Cart Pushing with a Mobile Manipulation System

Chinmayee Prabhakar
Robotics Engineering Department
Worcester Polytechnic Institute
Worcester, USA
cprabhakar@wpi.edu

Lalith Athithya Navaneetha Krishnan
Robotics Engineering Department
Worcester Polytechnic Institute
Worcester, USA
lnavaneethakrishnan@wpi.edu

Thira Patel
Robotics Engineering Department
Worcester Polytechnic Institute
Worcester, USA
tkpatel1@wpi.edu

Venkatesh VP
Robotics Engineering Department
Worcester Polytechnic Institute
Worcester, USA
vparsuram@wpi.edu

***Abstract*— The purpose of this project is to implement and compare the different navigation algorithms for a mobile manipulation system. This system includes a mobile base and two manipulator arm unit that work together to navigate through a dynamic environment while pushing a cart. The project is demonstrated in a 2D setting in which a non-holonomic robot pushes a holonomic cart. To solve the path planning problem, the study applies four distinct motion planning algorithms, namely A* (the baseline), ADA*, MHA*, and Bi2RRT*. These algorithms consider the movement (kinodynamic) constraints of the robot system and compute a collision-free trajectory between the start and endpoint of the robot's motion. We compare the performance of these algorithms on the system using various evaluation metrics. This project extends the work done in the paper, "Cart pushing with a mobile manipulation system: Towards navigation with moveable objects" through its non-holonomic implementation of the robot.**

***Index Terms*— Cart pushing, Mobile manipulation system, State Lattice, Moveable objects, Motion Planning**

I. INTRODUCTION

The ability to navigate and manipulate moveable objects in unstructured environments is a crucial capability for many real-world applications, such as logistics, warehousing, and manufacturing. Mobile manipulation systems that combine the mobility of a mobile platform with the dexterity of a manipulator have shown great potential for achieving these tasks. However, most existing approaches to mobile manipulation focus on manipulating static objects, and few have addressed the challenges of manipulating moveable objects.

Our work is mainly focused on implementing the novel approach presented in the paper, "Cart pushing with a mobile manipulation system: Towards navigation with moveable objects" where they have demonstrated mobile manipulation of moveable objects using a mobile manipulator system. Specifically, the focus is on the task of cart pushing, which involves manipulating a cart while navigating through an environment. We will replicate the research work by performing motion planning for a holonomic robot manipulator and a non-holonomic cart in a 2D environment. To do this, we will incorporate various state-of the art path planning techniques onto our simulation test platform and evaluate how well the algorithms perform. By conducting this analysis, we are taking a step forward in developing more advanced mobile manipulation systems that can adeptly manage moveable objects in real-world situations.

II. RELATED WORK

Several papers discuss novel algorithms built on top of conventional ones. Many of these focus on working in a cluttered environment with dynamic constraints. Scholz et al. (2011) [1] have discussed a planning and control approach to navigate the humanoid robot in a dynamic environment while it is pushing a cart. They developed this approach for an indoor office environment, with some narrow

passages. On board sensing helped detect dynamic obstacles and complete a safe navigation of the robot as well as the cart it is pushing. The testing of this method was done in a cluttered office with humans moving around. Certain waypoints were selected for the robot, and the global planner worked to find a safe path from the start to the goal. independent document. Please do not revise any of the currentdesignations.

Honerkamp et. Al (2022) [2], in their older work had proposed to decompose mobile manipulation tasks into a simplified motion generator for the end-effector in task space and a trained reinforcement learning agent for the mobile base to account for kinematic feasibility of the motion. They then introduced Navigation for Mobile Manipulation (N2M2) which extends their work to incorporate dynamic environments. This new approach has significantly decreased the reaction time of incorporating the dynamic obstacles by the planner.

Burget et al.(2016) [3] introduced Bidirectional Informed RRT* approach, for improving the efficiency of tasks performed in warehouses by mobile manipulators. The results of the path planning framework shows that it is way better than the Informed RRT* and Bidirectional RRT* algorithms. They further tested the algorithm for multiple scenarios and concluded that it is effective for both constrained and unconstrained environments.

