Motion Planning of Mobile Manipulator for Navigation Including Door Traversal

Keunwoo Jang, Sanghyun Kim, and Jaeheung Park, *Member, IEEE*

***Abstract*—Door traversal is an essential task for a mobile manipulator to achieve autonomous navigation in indoor environment. However, it remains challenging since it requires the coupled motion of the robot and door and is composed of several sub-problems such as approaching, opening, passing through, and closing the door. This letter proposes a planning framework that formulates these sub-problems as a single problem and computes a path for the mobile manipulator to navigate from the initial position, traverse through the door, and arrive at the target position. The proposed framework obtains the path of the whole-body configuration in two steps. First, the path for the pose of the mobile robot and the path for the door angle are computed by using the graph search algorithm. In graph search, an integer variable called *area indicator* is defined as an element of state, which indicates where the robot is located relative to the door. Especially, the area indicator expresses a process of door traversal. In the second step, the configuration of the manipulator is computed by the Inverse Kinematics (IK) solver from the path of the mobile robot and door angle. The proposed framework has a distinct advantage over the existing methods that manually determine several parameters such as which direction to approach the door and the angle of the door required for passage. The effectiveness of the proposed framework was validated through experiments with a mobile manipulator.**

***Index Terms*—Mobile manipulator, motion planning.**

I. Introduction

MOBILE manipulators have the extended workspace compared to fixed-base manipulators. In this respect, mobile manipulators can help people by performing various tasks such as delivery, household chores, etc. Since the chances of successfully executing such tasks depend on whether the target object is reachable or not, mobile manipulators need the ability to open and traverse doors which connect spaces. However, the problem of generating the motion of mobile manipulator is not simple because the mobile manipulator is a unified system and the door is an articulated object. This letter addresses the motion planning problem of traversing through the door and reaching the goal position for the mobile manipulator.

A. Related Works

Researches for door traversal have been actively conducted in recent decades. They can be categorized into two main approaches [1]: *sense-and-act* [2], [3], [4] and *plan-and-act* [5], [6], [7], [8], [9]. The former estimates geometric information of a door and simultaneously generates a motion that opens the door on the basis of sensory feedback information in real-time. The latter plans the motion that opens and traverses through the door given prior information of the door.

Specifically, in the field of *sense-and-act* approach, Lee et al. [2] developed a control system combining impedance control and teleoperation in order for a humanoid to open and traverse through the door. Karayiannidis et al. [3] developed control strategy that estimates the position of joint axis from the measured force and simultaneously opens articulated objects such as door and drawer. Stuede et al. [4] proposed a unified approach including impedance controller of mobile manipulator and robust door handle detection based on convolutional neural network. However, since these approaches rely on the local information of the environment, they may produce the control input stuck in local minima [1].

In the field of *plan-and-act* approach, Chitta et al. [5] developed a planner that efficiently generates the whole-body motion of the mobile manipulator to open both pull and push-type door by combining the graph search algorithm and IK solver. Furthermore, Gray et al. [6] extended [5] to enable the mobile manipulator to open the spring-loaded door by using the contact for holding the door. However, these methods [5], [6] can only deal with the problem of planning the motion that the robot approaches the door and opens it to a predefined door angle. Arduengo et al. [7] proposed a unified framework that includes detecting the door and handle, estimating the model of the door, and planning the motion for door opening. The framework utilizes a concept of task space region which represents the end-effector constraint. Burget et al. [8] proposed a planner that utilizes IK solver to compute the whole-body motion of humanoid satisfying the constraints enforced by articulated objects. Leidner et al. [9] proposed a framework that combines hybrid reasoning to produce feasible action and compliant control

Manuscript received 7 October 2022; accepted 24 April 2023. Date of publication 24 May 2023; date of current version 6 June 2023. This letter was recommended for publication by Associate Editor R. Triebel and Editor M. Vincze upon evaluation of the reviewers' comments. This work was supported in part by KIST Institutional Program under Grant 2E32283, in part by the National Research Foundation of Korea (NRF) grant funded by Korea Government (MSIT) under Grant 2021R1A2C3005914, and in part by the Ministry of Trade, Industry and Energy (MOTIE, Korea) under Grant 20018745 through Industrial Strategic Technology Development Program. *(Corresponding authors: Sanghyun Kim; Jaeheung Park.)*

Keunwoo Jang is with the Artificial Intelligence and Robotics Institute, Korea Institute of Science and Technology, Seoul 02792, South Korea (e-mail: jang90@kist.re.kr).

Sanghyun Kim is with the Department of Mechanical Engineering, Kyung Hee University, Suwon 17104, South Korea (e-mail: kim87@khu.ac.kr).

Jaeheung Park is with the Department of Intelligence and Information, ASRI, RICS, Seoul National University, Seoul 16229, South Korea, and also with the Advanced Institute of Convergence Technology, Suwon-si 17104, South Korea (e-mail: park73@snu.ac.kr).

A video clip of the experiments in various scenarios can be found at https://youtu.be/QN-NFQ18Z70.

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2023.3279612, provided by the authors.

