AROB Practical Work

Álvaro Casals Luna, Gonzalo Valero Domingo

***Abstract*— This paper addresses the motion planning problem for autonomous mobile robots using sampling-based algorithms. Specifically, it presents a comparative analysis between the standard Rapidly-exploring Random Tree Star (*RRT**) and the Informed *RRT** algorithm in 2D navigation scenarios.**

While standard *RRT employs a uniform sampling strategy over the entire configuration space, Informed *RRT** restricts the sampling domain to an admissible heuristic subset, geometrically defined as a prolate hyper-ellipsoid. Both planners are implemented within the ROS 2 framework and integrated into a navigation pipeline, including the use of costmaps for collision checking and path validation.**

The performance of the algorithms is evaluated in simulated environments using metrics such as path cost and computation time providing an experimental assessment of the impact of heuristic-based direct sampling on planning efficiency.

I. INTRODUCTION

A. Motivation

Autonomous navigation remains one of the most critical challenges in mobile robotics, requiring the ability to find safe and efficient trajectories in complex, unstructured environments. In recent years, algorithms such as *RRT** and Informed *RRT** have shown great promise in real-world applications, ranging from autonomous vehicles navigating urban environments to drones performing search and rescue missions. These applications highlight the importance of optimizing planning algorithms to ensure both safety and efficiency in dynamic, real-world conditions.

Classical deterministic approaches, such as A^* or Dijkstra, provide optimal solutions in grid-based discretizations. However, their computational complexity grows exponentially with the dimensionality of the configuration space, making them inefficient for high-resolution maps or robots with high degrees of freedom. To overcome these limitations, Sampling-based Motion Planning (SBMP) algorithms have emerged as the standard solution. These methods explore the connectivity of the configuration space ($\mathcal{X}$) by randomly sampling points and building a graph or tree, offering a trade-off between completeness and computational speed. However, ensuring that the generated path is not just feasible, but optimal, introduces significant algorithmic challenges that motivate this study.

B. Related Work

The evolution of sampling-based planners has been driven by the need to balance exploration speed with solution quality. The *Rapidly-exploring Random Tree* (*RRT*), introduced by LaValle [1], revolutionized the field by efficiently exploring high-dimensional spaces through a bias toward unexplored regions. While *RRT* is probabilistically complete (it finds a solution if one exists), it is not asymptotically optimal; the paths produced are often jagged and suboptimal, and they do not converge to the global optimum regardless of execution time.

To address this, Karaman and Frazzoli proposed the *RRT** algorithm [2], which introduces a "rewiring" procedure. By checking local neighborhoods and restructuring the graph when a shorter path is found, *RRT** guarantees asymptotic optimality. However, *RRT** suffers from a significant limitation: its sampling strategy remains uniform over the entire space $\mathcal{X}_{free}$. Even after a viable path is found, *RRT** continues to expend computational resources exploring regions that cannot possibly yield a better solution, leading to slow convergence rates in practice.

This inefficiency led to the development of heuristic-driven variants. Gammell et al. introduced *Informed RRT** [3], which fundamentally changes the sampling domain once an initial solution is found. By restricting the search to an "Informed Set", geometrically defined as a prolate hyper-ellipsoid with foci at x_{start} and x_{goal}, the algorithm focuses exclusively on the subset of $\mathcal{X}$ that can potentially improve the current solution. This approach maintains the theoretical guarantees of *RRT** while drastically reducing the time required to converge to the optimal path.

C. Project Description and Contributions

In this work, we present a practical implementation and comparative analysis of these three planning paradigms within a modern robotics framework. Unlike theoretical simulations often performed in MATLAB or standalone C++ scripts, this project integrates the algorithms directly into the ROS 2 Humble Navigation Stack (Nav2).

We developed a custom global planner plugin, adhering to the `nav2_core::GlobalPlanner` interface, which allows for the seamless switching between *RRT*, *RRT**, and *Informed RRT** strategies at runtime. The implementation handles the transformation between the continuous world coordinates required for ellipsoidal sampling and the discrete costmap data used for collision checking.

The main contributions of this paper are:

- Development of a ROS 2-native global planner plugin implementing *RRT*, *RRT**, and *Informed RRT**.
- Implementation of a direct ellipsoidal sampling strategy using coordinate transformation matrices to handle rotation and scaling relative to the start-goal axis.
- A rigorous experimental evaluation in Gazebo, comparing the algorithms in terms of path cost convergence, success rate, and time-to-optimality across environments with varying complexity.

