Human-Aware Adaptive Motion Planning with Lazy Tree Repair for Construction Robots

Samiha Tasnim[1], Inbae Jeong, Ph.D.[1], and Youjin Jang, Ph.D.[2]

***Abstract*— Human-related risk can change during robot navigation in shared construction environments, requiring the robot to adapt its motion while avoiding unnecessary replanning. This paper introduces an integrated adaptive path planning framework that combines human-activity-aware safety adaptation, a goal-rooted Anytime RRT* tree, and lazy edge evaluation for efficient planning. Human-risk information is used to update the robot velocity limit and human-centered risk region, allowing the planned motion to respond to changing interaction conditions. By reusing a goal-rooted search tree and evaluating only candidate-path edges affected by the updated human-centered risk region, the framework avoids repeatedly updating the entire tree and repairs the path only when necessary. The performance of the method is compared with Anytime RRT* Full Replanning and evaluated in terms of planning update time, checked path-edge ratio, robot velocity adaptation, and path length.**

I. INTRODUCTION

Automation in construction is increasing as robots are used for inspection, monitoring, material transport, and site navigation [1]. Unlike structured factory environments, construction workspaces are often cluttered, changing, and shared with workers. A mobile robot operating in such an environment may need to move through narrow passages, around stored materials, and near active workers [2]. Therefore, path planning should consider not only static obstacles but also worker-related risk that can change during navigation. Relevant factors include the worker's position relative to the current path, distance from the path, motion toward the robot or its planned route, and activity in the surrounding area [3], [4]. As these conditions change, the robot may need to reduce its velocity limit and enlarge the human-centered risk region.

Human-aware motion planning has shown that human information can be incorporated directly into the planning objective. Mainprice et al. used human-centered costmaps within a sampling-based planner to generate paths that account for human presence [5]. Faroni et al. incorporated safety-related velocity limitations into a path cost by estimating the slowdown caused by human proximity [6]. Sakcak and Bascetta combined human-motion prediction with RRTX and introduced a danger-based edge cost, evaluating the human-related cost only for edges intersecting the predicted human-motion region [7]. For execution-time navigation, however, a change in worker risk does not always require a new path. A worker may move closer to only one portion of the current route while the rest of the path and planning structure remain unaffected. In such cases, the planner should first determine whether the current candidate path is affected by the updated human condition before performing additional replanning. Reuse of previous planning information provides an efficient basis for this type of update. MP-RRT preserves useful portions of previous search trees after environmental changes [8], while RRTX maintains and repairs the same search graph throughout navigation. RRTX also uses a goal-rooted shortest-path structure, allowing previously sampled nodes and valid connections to remain useful as the robot moves [9]. Lazy planning further reduces computation by postponing edge evaluation until it is needed for a candidate solution. LazySP first identifies a candidate path using estimated edge costs and then evaluates selected edges of that path [10]. LLPT* combines this idea with lifelong sampling-based replanning by maintaining a goal-rooted structure, evaluating candidate-solution edges lazily, and repairing the existing tree when an evaluated edge becomes invalid [11]. Recent work has also combined online replanning with human-aware safety information. MARSHA uses a human-dependent safety-aware cost within a reactive replanning framework and reuses previous planning information while applying lazy cost evaluation to support real-time operation [12]. However, these approaches do not directly focus on a worker-risk-based update of the current candidate path, where only edges affected by the updated human-centered risk region are selected before deciding whether path repair is required. If the risk region affects only a small portion of the current path, evaluating a larger part of the planning structure can introduce unnecessary computation.

This paper presents a human-aware lazy path-update method for robot navigation in shared environments. The main contribution is the integration of human-risk-based safety adaptation with candidate-path-based lazy edge evaluation and reuse of a goal-rooted Anytime RRT* tree. When the human-risk condition changes, the robot velocity limit and human-centered risk region are updated, and only the affected candidate-path edges are re-evaluated. If the affected edges remain safe, the current path is maintained and the updated velocity limit is applied. If an affected edge becomes unsafe, the path is repaired using the existing planning structure instead of rebuilding the tree from the beginning. The main objective is to reduce unnecessary path-update computation while allowing the robot to adapt to changing human-risk conditions during navigation. The

[1]Samiha Tasnim and Inbae Jeong are with the Department of Mechanical Engineering, North Dakota State University,USA samiha.tasnim.1@ndsu.edu and inbae.jeong@ndsu.edu

[2]Youjin Jang is with the Department of Civil, Construction and Environmental Engineering, North Dakota State University y.jang@ndsu.edu

proposed method is evaluated using a quadruped robot connected through ROS in a mapped indoor environment and compared with an Anytime RRT* full-replanning baseline. The evaluation focuses on planning update time, checked path-edge ratio, robot velocity adaptation, and path length. Section II presents the proposed methodology. Section III describes the experimental setup and physical validation. Section IV presents the results and discussion, and Section V concludes the paper.

