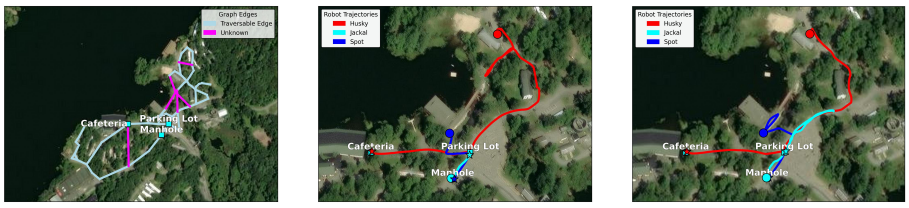
Fig. 4: MB 2-agent planning scenario with goals (stars) and start points (circles).

In the collaborative setting, agents must actively sense uncertain edges if it is deemed beneficial for the team.

MB-1 and MB-2: For the two scenarios at MB, we deploy a 2-agent team consisting of the Jackal and the Husky. In MB-1, the two robots start in the same region, as shown in Fig. 4, and the Husky is tasked with navigating towards the upper left corner of the park. The most direct route from the Husky's start to its goal goes through a forested region that may or may not be traversable. Since the Jackal moves faster than the Husky, it is beneficial for the Jackal to sense the edge through the forested region rather than the Husky. In MB-2, the edge through the forest, which is valuable to sense, remains the same, but the Husky and Jackal start on opposite sides of the forest. This scenario is interesting, because it tests the ability of our planner to model and use global information in situations where the robots may not have line-of-sight.

CB-1: We expand the use of global information in CB-1 and extend the scenario to include 3 agents: the Jackal, the Husky, and the Spot. All of the robots start at different non-line-of-sight locations, as shown in Fig. 5, and the slowest agent, the Husky, is tasked with traversing over 325 meters to its goal on the left. In this scenario, the satellite image shows a forested region and a bridge that may or may not be traversable. Shortcuts through either of these regions would enable the Husky to make progress more quickly towards its goal. However, sensing those edges is much more costly for the Husky than for the Spot or Jackal, since it is the slowest agent.

Results: Across all scenarios, we see that the faster robots divert from their own paths to sense information for the Husky in the collaborative setting. In the noncollaborative setting, all agents proceed to their individual goals, and all edges that are valuable for the Husky to sense (e.g., the forest edge in MB-1 and MB-2 and the bridge edge in CB-1) are sensed by the Husky itself. In CB-1, both of the faster agents, the Jackal and the Spot, divert from their optimal paths to collect information for the Husky. Since the Spot starts near the bridge, it senses that the bridge is untraversable before navigating to sense the forest edge closest to it. Meanwhile, the Jackal travels to sense the farther forest edge.



(a) The generated route graph with the uncertain regions near the bridge/forest area shown in pink.

(b) The non-collab. trajectory where the Husky has to sense the uncertain regions for itself.

(c) Collab. trajectory where the Jackal and Spot sense the edges for the Husky, allowing it to go directly.

Fig. 5: WP 3-agent planning scenario with goals (stars) and start points (circles).

We present a summary of the quantitative results from our 22 field trials in Table 1. In all cases, the execution time is reduced when the agents collaborate to reach their respective goals. For MB-1 and MB-2, our results demonstrate that our collaborative planner reduced team execution times by 13.3% and 24.6% as compared to the non-collaborative baseline, respectively. In the 3-agent case (CB-1), our execution time is reduced by 16.41%. The language command I issued for this scenario was "Husky, go to the cafeteria. Jackal, go to the parking lot. Enyo, go to the manhole", where Enyo was the name of the Spot. We also examine the total trial times (execution + planning) in relation to the planning times. In MB-1, the collaborative planner generated plans 7.8x more slowly than the non-collaborative planner, and resulted in a 0.5% increase in total trial times as compared to the baseline. However, in more complicated scenarios, MB-2 and CB-1, where the robots do not start within line of sight of each other, the collaborative planner outperforms in total trial times with a 3.5% decrease in total trial times, and a 12.2% decrease in total trial times as compared to the non-collaborative baseline.