D. Paper Organization

The remainder of this paper is organized as follows: Section II describes the motion planning algorithms and the theoretical foundations of the informed sampling strategy. Section III details the algorithmic implementation and the ellipsoidal sampling mathematics. Section IV presents the experimental results and quantitative metrics. Finally, Section V concludes the study and future research directions.

II. ALGORITHM DESCRIPTION

The motion planning problem is addressed using sampling-based methods that probabilistically explore the configuration space $\mathcal{X}$. This section outlines the theoretical foundations of the algorithms used, focusing on the transition from feasibility to asymptotic optimality.

A. Standard RRT*

The *Rapidly-exploring Random Tree Star* (*RRT**) builds upon the standard *RRT* by introducing a rewiring mechanism that guarantees asymptotic optimality. The algorithm incrementally constructs a tree $\mathcal{T}$ rooted at x_{start}. At each iteration, a sample x_{rand} is drawn from the free space $\mathcal{X}_{free}$. The nearest node in the tree, $x_{nearest}$, extends towards the sample to create x_{new}.

Unlike *RRT*, which simply adds the edge, *RRT** considers a set of neighbors $\mathcal{X}_{near}$ within a radius r that typically decreases as the tree grows, around x_{new}. It performs two key operations:

1) **Parent Selection:** It selects the parent from $\mathcal{X}_{near}$ that minimizes the total cost from the root:

$$x_{parent} = \underset{x \in \mathcal{X}_{near}}{\arg\min} \left(Cost(x) + ||x - x_{new}||_2 \right) \quad (1)$$

2) **Rewiring:** It checks if any node in $\mathcal{X}_{near}$ can be reached more efficiently through x_{new}. If $Cost(x_{new}) + ||x_{new} - x||_2 < Cost(x)$, the edge is rewired, ensuring the tree topology improves continuously.

B. Informed RRT*: Focusing the Search

Standard *RRT** suffers from the "uniform sampling curse": even after finding a feasible solution with cost c_{best}, it continues to sample the entire $\mathcal{X}_{free}$ with equal probability. This is inefficient as samples generated far from the solution path cannot improve the best known solution.

*Informed RRT** addresses this by restricting the sampling domain. Once an initial solution is found, the search is constrained to an *Informed Set* $\mathcal{X}_{\hat{f}} \subset \mathcal{X}$, defined by an admissible heuristic.

The overall logic of the proposed planner is summarized in Algorithm 1. The sampling strategy switches dynamically based on the existence of a solution (c_{best}).

Algorithm 1 Informed RRT^*

Require: $x_{start}, x_{goal}, \mathcal{X}_{free}, N_{max}$
$\mathcal{T} \leftarrow \{x_{start}\}$, $c_{best} \leftarrow \infty$
for $i = 1$ to N_{max} **do**
  if $c_{best} < \infty$ **then**
    $x_{rand} \leftarrow SampleEllipse(x_{start}, x_{goal}, c_{best})$
  else
    $x_{rand} \leftarrow SampleUniform(\mathcal{X}_{free})$
  end if
  $x_{nearest} \leftarrow Nearest(\mathcal{T}, x_{rand})$
  $x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$
  if $ObstacleFree(x_{nearest}, x_{new})$ **then**
    $r \leftarrow Radius(|\mathcal{T}|)$
    $X_{near} \leftarrow Near(\mathcal{T}, x_{new}, r)$
    $x_{min} \leftarrow ChooseParent(X_{near}, x_{new})$
    $\mathcal{T}.add(x_{new}, x_{min})$
    $Rewire(\mathcal{T}, X_{near}, x_{new})$
    if x_{new} near x_{goal} **then**
      Update c_{best}
    end if
  end if
end for

1) The Informed Set Definition: In a 2D Euclidean space with path length as the cost metric, the optimal path between x_{start} and x_{goal} constrained by a maximum cost c_{best} must lie within a specific geometric region. The heuristic function $\hat{f}(x)$ estimates the minimum cost-to-go from a state x:

$$\hat{f}(x) = ||x_{start} - x||_2 + ||x - x_{goal}||_2 \quad (2)$$

The Informed Set is formally defined as:

$$\mathcal{X}_{\hat{f}} = \{x \in \mathcal{X} \mid \hat{f}(x) \leq c_{best}\} \quad (3)$$

Geometrically, this inequality describes a prolate hyper-ellipsoid where:

- The foci are x_{start} and x_{goal}.
- The transverse diameter (major axis length) is equal to c_{best}.
- The theoretical minimum path length is the focal distance $c_{min} = ||x_{start} - x_{goal}||_2$.