II. Methodology

The proposed method updates the robot's safety behavior during execution using the human-risk value $r_h(t)$. This value adjusts the robot velocity limit and the human-centered risk region. The method is organized around risk-based safety adaptation, reuse of a goal-rooted planning tree, and lazy evaluation of the current candidate path.

A. Problem Setting

This paper considers a quadruped robot navigating in a human-shared workspace with static obstacles and a nearby worker. At time t, the current robot state is denoted by $x_r(t)$, and the fixed goal state is denoted by x_g. The worker position at time t is denoted by $p_h(t)$. At each planning update, the planner extracts a current candidate path $\pi_c(t)$ from $x_r(t)$ to x_g. During execution, the worker may move closer to the current candidate path, move toward the robot or the path, or work in an area that requires more caution. These changes can increase the risk around the current candidate path and may require the robot to reduce its velocity limit or update the path. The worker-related risk at time t is represented by a normalized human-risk value $r_h(t) \in [0, 1]$. This value reflects worker-related factors such as distance to the current candidate path, motion direction relative to the robot or path, and activity in the surrounding area. Human activity influences the planner indirectly through its effect on $r_h(t)$. Here, $r_h(t) = 0$ represents the nominal low-risk condition, while $r_h(t) = 1$ represents the highest-risk condition considered by the planner, and intermediate values represent increasing levels of risk. In a complete system, $r_h(t)$ can be estimated from worker position, motion, and activity information. In this study, $r_h(t)$ is provided as an input for planning adaptation. Based on $r_h(t)$, the planner updates the robot velocity limit and human-centered risk region, evaluates only the affected edges on the current candidate path $\pi_c(t)$, and repairs the path when those edges become unsafe.

B. Human-Risk-Based Safety Update

At each planning cycle, the human-risk value $r_h(t)$ updates two safety-related terms: the robot velocity limit $v_{\lim}(t)$ and the human-centered risk-region radius $\rho_h(t)$. Linear mappings are used to provide a simple and monotonic adaptation between the predefined safety bounds. As $r_h(t)$ increases, the velocity limit decreases and the risk-region radius increases. Similar human-aware planning approaches use human-related safety information to adapt robot motion or planning cost [6], [7]

$$v_{\lim}(t) = v_{\min} + (v_0 - v_{\min})\,(1 - r_h(t)) \tag{1}$$

$$\rho_h(t) = \rho_0 + (\rho_{\max} - \rho_0)\,r_h(t) \tag{2}$$

where v_0 is the nominal robot velocity, and $v_{\min}$ is the minimum safe velocity, with $v_0 \geq v_{\min}$. Here, ρ_0 is the nominal human-centered risk region radius, and $\rho_{\max}$ is the maximum radius, with $\rho_{\max} \geq \rho_0$. When $r_h(t) = 0$, the robot uses the nominal velocity v_0, and the human-centered risk region radius is ρ_0. When $r_h(t) = 1$, the velocity limit becomes $v_{\min}$, and the human-centered risk region radius becomes $\rho_{\max}$. Therefore, as the human-risk value increases, the robot slows down and the human-centered risk region becomes larger. The inner no-go region is defined by a fixed safety radius ρ_{safe} around the human, where robot motion is not allowed. A candidate-path edge is considered unsafe if $d(e, p_h(t)) \leq \rho_{\text{safe}}$. The outer human-centered risk region is used to identify candidate-path edges that require re-evaluation. As the human-risk value increases, this region becomes larger, allowing the planner to consider a larger portion of the candidate path for safety evaluation.

C. Goal-Rooted Anytime RRT* Tree Reuse

After the safety update, the planner reuses the existing Anytime RRT* tree instead of generating a new tree from the current robot state. The tree is rooted at the fixed goal x_g. This structure matches the execution setting because the robot state changes during motion, while the goal remains fixed. At each planning update, the planner connects the current robot state $x_r(t)$ to nearby nodes in the existing goal-rooted tree. It then extracts the current candidate path $\pi_c(t)$ from $x_r(t)$ to x_g. The basic RRT* sampling, parent selection, and rewiring process is not modified. The tree is used as a reusable planning structure. Since a human-risk update usually affects only a local region near the worker, most sampled nodes and tree edges remain valid. Reusing the existing tree avoids discarding useful planning information after every human-risk update.

