2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Planning for Autonomous Door Opening with a Mobile Manipulator

Sachin Chitta*, Benjamin Cohen†, Maxim Likhachev†,
* Willow Garage Inc., Menlo Park 94025, USA
{sachinc}@willowgarage.com
† Grasp Laboratory, University of Pennsylvania, Philadelphia PA 19104
{bcohen,maximl}@seas.upenn.edu

***Abstract*— Computing a motion that enables a mobile manipulator to open a door is challenging because it requires tight coordination between the motions of the arm and the base. Hard-coding the motion, on the other hand, is infeasible since doors vary widely in their sizes and types, some doors are opened by pulling and others by pushing, and indoor spaces often contain obstacles that limit the freedom of the mobile manipulator and the degree to which the doors open up. In this paper, we show how to overcome the high-dimensionality of the planning problem by identifying a graph-based representation that is small enough for efficient planning yet rich enough to contain feasible motions that open doors. The use of graph search-based motion planning enables us to handle consistently the wide variance of conditions under which doors need to be open. We demonstrate our approach on the PR2 robot - a mobile manipulator with an omnidirectional base and a 7 degree of freedom arm. The robot was successful in opening a variety of doors both by pulling and pushing.**

I. Introduction

A robot that needs to navigate autonomously indoors must be able to open doors. The ability to open doors widens drastically the area that the robot can reach and increases its utility significantly since it enables the robot to perform a larger set of tasks. Opening doors autonomously has found recent attention but still remains an unsolved problem for several reasons: it is hard to identify precisely the position and size of doors and handles, it is hard to autonomously compute the right approach to grasping and manipulating the handle, and finally it is hard to compute a coordinated arm-base motion that opens the door wide enough for the robot to navigate through it. In this paper, we concentrate on addressing the last problem.

Opening a door using a mobile manipulation platform typically involves tight coordination in between the motion of the base and the motion of the arm. This makes the problem high-dimensional and thus hard to plan for. On the other hand, hard-coding motion plans a-priori is even harder due to high variability in the conditions under which the doors may need to be opened: doors vary in their sizes and types, some doors are opened by pulling and others by pushing, spaces around the doors may often contain obstacles (e.g., furniture) that limit how wide the doors open and the space the robot can navigate in.

The contribution of this paper is that it proposes a novel graph-based representation of this planning problem. This representation offers a number of advantages. First, it can be used to represent a wide range of environments with arbitrary obstacles and varying door types and sizes. Second, the representation encodes the non-holonomic constraints of the base and allows us to find feasible arm motions that correspond to the plans generated using this representation. Finally, being a graph-based representation it can take advantage of the wide research on heuristic search-based planning and in particular of anytime and incremental searches suitable for planning and re-planning in partially-known environments in real-time [1], [2]. We demonstrate the effectiveness of our approach on the real mobile manipulation platform, PR2 robot (Figure 1), by using it to open autonomously a variety of doors, both the ones opened by pulling and pushing.

Fig. 1. The PR2 robot pulling open a door.

A. Related Work

Door opening has recently found widespread attention. There are two parts to the door opening problem: (a) detecting and identifying doors and handles and (b) opening the door. There has been extensive research into the task of identifying doors and handles([3]–[8]). Our door and handle recognition system is built on the work in [9] where a laser based perception was used to detect doors and handles in an indoor office environment. The detection system is robust

and provides complete information about the location of the handle in a local frame attached to the door.

The actual task of opening doors has been addressed using a variety of approaches dating back more than a decade [10], [11]. In [12] a set of behaviors for reliably opening a variety of doors are presented with over 87.5% success rate. The approaches rely on using a force sensor to detect the position of the handle on the door after which the door is pushed open using the arm. However, no results were presented for pulling doors open and the system was extremely slow (on the order of a few minutes to open a single door). In [13], a door opening approach using equilibrium point control was presented that could detect, navigate towards and push open doors in about 2 minutes. Again, no results were presented for pulling doors open. In [8], a cartesian impedance controller was used with a mobile manipulator to push doors open. Again, there was no attempt at pulling doors open. In [14], online estimation of the door model is combined with a hybrid system model to open doors. In [15], compliant control was used to open doors using a mobile manipulator. In [16], the robot arm operated the door handle to open the door. The base then drove through the door with the arm extended out in front of the robot to push the door open. No results were presented for opening doors by pulling on them. In [17], a comprehensive door opening system was presented that was able to open a series of doors consistently without any failures. Reactive controllers were used in the system to coordinate the motion of the arm and base with the door. This system, however, dealt only with opening doors by pushing them.