In dynamic environments, such as urban settings or search-and-rescue operations, robots must plan and adjust paths in real-time while avoiding moving obstacles. These conditions demand further optimization of planning algorithms to address issues such as time efficiency, adaptability, and safety.

2) Direct Sampling Strategy: The core innovation of Informed RRT^* is its ability to sample directly from this ellipsoidal subset. Instead of rejection sampling (which becomes inefficient as the ellipse narrows), the algorithm maps samples from a unit d-ball onto the hyper-ellipsoid.

As the algorithm runs and finds better paths, c_{best} decreases. This causes the volume of the hyper-ellipsoid to shrink, progressively focusing the random samples on the region containing the optimal homotopy class. If $c_{best} \to c_{min}$, the ellipsoid collapses to the line segment connecting

start and goal, maximizing convergence speed. The mathematical details of this transformation are described in the implementation section.

III. IMPLEMENTATION DETAILS

The proposed algorithms were implemented as a C++ plugin within the ROS 2 Humble ecosystem, specifically integrating with the Navigation 2 (Nav2) stack. This section details the software architecture, the specific algorithmic realizations, and the parameter configuration used.

A. ROS 2 System Architecture

Unlike standalone implementations, our planner is developed as a plugin adhering to the `nav2_core::GlobalPlanner` interface. This allows it to be dynamically loaded by the `planner_server` node at runtime, ensuring full compatibility with the standard ROS 2 navigation lifecycle.

1) Node and Topic Structure: To analyze the communication between nodes and topics in our navigation and control system, we used `rqt_graph`, a ROS 2 tool that provides a visual representation of node interactions. This tool helps identify connections and data flows within the system, enabling efficient analysis of its performance and potential areas for improvement.

The core logic resides within the `planner_server` node. As illustrated in Fig. 1, the planner interacts with the rest of the system through the following interfaces:

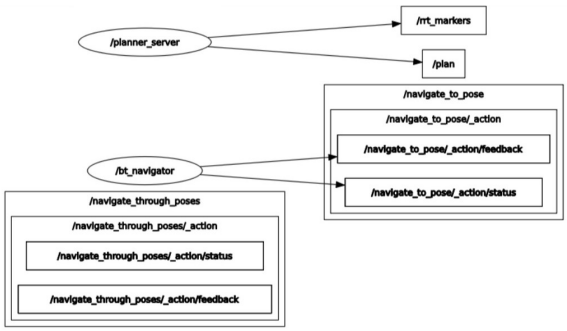


Fig. 1: ROS 2 Node Architecture. The graph shows the `/planner_server` node (where the plugin runs) publishing the calculated path to `/plan` and the debug visualizations to `/rrt_markers`.

- **Input Interfaces:** The planner receives the planning request via the `ComputePathToPose` action server. During real-time planning, the system uses sensor feedback (such as LiDAR or cameras) to continuously adjust the planned trajectory. The algorithms are designed to react quickly to detected obstacles, re-planning paths as necessary. It subscribes to the `/global_costmap/costmap` topic to receive updates about the environment's occupancy grid.
- **Output Interfaces:** The computed trajectory is returned as a `nav_msgs/Path` message through the `ComputePathToPose` action interface of the `planner_server`.
- **Visualization:** To facilitate debugging and performance analysis (as seen in Section IV), the internal state of the tree (nodes, edges, and informed ellipse) is published to the `/rrt_markers` topic using `visualization_msgs/MarkerArray`.

B. Algorithmic Realization

To bridge the gap between theory and efficient execution, two key mathematical components were implemented: direct ellipsoidal sampling and adaptive neighbor search.

1) Direct Ellipsoidal Sampling: For Informed RRT*, sampling from the heuristic subset $\mathcal{X}_{\hat{f}}$ is achieved by transforming a unit circle sample. Let x_{ball} be a sample from a unit d-ball. The informed sample $x_{informed}$ is computed as:

$$x_{informed} = C \cdot L \cdot x_{ball} + x_{center} \tag{4}$$

Where x_{center} is the midpoint between x_{start} and x_{goal}. $L = \text{diag}(a, b)$ is the scaling matrix based on the current best cost c_{best} and the focal distance c_{min}. C is the rotation matrix that aligns the ellipse with the start-goal axis:

$$C = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{5}$$

This transformation allows for $O(1)$ sampling complexity regardless of the ellipse's eccentricity.