D. Candidate-Path-Based Lazy Edge Evaluation and Repair

After the current candidate path $\pi_c(t)$ is extracted, the planner does not evaluate the full tree. The update is limited to the edges on the current candidate path that are affected by the updated human-centered risk region. Fig. 1 shows this process. The blue line represents the current candidate path $\pi_c(t)$, and the tree branches represent the reused goal-rooted tree. Only the candidate-path edges near the human-centered risk region are checked. Valid checked edges are kept, unsafe checked edges are pruned, and tree edges that are not part of the current candidate path are deferred. Let $\mathcal{E}_c(t)$ denote the set of edges on the current candidate path $\pi_c(t)$. Let $p_r(x)$ denote the robot position corresponding to state x. The minimum workspace distance between edge e and the worker position $p_h(t)$ is defined as

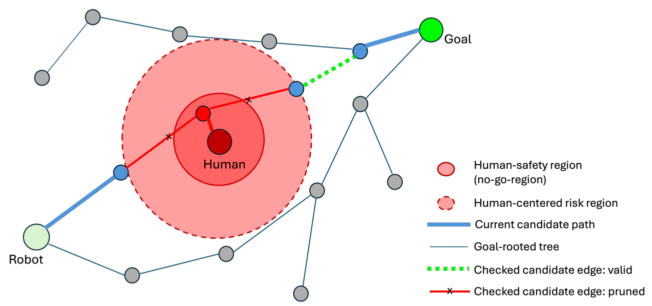


Fig. 1. Candidate-path-based lazy edge evaluation. The planner checks only the affected edges on the current candidate path $\pi_c(t)$. Valid edges are kept, unsafe checked edges are pruned, and non-path edges are deferred for future checking.

$$d(e, p_h(t)) = \min_{x \in e} \|p_r(x) - p_h(t)\| \tag{3}$$

The checked edge set is defined as

$$\mathcal{E}_{\text{check}}(t) = \{e \in \mathcal{E}_c(t) \mid d(e, p_h(t)) \leq \rho_h(t) + \delta\} \tag{4}$$

where $\rho_h(t)$ is the updated human-centered risk region radius and δ is a small buffer distance. The values are chosen so that $\rho_h(t) + \delta \geq \rho_{\text{safe}}$ for all human-risk levels. It ensures that any candidate-path edge entering the fixed safety region is included in $\mathcal{E}_{\text{check}}(t)$ and evaluated. This means that an edge is checked only if it is part of the current candidate path and lies close to the updated human-centered risk region. If $\mathcal{E}_{\text{check}}(t)$ is empty, the current candidate path is kept, and the robot follows the updated velocity limit. If $\mathcal{E}_{\text{check}}(t)$ is not empty, only those edges are re-evaluated. If all checked edges remain safe, the current path is kept. If an affected edge becomes unsafe, that edge is removed while the remaining valid tree structure is retained. The planner then continues the Anytime RRT* search from the retained tree to find a new valid path to the goal. This avoids rebuilding the tree from the beginning.

III. Experimental Setup and Physical Validation

The experiment was conducted using a physical quadruped robot integrated with ROS in a mapped indoor environment. The environment was used as a shared workspace to test the proposed planning method before evaluation in more complex construction environments. During the experiment, the goal-rooted Anytime RRT* tree was maintained in the ROS planning system and reused for candidate-path generation and path repair after each human-risk update. Fig. 2 shows the physical workspace and the corresponding occupancy-grid map in RViz, including the robot pose used for navigation. The evaluation used four main metrics: planning update time, checked path-edge ratio, robot velocity limit, and path length. Planning update time measures the computation required after a human-risk update. The checked path-edge ratio, defined as $|\mathcal{E}_{\text{check}}(t)|/|\mathcal{E}_c(t)|$, represents the portion of the current candidate path checked during an update. The robot velocity limit $v_{\text{lim}}(t)$ represents how the robot speed changes with the human-risk level. For comparison, an Anytime RRT* full-replanning method was used as the baseline. It used the same human-risk input and velocity