The approaches mentioned above do not carry out any planning and neglect collision avoidance with the door or with any part of the environment surrounding the door. In particular, when pulling open doors in situations where the mobile base of the robot has limited workspace, these systems are likely to collide either with the environment or with the door. Thus, planning for door opening is necessary in situations where a purely reactive controller would be unable to account for obstacles in the environment. A few planning based approaches have been presented earlier. In [18], a system for opening cabinet doors was presented where the motion of the arm of a mobile manipulator was explicitly planned using sampling based planners. However, the base was kept stationary throughout the door opening. This particular approach is not feasible with the robotic platform we use in our experiments since the workspace of the arm in front of the robot is not sufficient to open doors from a static position in front of the door.

Our approach can open doors consistently both by pulling or pushing doors open using a mobile manipulator. It accounts for obstacles by building a representation of the environment in realtime. The system accounts for collisions between the base of the robot and the door. It also plans a constrained path for the base to prevent collisions with obstacles in the environment (including walls). The constraint imposed on the base ensures that the door handle is always within the reachable workspace of the arm.

B. Structure of the paper

This paper is structured as follows. In Section II we present the PR2 robot used as an experimental platform for this task. We also present, in brief, the sensing systems that form an essential part of the door opening task. In Section III, we provide a system level description of our approach to door opening. In Section IV, we provide details about the motion planning approach used in this work. In Section V, we provide details about the implementation of this planner on a mobile manipulation platform and in Section VI, we present experimental results.

II. Hardware

The PR2 is a two-armed robot with an omni-directional base. It has an extensive sensor suite useful for mobile manipulation including a tilting laser scanner mounted on the head, two pairs of stereo cameras, an additional laser scanner mounted on the base and an IMU mounted inside the body. Encoders on each joint also provide continuous joint angle information. Each arm also has an additional camera mounted on the forearm. Each gripper has capacitive pressure sensors that can be used to detect the pressure applied by the gripper on an object.

Each arm has 7 degrees of freedom and the joints are torque controlled thus allowing for soft, compliant control. The extra degree of freedom allows the arm to achieve Cartesian goals while still retaining a redundant degree of freedom. This proves useful in extending the usable workspace of the arm, especially in cluttered environments. The base has four independently steered casters each of which has two independently driven wheels. The base is thus over-constrained and the different degrees of freedom must be properly coordinated for control.

The two sensors used for this work include the base laser and the scanning laser mounted on a tilting platform on the head. The base laser is only planar and operates at about ankle height above the ground but proves valuable in creating a 2D representation of the world for navigation. An additional planar laser is mounted on a tilting stage at shoulder level on the robot. The tilting laser can be used to create a complete 3D representation of the workspace in front of the robot.

III. System Architecture

Our approach to door opening has two components: (a) the door and handle detection and (b) planning and executing the door opening action. We use the approach described in [9] for door and handle detection. Our approach to the actual door opening task relies on a known model of the door. The model need not be extremely accurate but must specify the position of the handle of the door and its hinge in a common reference coordinate frame. The door and handle detection system [9] has access to a prior model of the door including its position, direction of rotation and hinge location from a topological database of the building. It utilizes this information as an initial guess to further refine its estimate of the door location, size and handle location using the tilting

laser scanner mounted on the head. The output from the door detection algorithm is thus a location for the handle and door including an estimate of where the hinge of the door is. This information, coupled with the rotation direction specified in the apriori model of the door serves as the input to the planning system used for door opening.

The door planning system utilizes the estimated door model provided by perception to execute a series of sub-tasks:

1) Move the base towards a position in front of the door so that the arm can reach the handle
2) Reach out, grasp the handle and unlatch the door
3) Determine from the model whether the door needs to be pushed or pulled open. Push or pull the door slightly to initiate the opening action.
4) Plan a coordinated arm-base motion to open the door. In the generated plan, the motion of the arm and base are coupled to ensure that the handle of the door stays within the reachable workspace of the arm, and the motion generated for the base is free of collisions with all obstacles including the door
5) If pulling the door open, check for collisions of the arm with the door. If collisions are detected, prune the plan until that point. Release the handle on one side of the door and move the arm so that it can grasp the handle on the other side of the door. This motion moves only the arm and keeps the base of the robot stationary. The motion of the arm must be collision-free.
6) If necessary, plan again the coordinated arm-base motion to complete the door opening process but this time by pushing the door.