	Non-Collaborative			Collaborative		
Scenario	Trial Times	Exec. Times	Plan. Times	Trial Times	Exec. Times	Plan. Times
WP (West Point 3-agent), 3 trials						
Average (s)	640.18	635.68	**4.5**	**561.58**	**531.35**	30.23
STD (s)	32.77	32.84	0.15	6.0	6.68	0.68
SEM (s)	23.17	23.22	0.1	4.24	4.72	0.48
MB-1 (Magazine Beach, 2-agent Scenario 1), 5 trials						
Average (s)	**261.9**	256.74	**5.16**	263.18	**222.51**	40.66
STD (s)	20.14	20.16	0.09	12.23	11.34	6.24
SEM (s)	10.07	10.08	0.04	6.12	5.67	3.12
MB-2 (Magazine Beach 2-agent, Scenario 2), 3 trials						
Average (s)	308.7	303.1	**5.6**	**298.0**	**228.4**	69.6
STD (s)	6.01	5.94	0.14	5.25	4.79	0.93
SEM (s)	4.3	4.2	0.1	3.71	3.38	0.66

Table 1: Planning results for real-world scenarios. Note: trial counts are for each of the non-collaborative and collaborative scenarios.

	1 Robot			2 Robots			3 Robots				Overall	Avg.
Model	**DL**	**SR**	**Syn.**	**DL**	**SR**	**Syn.**	**DL**	**SR**	**Syn.**	**U**	**Acc.**	**Time (s)**
CL-3.5-Sonnet [7]	100	50	100	90	50	100	70	50	100	70	78	2.76
CL-3.5-Haiku [7]	100	70	100	90	50	100	70	50	90	80	80	2.99
CL-3.7-Sonnet [7]	100	80	100	90	70	100	70	70	100	80	**86**	4.67
gpt-4o-mini [3]	100	10	100	90	0	80	70	0	80	60	59	4.28
gpt-4o [3]	100	50	100	90	50	100	70	10	100	60	73	1.41
Models run locally with 4bit quantization on a RTX 4090												
llama3.2-3B [8]	70	0	60	60	0	20	30	0	10	10	26	0.19
llama3.1-8B [8]	100	10	100	80	20	60	70	10	50	40	54	2.15
Gemma3-1B [4]	0	0	0	0	0	0	0	0	0	0	0	0.18
Gemma3-4B [4]	100	10	100	80	0	80	60	0	60	60	55	0.35
Gemma3-27B [4]	100	20	100	90	10	100	70	20	100	70	68	1.40
Qwen3-0.6B [9]	0	0	0	10	0	0	0	0	0	0	1	0.15
Qwen3-32B [9]	100	40	100	90	30	100	70	10	90	70	70	25.84

Table 2: Language Grounding Results. Values are in percentages, each category had 10 instructions.

3.3 Language Grounding

In addition to the language commands evaluated in the field, we conduct an offline study on grounding language to metric-semantic goals. Specifically, we generate a set of 100 language commands categorized by linguistic feature. Direct landmark instructions (DL) specify goals with direct reference to a specific semantic location in the map; we also evaluate instructions with synonyms (Syn.). Spatial relation instructions (SR) specify goals via spatial language, such as "go between". Under-specified instructions (U) do not include a specific agent assignment, requiring the model to assign the goals to agents. We group the instructions by the number of agents (λ) in the team. Table 2 shows a summary of the language results. The timing results for Anthropic Claude) (CL) and OpenAI (GPT) models refer to the API response time. All other models were evaluated locally using 4-bit quantized weights on an NVIDIA RTX 4090 via the *Ollama*[1] and `llama.cpp`[2] libraries.

The CL family of models is the most successful at the grounding task, with CL 3.7 achieving an 86% success rate. Across all models, SR instructions were challenging, especially when the models were tasked with creating goals for the full team. The Gemma3 family of models shows promise for grounding instructions locally on a robot platform with 68% success on the 27B parameter model.

