CS/ME 469: Machine Learning and Artificial Intelligence for Robotics
Assignment 1: Search and Navigation
Author: Jared Berry

Introduction

The topic of the following report is an implementation of the A* pathfinding algorithm, and the design of several subsystems that allow robust robot path following. The report will begin with a discussion of the A* algorithm's design, followed by a description of a path-following algorithm for a differential drive robot, culminating in a comparison between grid resolutions and an analysis of results.

The dataset used in this report is the provided ds1, which was used to place the obstacles in the robot's environment.

Section I corresponds to Questions 1-7, Section II corresponds to Questions 8-10, and Section III will discuss the remaining questions.

I. A* Algorithm Design and Implementation

A* is a heuristic-based pathfinding algorithm that is capable of finding the optimal path in a discretized environment. During the search process, it assigns nodes a score based on an informed sum; the true cost of navigating to a node is added to the estimated heuristic cost of traveling from that node to the goal. While the true cost is often a constraint or contextually informed, the heuristic function is an important design decision that will determine the converged path's optimality. The A* implementation in this report uses a simple but effective heuristic: Euclidean distance (Figure 1).

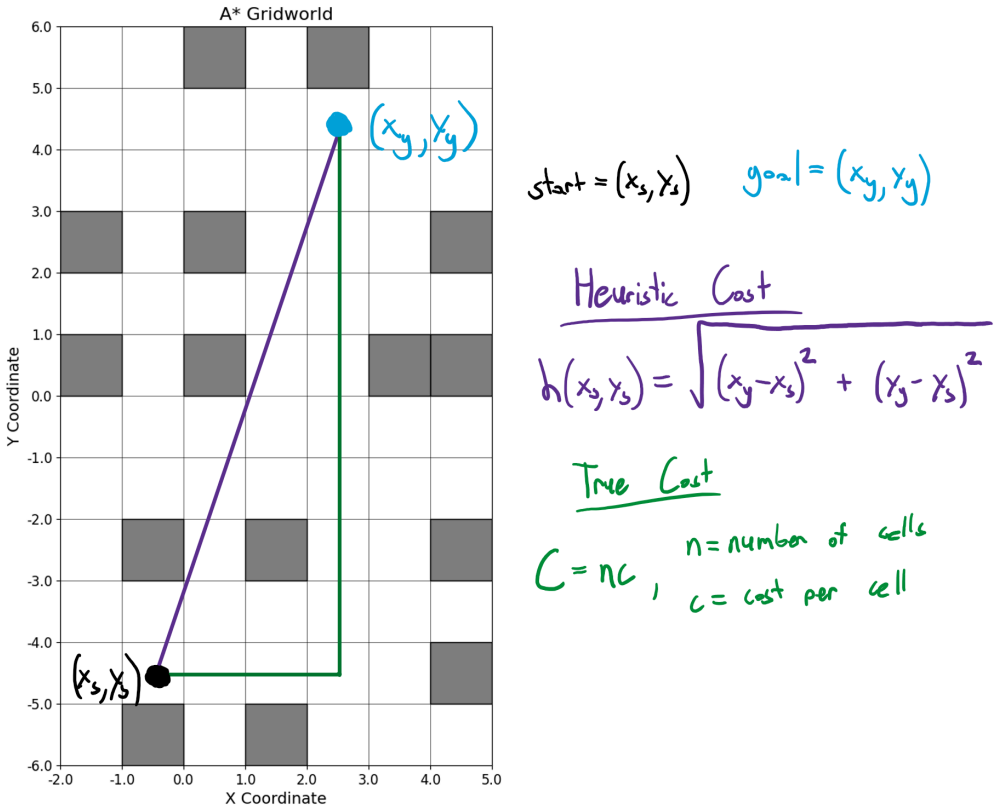

Figure 1: Euclidean distance heuristic and example of admissibility for Question 2. For the given problem and true cost, the heuristic cost will always be an underestimate.

For A* to converge to the optimal path, the estimated cost heuristic must be admissible. This means the heuristic must never overestimate the cost to the goal, because if an optimal path is overestimated it may never be pursued during the search process [1]. Euclidean distance is admissible because it minimizes the distance between two

points, meaning any other estimate of distance will be greater. In the given problem, the true cost is +1 to move into unoccupied neighboring cells (including diagonals), and +1000 to move onto obstacles. Because the true cost deals with distance (instead of energy or time), the Euclidean distance will always be an underestimate.

Because robots exist in a continuous domain, the environment must be discretized to a specified resolution. This specific A* implementation rounds starting and goal states to the resolution, and converts them to an internal integer representation to prevent floating point error. The bottom left corner of each cell is treated as that cell's coordinates in the A* node graph. A custom Node object tracks every node's position, cost, and lowest cost parent during search. The graph is traced in reverse to recover the converged path, and then converted back to the original floating point representation. The final plotted paths for the given example set of starting and ending points can be seen in Figure 2, with a coarse grid resolution (res=1.0 meters).