The last step gets invoked only if the door is opened by pulling, requires re-grasping the handle from one side of the door to another (i.e., re-grasping during step 5), and is not yet open enough for clear passage.

We subdivide these tasks further into two sets: (a) the act of reaching out and grasping the handle and opening the door slightly and (b) the act of opening the door using a coordinated motion of the base and the arm of the robot. The task of grasping the handle is described in detail in [17] and will not be discussed here. Instead, we will now describe in detail the components necessary to achieve the coordinated motion of the base and the arm to open the door.

Coordinated motion of the base and the arm is a difficult task and requires careful planning and execution. An important component of this task is the ability to generate collision-free paths for the base of the robot in order to move in a direction that allows for the door to open, a task that is further complicated by the need to have the gripper holding the door handle throughout this motion. Thus, the path of the base is *constrained* by this requirement on the arm. In addition, the goal for the motion of the base is not immediately obvious. The only constraint on the goal is that the base should move to a position where it can fully complete the door opening task.

One approach to this task would be to use a *whole body planner* to plan in the complete configuration space of the robot, *i.e.* in the space of planar base motion and 7 degree of freedom arm motion. The requirement that the gripper should always be on the door handle necessitates the implementation of a *constrained* planner that can deal with this constraint. Planning a constrained task in such a high dimensional space is computationally expensive.

Instead, we choose a lower-dimensional graph-based representation that allows us to compute a collision-free motion for the base of the robot that at any point of time keeps the handle of the door within the arm's reach. Given this property of the generated motion, we can then use inverse kinematics to compute the trajectory for the arm of the robot that goes along with the motion generated for the base and maintains the constraint that the gripper stays on the handle of the door. This partitioning of the problem into two planning spaces greatly reduces the computational efforts required for planning. It allows our system to generate plans for the task in very short times. We now describe in detail our graph-based representation and how we search it for low-cost motions.

IV. Coordinated Arm-Base Motion Planning

The base of the robot must execute a motion that moves it into a position where the door can be fully opened. There are two constraints acting on the entire path of the base: (a) the base cannot collide with any part of the environment and (b) the motion of the base must allow the gripper to stay on the handle so that the door can be pushed or pulled open by the arm. Note that the *final goal* for the base motion is not specified. The goal for the entire task, however, is to open the door.

Our approach is based on a graph-based representation. In the following we first describe how the graph itself is formed. We then describe the cost function used to set up the weights of the graph. We finish with the short description of the graph search algorithm we use to search the graph for low-cost solutions.

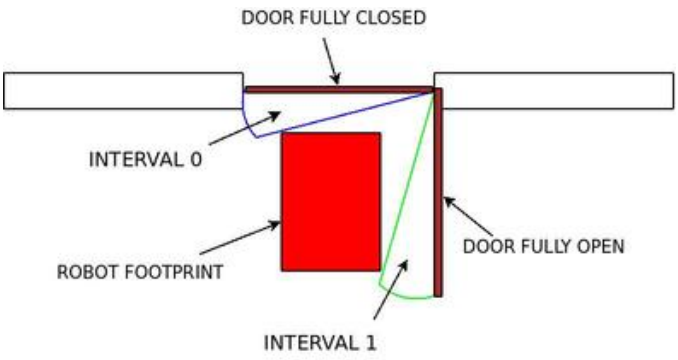

Fig. 2. A schematic representation of the door intervals. Door interval 0 contains all door positions that do not collide with the base and are connected to the fully closed door position. Door interval 1 contains all door positions that do not collide with the base and are connected to the fully open door position.

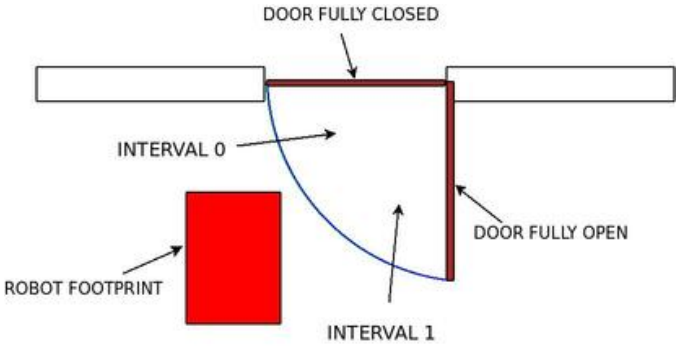

Fig. 3. Door interval 0 and door interval 1 now overlap, allowing the door to move freely between the fully open and fully closed positions.

