Reactive Constraint Relaxation for Urban Environment Navigation

Jinwoo Kim[1*], Keonyoung Koh[1*], Samuel Seungsup Lee[12], Yohan Park[1]
Daehyung Park[1**]

[1] Korea Advanced Institute of Science and Technology,
Yuseong-gu Daehak-ro 291,
Daejeon 34141, South Korea
[2] Seoul National University, Gwanak-gu Gwanak-ro 1,
Seoul 08826, South Korea

Abstract. This paper aims to develop an efficient and safe navigation framework under relaxable constraints. However, constraint-based navigation planning is challenging since the structure of the navigation environment and the existence of relaxable constraints are difficult to predict in advance. Furthermore, constraint relaxation priority varies depending on operation and task conditions. To address this, we introduce a novel reactive constraint-relaxation-and-planning method that autonomously identifies a constrained region to violate and redetermines a navigation plan that balances safety and efficiency in real time. We validate our approach through both quantitative and qualitative analysis using CARLA simulations, achieving a navigation success rate of 95% and reducing the navigation distance in the constraint region by 60% compared to scenarios with no constraint relaxation.

Keywords: safety-aware navigation, spatial constraints, constraint relaxation

1 Introduction

Advances in navigation platforms have significantly increased the prevalence of robots with enhanced mobility in urban environments [1–4]. The goal is to build an urban navigation planning framework that enables a robot to find the human-like safe but efficient path in complex environments. However, the diverse nature of urban areas raises the challenge of selecting a safe path while minimizing navigation costs since the environment structure is not known in advance.

As a solution, researchers often investigate reactive task-and-motion planning (TAMP) methods that find a task-wise complete and path-wise optimal plan in an iterative manner. For example, Li et al. introduce a reactive TAMP framework that leverages linear temporal logic (LTL) to find a sequence of action commands

* These authors contributed equally to this work
** Corresponding author: daehyung@kaist.ac.kr

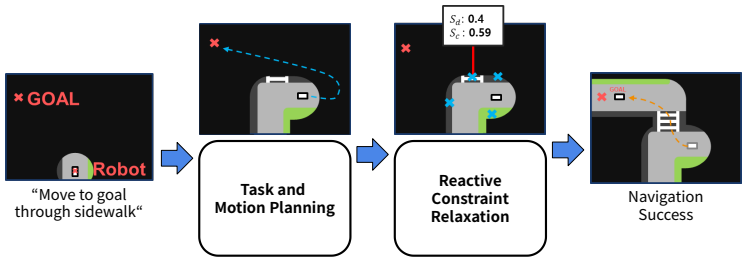


Fig. 1: Overall framework of the constraint relaxation-based navigation method.

satisfying logical specifications, given environmental changes [5]. However, real-world applications often fail to find a complete solution without passing through dangerous regions.

To resolve this issue, conventional studies predefine the priorities of constraint regions and suggested methods to reach destinations by selectively violating lower-priority constraints in unavoidable situations [6–10]. However, the prioritization of constraints is nontrivial, requiring a comprehensive understanding of both the environment and task goals.

To address these issues, we introduce a novel reactive constraint-relaxation-and-planning method that autonomously identifies a constrained region to temporally relax, enabling online replanning that balances safety and efficiency online. Our method builds on an LTL-based TAMP framework [5] to find a complete and optimal path within the current map. We improve the method by converting constrained regions into traversable areas, selecting the region with the highest relaxation score, when a constrained area significantly impedes efficient goal reaching, as illustrated in Fig. 1. Further, we propose a novel composite relaxation score based on distance and commonsense measures derived from large-scale language dataset. This score guides the generation of a relaxed path plan to enable effective, human-like navigation. We evaluate the safety and completeness of the proposed navigation method both qualitatively and quantitatively using an open-source simulator for autonomous urban driving.