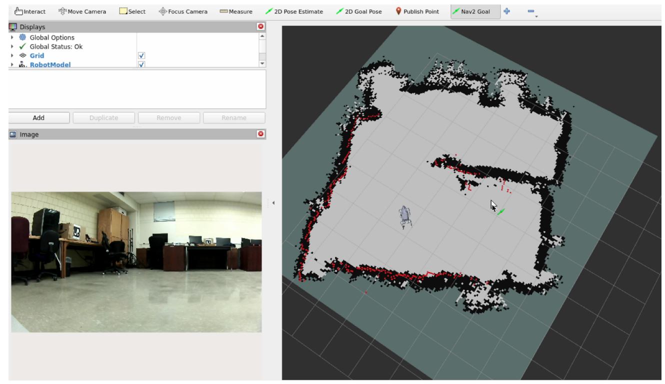
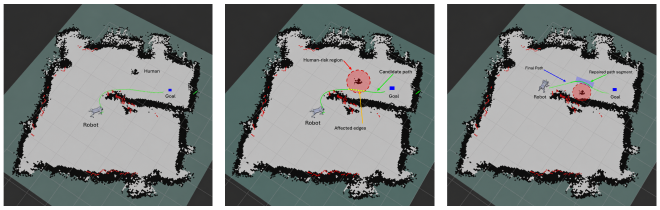


Fig. 2. Experimental environment for physical quadruped robot validation: indoor workspace captured by the robot camera (left) and corresponding mapped environment in RViz (right).

adaptation as the proposed method but rebuilt the planning tree after each risk update. In contrast, the proposed method reused the existing goal-rooted tree and checked only the candidate-path edges affected by the updated human-centered risk region. Both methods were tested under the same experimental conditions. During the physical experiment, the human position $p_h(t)$ and human-risk value $r_h(t)$ were manually provided to the ROS system. The human position was specified relative to the candidate path, while $r_h(t)$ represented the current human-risk condition. These inputs were used to update the human-centered risk region and robot velocity limit and to determine whether the current path should be kept or repaired. For the experiment, $v_0 = 0.40$ m/s and $v_{\min} = 0.20$ m/s were used, with $\rho_0 = 0.60$ m, $\rho_{\max} = 1.00$ m, $\rho_{\text{safe}} = 0.50$ m, and $\delta = 0.10$ m. The human-risk value $r_h(t)$ was varied from 0 to 1 to represent increasing risk conditions.

IV. Results and Discussion

Fig. 3 shows how the proposed method updates and repairs the candidate path when the human-risk condition changes. In Fig. 3(a), the human is away from the current candidate path $\pi_c(t)$, so the risk region does not affect the path and no edge is re-evaluated. In Fig. 3(b), the human moves closer to the path. The increased human-risk value $r_h(t)$ enlarges the human-centered risk region $\rho_h(t)$ and reduces the robot velocity limit $v_{\text{lim}}(t)$. The candidate-path edges inside the affected region are identified as $\mathcal{E}_{\text{check}}(t)$ and re-evaluated. If an affected edge becomes unsafe, that path segment is removed and repair is performed using the existing goal-rooted tree. Fig. 3(c) shows the repaired path. The planner keeps the unaffected tree structure and changes only the affected part of the route instead of rebuilding the complete planning tree. The proposed method was evaluated over 15 repeated trials using the physical quadruped robot. The same mapped environment, robot start position, fixed goal, human-risk conditions, and planning parameters were used in all trials. Anytime RRT* with full replanning was used as the benchmark. After each human-risk update, the benchmark generated a new planning tree from the current robot state, whereas the proposed method reused the existing tree and



(a) (b) (c)

Fig. 3. Candidate-path update and repair: (a) candidate path before the human-risk update, (b) affected candidate-path edges identified after the risk update, and (c) repaired path after an affected edge becomes unsafe.

TABLE I

COMPARISON OF PATH-UPDATE PERFORMANCE OVER 15 TRIALS.

Evaluation Metric	**Anytime RRT* Full Replanning**	**Proposed Method**
Planning update time	52.6 ± 4.8 ms	24.7 ± 3.2 ms
Checked path-edge ratio	-	$26.1 \pm 2.9\%$
Robot velocity limit	$0.40 \rightarrow 0.20$ m/s	$0.40 \rightarrow 0.20$ m/s