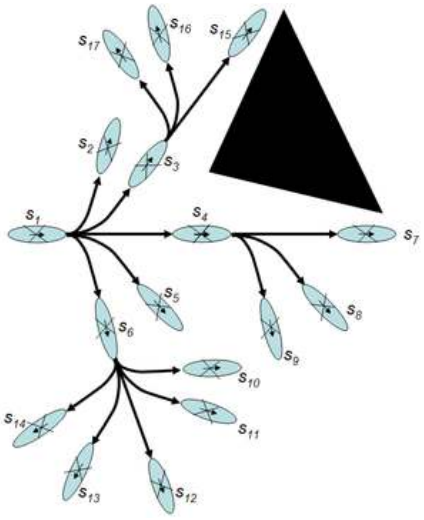

Fig. 4. A 3D (x, y, θ) lattice with at most five forward actions for each state and no backward actions. A full set of actions is shown for state s_1. For every state, this set of actions is translated and rotated appropriately, and all actions are checked for collisions by checking the footprint of the robot against the environment as the robot follows the action.

A. Graph Representation

1) State Variables: We first consider a three dimensional search space for the base representing the planar position degrees of freedom of the base (x, y, θ), where θ is the heading of the base. Note that the goal for the task, *i.e. opening the door* cannot be represented using this representation since it does not contain the door angle. We need to augment the state-space in order to represent the goal specification directly. The natural choice is to add an additional variable that represents the door angle itself. It turns out however that it is sufficient to use an even more compact representation of the door angle - using a single binary variable that we call *door interval*. We explain the meaning of *door interval* with an example.

Consider the door shown in Figure 2. For a given position of the base of the robot, all collision-free positions for the door belong to one of two intervals. One of these intervals is always connected to the fully closed position of the door, while the other interval is always connected to the fully open position of the door. When the robot moves back and out of the doorway, these two intervals overlap allowing the door to move freely between the fully closed and fully open positions (Figure 3). We label the interval connected to the fully closed position of the door as the 0 interval and the interval connected to the fully open door position as the 1 interval. Thus, each state in the search space can now be represented as (x, y, θ, d) where $d = 0, 1$ represents the interval that the door is currently in. The goal of having the door fully open can now be easily specified using the corresponding interval that the door should be in. A fully closed door will initially start off in the 0 interval. Once the door moves into the 1 interval, it is connected to the fully open position. While explicit representation of a state is always defined as (x, y, θ, d), we have the ability to reconstruct the set of door angles that are feasible given the pose of the robot. These are all the door angles for which the door handle is within the reach of the robot's arm and for which the door collides neither with environment nor with the robot. We denote this set of door angles as $\Lambda(s)$ for a state $s = (x, y, \theta, d)$. This set can always be constructed online for a given state s. This computation represents the most expensive part of our overall graph-search algorithm. Note however that this set is always finite since we discretize the set of possible door angles using a discretization of 1 degree.

We restrict our state-space to contain only those states s for which the set $\Lambda(s)$ is non-empty (in other words, for which there exist at least one door angle in the corresponding door interval whose handle is within the robot's reach). Using a binary variable to represent the door state reduces the size of the search space for motion planning. In contrast, a discretization of 1 degree would have resulted in 90 possible door angle states to search over for a door that opens through 90 degrees.

We can now formally define the set of goal states that need to be planned for. A state s is a goal state if it belongs to interval 1 and if $\Lambda(s)$ contains an angle that is closer than a threshold δ_d from the fully open door angle. In practice, we chose δ_d to be 5 degrees.

2) Transitions: In defining the transitions in between states, we followed a lattice-based planning representation [19], [20]. Lattice-based representation is a discretization of the configuration space into a set of states, and connections between these states, where every connection represents a (short-term) feasible path defined as a sequence of poses (see Figure 4 for an example of a lattice). As such, lattices provide a method for motion planning problems to be formulated as graph searches. However, in contrast to many graph-based representations (such as 4-connected or 8-connected grids), the feasibility requirement of lattice connections guarantees that any solutions found using a lattice will also be feasible. This makes them very well suited to planning for non-holonomic and highly-constrained robotic systems.