Digital Object Identifier 10.1109/LRA.2023.3279612

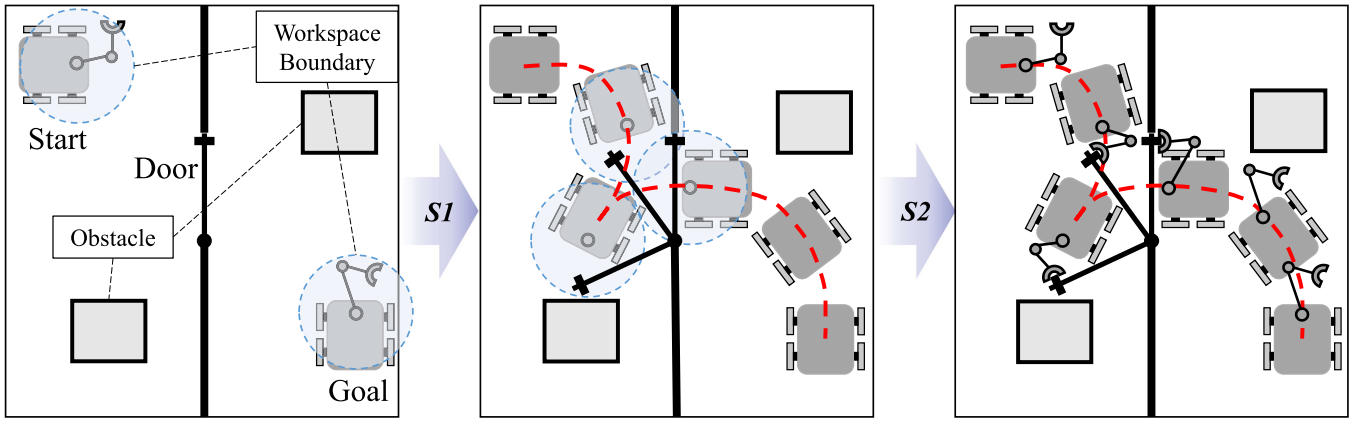

Fig. 1. Overview of the proposed framework. Given the information of the environment, door, and start and goal positions, ***S1*** is the step that computes the pose path of the mobile robot and angle path of the door. The paths are computed with the constraint that the position of the door handle has to be located inside the workspace of the manipulator as shown in blue dotted circles. Red dotted line shows the computed path of the mobile robot. Then, ***S2*** is the step that computes the path of the joint configuration of the manipulator by using the inverse kinematics solver. Combining the computed paths, the motion of the mobile manipulator is generated that the mobile manipulator approaches, opens, traverses through, closes the door, and eventually arrives at the goal position.

to execute whole-body manipulation task. These methods [5], [6], [7], [8] need to plan additional motion to traverse through the door because the goal of them is to open the door or articulated object to a predefined angle.

However, since the *plan-and-act* approach is suited for a structured environment, many researchers developed methods belonging to the *sequential sense-plan-and-act* approach [1] which exploits the environmental feedback information and generates the modified motion in real-time. Lee et al. [10] proposed model predictive control (MPC) framework that considers the aerial manipulator and the door as a whole articulated body and incorporates several constraints including self-collision and external collision. Recently, Ito et al. [11] proposed a learning-based method that predicts the behavior of the robot from the sensory data and generates the motion to open and pass through the door in real-time.

B. Overview of This Paper

In this letter, we propose a framework that belongs to the motion planning approach and addresses the problem for the mobile manipulator to navigate from the start position to the goal position, including passing through the door. Even though the framework is originally designed to plan the path off-line, which belongs to *plan-and-act* approach, the computed path from the framework can be utilized as a reference path of MPC belonging to *sequential sense-plan-and-act* approach. The search space of the problem is high dimensional. Also, the problem has to satisfy the constraint that the end-effector of the mobile manipulator grasps the door handle. In these respects, the problem demands expensive computational cost. To alleviate the cost, the framework is composed of two steps that plan separately for the mobile robot and the manipulator, as shown in Fig. 1. The framework first computes the paths of the mobile robot and door angle by using the graph search algorithm. Based on the obtained paths of the first step, the framework finds a collision-free path of the joint position of the manipulator by utilizing the IK solver. By merging these paths, the whole-body motion of the mobile manipulator is generated such that the robot navigates to the goal position after passing through the door.

The main contribution of the framework is a graph representation that formulates the navigation problem of the mobile manipulator in the low-dimensional state space and also handles an articulated object by checking the feasible transition of the states. Thus, the proposed framework can plan the path for all sub-problems by a single search, whereas the previous works [5], [6], [7], [8] can only plan the path for one sub-problem or a part of the sub-problems and thus require the additional path planning for the other sub-problems. Especially, one element of the state, called *area indicator*, has the integer value depending on the relative position between the robot and door. The area indicator allows to calculate the range of the door angle that the manipulator can reach, which eliminates the need to include the door angle directly in the state space.

The remainder of the letter is organized as follows. In Section II, the proposed framework consisting of two steps is explained. Section III describes the experimental validations and discussion of the proposed framework. Finally, the letter is concluded in Section IV.

II. Proposed Framework

This section describes the proposed framework which computes the collision-free path of the mobile manipulator passing through the door and thereafter reaching the goal position. Rather than searching for the entire joint space of the mobile manipulator, the proposed framework is designed to sequentially execute two steps denoted as ***S1*** and ***S2*** which generate the motion for the mobile robot and manipulator, respectively. The first step ***S1*** generates the collision-free motion of the robot and door angle, assuming that the manipulator maintains its home configuration except during door traversal. Next, the second step ***S2*** computes the synchronized motion of the manipulator for the generated motion of the mobile robot and door in ***S1***. In ***S2***, the motion of the manipulator is computed by using the IK solver while considering the collision between the manipulator and environment including the door and wall. The schematic description of the two steps is shown in Fig. 1.