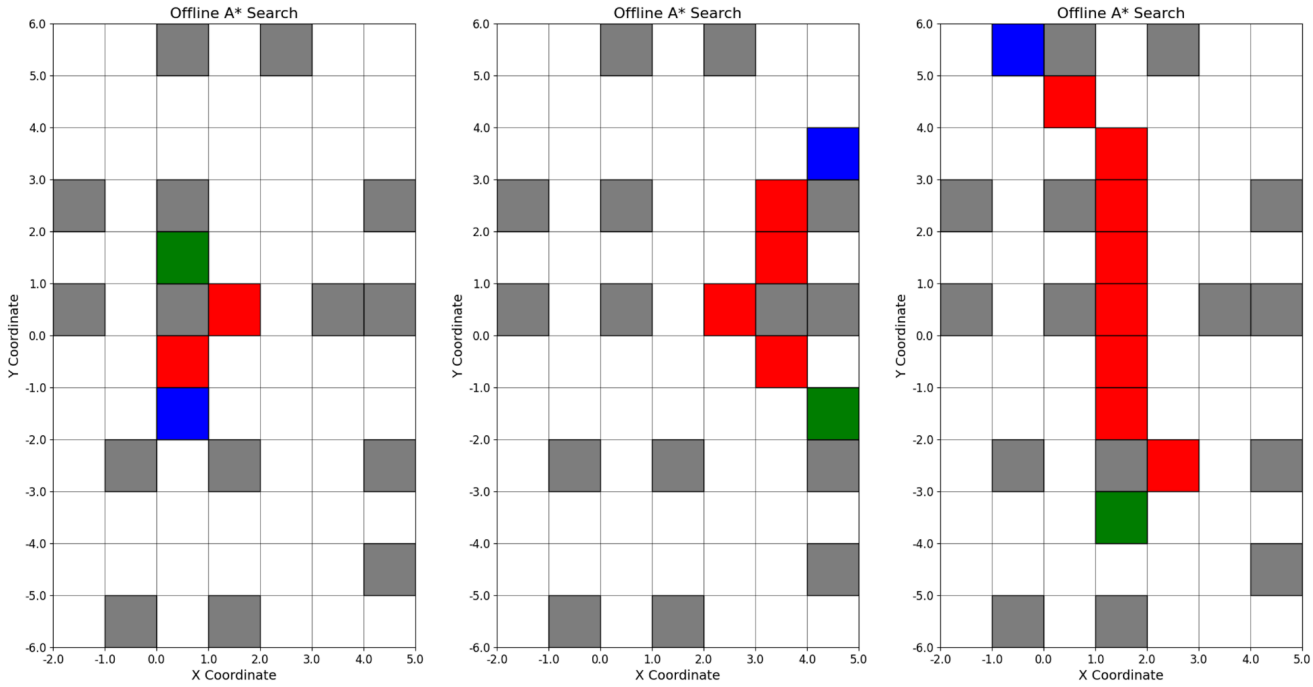

Figure 2: Offline A* examples for Question 3. The converged path is traced in red, while the starting and goal cells are marked in blue and green respectively.

While useful in some applications, offline A* alone is not practical for robotics. In order to explore an environment, a robot must physically follow the path, and will likely not have prior knowledge of obstacles. Much more relevant is an online A* implementation on a finer grid resolution, in which a robot physically explores its environment when expanding nodes on the A* graph.

Starting from an empty set of known obstacles, a naive A* path is planned and followed using the offline algorithm. If an obstacle is encountered while following the path, it's added to the set of known obstacles and the path is replanned with the updated set, starting from the robot's current position. This process is repeated iteratively until reaching the goal state. Figure A1 demonstrates the online variant's performance on the same set of start and goal positions as Figure 2.

Finer grid resolutions are also more realistic in robotics applications, as they more accurately represent the scale of the environment. To compensate for this finer resolution, the obstacle locations must be inflated to maintain their size. Figure 3 demonstrates the online A* algorithm's performance for the given set of starting and goal coordinates on a fine grid resolution (res=0.1 meters). Admittedly, this implementation is a "soft" online algorithm, as it doesn't simulate robot dynamics. The following section will discuss a more realistic version that includes such features.

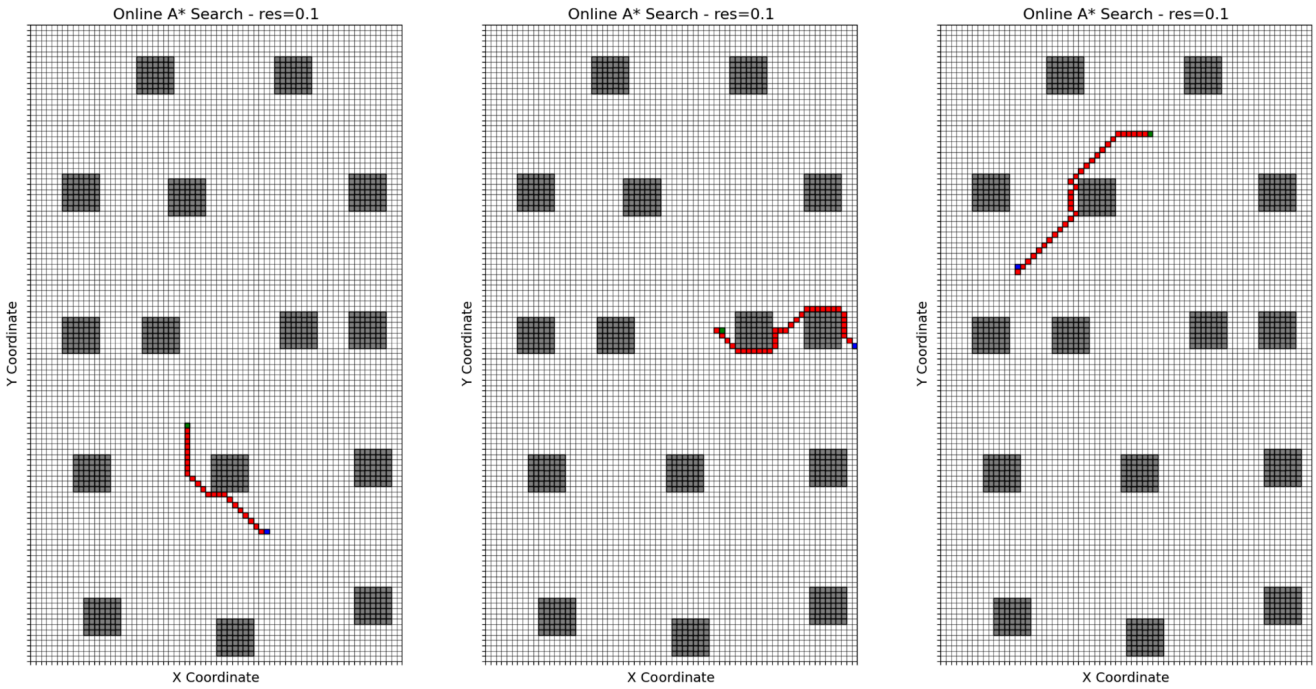

Figure 3: Online A* examples at a fine grid resolution (res = 0.1 meters) for Question 7.