4 Related Work

Multi-Agent Collaborative Planning: We model the problem of multi-agent navigation in unknown environments as a variant of the Canadian Traveller's Problem (CTP), which has been used as a tool to study the complexity of navigation on graphs with stochastic edges [10]–[13]. Other approaches have been developed for graph-based, risk-aware single-agent navigation under edge

[1] https://github.com/ollama/ollama

[2] https://github.com/ggml-org/llama.cpp

cost uncertainty, including stochastic graph generation from overhead imagery [14], and online planning with uncertain edge costs [15], [16]. Most recently, we developed an approach for generating high-quality multi-agent policies for CTPs that minimize team makespans for heterogeneous robot teams [1].

Constructing CTP Graphs: In this work, we proposed using a weighted, stochastic graph (i.e., a CTP graph) to represent the costs and traversability probabilities of different paths in the environment. Many other approaches have also used CTP graphs [17]–[19] to enable high-quality and robust planning under uncertainty. However, many of these approaches either use hand-designed graphs, randomly generated graphs, or graphs built from existing maps of the environment. Hand-designing these graphs is tedious, so we employ a method of automatically generating high-quality CTP graphs from prior information in this work. The approach that is most similar to ours is [20], which constructs Partial Covering Canadian Traveller Problem graphs from satellite images and also prunes stochastic edges that will not improve expected plan costs.

Language Grounding: There is a rich history of using natural language to command robots [21]. One formulation treats the problem as a semantic parsing or translation problem in which a language model maps an instruction to a structured symbolic representation that is congruent with an existing planner, such as lambda calculus [22], [23], temporal logic [24], planning domain definition language [25], or motion planning constraints [26], among others. More recently, an alternative approach has been to use LLMs or VLMs to directly map to action sequences [27], [28]. Our approach aligns more closely with the former in that we rely on an existing planner; however, we map to continuous coordinates as goals rather than an existing symbolic structure, perhaps most similarly to work that maps to non-symbolic motion planning constraints [29].

5 Conclusion

We present a hierarchical, uncertainty-aware multi-agent collaborative planning system that enables heterogeneous teams of robots to execute complex missions in large, outdoor environments. The system combines route graphs generated from overhead imagery, a natural language interface that grounds instructions to metric-semantic goals, and a collaborative planner that reasons over macro-actions to generate plans that minimize team makespan. A bilevel execution architecture allows abstract plans to be carried out in the real world, addressing challenges such as timing delays, unmodeled uncertainty, and robot navigation failures. We show that we can achieve an 86% grounding to goal success rate spanning 100 language instructions. Across 22 real-world trials, our approach consistently reduced execution time in comparison to a non-collaborative baseline, even when accounting for increased planning time. In future work, we would like to further improve our ability to handle mismatches between our planning abstractions and the real world. Updating our planning abstractions online in response to local sensor data, for example, would result in more efficient task execution, more efficient planning, and fewer catastrophic failures.

References

[1] M. Stadler, J. Banfi, and N. Roy, "Approximating the value of collaborative team actions for efficient multiagent navigation in uncertain graphs," in *International Conference on Automated Planning and Scheduling*, 2023.

[2] Y. Veys, M. S. Kurtz, and N. Roy, "Generating sparse probabilistic graphs for efficient planning in uncertain environments," in *IEEE International Conference on Robotics and Automation*, 2024.

[3] J. Achiam, S. Adler, S. Agarwal, *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[4] G. Team, A. Kamath, J. Ferret, *et al.*, "Gemma 3 technical report," *arXiv preprint arXiv:2503.19786*, 2025.

[5] B. Hedegaard, E. Fahnestock, J. Arkin, A. Menon, and T. M. Howard, "Discrete optimization of adaptive state lattices for iterative motion planning on unmanned ground vehicles," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021.

[6] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *IEEE International Conference on Robotics and Automation*, 2016.