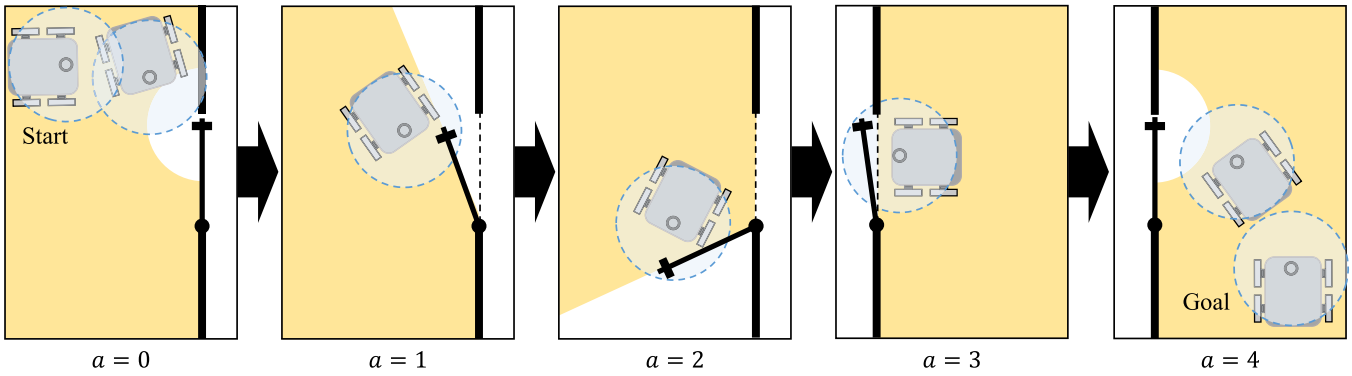

Fig. 2. Illustration of the area indicator. The value of the area indicator is decided by where the robot is located with respect to the door. The yellow area shows the area corresponding to each value of the area indicator. As the indicator changes from 0 to 4, the robot moves toward the target position after passing through the door.

A. *Computing Path for Mobile Robot and Door Angle - **S1***

To obtain the paths of both the mobile robot and door angle, the navigation problem of the mobile robot is formulated as a graph search. The graph search algorithm constructs the graph by propagating the states to their successors via actions. Then, the path with the minimum cost is found by searching for the constructed graph. To this end, the elements required for the graph search, including the state, action, and cost are described as follows.

1) State: The search space $\mathcal{S} \in \mathbb{R}^3 \times \mathbb{I}$ is four dimensional space. The state is expressed as $s = (x, y, \theta, a) \in \mathcal{S}$. (x, y, θ) is the pose of the mobile robot and a is called *area indicator*. The area indicator a expresses which area the current pose of the mobile robot belongs to. The value of the area indicator is determined according to the pose of the mobile robot and the angle of the door, as shown in Fig. 2.

In Fig. 2, the value of the area indicator is defined as 0 when the robot is far from the door. At this position, the door handle is outside the workspace of the manipulator. This implies the process of the robot approaching the door. Next, when the robot is located outside the line of the door and can reach the door handle, a is 1. When the robot is located inside the line of the door but does not yet cross over the doorsill, a is 2. As the robot moves outside the circle of the door and then inside it, the indicator changes from 1 to 2. At this point, two states can be generated that have the same pose of the mobile robot, but have two different indicators of 1 and 2. Then, when the robot is located beyond the doorsill but still can reach the door handle, a is 3. This indicates the process of the robot closing the door. Finally, when the robot is placed between the door and goal position, the value of a is defined as 4. This refers to the process of the robot reaching the goal position.

In fact, the indicator from 1 to 3 is an alternative of directly including the door angle as an element of the state. If the pose of the mobile robot is given, all possible angles of the door are checked and labeled as from 1 to 3. Then, the angles belonging to the same indicator are grouped into the range, as shown in Fig. 3. From a practical point of view, the range is computed by checking a finite set of the door angles obtained from discretizing the possible range of the door angle at a certain interval, e.g., 1° or 2 degrees.

The concept of the area indicator provides several advantages. First, it can handle both "pull" and "push"-type door. As the indicator changes from 1 to 3, it shows the process of the robot opening, passing through, and closing the "pull"-type door. On the contrary, as the indicator changes from 3 to 1, it expresses the task for "push"-type door. The indicator of 3 is different from the indicator of 1 in that the door angle does not decrease less than 0 degrees as in the right figure of Fig. 3. Second, all sub-tasks related to the door can be described. They include approaching the door, opening the door, traversing through the door, closing the door, and finally moving toward the goal position. Third, the state can be concisely expressed by including the area indicator without directly including the door angle as an element of the state.

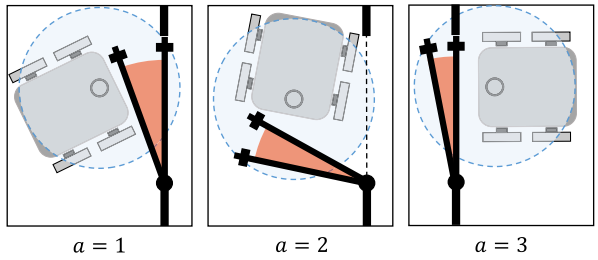

Fig. 3. Area indicator a from 1 to 3 implicitly shows the range of the reachable door angles. Given the pose of the mobile base, the range shown in red can be calculated by checking whether the door handle is within the workspace boundary of the manipulator and the collision between the door and the mobile robot.

2) Action: To propagate any state to its successor state, the action should be defined. Since the area indicator of the state is a variable that is automatically determined by the pose of the mobile robot, the action is defined only for the pose of the mobile robot. Thus, the concept of the *lattice* [12] is adopted in order to discretize the state space while considering non-holonomic constraints for the mobile robot, such as a car-like or differentially-driven mobile robot. After defining a set of feasible actions, as shown in Fig. 4, the lattice graph composed of the discretized states is constructed by executing the actions to the states.

Every transition between the two states is feasible when their area indicators are equal to 0 or 4 because the action of the robot is not related to the door. However, when the area indicator is 1 to 3, the reachable door angles of a state and its successor state should overlap, as shown in Fig. 5(a). This is because the

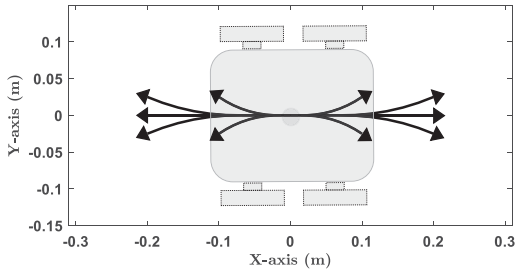

Fig. 4. Example of a set of feasible actions for the given pose of the robot.