II. Robot Controller and Path Following

While the online A* implementation in the previous section is technically more representative of a realistic robotics application than the offline version, the dynamics and control of a physical robot are not a consideration in its design. The following section will discuss the design of a controller for a differential drive robot that incorporates simulated process noise and a potential field setup for collision avoidance.

The simplified dynamics of a 2D differential drive robot consists of three states and two controls: the robot's position and heading in Cartesian space, controlled by forward and rotational velocity commands. These velocity commands are set proportionally to the straight line position and heading error to a specified destination point. The robot's dynamics are simulated using 4th-order Runge-Kutta (RK4) integration for higher accuracy and stability (Figure 4).

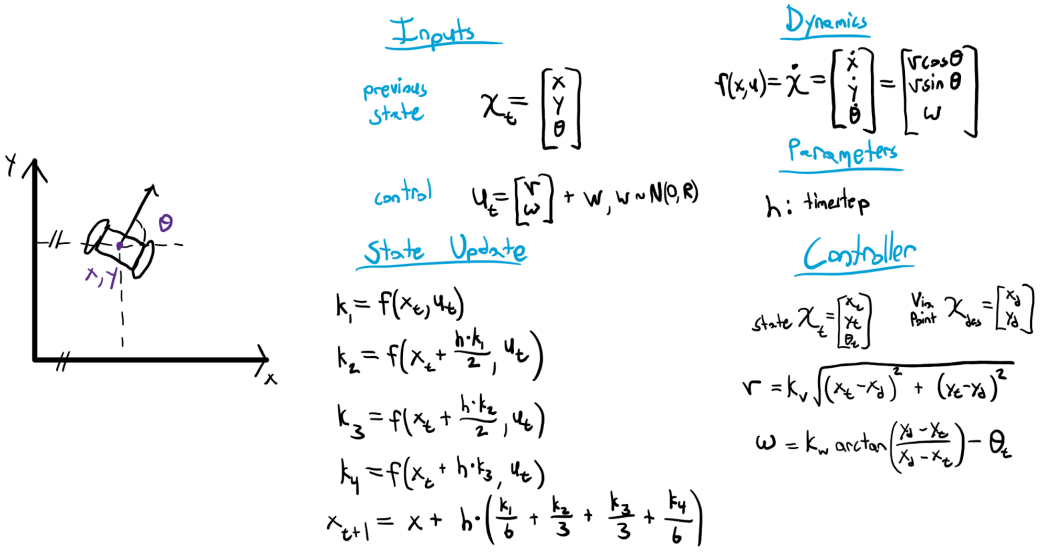

Figure 4: Robot motion model and controller for Question 8. Adapted from [2]

In the assigned problem there are two scenarios (Question 9 vs. Question 10): the robot must either follow a path found in the previously described online A* algorithm, or it must physically move during the online A* process. If a converged path from the online A* implementation is available, it's converted into a finer resolution trajectory of via points using third order spline interpolation. In this case, the robot's desired position is set to each point along the interpolated trajectory. If a full predetermined path isn't available, the robot follows the same naive online exploration process described in Section II, where the assigned path is updated iteratively upon encountering unseen obstacles. In this case, the straight line distance between the robot and the next cell in the naive A* path is linearly interpolated into a finer set of points that are followed in the same manner as the first case. Without interpolation the robot is still capable of following the path, but in a less direct manner. As interpolation becomes less sparse, the robot is able to adhere more strictly to the optimized A* path.

To make the controller more robust, it considers both process noise and obstacle avoidance. Process noise is sampled from a Gaussian centered at zero with specifiable variance, and added to the control signal. The variance is represented by an identity matrix multiplied by a scalar variance constant, p. Control velocities are then restricted according to acceleration limits and commanded velocities at the previous time step. Collision avoidance is accounted for with two methods, both inspired by potential fields. First, all via points within one grid cell of an obstacle are shifted away by a specified step size. Second, the robot itself is forced away from obstacles by shifting its commanded velocity vector proportionally to the inverse of the distance between the robot and nearby obstacles (within a certain distance threshold). These two strategies prevent collisions without any collision simulation or hardware sensing. Figure 5 compares the final trajectory of the simulated differential drive robot on the predetermined online A* path and the trajectory during real-time exploration.

