Autonomous Coverage Path Planning for Mobile Robot

Jiajun Li
Email: jil186@ucsd.edu
Department of ECE, UCSD

Tingyu Shi
Email: t5shi@ucsd.edu
Department of ECE, UCSD

***Abstract*—This paper presents the development of an autonomous robot system for coverage path planning (CPP) within a bounded environment. The objective was to design a system capable of navigating a 10ft $\times$ 10ft space with predefined landmarks, ensuring complete coverage using localization and control strategies. We employed a grid-based sweep algorithm integrated with a PID control module to achieve accurate motion. The performance of the system was evaluated through real-world experiments, and additional considerations were given to performance guarantees and SLAM-based mapping. Our results demonstrate the effectiveness of our approach in ensuring full coverage while maintaining localization accuracy.**

I. Introduction

The goal of this project is to develop a robot capable of autonomously navigating and covering a predefined area without relying on obstacle detection. The environment is a 10ft $\times$ 10ft space with AprilTag landmarks providing localization cues. Using insights from previous assignments, we implemented a coverage path planning algorithm combined with a motion control strategy to achieve efficient area coverage. This report details the system architecture, path planning methodology, implementation details, and experimental results.

II. System Architecture

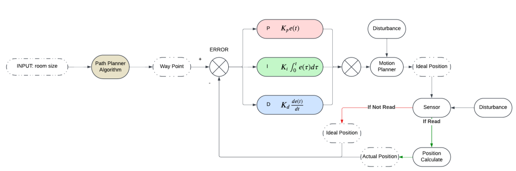

Fig. 1: System Architecture

Figure 1 illustrates the architecture of our system, which shares similarities with the one from our previous homework. However, the objective of this project differs significantly. Instead of finding the safest or shortest route to a specific goal, our focus is on designing a path that efficiently covers the entire area. While this new objective introduces a distinct challenge, the remaining components of the architecture remain consistent with those used previously.

III. Coverage Path Planning Algorithm

A. Algorithm Description

For the path planning algorithm, we used the Grid-based sweep algorithm. The first step is to discretize the space into different grids. Then, we use the center point of each grid to represent this grid. After discretizing the space, it can be considered as a 2D matrix, assume this matrix has N rows and M columns.

Algorithm 1 Grid-Based Sweep CPP Algorithm

Require: x_coords : *2D matrix with shape* $(N \times M)$ *representing x coordinates of all the grids*
Require: y_coords : *2D matrix with shape* $(N \times M)$ *representing y coordinates of all the grids*
$visited \leftarrow$ *2D boolean matrix with shape* $(N \times M)$ *full of False values*
$row\_idx, col\_idx \leftarrow N-1, 0$ ▷ Assume the car starts from left bottom and move upward
$waypoints \leftarrow []$
while $True$ **do**
  $visited[row\_idx, col\_idx] \leftarrow True$ ▷ Mark Visited Grid
  $x\_coord \leftarrow x\_coords[row\_idx,\ col\_idx]$ ▷ Extract X coordinate
  $y\_coord \leftarrow y\_coords[row\_idx,\ col\_idx]$ ▷ Extract Y coordinate
  $waypoints \leftarrow waypoints + [[x\_coord,\ y\_coord]]$ ▷ Update Waypoints
  if Robot Can Move to the Right Grid **then**
    $col\_idx \leftarrow col\_idx + 1$
  else if Robot Can Move to the Left Grid **then**
    $col\_idx \leftarrow col\_idx - 1$
  else if Robot Can Move to the Up Grid **then**
    $row\_idx \leftarrow row\_idx - 1$
  else
    $Break$
  end if
end while

Algorithm 1 shows the pseudo-code of our coverage path planning algorithm. The car can only move to a certain grid if that grid has never been visited. Figure 2 shows the planned path. The generated path is in a zigzag shape. Due to the noise in the localization, we also set certain distances between the path and the wall so that the car does not hit the wall. In Figure 2, the green line represents the safe distance between the path and the up/bottom wall, they are all 0.3m. The orange line represents the safe distance between the path and the left/right wall; they are all 0.24m. The black line in Figure 2 represents the planned path, the distance between