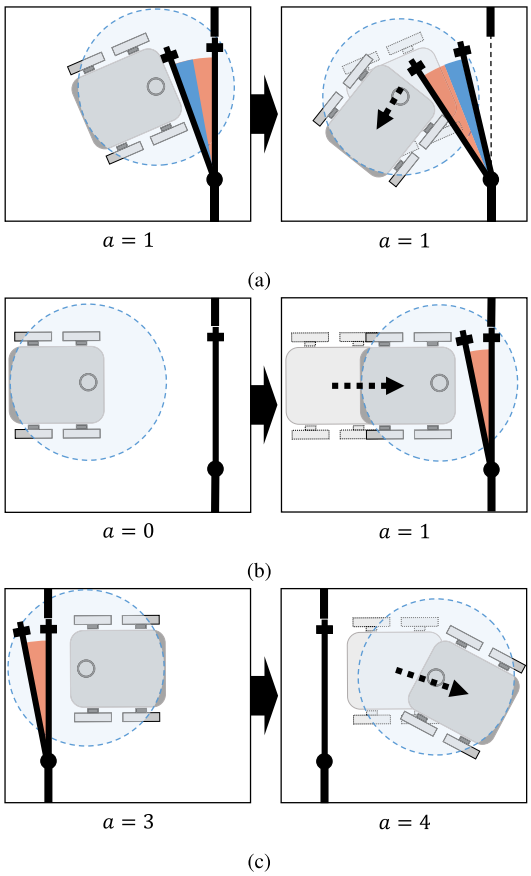

Fig. 5. Illustration of the state transitions depending on the value of the area indicator: (a) In the process of the robot opening the door, the ranges of the reachable angles for two consecutive states should overlap as shown in blue; (b) When the robot approaches the door to open it, i.e., for the area indicator of the successor state to be 1, the range of the door angle should contain 0 degrees; (c) When the robot closes the door and moves towards the goal position, i.e., for the area indicator of the successor state to become 4, the range of the door angle for the predecessor state should contain 0 degrees.

door angle should not change discontinuously in the process of the robot opening and closing the door. From a practical point of view, since the reachable angle range is calculated as a set of discretized angles not as the inequality bounds of the angle, at least one discrete value of the door angle should overlap to become a valid transition.

Furthermore, when the robot starts to open the door, i.e. the area indicator changes from 0 to 1, the successor state should contain 0 degrees in its set of the reachable door angles, as shown in Fig. 5(b). Likewise, when the robot finishes closing the door and starts to move toward the goal position, i.e. the area indicator changes from 3 to 4, the predecessor state whose area indicator is 3 should contain 0 degrees in its set of the feasible door angles, as shown in Fig. 5(c).

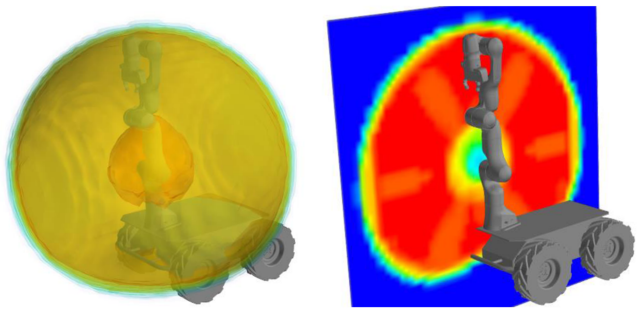

Fig. 6. Visualization of the reachability of our robot in OPENRAVE [13]. Left figure shows the colored surfaces depending on the density levels of the reachability. Right figure shows the surface cut by a vertical plane at the base of the manipulator (red: high reachability, blue: low reachability).

3) Cost: The transition cost from the state s to its successor state s_{succ}, denoted as $c(s, s_{succ})$, is expressed as

$$c(s, s_{succ}) = \begin{cases} c_{action} + c_{map} + c_{door} & \text{if } a = 1, 2, 3 \\ c_{action} + c_{map} & \text{otherwise} \end{cases} \quad (1)$$

where c_{action} is the cost to transit to the successor state by executing the predefined action of the mobile robot. This term is proportional to the weighted sum of the translational and rotational displacement [14]. Second, c_{map} is the cost based on the costmap which indicates how close the robot is to the obstacle [5]. Lastly, c_{door} is the cost to represent how properly the door handle is positioned from the base of the manipulator at the given pose of the mobile robot when the area indicator of the current state includes from 1 to 3. c_{door} is designed by utilizing the information of the reachability [13] which represents the quality of the pose of the end-effector based on the number of IK solutions. The reachability of our robot is illustrated in Fig. 6. Thus, c_{door} is defined as a second-order polynomial function of the distance from the base of the manipulator to the door handle. c_{door} has the minimum value at the region of high reachability as

$$c_{door} = K(d - d_{\min})^2, \quad (2)$$

where K is the positive coefficient and d is the distance between the base of the manipulator and door handle at the current pose of the mobile robot. Based on the reachability data, d_{min} is set as 0.4 m. If the robot can reach the door handle at several door angles, all costs for the corresponding door angles are calculated and the minimum cost is selected among them.

To guide the search for the graph, the heuristic should be defined. The heuristic is an estimated cost from the current state to the goal state. In the proposed framework, the heuristic $h(s)$ is designed as the sum of the estimated translational displacements from the current state to the goal state and computed as

$$h(s) = \sum_{i=a}^{4} d_i, \quad (3)$$

where d_i is the estimated displacement for the area indicator i, shown in Fig. 7, and a is the current state's area indicator. d_i implies the distance from the current state to reach the region of the next area indicator. For example, when the area indicator is

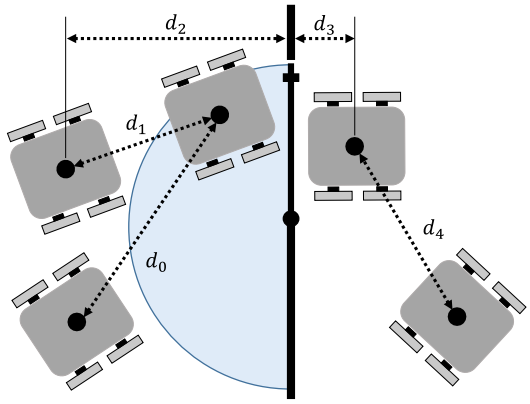

Fig. 7. Illustration of each term of the heuristic. When the area indicator is 0, d_0 is computed as the distance from the current pose to the pose in front of the door handle. d_1 is the distance from the current pose to the pose outside the circle swept by the door at the current heading angle of the robot. d_2 is the distance to the center of the door. d_3 is the distance to cross over the doorsill. d_4 is the distance to the target pose.