2) Adaptive Search Radius: To ensure asymptotic optimality while maintaining computational efficiency, we implemented a dynamic search radius r_{RRT*}. As the tree grows, the radius for finding neighbors shrinks, reducing the number of costly collision checks. The radius is calculated at each iteration n as:

$$r_n = \min\left(\gamma \left(\frac{\log n}{n}\right)^{1/d}, \eta\right) \tag{6}$$

Here, $d = 2$ is the dimension of the space. η is the maximum steering range, acting as an upper bound to prevent connections through obstacles. γ_{RRT} is a constant based on the free space volume, ensuring that the ball always contains a sufficient number of nodes to guarantee asymptotic optimality while preserving probabilistic completeness.

IV. EXPERIMENTAL RESULTS

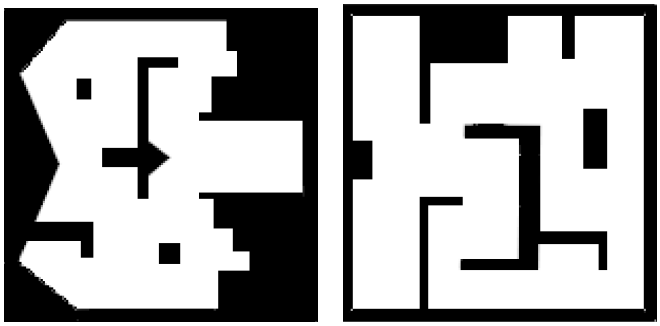
In this section, we present a comprehensive evaluation of the standard *RRT** and Informed RRT^* algorithms.

A. Experimental Setup and Testing Environment

The performance of the proposed algorithms was evaluated in two distinct simulated environments, designed to challenge different aspects of autonomous navigation such as obstacle avoidance in cluttered environments and path planning through narrow corridors.

1) Simulation Environments: The maps used for validation are depicted in Fig. 2.

- **Map A (Fig. 2a):** Represents a structured environment with clearly defined sections and static obstacles, evaluating the algorithm's capability to identify optimal paths within multiple homotopy classes.
- **Map B (Fig. 2b):** Represents a complex, maze-like environment with U-shaped structures and multiple potential paths to reach the goal, testing the algorithm's ability to identify optimal routes among different homotopy classes.

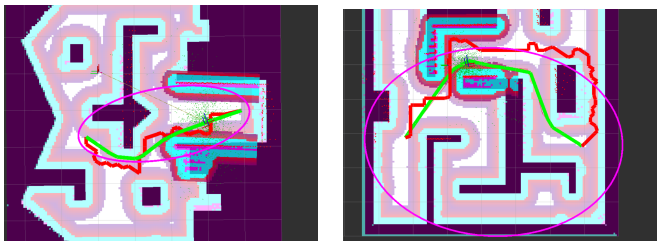


(a) Scenario A: Structured environment. (b) Scenario B: Complex maze layout.

Fig. 2: Layout of the two distinct maps used to validate the path planning algorithms.

2) Software and Hardware Stack: The simulations were conducted using the *Gazebo* physics engine for realistic dynamics and *RViz* for real-time visualization of the robot's sensor data and mapping process. The planning framework was implemented in C++ within the ROS 2 (Robot Operating System) Humble ecosystem.

The robot model employed is the *TurtleBot3 Waffle*, a differential drive mobile robot equipped with a 360° LiDAR sensor for localization and obstacle detection. Fig. 3 shows snapshots of the robot's performance in RViz during the simulation. These images capture the robot's movement, the planned path, and the sensor data in real-time as the algorithm navigates through the test environments.



(a) Navigation in Scenario A. (b) Navigation in Scenario B.

Fig. 3: Real-time simulation snapshots in RViz showing the robot's footprint, costmap layer, and the generated global path (red line).

3) Algorithm Configuration: To ensure a fair comparison, all planners shared the same configuration parameters regarding collision checking and kinematic constraints. These parameters, which are provided in meters, are then normalized within the algorithm by dividing them by the resolution value. The resolution is fixed at 0.05 meters per cell, meaning that a distance of 1 meter corresponds to 20 cells in the grid. This scaling ensures that the algorithm operates with consistent spatial precision across all planners. Table I summarizes the key parameters used during the experiments.