The PR2 robot is capable of rotating in place, moving sideways and moving forwards and backwards while rotating. We therefore included all of these actions as possible transi-

tions used in the lattice. During the search (planning), these actions are translated and rotated for each state encountered by search, and the successors of the state are computed as the corresponding end configurations of these actions. The planner also checks all actions against collisions by checking the footprint of the action (constructed from the footprint of the robot) against the map of the environment.

Unlike in the usual lattice-based representation however, our transitions also include the transitions for the door interval d. We define these transitions as follows: a state s contains a transition to any state s' such that $d(s) \neq d(s')$ if and only if $\Lambda(s) \cap \Lambda(s') \neq \emptyset$ and x, y, θ variables for states s and s' are the same. In other words, the transition between s and s' whose door intervals are different exists if and only if the robot is out of the way of the door and can reach the door handle (Figure 3).

Finally, a transition between any two states s and s' whose door intervals are the same exists if and only if this transition corresponds to one of the actions in the lattice *and* for each two subsequent poses of the robot along this action the corresponding Λ intervals are overlapping. The latter condition ensures that as the robot executes any action in our lattice, it will be able to maintain its gripper on the door handle.

B. Cost function

The cost used for the planner is a combination of a 2D cost that represents the distance of the base to the nearest obstacle and an arm configuration-based cost which tries to place the arm in a *comfortable* position with respect to the base.

1) 2D Costmap: In order to execute collision-free navigation, the robot needs to be aware of the world around it. However, it is computationally expensive to store and process at reasonable speeds a detailed and high resolution representation of the entire world in 3D space. Thus, we choose to essentially treat the navigation problem as a 2 dimensional problem by projecting 3 dimensional information onto a 2 dimensional grid. This process is described in more detail in [21]. The result of this process is a 2D costmap where the cost represents the distance to the closest obstacle.

Since the robot will be moving around continuously during navigation, it is important to project the sensor information into a *fixed coordinate frame*. Since the door opening task is localized to a small area, we choose to work in a local odometric frame. Although odometry drifts over large distances, the error accumulated due to odometry is very small for the short distances that the robot has to travel in performing the door opening task. Figure 5 shows a snapshot of the costmap while the robot is positioned in front of a door. During planning, the footprint of any transition encountered by our graph search while constructing the lattice is convolved with the 2D costmap and the cost of the transition is computed.

2) Arm-based cost: The position of the robot base is constrained by the requirement that the gripper remains on the door handle throughout the door opening motion. This constraint can also be interpreted as a requirement that the door handle should be within the reachable workspace of the robot. In addition to the hard constraint, we also prefer that the arm is able to reach the door handle *comfortably*, *i.e.* the door handle should be well within the workspace of the arm and not at the boundaries where joint limits may limit the range of motion available to the arm. We enforce this preference by imposing an additional cost based on the position of the door handle in the shoulder frame of the arm of the robot. The minimum cost is imposed when the door handle is at a nominal distance from the shoulder. Greater costs are imposed when the door handle is further away or closer than this nominal distance. Thus, to compute the cost of any transition, the planner first iterates over all the poses that make up this transition, and for each pose p, iterates over all the possible door angles in the set $\Lambda(p)$ and picks the angle with the minimum cost. The overall cost of the transition is set to the cost associated with the pose p' that has the highest cost minimized over the angles in $\Lambda(p')$. This forces the planner to prefer motions that keep the door handle close to a nominal distance from the shoulder of the robot. Note that we are imposing a soft constraint here, i.e. the door handle need not be at a *fixed* distance from the shoulder but can be at any position within comfortable reach of the end-effector. This has the added benefit of reducing the risk of collisions between the arm and the door since the door is always kept a minimum distance away from the shoulder of the robot.

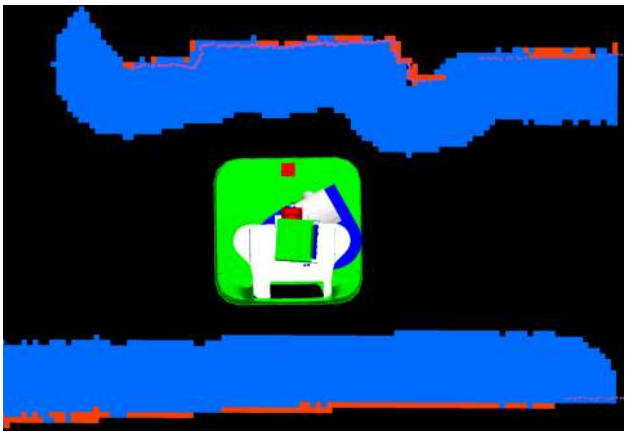

Fig. 5. A typical 2D costmap for a robot in front of a door.