0, d_0 is calculated as the displacement from the current pose to the closest pose to the door handle.

4) Search: Based on the designed components of the graph including the state, action, and cost, the graph is constructed and searched in order to obtain the optimal path. The initial state s_0 is defined as $(x_0, y_0, \theta_0, a_0)$ where (x_0, y_0, θ_0) is the initial pose of the mobile robot and a_0 is its area indicator. The goal state s_g is defined as $(x_g, y_g, \theta_g, a_g)$ where (x_g, y_g, θ_g) is the target pose of the mobile robot and a_g is its area indicator. The search algorithm used in this letter is Anytime Repairing A* (ARA*) [15]. The ARA* quickly finds the initial solution path which can be sub-optimal, then refines it towards the optimal path for the remaining planning time. The solution path can be obtained by using the other variants of A* algorithm, such as Anytime Dynamic A* [16] and Lifelong Planning A* [17].

Before the search, the information should be provided including the radius of the workspace of the manipulator, 2D map information including the position of the door joint and door handle, the type of door, the joint range of door, and the size of the mobile robot. The output of the search is the path from the start pose to the target pose of the mobile robot with the corresponding area indicators. Since the area indicator represents a finite set of valid door angles, the door angle with the minimum cost is selected after they are evaluated by c_{door}. Consequently, the paths for the mobile robot and door angle can be obtained.

B. Computing Path for Arm Configuration - *S2*

In order to compute the path for the joint position of the manipulator, the path for the position of the door handle is first computed from the path for the door angle obtained in ***S1*** with the given information of the door model. Then, the joint position of the manipulator grasping the door handle is computed by the IK solver at the corresponding pose of the mobile robot. To compute a collision-free IK solution, the kinematic model of the manipulator should be provided in advance. Since the manipulator with more than 7 degrees of freedom (DOFs) has many IK solutions, the IK solution is selected such that it is furthest from the joint limits and closest to the previous IK solution. At this point, we do not consider the case that the IK solution does not exist because we assume that the IK solution grasping the door handle always exists if the door handle is within the workspace boundary of the arm. It may fail to compute the IK solution even though the workspace boundary is computed conservatively based on the reachability data. However, this limitation, as also mentioned in [18], is not handled for now and will be dealt with in future work, including the analysis of the reachability. On the other hand, the end-effector of the manipulator is set to move from the outer door handle to the inner door handle when the line of the door meets the mounted position of the manipulator, as shown in Fig. 8. This regrasping motion can be planned by sampling-based motion planners, such as rapidly-exploring random trees [19]. The regrasping motion also includes connecting the IK solution close to the joint limit caused by iterative IK computations to the new IK solution.

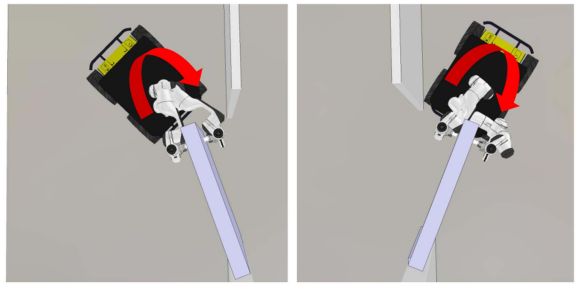

Fig. 8. Regrasping the door handle on the other side of the door is necessary in case of (a) pull-type door and (b) push-type door.

Consequently, the whole body motion of the mobile manipulator is generated by combining the path for the pose of the mobile robot, obtained in ***S1***, and the path for the joint position of the manipulator, obtained in ***S2***.

III. Results

The proposed framework was validated through various simulations and real experiment using both the differentially-driven mobile manipulator consisting of the four-wheel mobile base *Husky* (*Clearpath Robotics. Co.*) and the 7-DOFs manipulator *Panda* (*Franka Emika. Co.*) and omni-directional mobile manipulator called *KUKA omniRob*. The specification of the computer is $i7$ 4.2 GHz with 16 GB RAM. The external libraries were utilized, including SBPL [14] for the graph search algorithm in ***S1*** and TRAC-IK [20] for computing IK solution in ***S2***. The video clips of the experiments are available at.[1]

A. Application to Pull and Push-Type Door

Motion planning for our mobile manipulator was performed by using the proposed framework in order to validate that it is possible to address the navigation problem including the tasks for both pull and push-type doors. The snapshots for them are shown in Fig. 9. Fig. 9(a) shows the result of the pull-type door. The inputs to the framework were the start state $s_0 = (1.0, 4.0, 0.0, 0)$ and goal state $s_g = (4.0, 1.0, 0.0, 4)$. The door length is set as 1.0 m. The computation time was recorded as 4.56 s for ***S1*** and 0.71 s for ***S2***. On the other hand, the snapshots of the result for the push-type door are depicted in Fig. 9(b). The start and goal state were defined as $s_0 = (1.0, 1.0, 0.0, 0)$

[1][Online]. Available: https://youtu.be/QN-NFQ18Z70.