The parameters in Table I were selected based on standard values in the literature and preliminary testing, aiming to balance planning performance and computational cost. Their influence is later analyzed in Experiment 4.

TABLE I: Experimental Configuration Parameters

Parameter	Value	Description
Max. Samples (N)	5000	Maximum iterations per planning cycle
Goal Threshold (d_{th})	0.5 m	Max distance to connect node to goal
Steering Range (η)	0.2 m	Maximum expansion steering range (max_dist)
Rewiring Constant (γ)	1.0	RRT* ball radius factor
Map Resolution	0.05 m	Grid cell size

B. Experiment 1: Path Cost Convergence and Efficiency

This experiment evaluates the ability of each algorithm to find optimal paths as computational resources increase. We analyze convergence with respect to both the number of samples and execution time.

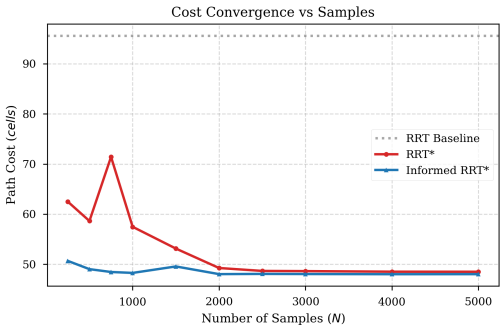
In experiments with larger maps, the number of samples required for efficient convergence increases significantly. However, in smaller maps or environments with simpler obstacles, the algorithms converge much faster. These findings highlight the importance of adjusting parameters according to the specific characteristics of the environment.

1) Results Analysis: Fig. 4 illustrates the evolution of path cost (Euclidean distance). The baseline *RRT* algorithm produces suboptimal paths with a high average cost (≈ 95 cells). While standard *RRT** improves upon this, refining the solution to ≈ 48.5 cells after $N = 5000$, Informed RRT^* exhibits superior efficiency. It converges to a near-optimal cost of ≈ 48 cells significantly faster, achieving a solution quality at $N = 2000$ that standard *RRT** struggles to match even at maximum samples.

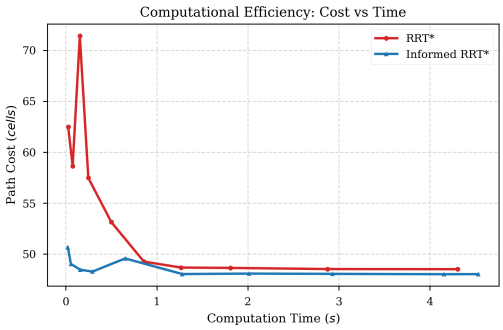
When analyzing time efficiency (Fig. 4b), the results highlight a significant advantage for the informed strategy. Informed RRT^* reaches a path cost of ≈ 48 cells in under 0.5 seconds, whereas standard *RRT** requires over 4 seconds to approach a similar cost. This represents a substantial improvement in time-to-optimality, suggesting that the reduction in search space outweighs the cost of heuristic calculations.

C. Experiment 2: Global Performance Improvement

To quantify the overall performance gains and robustness of the proposed approach, we conducted a statistical analy-

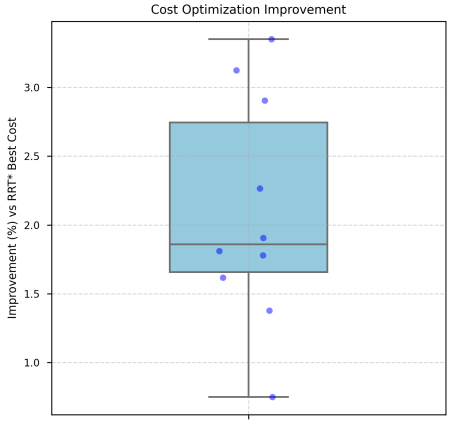


(a) Cost vs Samples (N)

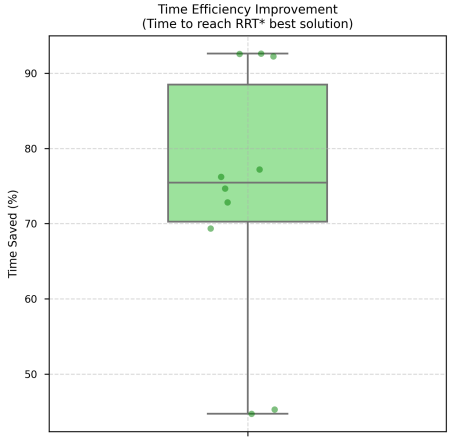


(b) Cost vs Time (s)

Fig. 4: Convergence Analysis. (a) Path cost evolution relative to the number of samples. (b) Computational efficiency correlating cost with execution time.