2 Related Work

2.1 Path Planning-Based Safety-Aware Navigation

Traditional planning methods often utilize cost-based search algorithms (e.g., Dijkstra or A* algorithms) to safely navigate around areas. However, accurately modeling such costs is challenging, particulary when environmental information is sparse. This significanly lowers the completeness of planning approach in real world. Alternatively, recent studies often estimate the traversability of regions, including risky areas (e.g., stairs, narrow aisles, and grass), to enable more robust path planning. To assess traversability, these studies leverage various sensor data [11–13] or adopt vision-based self-supervised learning techniques [14, 15].

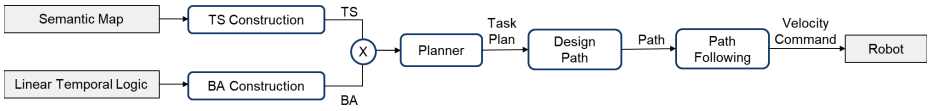


Fig. 2: Path planning block diagram. Given semantic map and linear temporal logic (LTL) inputs, the high-level planner devises a task plan, and the low-level path and motion planner generates a velocity command.

However, the difficulty in intuitively understanding the computed traversability limits its application to urban environments with frequent unpredicted behavior. In addition, traversable areas may require additional safety consideration based on social conventions.

In this paper, we also evaluate the navigation risk of constrained areas by a commonsense-based score to consider semantic safeties as well as physical risks, and express the evaluation result in formal language as a task command.

2.2 Task Planning-Based Safety-Aware Navigation

Task planning-based research handles hazardous areas by defining constraints using high-level formal language for intuitive representation and constraint avoidance [16–19]. If necessary, violation or dynamic modification of formal language-given constraints can help to achieve the task goal [20].

First, the minimum constraint violation method finds a path that violates as few high-priority constraints as possible. This method defines the constraint priority as a dictionary expression [9, 10], cost [7], or hard-soft constraint [6, 8] to find the optimal path that violates the least number of constraints. As defining constraint prioritization requires prior environmental understanding, existing work utilizes predefined constraints and priorities from various regulations.

Next, constraint modification methods minimally adjusts constraints at the task level to generate a feasible plan by considering combinations of constraints to find a new feasible constraint set that enables the execution [21–23]. Unlike violation, constraint modification does not require the prioritization of constraints to be defined in advance, but instead requires a computationally intensive combinatorial consideration of constraints.

In this paper, we take the advantage of formal language to define constraints, and propose a reactive constraint-relaxation methodology that relaxes constraint regions by modifying the formal language-given task command. We evaluate the constraint regions using distance-based and commonsense-based scores.

3 Path Planning Methodology

We employ a path planning method to obtain movement commands that satisfy sequential tasks and constraints, as illustrated in Fig. 2. This method proceeds through three stages: (1) semantic map and linear temporal logic (LTL) input,

(2) their compression to transition system (TS) and Büchi automaton (BA), and (3) product automaton (PA)-based path planning.

3.1 Inputs

Semantic Maps: A semantic map represents a meaningful structure and feature of the environment recognizing ground labels and obstacles from sensors. The map also depicts areas in which constraint relaxation occurs according to user commands.

Linear Temporal Logic (LTL): LTL is a logic that models time as an infinitely expandable sequence of states, and is used in this paper to express sequential tasks and constraints, with formula syntax as:

$$\varphi = \pi \,|\, \neg\varphi \,|\, (\varphi_1 || \varphi_2) \,|\, X\varphi \,|\, (\varphi_1 U \varphi_2), \tag{1}$$

where $\pi \in U$ is an atomic proposition, and temporal operators & (conjunction) and F (finally) are derived from logic operators [24].

3.2 Compression Graphs

Transition System (TS): As in Definition 1 [5], transition system is a graph that abstracts the semantic map's various regions, their interconnections, and the robot's position among them.

Definition 1. ***(Transition System)*** *The transition system is defined as* $\mathrm{TS} = (S, s_0, A, \delta_t, \Pi, L)$*, where* S *is a finite state set* ($s \in S$)*,* s_0 *the initial state,* A *the action space,* $\delta_t : S \times A \to S$ *the transition between states,* Π *the atomic proposition set, and* $L : S \to 2^{\Pi}$ *the labeling function for states where the atomic proposition is true (details are in [5]).*

Fig. 3: Transition system formulation process. A region growing algorithms clusters regions with identical labels. A transition system then represents the connections.