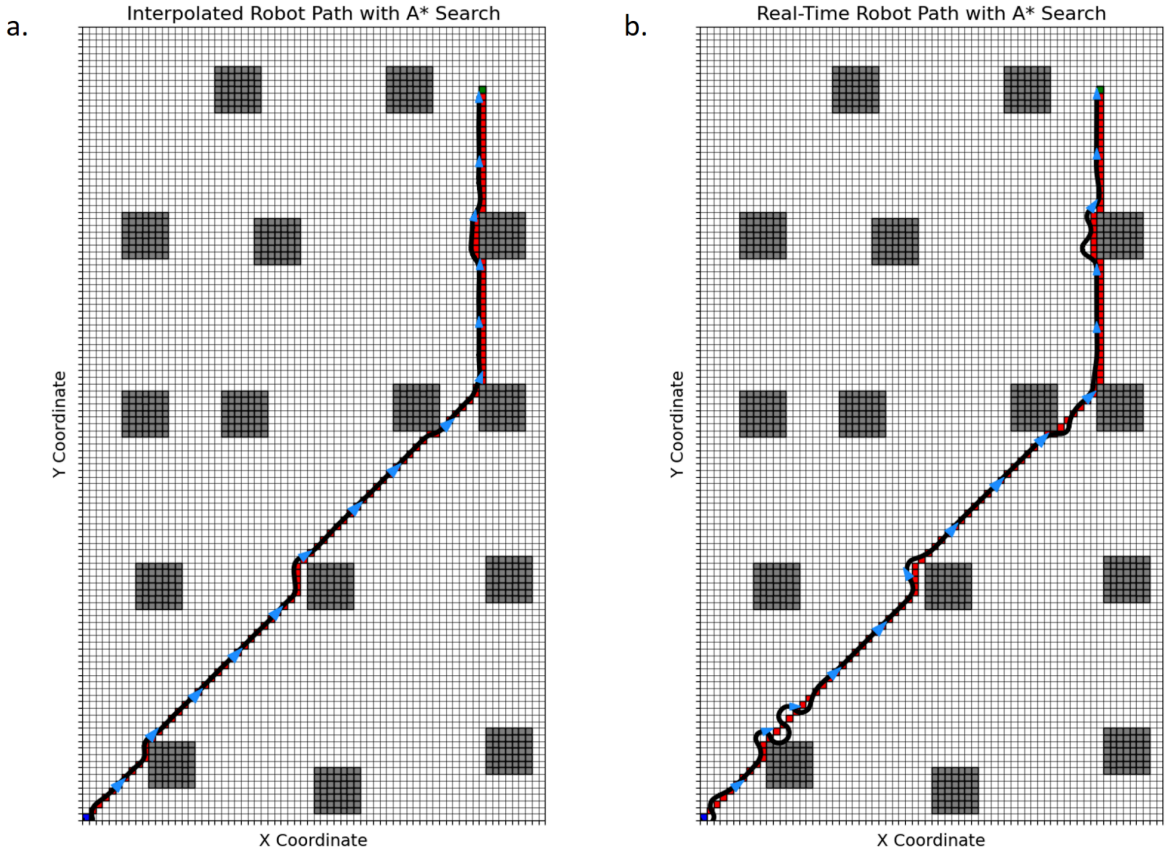

Figure 5: Final robot trajectory during path following. a) Performance with a predetermined path from the online A* algorithm, using high resolution spline interpolation. b) Recorded trajectory during real-time exploration with simulated dynamics for the same start and goal coordinates.

Final figure outputs for Questions 9 and 10 can be found in figures A2 and A3, respectively. The path in Figure 5 was determined to be a more useful comparison due to the length of the planned path. In all figures displaying the robot trajectory, the path followed by the robot is plotted in black and the robot's position and orientation are plotted in intervals along the path with light blue arrows.

Collision avoidance was the most time consuming feature to implement in this section. In a hardware system, there would be a greater selection of sensors that could be used to detect collisions, and more importantly, determine when paths are clear. Using a vision or ultrasonic sensor in this case would simplify the control problem because vectors to obstacles would not necessarily need to be continually computed in order to avoid them. While the current collision avoidance is functional, it would be more useful if it also tracked detected obstacles not directly intersecting with the planned path. There are some cases in the current implementation where an obstacle is not directly impeding the robot, but the differential drive dynamics bring it dangerously close to an impact. With more fleshed out sensing, the potential field-inspired collision avoidance system would be more useful.

III. Analysis and Discussion of Noise and Resolution

The final section of this report will analyze the effects of noise on the path-following robot controller, and demonstrate its robustness. This section will also compare the effects of varying grid resolutions on the final performance of both the A* pathfinding and the simulated path following.

Noise is inherent in any physical system, but its effects can be mitigated through smart design decisions in a robot's controller. The controller in this report uses an interpolated via point trajectory to reduce the distance between each subsequent point along a trajectory, effectively providing denser feedback for closed-loop control. Increasing the amount of feedback makes the controller more robust to process noise, resulting in acceptable performance for several orders of magnitude of noise levels. A noise experiment was performed to determine how varying the amount of noise in the system affects robot path following. The ground truth trajectory is the final trajectory from the noiseless real-time A* algorithm. Trials were run with moderate noise (p=0.01) and high noise (p=0.1). Representative metrics in Table 1 demonstrate the controller's consistent performance across noise levels. Visualizations of this experiment can be seen in Figure A4.

Table 1: Representative statistics for noise level comparison experiment. Metrics used are cumulative root mean squared error (RMSE) and Pearson correlation coefficient (CORR), and are in reference to the ground truth.

Metric	Moderate Noise (p=0.01)	High Noise (p=0.1)
RMSE x	1.17	2.35
RMSE y	1.71	4.81
RMSE θ	5.12	6.23
CORR x	0.998	0.960
CORR y	0.999	0.987
CORR θ	0.238	0.115

Discretization is an inescapable issue when using A* for path finding in robotics. While it guarantees optimality in its discrete operational space, there is no way for it to do so in a continuous domain unless its grid resolution approaches infinity. Thus, grid resolution can also be an important design decision for search and path-following algorithms. While finer resolutions may offer more precision, they also require more computation. Coarse resolutions may offer improved convergence speed, but the discretized representation of the optimal path may not hold up well in continuous space. For the implementation in this report, only the A* path planning is done at specifiable resolution. For collision checking and motion model simulation, a finer resolution with inflated

obstacle sizes is used for improved precision during the potential field velocity calculations and to improve overall stability.