(a) Cost Improvement (%)



(b) Time Efficiency (%)

Fig. 5: Global performance metrics across 10 experiments. Boxplots show the distribution of improvements of Informed RRT^* over the standard baseline.

sis based on aggregated results from 10 distinct planning queries. This evaluation focuses on two key performance indicators: the reduction in path cost relative to sample size and the computational time required to reach a near-optimal solution.

1) Statistical Discussion: Fig. 5 presents the distribution of improvements.

- **Cost Optimization:** The analysis reveals a mean cost improvement of 2.09%. While modest, this is consistent with theoretical expectations: as $N \to \infty$, both algorithms are asymptotically optimal and will converge to the same global minimum. The improvement reflects that Informed RRT^* is simply further along the convergence curve for finite samples.
- **Time Efficiency:** The time efficiency improvement averages 73.78%. This metric signifies that, on average, Informed RRT^* requires approximately 74% less computational time to achieve a solution quality equivalent to the best result obtained by standard *RRT**. This drastic reduction in computational overhead is the critical validation for real-time robotic applications.

D. Experiment 3: Spatial Search Distribution

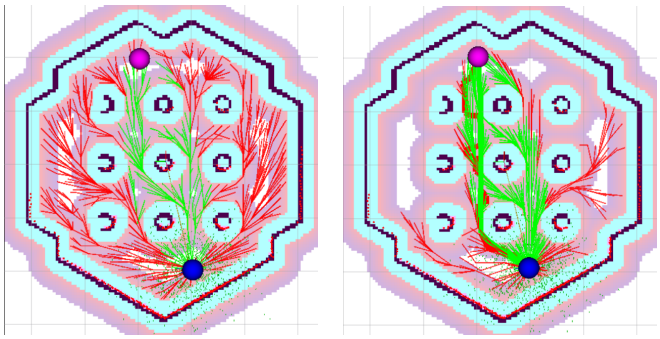
The efficiency gains observed in the previous experiments are visually explained by the spatial distribution of the search trees. Fig. 6 provides a qualitative comparison.

Standard *RRT** expands uniformly, wasting resources on exploring corners of the map far from the optimal path (Fig. 6a). In contrast, Informed RRT^* (Fig. 6b) demonstrates the focused search, where the tree grows almost exclusively within the prolate hyper-ellipsoid connecting the start and goal. This targeted exploration is the primary factor driving the observed improvements in convergence speed.

E. Experiment 4: Hyperparameter Sensitivity Analysis

1) Quantitative Analysis: We analyzed how performance varied with changes in key configuration parameters, specifically the steering range (η) and the rewiring radius factor (γ). This experiment aimed to assess the trade-off between exploration speed and path precision.

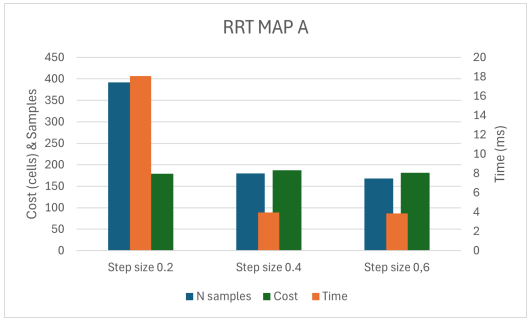
- **Impact on *RRT*:** As seen in Fig. 7, increasing η from 0.2 to 0.6 reduces the number of samples needed and improves execution speed, as nodes are connected at greater distances. However, this comes at the cost of precision, potentially increasing the final path length due to the inability to navigate tight corners optimally.