As in Fig. 3, a region growing algorithms clusters regions with identical labels. Initially positioned in region r_5 with state s_0, the robot triggers action A upon transitioning between regions. The labeling function L ensures the state truth of the robot's current region.

Büchi Automaton (BA): The Büchi automaton (Definition 2 [5]) represents the sequential tasks and constraints of linear temporal logic as a graph. A robotic system determines whether its given commands are satisfied with Büchi automata [25].

Definition 2. ***(Büchi Automaton)*** BA $= (Q_b, q_{b,0}, \Sigma, \delta_b, F_b)$*, where* Q_b *is the finite state set(*$q_b \in Q_b$*),* $q_{b,0}$ *the initial state,* $\Sigma = 2^{\Pi}$ *the LTL symbol set,* $\delta_b : Q_b \times \Sigma \to 2^{Q_b}$ *the transition between states, and* $F_b \subseteq Q_b$ *the accept state set (details are in [5]).*

Traversal of the regions associated with an edge allows movement between nodes, where an edge labeled with 1 denotes all possible regions. Failure to traverse an appropriate region results in a constraint violation of the Büchi automaton.

3.3 Planning

Product Automaton (PA): The product automaton is a composite graph formed from the transition system and the Büchi automaton, defined as a tuple in Definition 3 [5].

Definition 3. ***(Product Automaton)*** PA $=$ TS $\times$ BA $= (Q_p, q_{p,0}, \Sigma, \delta_p, F_p)$*,* $Q_p = S \times Q_b (q_p = (s, q_b) \in Q_p)$*, where* $q_{p,0} = (s_0, q_{b,0})$ *is the initial state,* $\Sigma = 2^{\Pi}$ *the LTL symbol set,* $\delta_p : Q_p \to Q_p$ *the connectivity between states, and* $F_p \subseteq Q_p$ *the set of accept states (details are in [5]).*

The combination of the transition system and the Büchi automaton allows simultaneous consideration of the environment and tasks. Through graph exploration from the initial state $(s_0, q_{b,0})$ to $(s_F, q_{b,F})$ where $q_{b,F} \in F_b$, the robot identifies a task plan $\xi = \{(s_i, q_{b,i})\}_{i=0,1,\ldots,F}$ that satisfies the command φ.

Path Planning: Through path planning, the robot executes its given task for each region or goal. First, the robot creates a cost map by inflating all regions and obstacles on the semantic map to the robot size. The robot then employs a global search algorithm [26] to plan the path from the robot's current position to the goal, and a local search algorithm [27] to traverse the path while outputting velocity commands for avoiding constrained regions and obstacles.

4 Reactive Constraint Relaxation Methodology

4.1 Overview

The path planning method includes a reactive constraint-relaxation process, indicated by the red line in Fig. 4. The method proceeds through three stages: (1) assessing task difficulty, (2) scoring constraint regions, and (3) relaxing constraints and re-planning.

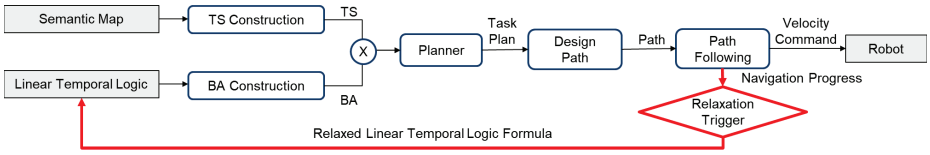


Fig. 4: Reactive constraint relaxation methodology block diagram.

4.2 Constraint Relaxation Threshold

Constraint relaxation occurs when goal-reaching is difficult under the existing constraints. The method determines task difficulty by whether the difference between the current distance d_{robot} to the goal and the minimum Manhattan distance to the goal d_{min} exceeds a threshold,

$$d_{robot} > d_{min} + threshold. \tag{2}$$

4.3 Score Evaluation

Distance: We use the Manhattan distance $d_{constraint}$ from the constraint region center to the goal to account for urban environment features such as buildings and roads. As in Eq. (3), the distance score S_d,

$$S_d = 1 - \frac{d_{constraint}}{2l}, \tag{3}$$

has a value between 0 and 1 based on $d_{constraint}$ and the map size l. The distance score is proportional to the negative of the distance $d_{constraint}$, indicating how useful it was in reaching the goal.