Adiyatov et al. (2017) [4] have developed a new approach based on the RRT* algorithm, which is memory efficient, called RRT*FN dynamic. In their previous work they introduced RRT*FN algorithm, and have built the RRT*FN Dynamic algorithm on top of that, with the goal of being able to plan for an environment with dynamic variables. This algorithm for tested on a non-holonomic dynamic robot as well as an industrial manipulator, and proved that RRT*FN Dynamic algorithm worked for most of the test cases, and was more efficient than the RRT* as well as the RRT*FN. Another approach of dealing with dynamic obstacles was developed by Likhachev et al. (2005) [5] in the form of a new algorithm called Anytime Dynamic A*, which incrementally repairs its previous solution.

Lamiraux et al. (2004) [6] introduced an approach for path optimization. They later applied the method to a non-holonomic robot in a highly cluttered environment.

III. PROPOSED METHOD

Implement A* (baseline), and advanced algorithms Multi-Heuristic A*, ADA* and Bi2RRT* for the navigation of a robot with a nonholonomic mobile base pushing a holonomic cart in a cluttered and simulated dynamic environment.

A. Algorithms

A* algorithm is a pathfinding algorithm the shortest path between two nodes in a network can be discovered using the A* method, a popular search tool. It is founded on a heuristic search technique that combines the cost of the journey from the starting node to a neighboring node's actual node and the cost of the path from the neighboring node's estimated node to the destination node. An open set and a closed set of nodes are kept in the method. While nodes in the closed set have already been reviewed, those in the open set are candidates for evaluation. The initial node is first added to the open set by the algorithm. After that, it adds each of the nearby nodes to the open set and calculates the cost of the path from the beginning node to each of them. The method chooses the open set node with the lowest cost and includes it in the closed set. After the desired node is reached or the open set is empty, the process is repeated after evaluating the selected node's neighbors. The method returns the path with the lowest cost if the goal node is reached. If no path was identified, it returns a message to the user. Robotics, video games, and other fields that call for locating the shortest path in a graph frequently use the A* algorithm.

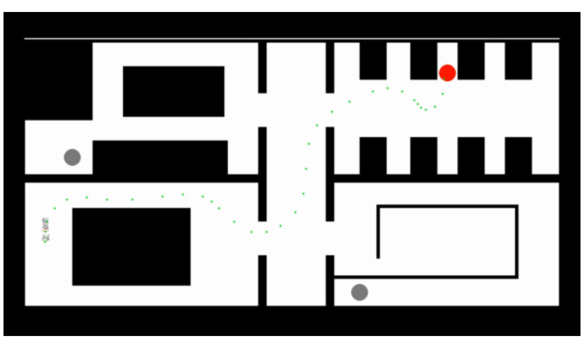

Fig.1: Baseline Algorithm A*

ADA* algorithm is an extension of the A* algorithm that dynamically adjusts the weight given to the heuristic function during the search process. The weight adjustment is based on the actual cost of the path from the starting node to the current node. This adaptive weight adjustment improves the efficiency of the algorithm. ADA* is more efficient in situations where the heuristic function is inaccurate, or the search space is large and complex, and it also ensures thorough exploration of the search space. Like A*, the algorithm returns the path with the lowest cost, but the computational resources required may be higher than A* due to the additional weight adjustment calculations. ADA* achieves this by storing information about the previous search in a search tree, which is updated as new information becomes available.

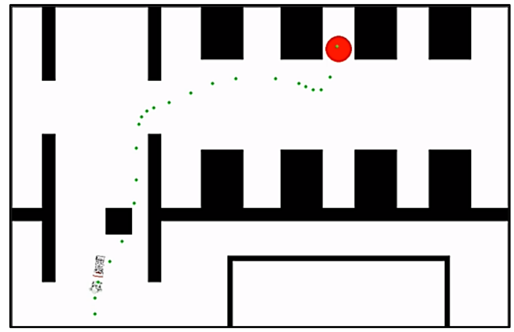

Fig.2: Initial Path of ADA* (a)

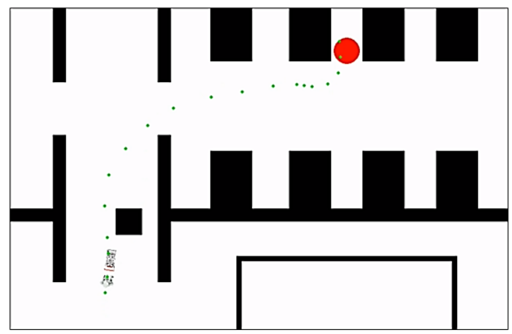

Fig.2: Final Path of ADA* (b)