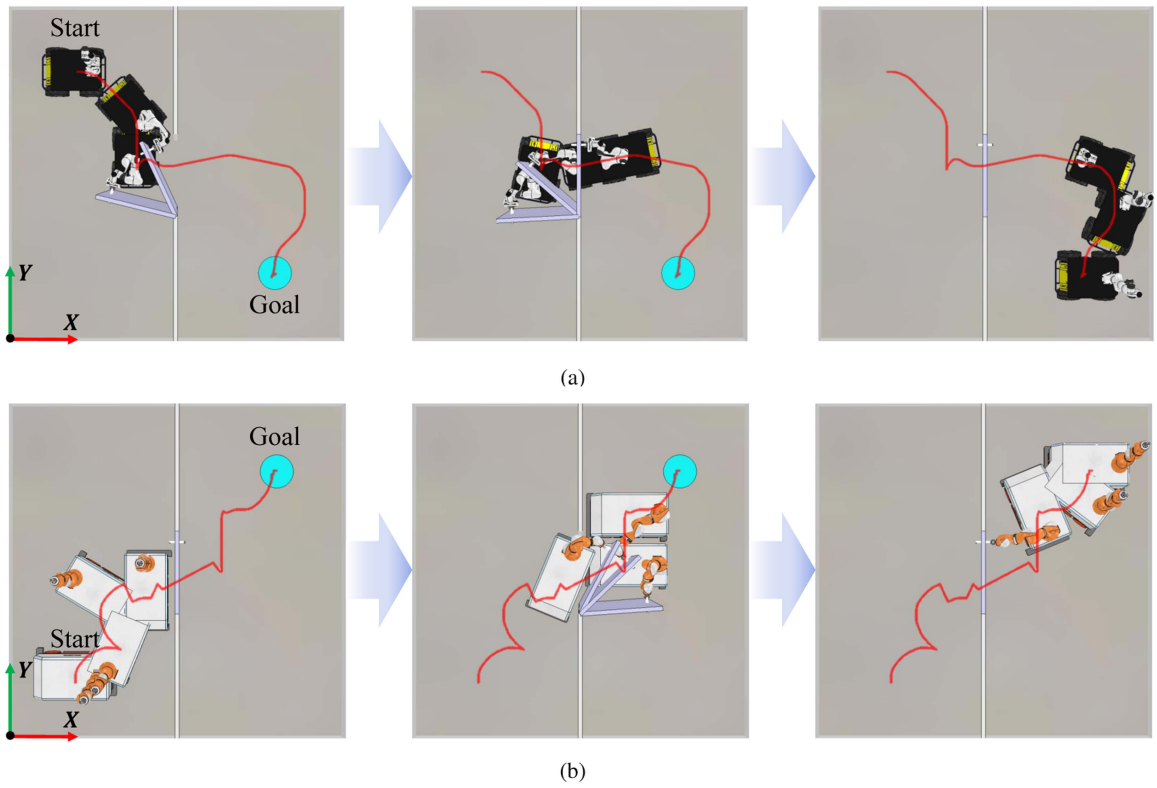

Fig. 9. Snapshots of simulation from the proposed framework for (a) pull-type door and (b) push-type door. Red lines indicate the path of the mobile robot.

and $s_g = (4.0, 4.0, 0.0, 4)$, respectively. The planning time for ***S1*** was 0.55 s and the computation time for ***S2*** was 0.9 s. In the case of the push-type door, the path for the area indicator started at 0, changed to 3 when the robot was in front of the door, 2 when the robot is located inside the line of the door, 1 when the robot is outside the line of the door, and 4 when the robot moves towards the target position. Therefore, the results demonstrate that the motion of the mobile manipulator executing the tasks for both pull and push-type doors can be planned by the proposed graph search using the concept of the area indicator.

B. Comparison With Separate Planning by Existing Works

The previous works [5], [6], [7] handled single or few sub-problems related to the door. As mentioned in Section I-B, one can utilize the separate planning that addresses each sub-problem and combines each path of sub-problem into a unified one. Even though the separate planning consisting of the existing methods [5], [15] can solve the navigation problem addressed in the proposed framework, it has several limitations. To show this, an additional experiment was performed to solve the problem of ***S1*** by the separate planning based on the graph search. The problem of ***S1*** can be divided into 4 sub-problems denotes as *SP1*, *SP2*, *SP3*, and *SP4* : *SP1* - Depart from the initial position and approach the door, *SP2* - Open the door to the desired angle, *SP3* - Pass through the door and close it, and *SP4* - Arrive at the target position. To obtain the continuous path, the final position of the path for each sub-problem was set as the start position for the next sub-problem. The state spaces of *SP1* and *SP4* were three-dimensional which are the poses of the mobile robot since *SP1* and *SP4* were the navigation problem to reach the target pose of the mobile robot. The start pose of *SP1* was the same pose in Section III-A and the target pose of *SP1* was determined by randomly sampling the pose where the door handle is within the workspace boundary of the manipulator. The state spaces of *SP2* and *SP3* were the same as [5], which include the pose of the mobile robot and binary variable compactly representing the door angle. The desired door angle of *SP2* was defined as 135 degrees. For fair comparison, the action and cost designed in Section II-A were utilized for constructing the graph. The ARA* was adopted for the graph search. The maximum computation time was limited as 100 s. The graph search of the separate planning for both pull and push-type door was conducted for 25 trials, respectively. The results of the trials are summarized in Table I. Table I reports the mean values with standard deviation of success rate, planning time, cost, number of states, and path length for the separate planning, as well as the results from the proposed algorithm.