Commonsense Knowledge: We assign commonsense-based scores to constraints using word similarities from the large-scale language dataset ConceptNet Numberbatch [28], employed by former studies for commonsense scoring [29].

After finding the word similarity between constraint labels and the roads, we normalize the similarity values. As in Eq. (4), the commonsense score S_c,

$$S_c = \frac{w_{sidewalk} - w_{road} + 1}{2}. \tag{4}$$

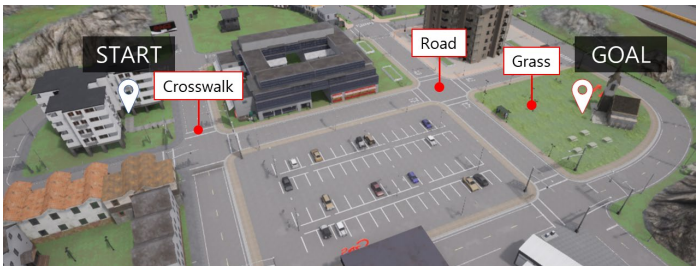
has a value between 0 and 1 based on the word similarity $w_{sidewalk}$ to sidewalks and w_{road} to roads. The commonsense scores for crosswalks, grass, and roads were 0.59, 0.44, and 0.20, respectively.

4.4 Constraint Relaxation Method

We relax the constraint with the highest score to establish a new path plan through the constrained region. After scoring the constraints with S_d and S_c, we find the weighted sum S_s,

$$S_s = w * S_d + S_c. \tag{5}$$

We modify the LTL to allow passage through the constraint region with the highest weighted sum and reconstruct the Büchi automaton and product automaton according to the modified LTL. Finally, based on the reconstructed product automaton, we perform graph exploration to establish a new task plan that passes through the relaxed constraint region.



(a) Qualitative evaluation environment

(b) Village environment

(c) City environment

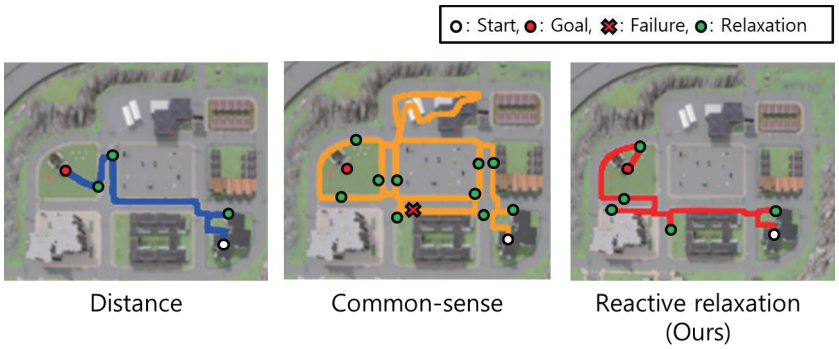
Fig. 5: Evaluation environments. (a) Qualitative evaluation environment with crosswalk, road, and lawn constraints. (b) Quantitative evaluation village environment. (c) Quantitative evaluation downtown environment.

5 Performance Evaluation

5.1 Experiment Settings

Simulation environment: As in Fig. 5, we conducted qualitative and quantitative evaluations for a navigation task in a 3D CARLA simulation environment [30]. The initial constraint was to travel on sidewalks only for all evaluation settings.

(a) Qualitative comparison of constraint-relaxation methods



(b) Qualitative compairson of scoring methods

Fig. 6: Comparison results with (a): other constraint-relaxation methods and (b): scoring methods.

Navigation system: We evaluated the reactive constraint-relaxation method's performance using a pedestrian agent with the CARLA simulator. The agent used semantic LiDAR to acquire a semantic point cloud for the surrounding environment, then classified the ground into four labels: sidewalk, road, crosswalk, and grass. The agent then created a combined map from the semantic and metric maps.