Bi^{2}RRT* is an extension of informed RRT* which has two ellipsoidal subsets which are used to perform informed sampling of a mobile manipulator in configuration space. Bi^{2}RRT* is used to plan the mobility of a robot in a workspace. Two Rapidly Exploring Random Tree (RRT) is used by the robot in the method, which simultaneously explores the workspace from the robot's initial configuration and the desired configuration. The method uses a cost function to assess the quality of each branch in the trees and determines a path that avoids collisions for the robots. Sampling aids the algorithm in avoiding workplace impediments and favors research in low-cost zones. To prevent the robots from colliding with the obstacles as they move toward their destination configurations, collision detection is also used. The algorithm periodically prunes the trees to get rid of extra branches and boost planning effectiveness. The program also looks for a path that crosses both trees to connect the two trees. The algorithm returns the path as the answer to the motion planning problem if a path was located. The algorithm keeps looking around the workspace until the trees become near enough to one another to allow a connection if a connection between the trees cannot be found. The algorithm offers a probabilistic completeness guarantee, which means that if a solution exists, it will ultimately find it. However, in complex workspaces, the process may be computationally expensive and take a long time to find a solution.

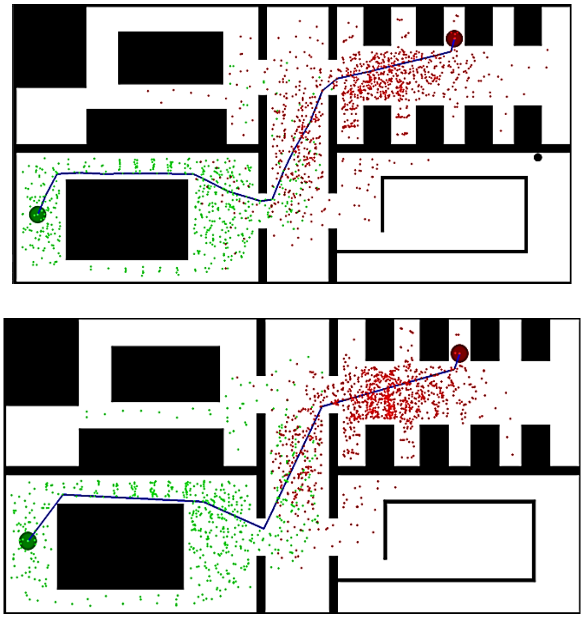

Fig.3: Bi2RRT*

Multi-Heuristic A* (DMHA*) is a variant of the A* algorithm. Basically, Multi-Heuristic A* (MHA*) expands the original A* algorithm by employing several heuristics to direct the search. Each heuristic function in MHA* calculates an estimate of the distance between a starting state and a destination state, and the algorithm then combines these estimates to direct the search. The core notion of MHA* is to direct the search using a variety of heuristics, each of which offers a unique viewpoint on the state space. Enabling the algorithm to avoid local minima and more thoroughly explore the search space, can increase search efficiency. MHA* has been shown to outperform other A* variants in certain scenarios, such as in high-dimensional search spaces.

These algorithms will be tested in a simulated environment in Pygame.

For the search-based method, which is A*, ADA* and MHA*, the state lattice will be generated using forward propagation by varying the control inputs, liner velocity, steering angle, and the cart angle within a specified range in each state. This will generate the trajectories from a point that will form the child nodes, these nodes will be further propagated to create a graph. In those graphs, we implement our algorithms.

For the sampling-based method, Bi^{2}RRT* the points are sampled from each node and find the nearest neighbors and connect each node to the nearest neighbors using feasible trajectories.

We expect ADA* to perform better than A* in our dynamic environment due to its ability to generate heuristics and perform better. MHA* will help us to have different heuristics for the position and orientation of the robot and the cart which might be helpful in finding paths.

IV. EXPERIMENT

We're going to use Pygame to create a simulation of a hospital environment that includes narrow corridors and moving obstacles, like people. We'll test and compare different motion planning algorithms to see how well they perform using evaluation metrics.