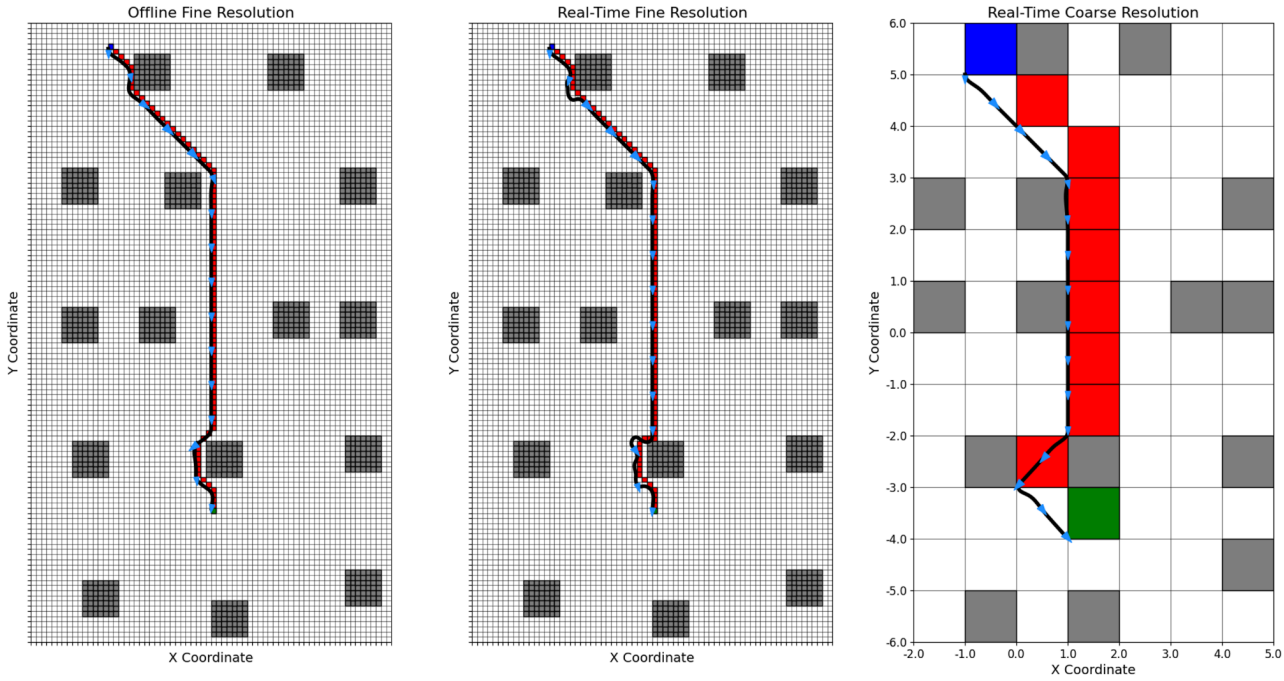

Figure 6: Planned path and resulting robot trajectories at different A* grid resolutions for the same start and goal positions. From left to right: offline ground truth, real-time fine resolution, and real-time coarse resolution.

A comparison of the final robot trajectory at different resolutions can be seen in Figure 6. While all A* paths follow the same general shape, the finer resolution clearly allows for a more precise representation of obstacles and the environment. This allows for more control over the robot's behavior, start, and goal. Conversely, the coarse resolution grid clearly has the advantage in speed, requiring both less runtime and fewer total simulated timesteps than the fine resolution. Table 2 contains representative metrics calculated from the robot's final trajectory and performance at res=1.0 meters and res=0.1 meters. The controller's performance on the path found in the offline A* algorithm without simulated process noise is treated as the most optimal case, i.e., the ground truth.

Table 2: Statistics calculated during the grid resolution experiment.

Metric	Fine Resolution (res=0.1 m)	Coarse Resolution (res=1.0 m)
RMSE x	1.05	4.30
RMSE y	1.71	4.67
RMSE θ	4.23	3.61
CORR x	0.998	0.884
CORR y	0.999	0.999
CORR θ	0.465	0.645
Runtime [seconds]	0.442	0.393
Total Number of Timesteps	1,100	967

Figures A5 and A6 contain additional visualizations comparing grid resolutions for Question 11.

While it does incorporate collision avoidance and simulated process noise, the path planner and controller described in this report admittedly operate on the basis of several unrealistic assumptions. Primarily among them is the assumption that the environment is entirely static. Moving obstacles would significantly decrease the efficiency of the online A* algorithm, as the set of known obstacles would be constantly shifting. This would potentially require more naive planning iterations, which would quickly become computationally unrealistic. This drawback could be mitigated by instead using a path planning algorithm that supports dynamic environments, such as D* or Field D*.

Another unrealistic assumption is the model of the robot's physical body (or rather, the lack thereof). A physical robot would take up some nonzero volume of space in its environment, which significantly increases the complexity of the controller formulation. The controller would need to consider collisions of the robot's geometry instead of just point collisions, and command wheel velocities instead of simplified forward and rotational velocities. Because the robot is currently represented as a point in space with simulated differential drive dynamics, it is capable of performing tight turns that a physical robot would not be capable of. A more accurate simulation would give the robot's geometry additional consideration.

Conclusion

This report has described the design and implementation of the A* path finding algorithm, and the simulation and control of a 2D differential drive robot. Consideration is given to process noise and collision avoidance, more accurately representing a robot in a physical environment. A comparison is made between varying noise levels and discretization resolutions, ending with a discussion of potential design tradeoffs and limitations.

Citations

[1] Y.-B. Jia, *Lecture Notes for Principles of Artificial Intelligence (COMS 4720/5720)*, Iowa State University, Ames, IA, USA.

[2] J. Berry, *CS/ME 469: Machine Learning and Artificial Intelligence for Robotics - Assignment 0: Filtering Algorithms*, course report, Northwestern University, Evanston, IL, USA, 2025.