[7] Anthropic, "The Claude 3 model family: Opus, Sonnet, Haiku," Anthropic, Tech. Rep., Mar. 2024, Accessed on June 4, 2025. [Online]. Available: https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf.

[8] A. Grattafiori, A. Dubey, A. Jauhri, *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[9] A. Yang, A. Li, B. Yang, *et al.*, "Qwen3 technical report," *arXiv preprint arXiv:2505.09388*, 2025.

[10] C. H. Papadimitriou and M. Yannakakis, "Shortest paths without a map," *Theoretical Computer Science*, vol. 84, no. 1, 1991.

[11] E. Nikolova and D. R. Karger, "Route planning under uncertainty: The canadian traveller problem.," in *AAAI Conference on Artificial Intelligence*, 2008.

[12] Z. Bnaya, A. Felner, and S. E. Shimony, "Canadian traveler problem with remote sensing.," in *International Joint Conference on Artificial Intelligence*, 2009.

[13] P. Eyerich, T. Keller, and M. Helmert, "High-quality policies for the canadian traveler's problem," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 24, 2010.

[14] L. Murphy and P. Newman, "Risky planning on probabilistic costmaps for path planning in outdoor environments," *IEEE Transactions on Robotics*, vol. 29, no. 2, 2012.

[15] D. Dey, A. Kolobov, R. Caruana, E. Kamar, E. Horvitz, and A. Kapoor, "Gauss meets canadian traveler: Shortest-path problems with correlated natural dynamics," in *International Conference on Autonomous Agents and Multi-agent Systems*, 2014.

[16] J. J. Chung, A. J. Smith, R. Skeele, and G. A. Hollinger, "Risk-aware graph search with dynamic edge cost discovery," *The International Journal of Robotics Research*, vol. 38, no. 2-3, 2019.

[17] M. Kneebone and R. Dearden, "Navigation planning in probabilistic roadmaps with uncertainty," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 19, 2009.

[18] J. Guzzi, R. O. Chavez-Garcia, L. M. Gambardella, and A. Giusti, "On the impact of uncertainty for path planning," in *Proceedings of the International Conference on Robotics and Automation*, 2019.

[19] H. Guo and T. D. Barfoot, "The robust canadian traveler problem applied to robot routing," in *Proceedings of the International Conference on Robotics and Automation*, 2019.

[20] Y. Huang, H. Dugmag, T. D. Barfoot, and F. Shkurti, "Stochastic planning for asv navigation using satellite images," in *Proceedings of the International Conference on Robotics and Automation*, 2023.

[21] S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek, "Robots that use language," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, 2020.

[22] Y. Artzi and L. Zettlemoyer, "Weakly supervised learning of semantic parsers for mapping instructions to actions," *Transactions of the association for computational linguistics*, vol. 1, 2013.

[23] J. Dzifcak, M. Scheutz, C. Baral, and P. Schermerhorn, "What to do and how to do it: Translating natural language directives into temporal and dynamic logic representation for goal management and action execution," in *IEEE International Conference on Robotics and Automation*, 2009.

[24] Y. Chen, J. Arkin, C. Dawson, Y. Zhang, N. Roy, and C. Fan, "Autotamp: Autoregressive task and motion planning with llms as translators and checkers," in *2024 IEEE International conference on robotics and automation (ICRA)*, IEEE, 2024.

[25] Y. Xie, C. Yu, T. Zhu, J. Bai, Z. Gong, and H. Soh, "Translating natural language to planning goals with large-language models," *arXiv preprint arXiv:2302.05128*, 2023.

[26] T. M. Howard, S. Tellex, and N. Roy, "A natural language planner interface for mobile manipulators," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2014.

[27] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *International Conference on Machine Learning*, PMLR, 2022.

[28] M. Ahn, A. Brohan, N. Brown, *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[29] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei, "Voxposer: Composable 3d value maps for robotic manipulation with language models," *arXiv preprint arXiv:2307.05973*, 2023.