

Implementation of A* Motion Planning in ROS 2 for Obstacle Avoidance in a 2D Occupancy Grid

- Mohamed Eljahmi
- RBE 550: Motion Planning
- Worcester Polytechnic Institute (WPI)
- Fall 2025




Motivation and Problem

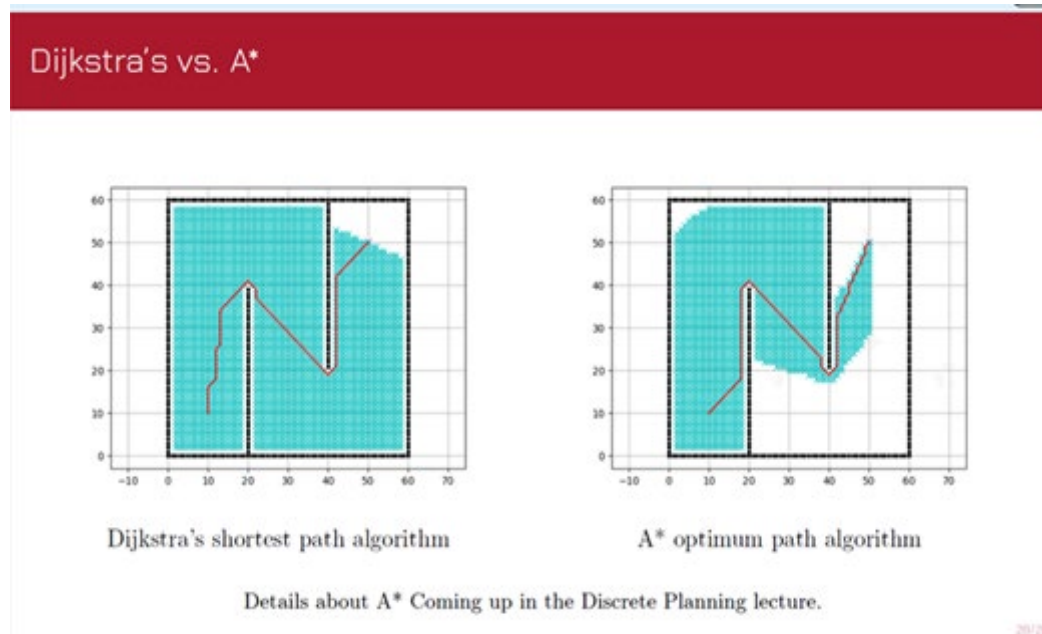
- Motion planning is essential for safe, efficient robot navigation
- Applications: warehouses, aerial robotics, surgical robots
- Motivating scenario: warehouse robot navigating shelves
- Challenge: Efficient, collision-free path planning on 2D occupancy grids



Background: Graph-Based Search Methods

- Classical search methods like Breadth-First Search and Depth-First Search are complete but very inefficient in large spaces.
- Dijkstra's algorithm improves this by assigning costs to edges and always expanding the lowest-cost node. This guarantees an optimal path, but it still explores in all directions, which can waste time.
- A* builds on Dijkstra by adding a heuristic. The heuristic guides the search toward the goal, making it much faster while still giving an optimal path if the heuristic is admissible.
- This diagram shows the key difference — Dijkstra expands evenly everywhere, while A* focuses the search in the direction of the goal. 

Visual Comparison between A* and Dijkstra



As shown in the diagram, Dijkstra expands evenly everywhere, while A-star focuses more in the direction of the goal.




Origin of the Algorithms

- Dijkstra's algorithm was invented in 1956 by Edsger W. Dijkstra, a Dutch scientist. He came up with it during a trip in Amsterdam, while thinking about how to find the shortest routes.

Later, in 1968, Hart, Nilsson, and Raphael created A star at SRI International. They extended Dijkstra's work by adding heuristics, which made the algorithm much more efficient for guiding searches toward the goal



Proposed Methods

- Framework in ROS 2 : modular node-based design
- Algorithm: A* (baseline) with configurable heuristics
- Inputs: occupancy grid (/map), start and goal poses
- Outputs: planned path (/planned_path), visualization in RViz
- Planned Figure: occupancy grid + computed path (to be added later) 

Goals and Evaluation

- **Path Planning:** Compute collision-free paths using A*
- **ROS 2 Integration:** Use standard message types
- **Visualization:** RViz display (occupancy grid, path, search expansion)
- **Evaluation Metrics:**
 - Path length
 - Nodes expanded
 - Computation time
- **Extensions:** BFS, DFS, Dijkstra, real hardware tests



Schedule and Expected Results

- Weeks 1–2: Literature review, ROS 2 setup
- Weeks 3–4: A* implementation (not started yet)
- Weeks 5–7: ROS 2 integration, testing
- Weeks 8–9: Evaluation (synthetic environments)
- Week 10: Results, video, final report
- Expected outcome: reproducible ROS 2 package, clear visualization of A*