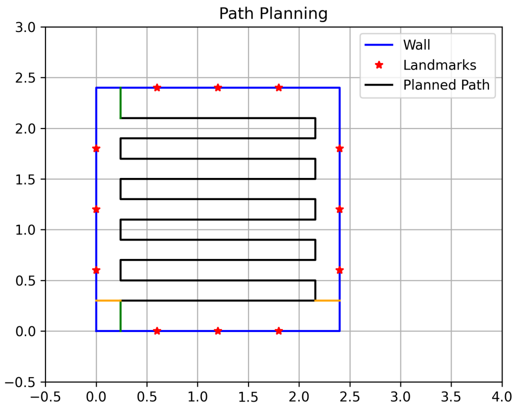

Fig. 2: Planned Path

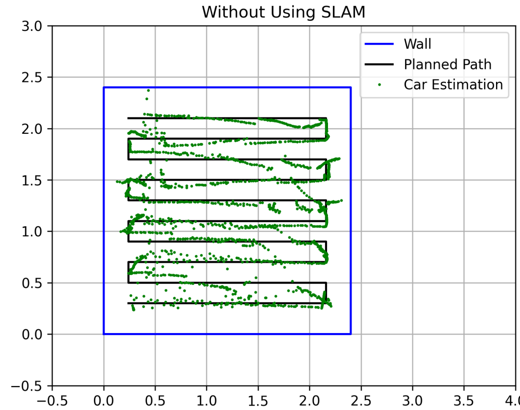

Fig. 3: Car Motion vs. Planned Path

any consecutive horizontal lines is 0.2m, which is the same as the car's width. By doing this, we can maximize our area coverage.

B. Coverage Guarantee

As shown in algorithm 1, the algorithm stops if and only if the car cannot move left, right, and up. By designing in this way, the algorithm can guarantee that every discretized grid can be visited. Assume that the square space has a side length of S, the car's width is W, and the square space is vertically separated into N rows, then as long as we can ensure that $\frac{S}{N} \leq W$, we can say that this algorithm can reach 100% coverage. However, this needs accurate motion control and localization.

IV. PID Control and Motion Module

For each waypoint, the desired location and pose are first set. The error is then calculated using either the position and pose estimated by the april detection node or those estimated by the algorithm. This error is processed by the PID controller, which generates a twist signal. This signal serves as input for the MPI twist control node and also provides feedback for the closed-loop system when no tag is detected. The MPI twist control node calculates and sends PWM signals to the four motors using the kinematic model. These motors generate the robot's real-world motion, which is captured by the camera. Finally, if the april detection node detects at least one tag, the current state is estimated based on the results provided by the april detection node.

V. Results and Evaluation

Figure 3 illustrates the robot's motion along the planned path. The robot successfully follows the trajectory with minor deviations due to localization noise. The system achieves nearly 100% coverage while maintaining safe distances from the environment boundaries.

VI. Future Work

While our system successfully achieves full coverage path planning within a predefined environment, several enhancements can be made to improve performance and robustness. Future work will explore the following areas:

- **Dynamic Obstacle Avoidance:** Implementing real-time obstacle detection and avoidance using sensor fusion techniques such as LiDAR and depth cameras.
- **Enhanced Localization:** Improving localization accuracy by integrating sensor fusion algorithms, such as Kalman Filters or Particle Filters, to mitigate errors in pose estimation.
- **Adaptive Path Planning:** Developing an adaptive path planning algorithm that dynamically adjusts the coverage strategy based on environmental changes and real-time data.
- **SLAM Integration:** Incorporating a custom SLAM (Simultaneous Localization and Mapping) algorithm to enable the robot to build a map of the environment while performing coverage tasks.
- **Hardware Optimization:** Evaluating and optimizing the motor control system to improve energy efficiency and motion stability.
- **Multi-Robot Coordination:** Extending the system to support multiple robots working collaboratively to achieve efficient coverage in larger areas.

These enhancements will contribute to a more robust and intelligent robotic system, expanding its applicability to real-world scenarios beyond controlled environments.

VII. Conclusion

This work presents a coverage path planning system for an autonomous robot using grid-based planning and PID control. Our approach effectively ensures full area coverage in a predefined environment while leveraging localization for situational awareness. Future work will explore improved localization techniques and dynamic obstacle avoidance.