Appendix

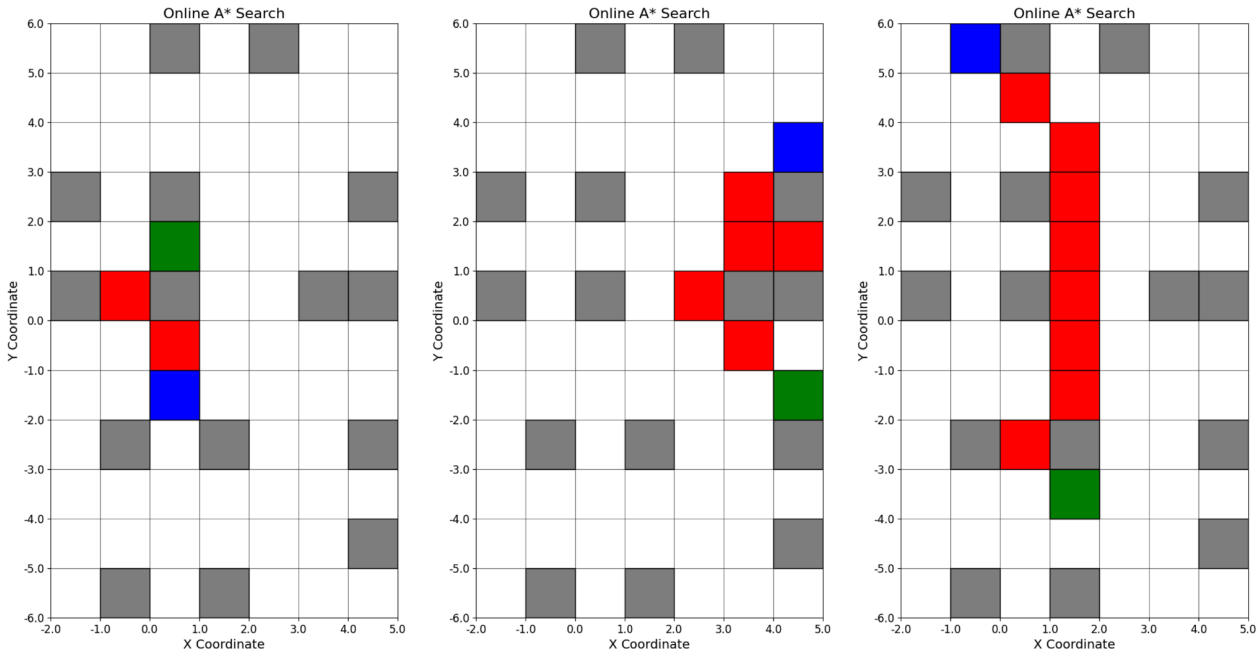

Figure A1: Plotted results from the online A* algorithm for Question 5.

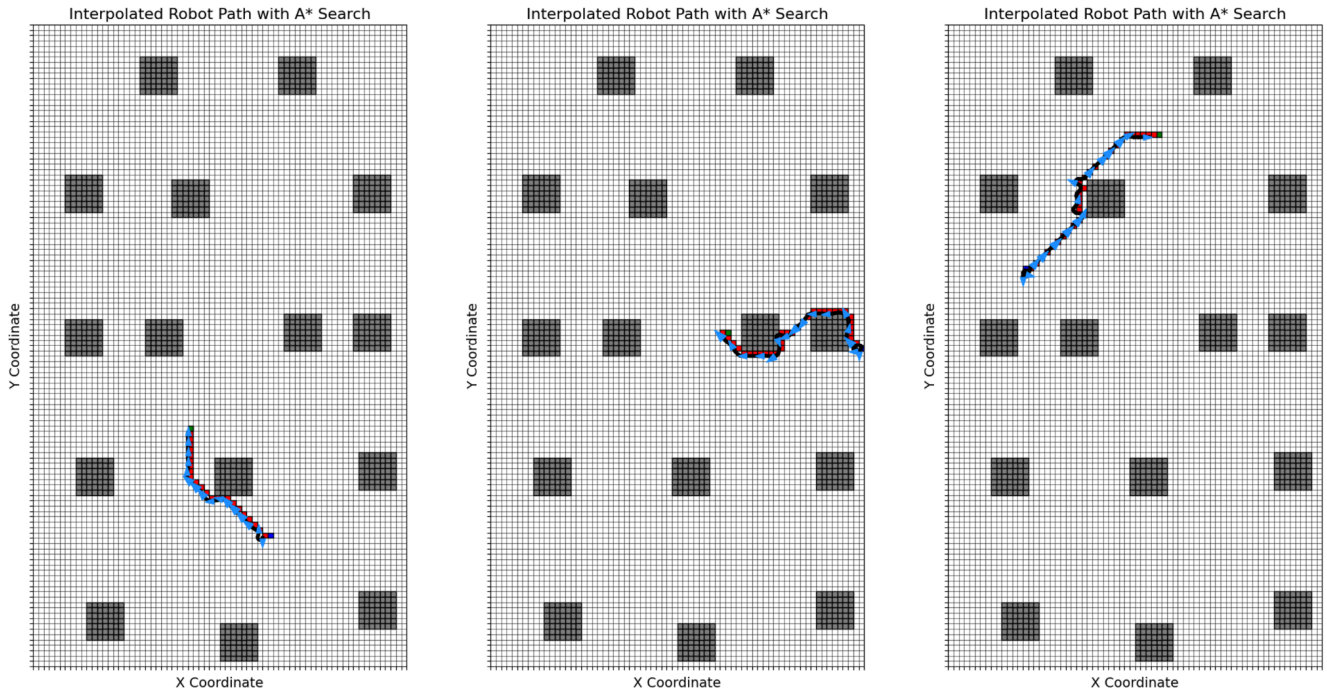

Figure A2: Plotted results from using the robot controller on the online algorithm paths for Question 9.