C. Graph Search

Given a graph we defined above and a cost function associated with each action, we need an efficient method for searching it for a solution path. A* search is perhaps one of the most popular methods for doing this [22]. It utilizes a heuristic to focus the search towards the most promising areas of the search space. While highly efficient, A* aims to find an optimal path which may not be feasible given time constraints and the dimensionality of the problem. To cope with limited deliberation time, we use an anytime variant of A* - Anytime Repairing A* (ARA*) [1]. This algorithm generates an initial, possibly suboptimal solution

quickly and then concentrates on improving this solution while deliberation time allows. The algorithm guarantees completeness (for a given graph) and provides bounds on the suboptimality of the solution at any point of time during the search. Furthermore, this bound, denoted by ϵ, can be controlled by a user. In all of our experiments, we set the initial ϵ to 5.0 implying that the cost of the returned solution can be no worse 5.0 times the cost of an optimal solution (even though the optimal solution is not known). In addition, practically in all of our experiments, ARA* was able to decrease the bound on suboptimality to 1.0 within the time we allocated for planning.

The heuristics estimate the amount by which the door still needs to be opened before it is considered to be fully open. That is, for any state s, let $\lambda(s)$ represent the *least cost* door angle corresponding to the position of the robot in that state. Further, let λ_{open} represent the fully open door angle. The heuristic is then proportional to $|\lambda(s) - \lambda_{open}|$.

V. Implementation

The output from the planner is a path for the base of the robot along with the interval that the door is expected to be in for every position of the base in the path. To compute the door position corresponding to every base position in the path, we choose to use the *least cost* door angle for that position. This represents a comfortable position where the arm can easily reach the door handle without getting close to joint limits. This information completely defines the postion of the base and the door along every step in the path. However, it is possible that the desired door opening angles may *stutter* as the robot opens the door, i.e. there may be several small back-forth motions of the door as it is opened.

This completely defines the trajectory of the two handles on either side of the door which in turn describes the pose of the end-effector for every position of the base in the planned trajectory. Since the poses of the base and the handle are completely known throughout the path, this implies that a desired cartesian path for the end-effector can also be computed easily. This desired path is passed through a inverse kinematics solver to get a desired joint space path for the arm. Since the robot arm is redundant, the inverse kinematics solver requires an initial guess for one of the joint angles. It then computes an analytical solution for the other 6 degrees of freedom by discretizing and searching over the entire reachable space for this specified joint.

The solution for a waypoint in the path is used to seed the search for the next waypoint. It is possible that the inverse kinematics solution may exhibit *divergence*, i.e. two consecutive solutions along the desired trajectory may be far apart. However, in practice, restricting the door handle to be within a *comfortable* reach of the shoulder of the robot implies that the arm always has access to a large part of the workspace throughout its motion. This minimizes the risk of divergent inverse kinematics solutions.

We now describe in detail the process of executing the paths generated by the planner on both the arm and the base of the robot to get the required coordinated motion necessary for opening the door.

A. Trajectory Generation

The path returned by the planner is time-parameterized before being passed onto the controllers. A crucial goal of this *trajectory generation* step is to create a smooth motion of the door and robot's base and arm. The trajectory generation only applies smoothing in the velocity space, the desired path of the robot remains unchanged. This process restricts the velocities of the robot's joints to ensure safe execution of the door-opening task. The input to the trajectory generation process is the path for the base generated by the planner and the door path computed from this information. The output of the trajectory generation module is a time parameterized trajectory of base and door positions that is passed to a lower-level trajectory controller.

B. Control

The output from the trajectory generation process is a time-parameterized trajectory that specifies positions, velocities for the base of the robot, the door and the arm. Separate controllers drive the base and the arm along the path specified by the planning system. The low-level base and arm controllers operate inside a dedicated realtime framework where controllers run at a 1KHz frequency. This allows the trajectory controllers to execute tightly coordinated motions plans for both the base and arm of the robot.