Regarding the result of the pull-type door in Table I, the success rate of planning the problem of ***S1*** by the separate planning within 100 s was recorded as 80 %. The failure of the search occurred because the initial pose of *SP2* was randomly sampled. Specifically, the sampled pose might be possible to solve *SP2*, but could arrive at the unfavorable pose, which is the initial pose of *SP3*. Furthermore, *SP3* included narrow passage and considered the constraint that the door handle is within the workspace of the arm. These features made it difficult to search for the path of *SP3* within the predefined time period. The proposed framework required a planning time of 4.56 s, whereas the separate planning required an average planning time of 39.67 s. The path cost computed from our framework was 99266, whereas the path cost obtained from separate planning was 150925 on average, which is 1.52 times more than the proposed framework. The number of states composed of the

TABLE I
COMPARISON WITH THE SEPARATE PLANNING

Door type	Pull-type door						Push-type door					
Approach	Separate planning					Ours	Separate planning					Ours
Success rate (%)	80					100	76					100
Planning time (s)	*SP1*	*SP2*	*SP3*	*SP4*	Total	4.56	*SP1*	*SP2*	*SP3*	*SP4*	Total	4.40
	1.53 ±1.59	0.37 ±0.12	36.83 ±39.44	0.94 ±0.79	39.67 ±39.02		1.74 ±1.16	1.75 ±1.54	5.90 ±11.73	0.81 ±0.43	10.2 ±11.79	
Cost	31195 ±7648	26393 ±11351	47110 ±16801	46227 ±4602	150925 ±26290	99266	35417 ±5171	23036 ±9641	50331 ±19176	33673 ±7737	142457 ±24501	88507
Number of states	21.6 ±5.7	7.6 ±2.9	18.8 ±4.3	26.1 ±6.8	74.0 ±9.9	35	19.2 ±2.5	12.4 ±5.1	13.1 ±3.8	15.5 ±3.2	60.2 ±8.1	47
Path length (m)	7.68±0.92					6.34	7.96±1.12					6.24

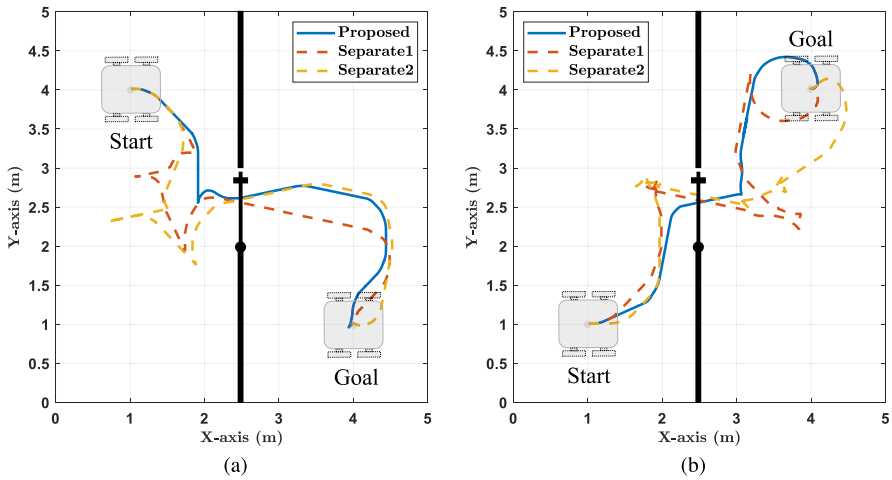

Fig. 10. Paths of the mobile robot obtained by the proposed framework and separate planning for (a) pull-type door and (b) push-type door. The blue line indicates the path from the proposed framework. The dotted lines show the paths from the separate planning.

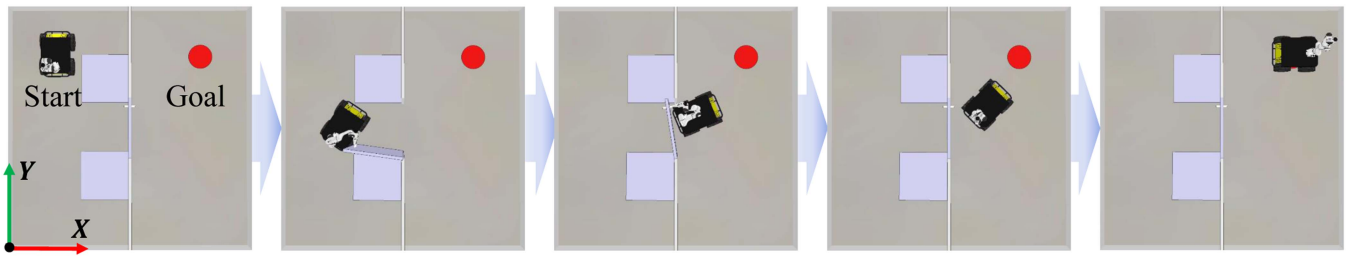

Fig. 11. Snapshots of the simulation experiment are depicted for pull-type door with obstacles. Two square-shaped obstacles are placed in front of the door.

solution path from the proposed method is less than half that of the separate planning. The length of the path from our framework was obtained as 6.34 m, which 1.34 m shorter than the path from the separate planning.

In the result for the push-type door, the success rate of solving ***S1*** was recorded as 76 % in the separate planning. The failure cases were also due to the influence of the randomly sampled initial pose for *SP2*. The planning time of the proposed framework was recorded as 4.40 s, while that of the separate planning was recorded as 10.2 s on average. The path cost computed from the separate planning was 142457, which is about 1.61 times more than our framework. The number of states consisting of the solution path from the proposed framework was 47, whereas the separate planning had 60.2 states on average. In terms of path length, the proposed framework computed the path of 6.24 m, while the path from the separate planning is measured as 7.96 m on average.

Additionally, to visualize the difference of the solution path, three paths including two example paths obtained by the separate planning and one path obtained by our framework are shown in Fig. 10. As shown in Fig. 10, the separate planning computed a more complex path for the mobile robot to open and pass through the door than our method because the separate planning randomly samples the initial pose of *SP2* and defines the desired door angle of *SP2* as a fixed value. Furthermore, to show the effectiveness of the proposed algorithm, the experiment in a complex environment was additionally conducted and snapshots of the result are shown in Fig. 11.