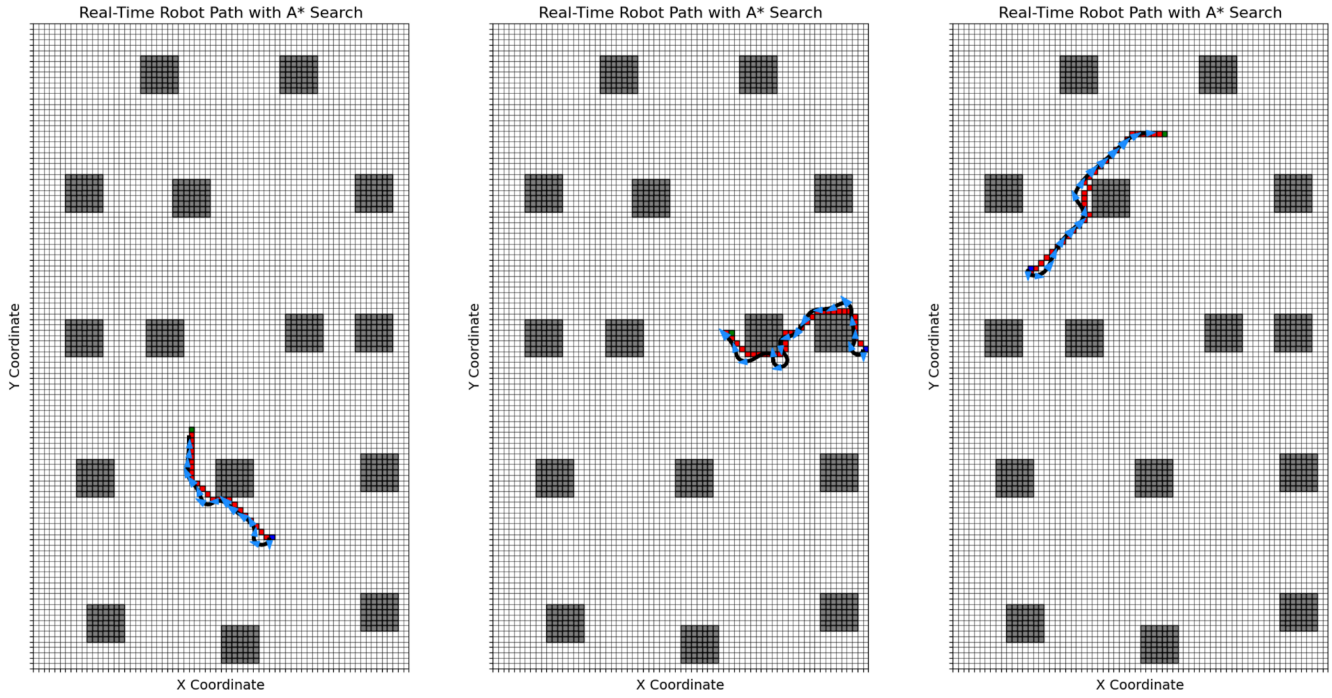

Figure A3: Plotted results from real-time exploration for Question 10.

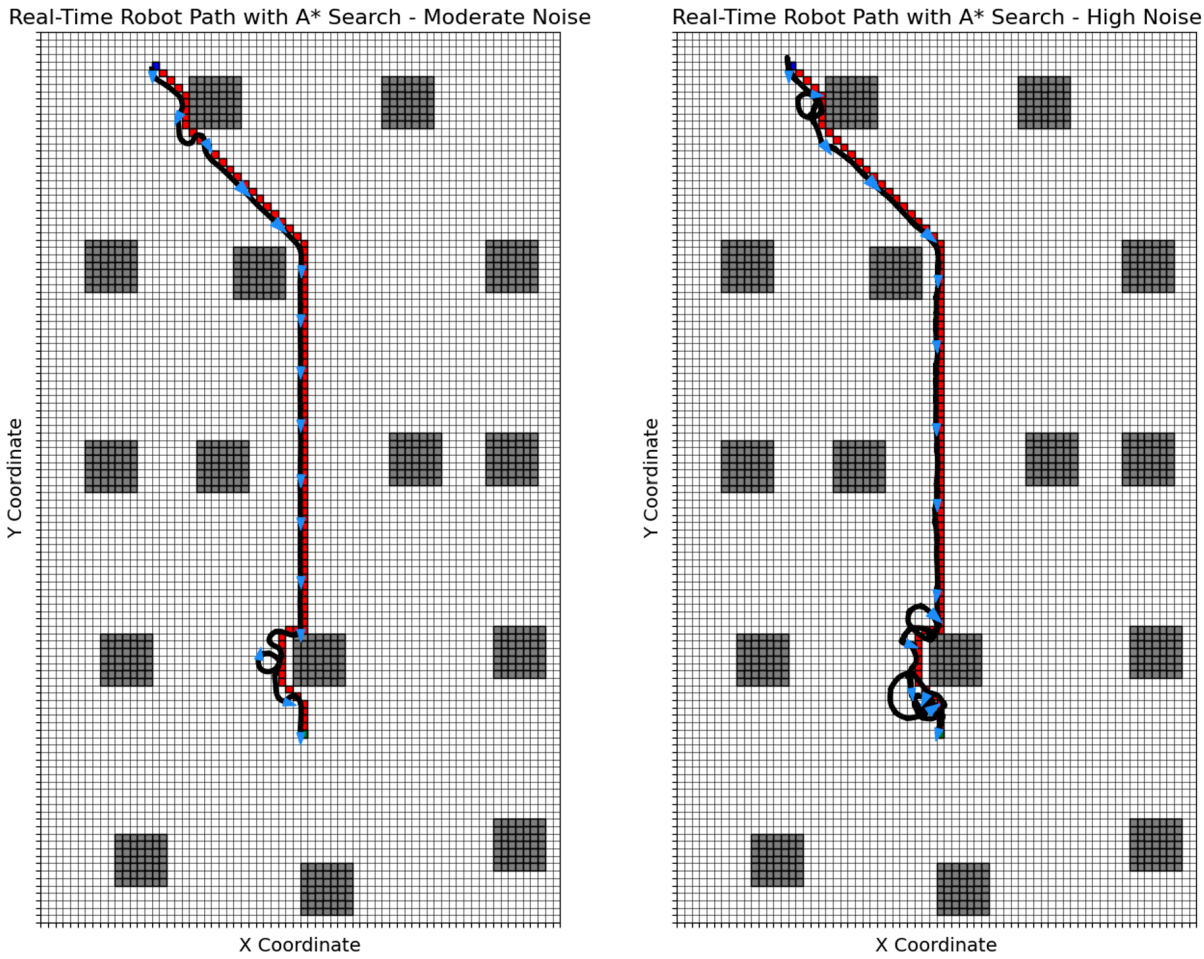

Figure A4: Visualization of path comparisons from the noise experiment.