C. Arm planning

As mentioned in Section III, the arm may collide with the door while pulling the door back. This case is easily detectable by performing a simple collision check on the arm of the robot as it opens the door. If a point in the path corresponds to a base position where the arm cannot grasp the handle without coming into contact with the door, the path is pruned beyond that point. The system then creates a motion plan to move the gripper off the handle on one side of the door with the goal of grasping the handle on the other side of the door. This arm planning was implemented using a sampling based motion planning approach [23]. Once the robot grasps the handle on the other side of the door, it uses the door opening planner to create the motion plan required to finish the door opening task. In most cases, this last action involves only a small motion of the door where the robot uses its arm to push the door open while keeping the base stationary.

It is possible that there may be no solution to allow the arm to regrasp the handle on the other side of the door without moving the base. We do not handle this situation for now and aim to address it in the future by using a base navigation planner to move the robot to a position where it can reach the handle on the other side and finish the door opening motion.

VI. Experimental Results

The door planner was implemented on the PR2 mobile manipulation robot. It was tested on both kinds of door

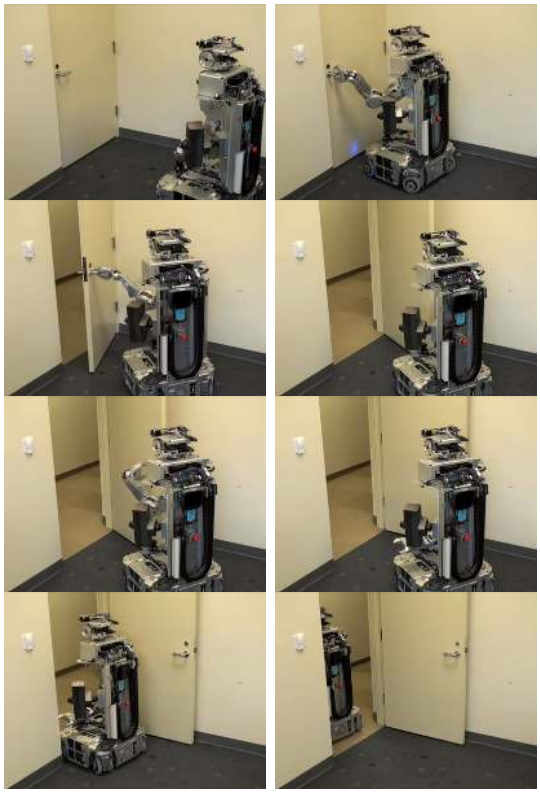

Fig. 6. Pulling doors open.

Fig. 7. Pushing doors open.

Planning Time until First Soln. (s) (ϵ =5.0)	Solution Cost of First Soln. (ϵ =5.0)	Planning Time until Final Soln. (s) (ϵ =1.0)	Number of Expands until Final Soln. (ϵ =1.0)	Solution Cost of Final Soln. (ϵ =1.0)
5.18	517,659	29.59	2,303	517,659
3.36	542,036	34.51	2,706	540,083
4.22	574,174	25.90	2,024	518,604
3.61	571,961	28.62	2,216	543,418

TABLE I

PLANNING TIMES FOR PUSHING DOORS OPEN. LEFT TWO COLUMNS ARE FOR GETTING A FIRST SOLUTION, AND RIGHT THREE COLUMNS ARE FOR GETTING A FINAL SOLUTION.

opening tasks: pulling and pushing doors open. The tests were setup with the robot placed in front of a candidate door. A rough estimate of the door model was provided to the robot which it then refined by detecting the door and handle using its laser scanner. The robot then moved towards the door and used its arm to open it slightly. The door planning system then planned a path for the base to open the door. The plan was then filtered and executed using trajectory controllers for the base and the arm.

Multiple runs were carried out for door opening using pulling and pushing. Snapshots of the PR2 pulling open a door are presented in Figure 6 while those of the robot pushing a door open are presented in Figure 7. In 5 different runs, the planner was able to consistently push open doors to different rooms. In addition, in 5 different trials, it was able to successfully pull open the door in a single room.[1]

Planning times and the number of nodes expanded for different runs are listed in Table I and Table II. Pulling doors takes on average less time to plan for than pushing doors open. The robot is essentially moving through open space while pulling doors open but is going through a narrow doorway when pushing doors open. The planned trajectory for pushing doors open contains states that are close to the door and the door frame. These states have higher costs because of their proximity to obstacles. Finding an optimal plan requires the planner to expand a lot of states with higher costs when pushing doors open as compared to pulling doors open and this makes plan computation much more expensive.