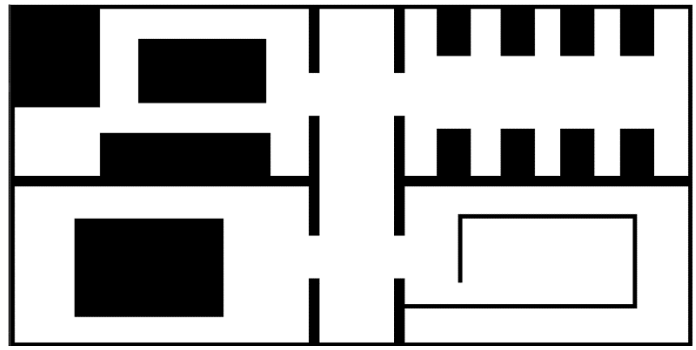

Fig.4: The Test Map

A. *Environment*

The robot navigation will be planned in 4-DOF configuration space using motion trajectories to achieve transition between nodes. The motion trajectories are generated by modifying the 3 control inputs:

- Velocity
- Steering angle
- Cart Angle

B. *Evaluation Metrics*

The advanced algorithms will be evaluated against the baseline algorithm using the following metrics:

- Clearance with respect to obstacles
- Length of path traversed
- Smoothness of path
- Time to compute a viable path
- Computation Complexity
- The algorithm performance (complexity, optimality, completeness)

V. PRELIMINARY RESULTS

The kinematic model of the non-holonomic robot pushing the holonomic cart has been implemented in Pygame and the robot is moving with the a fore mentioned motion trajectories. The algorithms are tested for the above pictured test map. We've established the dynamics of the system and implemented motion primitives to control the robot's movement. We use forward propagation to create the state lattice by making small changes to the control inputs, resulting in different trajectories originating from the point that serves as child node. This process is repeated to create a graph and A* algorithm is implemented on this graph, to generate a collision-free path from the starting point to the goal point.

The robot is able to follow this path and reach the goal in a reasonable amount of time for our basic map, reaching an optimal path under the defined resolution of the control inputs.

VI. TASK DIVISION

Task	Timeline	Team Member
Literature Review	February 10 – February 25	Chinmayee, Lalith, Thira, Venkatesh
Creating the Simulation Environment		Lalith
Creating the robot model for simulation	February 25 – March 4	Thira, Venkatesh
Kinematic model of the robot and cart		Chinmayee
Motion Planning Algorithm Implementation: A*	March 4 – March 11	Venkatesh, Thira
Motion Planning Algorithm Implementation: ADA*	March 14 – March 22	Thira, Chinmayee
Motion Planning Algorithm Implementation: Dynamic Multi-Heuristic A*	March 24 – March 31	Venkatesh, Lalith
Motion Planning Algorithm Implementation: BI^{2}RRT*	April 1 – April 7	Lalith, Chinmayee
Algorithm Evaluation	April 7 – April 21	Chinmayee, Lalith
Report	April 21 – April 28	Chinmayee, Lalith, Thira, Venkatesh
Unity*	April 28 – May 5	

REFERENCES

[1] J. Scholz, S. Chitta, B. Marthi, and M. Likhachev, "Cart pushing with a mobile manipulation system: Towards navigation with moveable objects," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, pp. 6115–6120. doi: 10.1109/ICRA.2011.5980288.

[2] D. Honerkamp, T. Welschehold, and A. Valada, "N^{2}M^2: Learning Navigation for Arbitrary Mobile Manipulation Motions in Unseen and Dynamic Environments," Jun. 2022, [Online]. Available: http://arxiv.org/abs/2206.08737

[3] F. Burget, M. Bennewitz, and W. Burgard, "BI2RRT∗: An efficient sampling-based path planning framework for task-constrained Mobile manipulation," in *IEEE International Conference on Intelligent Robots and Systems*, Nov. 2016, vol. 2016-November, pp. 3714–3721. doi: 10.1109/IROS.2016.7759547.

[4] O. Adiyatov and H. A. Varol, "A novel RRT∗-based algorithm for motion planning in Dynamic environments," in *2017 IEEE International Conference on Mechatronics and Automation, ICMA 2017*, Aug. 2017, pp. 1416–1421. doi: 10.1109/ICMA.2017.8016024.

[5] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime Dynamic A*: An Anytime, Replanning Algorithm." [Online]. Available: www.aaai.org

[6] F. Lamiraux, D. Bonnafous, and O. Lefebvre, "Reactive path deformation for nonholonomic mobile robots," *IEEE Transactions on Robotics*, vol. 20, no. 6, pp. 967–977, 2004, doi: 10.1109/TRO.2004.829459.