Safer Online Replanning for Construction Robots via NLP-Informed Potential Fields and BIM Semantics

Johannes Mootz[1,2*], Mani Amani[1,3*], and Reza Akhavian[1]

***Abstract*— Autonomous navigation in construction environments is particularly challenging due to dynamic obstacles and uncertain surroundings. While recent advances in Building Information Modeling (BIM)-based planning have leveraged spatial and semantic information to improve navigation, most prior work assumes precise localization of the BIM model to enable global path planning. In contrast, this paper introduces an online replanning framework that registers obstacles on discovery within BIM and replans according to the updated semantic map. Our method integrates object-aware path planning by utilizing large language models (LLMs) to extract semantic danger sentiments from BIM-annotated objects and their spatial information about the mission environment. Additionally, we demonstrate practical feasibility by integrating a path tracking control, ensuring generated paths are not only safer but also realistically executable by mobile robots. Experimental results demonstrate an improved obstacle avoidance by 2.8× compared to traditional A* algorithms in dynamically updated environments.**

I. Introduction

Robotic systems are becoming increasingly important in the construction industry due to their potential to improve safety, productivity, and precision [1]. However, the dynamic and unstructured nature of construction sites poses considerable challenges, particularly for on-site deployment [2]. One of the relevant challenges that must be addressed is path planning and execution. In construction robotics, path planning often involves a layered process: modeling the environment into a grid or graph structure, using global search algorithms for decision-making, and translating the result into motion commands [3]. Recent approaches have used BIM to generate detailed environmental maps for navigation, enabling A* algorithms to compute efficient paths within semantically rich building layouts [4]. Other approaches have combined BIM-based global navigation with local adjustments to handle dynamic environments [5]. However, those methods either rely on pre-aligned static BIM maps or strictly separate global BIM-based waypoints from local reactive planners, thus not actively updating global paths in response to newly detected obstacles. Given the dynamic nature of construction environments, the accurate localization of moving obstacles can be challenging. Integrating global and local path planning is essential for autonomous mobile robots to navigate effectively. Global planners establish an initial route using known environmental data, while local planners adapt to dynamic obstacles in real time and ensure safe and efficient movement [6].

Previous works by the authors have demonstrated the feasibility of global path planning using BIM-based semantic maps combined with object-avoidance heuristics [7], [8]. Such approaches assumed accurate localization of BIM elements and focused primarily on global planning frameworks. In contrast, this paper introduces an online replanning framework that registers obstacles on discovery within BIM and replans according to the updated semantic map. By leveraging the semantic knowledge of existing BIM families and dynamically updating potential heuristics with the Multi-Heuristic A* (MHA*) algorithm upon discovering previously unseen obstacles, our approach significantly enhances safety and context-awareness. Beyond planning, this work incorporates a practical low-level control strategy (proportional-integral or PI-based path tracking), thus bridging the gap between theoretical path generation and feasible robot deployment. Experimental results in simulated construction environments demonstrate that our online replanning algorithm maintains safer paths, quantified by increased minimum distances to obstacles.

II. Problem Formulation

We consider a mobile robot navigating a 2D plane discretized into a grid of cells. Let

$$\mathcal{S} = \left\{(x_i, y_j) \;\middle|\; i \in \{1, \ldots, N_x\},\ j \in \{1, \ldots, N_y\}\right\} \quad (1)$$

denote the set of all possible grid points (or nodes) in the environment. Some of these cells may be fully occupied by obstacles.

Initially, the robot plans a path assuming a known obstacle map $\mathcal{O}_0 \subset \mathcal{S}$. The optimal path at time $t = 0$ is given by:

$$\pi_0^* = \arg \min_{\pi \in \Pi(\mathcal{S} \setminus \mathcal{O}_0)} J(\pi) \quad (2)$$

where $\Pi(\cdot)$ denotes the set of feasible paths in the obstacle-free space, and $J(\pi)$ is a cost functional that jointly evaluates both path efficiency (e.g., length, curvature) and safety (e.g., distance to nearest obstacle).