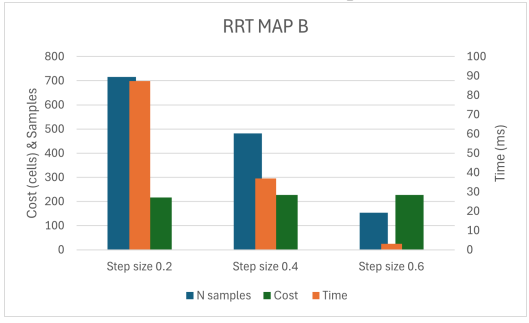
(a) Standard RRT^* (b) Informed RRT^*

Fig. 6: Visual comparison of tree expansion, where blue point is the start node and pink point is the end node. The green branches are the nodes that are inside the optimal ellipse. (a) Uniform exploration covering the entire map. (b) Ellipsoidal containment focusing density on the relevant region.

- **Impact on *RRT**:** Similar to *RRT*, increasing the steering range improves speed. Critically, the path length remains stable (Fig. 8), reflecting the algorithm's rewiring capability to maintain optimality. Additionally, increasing the ball radius size (γ) expands the set of potential parents (Fig. 9). While this theoretically improves path quality, it significantly increases execution time due to the higher number of collision checks required per iteration.



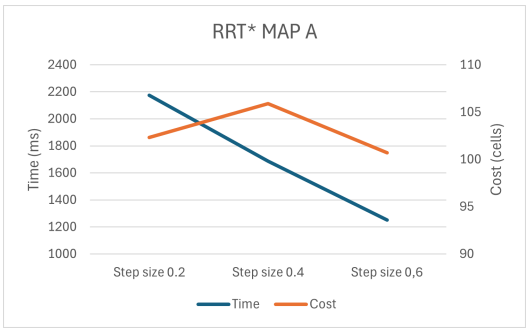
(a) *RRT* Metrics (Map A)



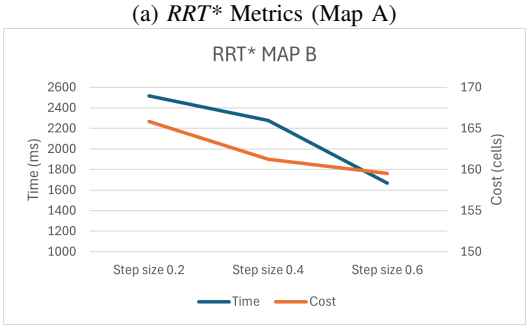
(b) *RRT* Metrics (Map B)

Fig. 7: *RRT* performance metrics with different steering range values.

2) Qualitative Analysis: To visualize these effects, Fig. 10 presents the generated paths for varying steering ranges.



(a) *RRT** Metrics (Map A)



(b) *RRT** Metrics (Map B)

Fig. 8: *RRT** performance metrics with different steering range values.

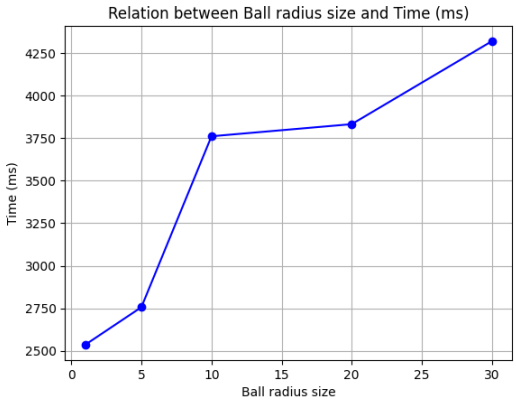
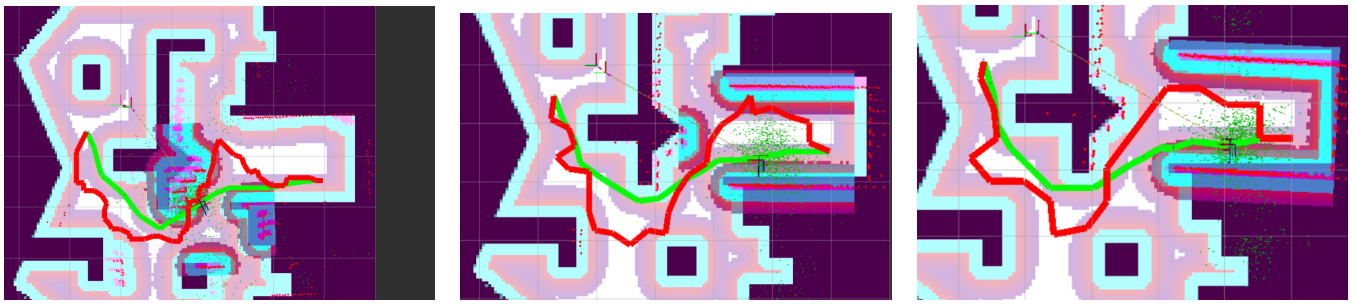


Fig. 9: Relation between ball radius size and execution time.