Consequently, it was validated through the results in Sections III-A and III-B that the proposed framework computed

Fig. 12. Snapshots of the real experiment results from the proposed framework for pull-type door.

the path reliably and efficiently due to the following characteristics. First, the complexity of the state was reduced by defining the area indicator expressing the range of the door angle rather than directly including the door angle as a component of the state. Second, the graph was searched efficiently because the heuristic in (3) properly describes the estimated displacement from the current pose to the goal pose. Especially, since the heuristic includes the estimated distance from the current pose to the region where the state has the next area indicator, the state can be easily propagated to the successor state with the next area indicator.

C. Experiment With Real Robot

The proposed framework was applied to the scenario in real indoor environment. Based on the computed path from the proposed framework, the trajectory was generated considering the velocity limits of the manipulator and the mobile robot. The robot was controlled by using the quadratic programming-based controller such as [21]. The snapshots of the experiment, shown in Fig. 12, indicate that the robot successfully executed the tasks including approaching the door, passing through the door, and reaching the target position.

IV. Conclusion

This letter proposes a framework to solve the navigation problem including door traversal for the mobile manipulator. The framework computes the whole-body motion of the mobile manipulator in two steps, instead of exploring the whole-body configuration space of the mobile manipulator. The first step is to formulate the navigation problem including the process of passing through the door as a graph search problem and find the path of the mobile robot and the angular path of the door by the graph search algorithm. In particular, the search space is reduced by introducing a component of the state, called area indicator, which compactly expresses the range of the reachable door angles. Also, the area indicator implicitly shows the process of approaching, opening, traversing through, closing the door, and reaching the target position. In the second step, the path for the arm configuration grasping the door handle is computed by the IK solver. The effectiveness of the proposed framework was demonstrated through several simulations and real experiments with the differentially-driven mobile manipulator. Since the framework is based on off-line planning, future work is to extend the framework to deal with uncertainty and changes in the environment based on sensory feedback information.

References

[1] D. Kappler et al., "Real-time perception meets reactive motion generation," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 1864–1871, Jul. 2018.

[2] J. Lee et al., "Upper-body impedance control with variable stiffness for a door opening task," in *Proc. IEEE/RAS Int. Conf. Humanoid Robots*, 2014, pp. 713–719.

[3] Y. Karayiannidis, C. Smith, F. E. V. Barrientos, P. Ögren, and D. Kragic, "An adaptive control approach for opening doors and drawers under uncertainties," *IEEE Trans. Robot.*, vol. 32, no. 1, pp. 161–175, Feb. 2016.

[4] M. Stuede, K. Nuelle, S. Tappe, and T. Ortmaier, "Door opening and traversal with an industrial cartesian impedance controlled mobile robot," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 966–972.

[5] S. Chitta, B. Cohen, and M. Likhachev, "Planning for autonomous door opening with a mobile manipulator," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2010, pp. 1799–1806.

[6] S. Gray, S. Chitta, V. Kumar, and M. Likhachev, "A single planner for a composite task of approaching, opening and navigating through non-spring and spring-loaded doors," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 3839–3846.

[7] M. Arduengo, C. Torras, and L. Sentis, "Robust and adaptive door operation with a mobile robot," *Intell. Serv. Robot.*, vol. 14, pp. 409–425, 2021.

[8] F. Burget, A. Hornung, and M. Bennewitz, "Whole-body motion planning for manipulation of articulated objects," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 1656–1662.

[9] D. Leidner, A. Dietrich, F. Schmidt, C. Borst, and A. Albu-Schäffer, "Object-centered hybrid reasoning for whole-body mobile manipulation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 1828–1835.

[10] D. Lee, H. Seo, D. Kim, and H. J. Kim, "Aerial manipulation using model predictive control for opening a hinged door," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 1237–1242.

[11] H. Ito, K. Yamamoto, H. Mori, and T. Ogata, "Efficient multitask learning with an embodied predictive model for door opening and entry with whole-body control," *Sci. Robot.*, vol. 7, no. 65, 2022, Art. no. eaax8177.

[12] M. Likhachev and D. Ferguson, "Planning long dynamically feasible maneuvers for autonomous vehicles," *Int. J. Robot. Res.*, vol. 28, no. 8, pp. 933–945, 2009.

[13] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. dissertation, Robot. Inst., Carnegie Mellon Univ., Pittburgh, PA, USA, Sep. 2010.

[14] "Search-based planning library," [Online]. Available: https://github.com/sbpl/sbpl

[15] M. Likhachev, G. J. Gordon, and S. Thrun, "ARA∗ : Anytime A∗ With Provable Bounds on Sub-Optimality," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, Cambridge, MA, USA, MIT Press, 2003, pp. 767–774.

[16] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime dynamic A*: An anytime, replanning algorithm," in *Proc. 15th Int. Conf. Int. Conf. Automated Plan. Scheduling*, 2005, pp. 262–271.

[17] S. Koenig, M. Likhachev, and D. Furcy, "Lifelong planning A*," *Artif. Intell.*, vol. 155, no. 1, pp. 93–146, 2004.

[18] S. Tonneau, N. Mansard, C. Park, D. Manocha, F. Multon, and J. Pettré, "A reachability-based planner for sequences of acyclic contacts in cluttered environments," in *Proc. Int. Symp. Robot. Res.*, 2018, pp. 287–303.

[19] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Dept. of Comput. Sci., Iowa State University, Ames, IA, USA, Tech. Rep. TR 98-11, 1998.

[20] P. Beeson and B. Ames, "TRAC-IK: An open-source library for improved solving of generic inverse kinematics," in *Proc. IEEE-RAS 15th Int. Conf. Humanoid Robots*, 2015, pp. 928–935.

[21] K. Jang, S. Kim, S. Park, J. Kim, and J. Park, "Weighted hierarchical quadratic programming: Assigning individual joint weights for each task priority," *Intell. Serv. Robot.*, vol. 15, pp. 475–486, 2022.