As the robot navigates, it actively observes the environment. Upon detecting new obstacles $\Delta\mathcal{O}_t$ at time $t > 0$, which were previously unknown (i.e., $\Delta\mathcal{O}_t \cap \mathcal{O}_{t-1} = \emptyset$), it updates the obstacle set:

$$\mathcal{O}_t = \mathcal{O}_{t-1} \cup \Delta\mathcal{O}_t \quad (3)$$

[1]Department of Civil, Construction, and Environmental Engineering, San Diego State University, San Diego, CA, United States

[2]Department of Mechanical and Aerospace Engineering, University of California San Diego, San Diego, CA, United States

[3]Department of Electrical and Computer Engineering, University of California San Diego, San Diego, CA, United States

*These authors contributed equally to this work

and resolves the online optimization problem:

$$\pi_t^* = \arg \min_{\pi \in \Pi(\mathcal{S} \setminus \mathcal{O}_t)} J(\pi) \tag{4}$$

This optimization is resolved using the MHA* method detailed in the next section.

III. Mathematical Framework

A. Repulsive potential fields

To define the safety cost heuristic, we construct repulsive potential fields. This builds on the planning framework introduced in previous studies [7], [8], which provides more detailed information on the underlying concepts. The following sections summarize the key concepts relevant to this work. To represent the environment, we utilize BIM, a widely adopted standard in construction that includes both spatial geometry and semantic object data. The BIM model serves as a starting point and is projected onto a 2D grid-based map suitable for path planning. During this process, object instances are classified by type, and their geometric extents are projected onto the 2D plane. Each object is assigned a risk score, reflecting the hazard the object may pose during navigation. This score is used to define a repulsive potential field over the grid. The field value is computed at each grid point based on its proximity to the object and the associated danger level. For an object set $\{\mathcal{M}\}$, the repulsive field at a grid cell (x, y) is defined as:

$$f_{\text{rep}}(x, y, \mathcal{M}) = \begin{cases} k_{rep} \cdot \exp\left(-D(x, y, \mathcal{M})\right), & D < D_{\max} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

Here, $D(x, y, \mathcal{M})$ is the minimum Euclidean distance from (x, y) to the object's occupied region M, and k_{rep} is the object's repulsion coefficient. We set $D_{\max}$ as the practical sensing radius (e.g., 2 m). Initially, k_{rep} is uniform for all objects, subsequently scaled by semantic risk scores obtained via NLP-based assessments. The total repulsive potential at a point is computed as the sum over all objects:

$$G(x_i, y_j) = \sum_k f_{\text{rep}}(x_i, y_j, \mathcal{M}_k) \tag{6}$$

The result is a dense potential field over the environment, where high-risk areas create strong repulsive forces that discourage paths from passing nearby. The total potential $G(x_i, y_j)$ serves as a heuristic in the path planning algorithm.

However, prior works often assume reasonable localization and a static environment where obstacles are known in advance and do not change over time. In contrast, this work removes these assumptions and registers obstacles upon exploration and discovery. Leveraging the labeled object models available in BIM, we formulate a classification problem in which newly discovered obstacles, identified through sensor observations, such as an RGB-D camera, are assigned accordingly. This information is then used to dynamically update the environment representation and resolve the path planning problem, minimizing the revised optimization objective in response to the newly perceived surroundings.

B. Risk modulation with NLP given BIM family labels

Given the semantic label of a BIM family, we infer the appropriate level of avoidance a robot should exhibit. We use GPT-3.5 with structured prompts describing object characteristics (e.g., financial value, dynamism, hazard type) to output a normalized danger coefficient (0–1). For instance, a forklift might receive a danger coefficient of 0.9 versus 0.3 for a stationary beam. These coefficients directly modulate each object's repulsive coefficient through $\alpha * k_{rep}$, where α is the scaling factor derived from the LLM to reflect implicit semantic risk and thereby adjusts the planner's behavior [8].

C. Multi-Heuristic A* path planning

We use a graph-based path planning method to compute a safe and efficient trajectory. Each node within the graph has an associated distance-to-goal cost and potential field. We adopt the MHA* algorithm, an extension of the classical A* that enables the simultaneous use of multiple heuristics. In standard A*, the cost of reaching a node n is given by:

$$f(n) = g(n) + h(n) \tag{7}$$

With $f(n)$ as the total cost of each node, $g(n)$ denotes the cost of the start node to the current node, and $h(n)$ is the heuristic of the current node [8]. Using Euclidean distance alone can lead to overly conservative paths or unsafe shortcuts. MHA* effectively balances navigation by handling multiple heuristics, such as safety heuristics, to ensure safe and reasonably efficient paths. To guarantee optimality, $h(n)$ must be admissible, meaning it never overestimates the true cost. MHA* generalizes this formulation by allowing multiple heuristics to guide the search, where only one (called the anchor) must be admissible. In our case, we define: (1) an admissible heuristic based on Euclidean distance to the goal, (2) a safety heuristic based on the repulsive potential field $G(x, y)$. Original formulation and analysis of MHA* can be found in [9].