evaluated only the affected candidate-path edges. Both methods used the same human-risk input and velocity adaptation. Table I summarizes the path-update performance over 15 physical trials. All quantitative results are reported as mean $\pm$ standard deviation over the trials. The proposed method achieved a mean planning update time of 24.7 ± 3.2 ms, compared with 52.6 ± 4.8 ms for Anytime RRT* with full replanning, giving an approximately 53.0% reduction in update time. This difference comes from how the two methods respond to a human-risk update. Anytime RRT* rebuilds the planning tree from the current robot state, while the proposed method keeps the existing goal-rooted tree and checks only the part of the candidate path affected by the updated human-centered risk region. On average, the proposed method re-evaluated only $26.1 \pm 2.9\%$ of the candidate-path edges during an update. The remaining path edges and valid parts of the tree were kept without additional checking. This was particularly useful when the human-risk change affected only a small part of the route, because the rest of the path could be kept without additional checking. The checked path-edge ratio is not reported for Anytime RRT* because it generates a new planning tree after each update instead of checking selected edges of the existing path. The repeated trials also showed how the proposed method behaved under different update outcomes. The current candidate path was maintained in 9 of the 15 trials because the affected edges remained safe. In the other 6 trials, at least one affected edge became unsafe and path repair was performed using the retained tree. For these repair trials, the remaining path length increased from 4.18 ± 0.21 m before repair to 4.30 ± 0.22 m after repair, corresponding to an average increase of approximately 2.9%. This depicts that the planner was able to avoid the unsafe segment while making only a small change to the remaining route.

V. CONCLUSION

This paper presented a human-aware adaptive path planning method for robot navigation in shared environments. The method adapts the robot velocity limit and human-centered risk region based on the human-risk value while reusing the existing goal-rooted planning structure. Only candidate-path edges affected by the updated risk region are re-evaluated, and path repair is performed when an affected edge becomes unsafe. Physical experiments showed that the proposed method reduced planning update time compared with Anytime RRT* with full replanning while evaluating only the affected portion of the candidate path. The experiments also demonstrated both path-maintenance and path-repair behavior under changing human-risk conditions. The current study focuses on the planning response using human-risk information as an input. Future work will include real-time human tracking and risk estimation, as well as evaluation in more complex construction environments with multiple workers.

ACKNOWLEDGMENT

The authors would like to acknowledge the financial support from ND EPSCoR for this research.

REFERENCES

[1] L. Zeng, S. Guo, J. Wu, and B. Markert, "Autonomous mobile construction robots in built environment: A comprehensive review," *Developments in the Built Environment*, vol. 19, p. 100484, 2024.

[2] S. Halder, K. Afsari, E. Chiou, R. Patrick, and K. A. Hamed, "Construction inspection & monitoring with quadruped robots in future human-robot teaming: A preliminary study," *Journal of Building Engineering*, vol. 65, p. 105814, 2023.

[3] C.-J. Liang, X. Wang, V. R. Kamat, and C. C. Menassa, "Human–robot collaboration in construction: Classification and research trends," *Journal of Construction Engineering and Management*, vol. 147, no. 10, p. 03121006, 2021.

[4] M. Wu, J.-R. Lin, and X.-H. Zhang, "How human-robot collaboration impacts construction productivity: An agent-based multi-fidelity modeling approach," *Advanced Engineering Informatics*, vol. 52, p. 101589, 2022.

[5] J. Mainprice, E. A. Sisbot, L. Jaillet, J. Cortés, R. Alami, and T. Siméon, "Planning human-aware motions using a sampling-based costmap planner," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2011, pp. 5012–5017.

[6] M. Faroni, M. Beschi, and N. Pedrocchi, "Safety-aware time-optimal motion planning with uncertain human state estimation," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 12 219–12 226, 2022.

[7] B. Sakcak and L. Bascetta, "Safe motion planning for a mobile robot navigating in environments shared with humans," in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2022, pp. 2074–2079.

[8] M. Zucker, J. Kuffner, and M. Branicky, "Multipartite rrts for rapid replanning in dynamic environments," in *2007 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2007, pp. 1603–1609.

[9] M. Otte and E. Frazzoli, "RRTX: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *The International Journal of Robotics Research*, vol. 35, no. 7, pp. 797–822, 2016.

[10] C. M. Dellin and S. S. Srinivasa, "A unifying formalism for shortest path problems with expensive edge evaluations via lazy best-first search over paths with edge selectors," in *Proceedings of the Twenty-Sixth International Conference on Automated Planning and Scheduling (ICAPS)*, 2016.

[11] L. Huang and X. Jing, "Asymptotically optimal lazy lifelong sampling-based algorithm for efficient motion planning in dynamic environments," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 8861–8867.

[12] C. Tonola, M. Faroni, S. Abdolshah, M. Hamad, S. Haddadin, N. Pedrocchi, and M. Beschi, "Reactive and safety-aware path replanning for collaborative applications," *arXiv preprint arXiv:2503.07192*, 2025, arXiv:2503.07192v2.