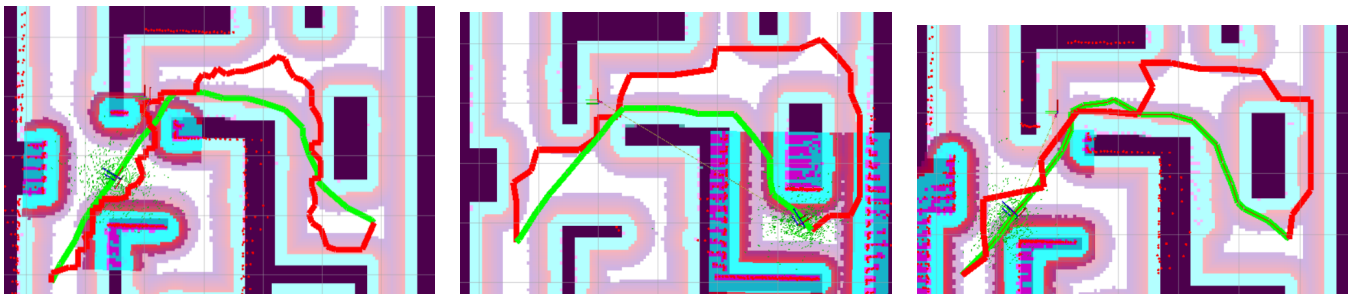
The standard *RRT* (red path) shows significant degradation in path quality as the steering range increases, producing jagged trajectories that may overshoot optimal turns. Conversely, the *RRT** (green path) maintains a smooth, near-optimal trajectory regardless of the steering step, demonstrating its robustness. This confirms that while larger steps accelerate exploration, the rewiring mechanism is essential for maintaining path quality in constrained environments.

V. CONCLUSIONS

This work presented a comparative analysis of sampling-based motion planning algorithms implemented within the ROS 2 framework. The study focused on evaluating the impact of admissible heuristic sampling in Informed *RRT** compared to the uniform exploration strategy of standard



(a) Map A: Range 0.2 (b) Map A: Range 0.4 (c) Map A: Range 0.6



(d) Map B: Range 0.2 (e) Map B: Range 0.4 (f) Map B: Range 0.6

Fig. 10: Qualitative comparison of path generation as steering range (η) increases. *RRT* (red) degrades in quality, while *RRT** (green) maintains optimality.

*RRT**.

The experimental results indicate that restricting the sampling domain to a prolate hyper-ellipsoid accelerates convergence toward near-optimal solutions. In particular, Informed RRT^* consistently achieved comparable solution quality with a significantly reduced computation time, highlighting the benefits of focusing the search on informative regions of the configuration space during the optimization phase.

Nevertheless, several limitations must be acknowledged. The efficiency gains of Informed *RRT** are only observed after an initial feasible solution is found. In environments dominated by narrow passages, where initial exploration is the primary bottleneck, the algorithm behaves similarly to standard *RRT**. To mitigate this, hybrid approaches could be employed, such as combining Informed RRT^* with PRM (Probabilistic Roadmap) to improve the exploration of the configuration space, or adjusting the sampling strategies to focus on narrower passage areas. Additionally, the performance of Informed RRT^* relies on the suitability of the Euclidean heuristic as a cost-to-go estimate. This heuristic may not be ideal in scenarios involving complex kinematic constraints or non-uniform cost functions. In such cases, more advanced heuristics could be used that take the robot's kinematic constraints and the non-uniform nature of the environment into account.

A. Future Work

Future work will aim to address these limitations by:

- Extending the planner to three-dimensional configuration spaces ($SE(3)$) for aerial robotics applications.
- Incorporating kinematic constraints to handle non-holonomic vehicles, where Euclidean heuristics may be insufficient.
- Integrating the global planner with a local trajectory optimization method, such as TEB, to improve performance in dynamic environments.

References

[1] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Computer Science Dept., Iowa State University, Tech. Rep. TR 98-11, 1998.

[2] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[3] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed rrt*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 2997–3004.