Reactive constraint-relaxation method parameters: We set the threshold value for Eq. (2) to 10 m, and the constraint score weight w to 3.887, indicating that the grass constraint was to be relaxed if no crosswalks are within 30 m of the agent.

5.2 Qualitative Evaluation

Given the command to travel from the apartment to the church along the sidewalk, the reactive constraint-relaxation method relaxed the crosswalk constraint 4 times and the grass constraint once. (Video link: https://youtu.be/Gpro85V-zBk)

Table 1: Quantitative evaluation results. The best and second-to-best results are in red and blue, respectively.

Env.	Method	Success Rate (%)	Constraint Violation Distance (m)			
			Road	Crosswalk	Grass	Total
Village	Random relaxation	91	66.43	18.46	85.05	169.94
	Full relaxation	96	33.27	3.12	72.22	108.62
	Zero relaxation	10	0.01	0.00	0.08	0.09
	Reactive relaxation (Ours)	95	12.53	22.81	76.15	111.49
Downtown	Random relaxation	90	127.93	26.68	-	154.61
	Full relaxation	95	63.06	4.24	-	67.30
	Zero relaxation	22	0.00	0.00	-	0.00
	Reactive relaxation (Ours)	95	20.10	41.85	-	61.94

Comparisons: The reactive constraint-relaxation method was compared with zero-relaxation and full-relaxation methods in Fig. 6. While the zero-relaxation method failed to reach the goal and the full-relaxation method took a dangerous road-crossing path, our method reached the goal by relaxing safe constraints such as crosswalks and grass. Similarly, distance-only scoring took a dangerous road-crossing path and commonsense-only scoring failed to find constraints to relax to reach the goal, indicating both distance and commonsense scores must be considered when relaxing constraints.

5.3 Quantitative Evaluation

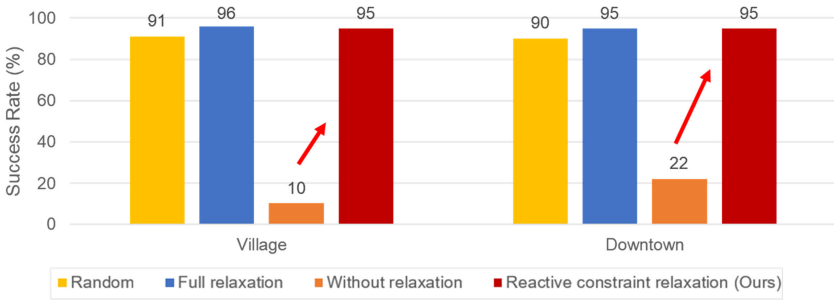
The results of the quantitative evaluation are shown in Table 1.

Comparisons: Fig. 7 compares the navigation success rate and the distance traveled in constraint-violated areas for the reactive constraint-relaxation method with those of random-relaxation and full-relaxation methods. In Fig. 7-(b), the left-hand chart graphs the distance for all violated constraints, while the right-hand chart graphs the distance for violated road constraints only.

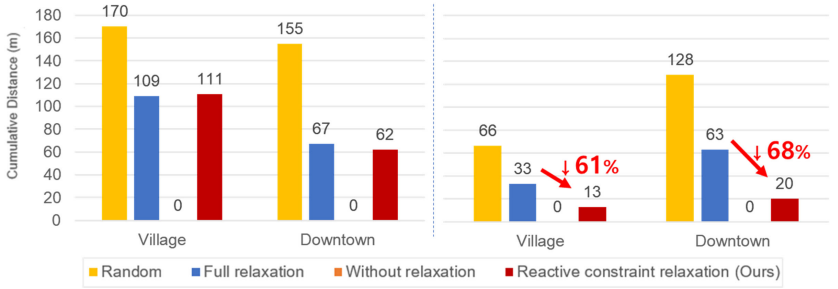
While most driving tasks failed without constraint-relaxation, the reactive constraint-relaxation method achieved similar success rates to full relaxation. Although the constraint violation distances for all constraints were similar for the reactive constraint-relaxation method and full-relaxation method, our method decreased the violation distance for roads by over 60%. These results indicate that reactive constraint-relaxation is necessary for navigation completeness and safety.