D. Path Following for Wheeled Mobile Robots

Path-following in mobile robotics is a well-studied problem, with solutions ranging from basic control laws to sophisticated nonlinear and model-based strategies [10], [11]. To follow a planned path generated by MHA*, we adopt a low-level control strategy suitable for differential-drive robots. This skid steer UGV can be modeled as a unicycle [12]. The kinematic model is given by (8) and it includes the configuration $\mathbf{p} = [x, y, \theta]^\top$ and control input $\mathbf{u} = [v, \omega]^\top$.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{8}$$

The high-level planner provides a discrete sequence of waypoints $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$, which are interpolated using splines to ensure smooth tracking. Given the robot's current position $\mathbf{p} = [x, y, \theta]^\top$ we find the closest path target:

$i_{\text{closest}} = \arg\min_i \|\mathbf{p} - \mathbf{p}_i\|$. The control target is selected by a fixed-step lookahead rule: $i_{target} = i_{closest} + \delta$. The error state is denoted as $\mathbf{e} = [e_x, e_y, e_\theta]^\top$. Path following is performed in the robot's local frame using a feedforward velocity v_{FF} plus proportional-integral (PI) feedback control command for the forward velocity in (9) and a PI control law to compute the angular velocity command as in (10) [13].

$$v = v_{FF} + k_{P,v} e_v + k_{I,v} \int e_v \, dt \tag{9}$$

$$\omega = k_{P,w} e_\theta + k_{I,w} \int e_\theta \, dt \tag{10}$$

The controller's proportional gains $k_{P,v}$, $k_{P,w}$ and integral gains $k_{I,v}$ $k_{I,w}$ are tuned to achieve desired behavior. This control law is simple to implement, effective for smooth trajectories and suitable for simulation or deployment on skid-steered platforms. The velocity commands $[v, \omega]^\top$ can be mapped directly to wheel velocities $[v_{left}, v_{right}]^\top$ using a one-to-one relationship.

IV. Methodology

The proposed framework imports a BIM file into the Unity game engine simulation environment. Object metadata is extracted and structured as a JavaScript Object Notation (JSON) file, which is provided as input to an LLM. Based on semantic interpretation and prompt context, the LLM assigns risk-based coefficients to each object, forming a repulsive potential field. This field serves as a heuristic in the multi-heuristic A* (MHA*) planner for global path generation.

Obstacle detection is performed in simulation using Unity's Raycast module, which enables real-time sensing by projecting rays from a virtual sensor toward objects in the environment. In simulation, detected objects via Unity's Raycast are directly matched to their known BIM identifiers. In real-world deployments, a visual recognition module would be necessary, which we currently assume as adequate. An object is classified as an obstacle if it falls within a 60° field of view relative to the sensor's forward direction. Upon detection, online replanning is triggered using the MHA* algorithm to update the planned path. For local control, a PI controller tracks the planned path by converting position errors into velocity commands for a unicycle-model robot. The overall system pipeline is illustrated in Figure 1. The final minimum distances to each detected obstacle is recorded as a safety performance metric. The controller gains listed in Table (I) were tuned through simulation trials to result in a desired behavior.

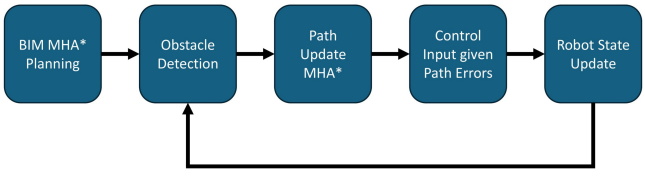

Fig. 1. Overview of the proposed planning and control framework

TABLE I
Control Parameters

v_{limit}	w_{limit}	v_{FF}	$k_{P,v}$	$k_{I,v}$	$k_{P,w}$	$k_{I,w}$
0.15 m/s	0.75 m/s	0.1 m/s	0.2	0.01	1.5	0.01