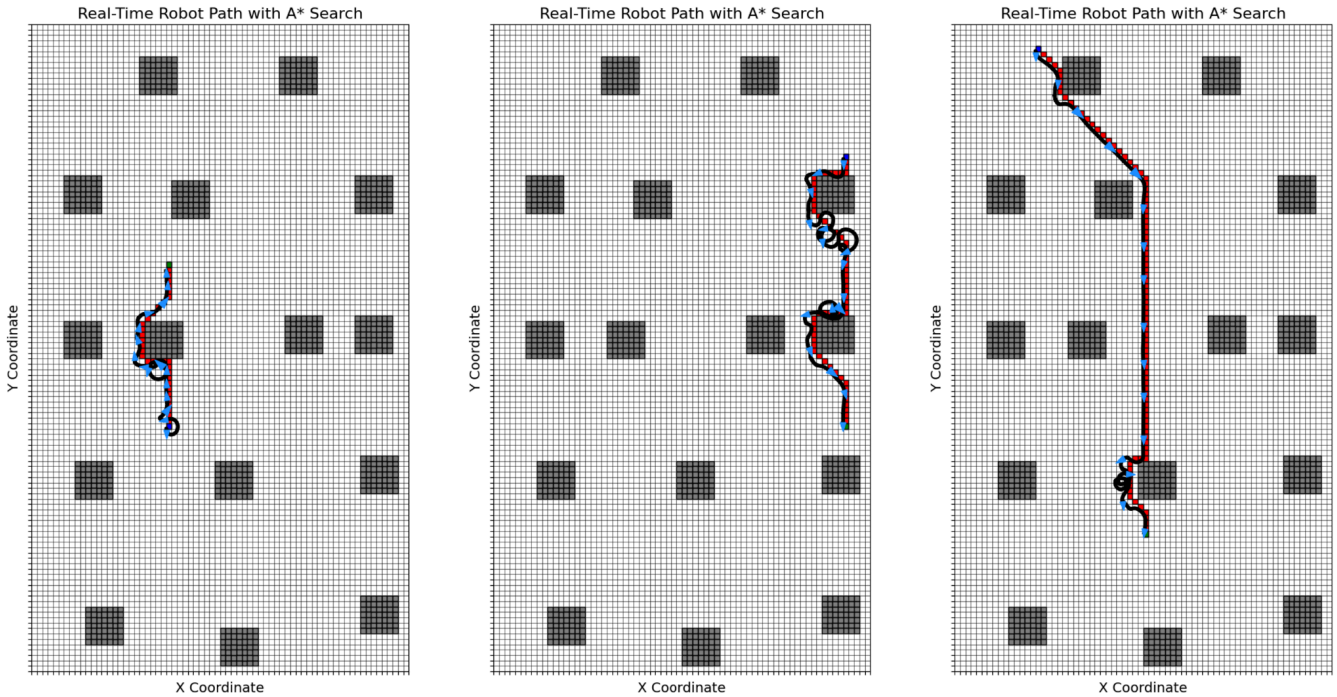

Figure A5: Visualization of real-time exploration on fine grid for Question 11.

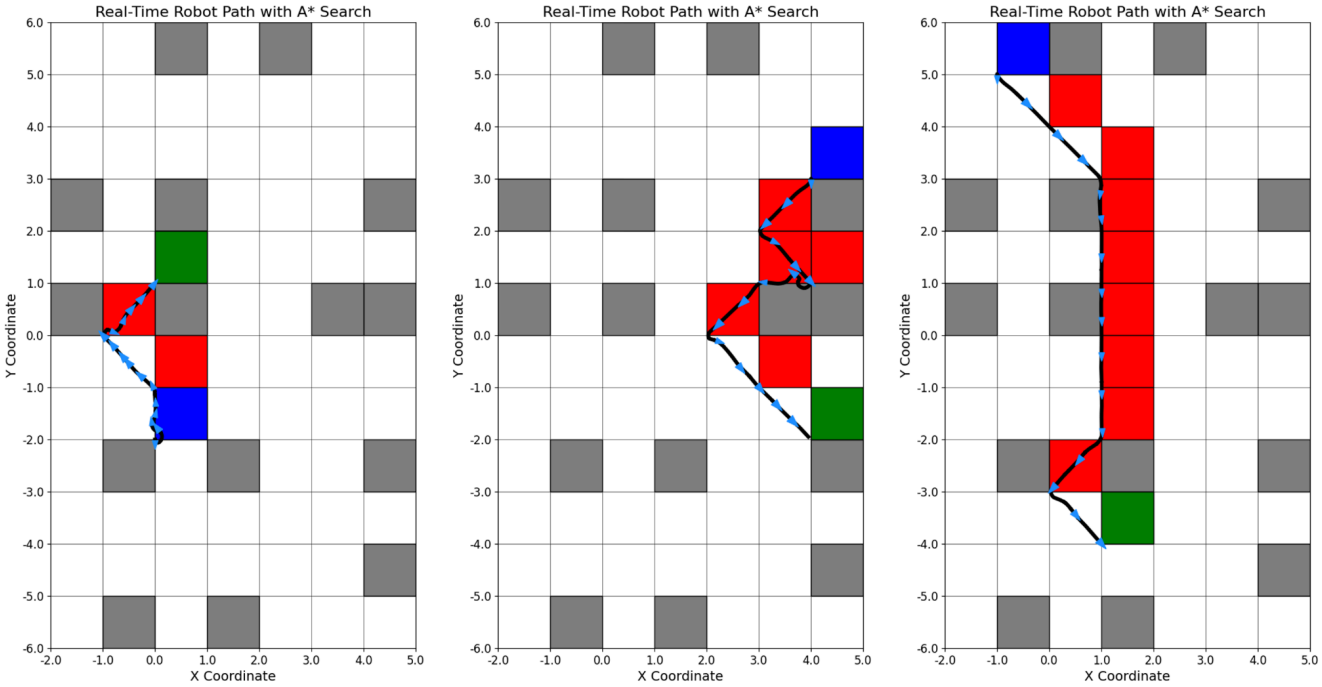

Figure A6: Visualization of real-time exploration on coarse grid for Question 11.