6 Conclusion

We introduced a novel urban navigation approach that reactively relaxes constrained regions for efficient and safe navigation. Our method shows a navigation



(a) Quantitative comparison of navigation success rate



(b) Quantitative comparison of constraint violation distance

Fig. 7: Comparison results for (a): navigation success rate and (b): constraint violation distance.

success rate of 95% for multiple simulation environments. In addition, compared to a full-relaxation method, our method traveled 60% less distance on roads, which are semantically the most dangerous constraint to relax.

Lastly, we seek to expand the current method with various data-driven models to autonomously define constraints, leveraging inverse constraint learning [31, 32] and large language models [33]. Further, we plan to incorporate natural-language based safety space indicators [34] and conduct real world evaluations.

Acknowledgement. This research was supported in part by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2024-RS-2024-00437102) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation) and in part by the KAIST Convergence Research Institute Operation Program.

References

1. Arkin, J., Park, D., Roy, S., Walter, M.R., Roy, N., Howard, T.M., Paul, R.: Multimodal estimation and communication of latent semantic knowledge for robust

execution of robot instructions. International Journal of Robotics Research **39**(10-11) (2020) 1279–1304
2. Howard, T., Stump, E., Fink, J., Arkin, J., Paul, R., Park, D., Roy, S., Barber, D., Bendell, R., Schmeckpeper, K., et al.: An intelligence architecture for grounded language communication with field robots. Field Robotics (2022)
3. Kim, D., Kim, J., Cho, M., Park, D.: Natural language-guided semantic navigation using scene graph. In: Proceedings of the International Conference on Robot Intelligence Technology and Applications (RiTA), Springer (2022) 148–156
4. Kim, D., Kim, Y., Jang, J., Song, M., Choi, W., Park, D.: Sggnet 2: Speech-scene graph grounding network for speech-guided navigation. In: Proceedings of the International Conference on Robot and Human Interactive Communication (RO-MAN), IEEE (2023) 1648–1654
5. Li, S., Park, D., Sung, Y., Shah, J.A., Roy, N.: Reactive task and motion planning under temporal logic specifications. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). (2021) 12618–12624
6. Rahmani, H., O'Kane, J.M.: What to do when you can't do it all: Temporal logic planning with soft temporal logic constraints. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). (2020)
7. Vasile, C.I., Tumova, J., Karaman, S., Belta, C., Rus, D.: Minimum-violation scltl motion planning for mobility-on-demand. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). (2017)
8. Lahijanian, M., Maly, M.R., Fried, D., Kavraki, L.E., Kress-Gazit, H., Vardi, M.Y.: Iterative temporal planning in uncertain environments with partial satisfaction guarantees. Transactions on Robotics **32**(3) (2016) 583–599
9. Castro, L.I.R., Chaudhari, P., Tůmová, J., Karaman, S., Frazzoli, E., Rus, D.: Incremental sampling-based algorithm for minimum-violation motion planning. In: Proceedings of the IEEE Conference on Decision and Control (CDC). (2013)
10. Tůmová, J., Hall, G.C., Karaman, S., Frazzoli, E., Rus, D.: Least-violating control strategy synthesis with safety rules. In: Proceedings of the International Conference on Hybrid Systems: Computation and Control (HSCC). (2013) 1–10
11. Ginting, M.F., Kim, S.K., Peltzer, O., Ott, J., Jung, S., Kochenderfer, M.J., a. Agha-mohammadi, A.: Safe and efficient navigation in extreme environments using semantic belief graphs. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). (2023) 5653–5658
12. Bouman, A., Ginting, M.F., Alatur, N., Palieri, M., Fan, D.D., Touma, T., Pailevanian, T., Kim, S.K., Otsu, K., Burdick, J., et al.: Autonomous spot: Long-range autonomous exploration of extreme environments with legged locomotion. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). (2020) 2518–2525
13. Dang, T., Tranzatto, M., Khattak, S., Mascarich, F., Alexis, K., Hutter, M.: Graph-based subterranean exploration path planning using aerial and legged robots. Journal of Field Robotics **37**(8) (2020) 1363–1388
14. Kahn, G., Abbeel, P., Levine, S.: Badgr: An autonomous self-supervised learning-based navigation system. IEEE Robotics and Automation Letters (RA-L) **6**(2) (2021) 1312–1319
15. Frey, J., Mattamala, M., Chebrolu, N., Cadena, C., Fallon, M., Hutter, M.: Fast traversability estimation for wild visual navigation. In: Proceedings of Robotics: Science and Systems (RSS). (2023)
16. Kress-Gazit, H., Lahijanian, M., Raman, V.: Synthesis for robots: Guarantees and feedback for robot behavior. Annual Review of Control, Robotics, and Autonomous Systems **1** (2018) 211–236