V. Experiments and Validation

The BIM file is converted from a Revit file into an FBX file and loaded into the simulator as shown in Figure 2, along with the initial planned path without prior knowledge of obstacle positions. Obstacles are only detected when they enter the FOV of the robot and are then used for updating the path. Figure 3 shows the following replanning sequence. As the robot navigates, newly detected obstacles trigger local turns and global replanning, resulting in a safer path that avoids high-risk regions. A video demonstration of the path-following behavior is available: Pathfinding Mobile Robot

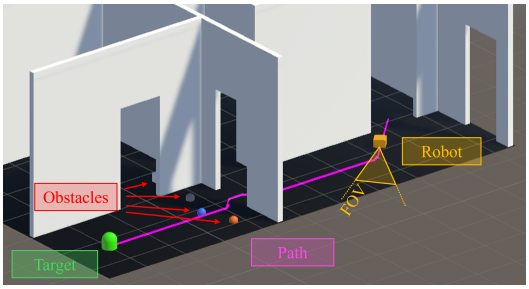

Fig. 2. Simulation environment of a construction site modeled in Unity, with BIM data imported from a Revit file. It shows the robot (yellow), obstacles (red), path (pink) and target (green)

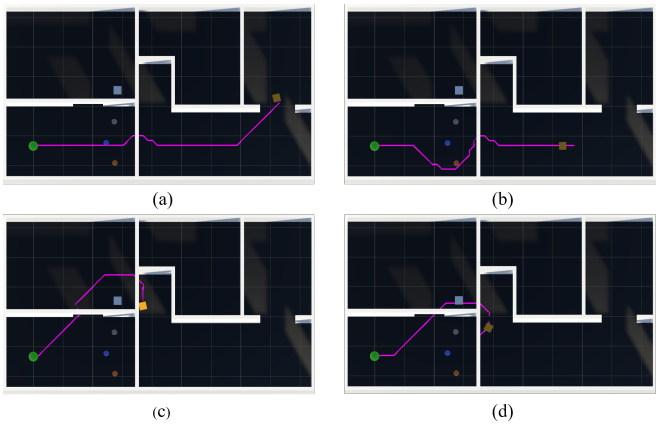

Fig. 3. Updated path based on the detection of obstacles: (a) Sequence 1, (b) sequence 2, (c) sequence 3, (d) sequence 4. With the detection of new obstacles, the robot adjusts the plan online, resulting in a longer but safer path.

For baseline comparison, we evaluated standard A* path planning without obstacle detection. As it relies solely on distance-based planning, it does not actively seek safer paths around newly detected obstacles. Table II reports the minimum distance to unseen obstacles, which are the BIM families that we find during navigation. The baseline A*

closely passes obstacles (min clearance 0.26 m), while our method proactively maintains safer clearances (0.74 m) by dynamically adjusting paths earlier. Although this results in longer paths, it significantly enhances safety—a critical trade-off in dynamic construction settings. In other words, our planning framework achieves a 2.8x improvement in object avoidance compared to baseline A*. In addition, Table III presents the root-mean-square error (RMSE) in forward, lateral, and angular tracking, confirming the effectiveness of the low-level controller in following the planned path.

TABLE II

TABLE II: OBSTACLE AVOIDANCE METRICS COMPUTED BASED ON 4 DETECTED BIM OBSTACLES

Method	**Min. Distance (m)**	**Path length (m)**
Standard A*	0.26	5.74
BIM-based MHA*	0.74	9.58

TABLE III

RMSE METRIC FOR FORWARD x_e, LATERAL y_e, AND ANGULAR θ_e ERROR

RMSE x_e	**RMSE** y_e	**RMSE** θ_e
0.0886 (m)	0.1181 (m)	25.19 (deg)

The experimental results confirm that the proposed framework generates a safe, dynamically adaptive path in a simulated construction environment using partial visibility guidance. As the robot progressively detects previously unseen obstacles, the planner successfully updates the path to avoid hazards, demonstrating the benefits of online replanning with real-time sensing. The minimum distance to obstacles demonstrates that safety is maintained throughout the navigation process despite the increase in path length. This trade-off is consistent with the intended design of the MHA* planner, which prioritizes risk avoidance over shortest path optimization.

VI. CONCLUSION

This work presents an online replanning framework for mobile robots operating in BIM-enabled environments. By exploiting semantic object data from BIM, contextual risk assessment through LLMs, and multi-heuristic A* planning, the proposed system effectively generates safe and adaptive navigation paths. Simulation results demonstrate the framework's ability to adjust paths in response to newly observed obstacles, while maintaining safe distances and tracking accuracy through a PI-based controller.