17. Vasile, C.I., Li, X., Belta, C.: Reactive sampling-based path planning with temporal logic specifications. International Journal of Robotics Research **39**(8) (2020) 1002–1028
18. Kantaros, Y., Malencia, M., Kumar, V., Pappas, G.J.: Reactive temporal logic planning for multiple robots in unknown environments. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). (2020)
19. Amorese, P., Lahijanian, M.: Optimal cost-preference trade-off planning with multiple temporal tasks. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). (2023) 2071–2077
20. Kamale, D., Karyofylli, E., Vasile, C.I.: Automata-based optimal planning with relaxed specifications. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). (2021) 6525–6530
21. Lahijanian, M., Kwiatkowska, M.: Specification revision for markov decision processes with optimal trade-off. In: Proceedings of the IEEE Conference on Decision and Control (CDC). (2016) 7411–7418
22. Kim, K., Fainekos, G.E., Sankaranarayanan, S.: On the minimal revision problem of specification automata. International Journal of Robotics Research **34**(12) (2015)
23. Fainekos, G.E.: Revising temporal logic specifications for motion planning. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). (2011) 40–45
24. Kress-Gazit, H., Lahijanian, M., Raman, V.: Synthesis for robots: Guarantees and feedback for robot behavior. Annual Review of Control, Robotics, and Autonomous Systems **1** (2018) 211–236
25. Baier, C., Katoen, J.P.: Principles of model checking. MIT Press (2008)
26. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik **1** (1959) 269–271
27. Fox, D., Burgard, W., Thrun, S.: The dynamic window approach to collision avoidance. IEEE Robotics and Automation Magazine (RA-M) **4**(1) (1997) 23–33
28. Speer, R., Chin, J., Havasi, C.: Conceptnet 5.5: An open multilingual graph of general knowledge. In: Proceedings of the National Conference on Artificial Intelligence (AAAI). Volume 31. (2017)
29. Chen, J., Li, G., Kumar, S., Ghanem, B., Yu, F.: How to not train your dragon: Training-free embodied object goal navigation with semantic frontiers. In: Proceedings of Robotics: Science and Systems (RSS). (2023)
30. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: An open urban driving simulator. In: Proceedings of the Conference on robot learning (CoRL), PMLR (2017) 1–16
31. Park, D., Noseworthy, M., Paul, R., Roy, S., Roy, N.: Inferring task goals and constraints using bayesian nonparametric inverse reinforcement learning. In: Proceedings of the Conference on robot learning (CoRL), PMLR (2020) 1005–1014
32. Jang, J., Song, M., Park, D.: Inverse constraint learning and generalization by transferable reward decomposition. IEEE Robotics and Automation Letters (RA-L) **9**(1) (2023) 279–286
33. Kim, Y., Kim, D., Choi, J., Park, J., Oh, N., Park, D.: A survey on integration of large language models with intelligent robots. Intelligent Service Robotics (2024)
34. Kim, D., Oh, N., Hwang, D., Park, D.: Lingo-space: Language-conditioned incremental grounding for space. In: Proceedings of the National Conference on Artificial Intelligence (AAAI). Volume 38. (2024) 10314–10322