The current validation relies exclusively on Unity simulation using simplified perception (Raycast sensors). While this enables controlled evaluations, real-world scenarios involve complexities such as sensor noise, varying lighting conditions, and partial occlusions. Prior research [14] has demonstrated accurate semantic segmentation in cluttered construction scenes, suggesting real-world feasibility. However, future practical deployment requires robust perception modules and SLAM-based localization to handle realistic uncertainties. These critical aspects form our future research directions. Also, future research will incorporate realistic semantic perception and SLAM techniques to address current assumptions of perfect perception and localization. Additionally, precomputing or caching LLM-derived risk assessments is proposed to facilitate real-time, real-world deployments. Dynamic obstacle avoidance will be addressed through the integration of appropriate control strategies, such as control barrier functions (CBFs).

ACKNOWLEDGMENT

The presented work has been supported by the U.S. National Science Foundation (NSF) CAREER Award through grant No. CMMI 2047138, and grant No. DUE 1930546. The authors gratefully acknowledge the support from the NSF. Any opinions, findings, conclusions, and recommendations expressed in this paper are those of the authors and do not necessarily represent those of the NSF.

REFERENCES

[1] A. Attalla, O. Attalla, A. Moussa, D. Shafique, S. B. Raean, and T. Hegazy, "Construction robotics: review of intelligent features," *International Journal of Intelligent Robotics and Applications*, vol. 7, no. 3, pp. 535–555, 2023.

[2] S. Parascho, "Construction robotics: from automation to collaboration," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 6, no. 1, pp. 183–204, 2023.

[3] S. Fu, D. Yang, Z. Mei, and W. Zheng, "Progress in construction robot path-planning algorithms," *Applied Sciences*, vol. 15, no. 3, p. 1165, 2025.

[4] D. Wang, B. Liu, H. Jiang, and P. Liu, "Path planning for construction robot based on the improved a* algorithm and building information modeling," *Buildings*, vol. 15, no. 5, p. 719, 2025.

[5] R. G. Braga, M. O. Tahir, S. Karimi, U. Dah-Achinanon, I. Iordanova, and D. St-Onge, "Intuitive bim-aided robotic navigation and assets localization with semantic user interfaces," *Frontiers in Robotics and AI*, vol. 12, p. 1548684, 2025.

[6] J. R. Sánchez-Ibáñez, C. J. Pérez-del Pulgar, and A. García-Cerezo, "Path planning for autonomous mobile robots: A review," *Sensors*, vol. 21, no. 23, p. 7898, 2021.

[7] M. Amani and R. Akhavian, "Bim-based safe and trustworthy robot pathfinding using scalable mha* algorithms and natural language processing," *arXiv preprint arXiv:2411.15371*, 2024.

[8] M. Amani and R. Akhavian, "Bayesian bim-guided construction robot navigation with nlp safety prompts in dynamic environments," *arXiv preprint arXiv:2501.17437*, 2025. [Online]. Available: https://arxiv.org/abs/2501.17437

[9] S. Aine, S. Swaminathan, V. Narayanan, V. Hwang, and M. Likhachev, "Multi-heuristic a*," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 224–243, 2016. [Online]. Available: https://doi.org/10.1177/0278364915594029

[10] P. Morin and C. Samson, "Motion control of wheeled mobile robots." *Springer handbook of robotics*, vol. 1, pp. 799–826, 2008.

[11] N. Hung, F. Rego, J. Quintas, J. Cruz, M. Jacinto, D. Souto, A. Potes, L. Sebastiao, and A. Pascoal, "A review of path following control strategies for autonomous robotic vehicles: Theory, simulations, and experiments," *Journal of Field Robotics*, vol. 40, no. 3, pp. 747–779, 2023.

[12] L. S. V. Boas, T. L. Santos, and A. G. Conceicao, "A simplified trajectory tracking control based on linear design for skid-steered wheeled ugvs," *Journal of the Franklin Institute*, vol. 361, no. 4, p. 106660, 2024.

[13] K. M. Lynch and F. C. Park, *Modern robotics*. Cambridge University Press, 2017.

[14] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3d semantic parsing of large-scale indoor spaces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.