[1]The door detection algorithm used for the experiments uses 3D geometric information to detect doors by looking for planar regions that match a model of the door. This required door planes to be offset from the walls around them. The detector was able to detect only one door reliably when the robot was inside a room and needed to pull on the door to open it. However, door detection is not the primary focus of this work and more detailed experiments with different doors will be performed in the future once the door detection algorithm is updated to detect different types of doors.

VII. CONCLUSION

In this paper, we have proposed a novel graph-based representation for the problem of planning to open doors. We have successfully demonstrated our approach on the PR2 mobile manipulation platform with the ability to consistently open doors both by pushing and pulling. We showed that our compact graph-based representation of the problem can be used to dramatically improve the efficiency of planning for the originally high-dimensional constrained task. Our approach allowed us to avoid collisions in between the base of the robot and the environment while carrying out this constrained task. In addition, we were able to use a coordinated motion of our base and arms to generate smooth motions for opening doors.

We believe that the use of a compact representation will

Planning Time until First Soln. (s) (ϵ =5.0)	Solution Cost of First Soln. (ϵ =5.0)	Planning Time until Final Soln. (s) (ϵ =1.0)	Number of Expands until Final Soln. (ϵ =1.0)	Solution Cost of Final Soln. (ϵ =1.0)
2.43	366,711	6.77	626	256,914
2.15	233,445	3.95	633	233,445
0.56	180,076	0.87	95	180,076
0.11	224,870	0.25	16	103,011
0.20	126,036	0.27	27	126,036
0.24	138,264	0.37	33	138,264
0.20	125,728	0.33	16	125,728

TABLE II

PLANNING TIMES FOR PULLING DOORS OPEN.

allow this approach to be applicable to other robotic systems as well. We have made only simple assumptions about the geometry of the manipulator on the mobile platform in developing our approach. Our approach does require good sensor coverage around the robot to detect all possible obstacles before the door opening is initiated. In addition, as the robot executes the motion, it has limited sensor coverage of the area behind it and may hit dynamic obstacles. We aim to use an additional stereo camera mounted on the head to provide sensor coverage behind the robot. Our system also does not handle errors that may arise while opening the door, e.g. a person holding the door closed. Further, we ignore the dynamics of the door itself. This may lead to situations where the door keeps swinging after being partly opened and hits the base. We aim to address this in the future by exploring more compliant controllers and implementing reactive collision avoidance for the base of the robot.

In the future, we aim to extend our approach to opening and closing more complex doors and movable parts of a domestic environment including fridge doors, spring-loaded doors, kitchen drawers and cabinet doors. Opening spring-loaded doors in particular may require the development of a more complicated planning framework that allows the robot to switch between the two arms and use one to prop the door open. We also aim to incorporate search-based motion planning into other parts of the door opening system including the part that grasps the handle. Finally, we would like to make use of incremental graph search techniques to be able to react quickly to the presence of humans in the workspace of the robot arm.

Acknowledgments The research efforts of Benjamin Cohen and Maxim Likhachev were supported by Willow Garage. We would also like to thank the members of the Willow Garage development team for their help.

REFERENCES

[1] M. Likhachev, G. Gordon, and S. Thrun, "ARA*: Anytime A* with provable bounds on sub-optimality," in *Advances in Neural Information Processing Systems (NIPS) 16*. Cambridge, MA: MIT Press, 2003.

[2] A. Stentz, "The focussed D* algorithm for real-time replanning," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1995, pp. 1652–1659.

[3] D. Anguelov, D. Koller, E. Parker, and S. Thrun, "Detecting and modeling doors with mobile robots," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2004.

[4] J. K. T. Alexander Andreopoulos, "A Framework for Door Localization and Door Opening Using a Robotic Wheelchair for People Living with Mobility Impairments," in *Robotics: Science and Systems, Workshop: Robot Manipulation: Sensing and Adapting to the Real World, Atlanta, June 30*, 2007.

[5] A. Andreopoulos and J. Tsotsos, "Active Vision for Door Localization and Door Opening using Playbot: A Computer Controlled Wheelchair for People with Mobility Impairments," *Computer and Robot Vision, 2008. CRV '08. Canadian Conference on*, pp. 3–10, May 2008.

[6] E. Aude, E. Lopes, C. Aguiar, and M. Martins, "Door Crossing and State Identification Using Robotic Vision," in *8th International IFAC Symposium on Robot Control (Syroco 2006), Bologna, Italy, September*, 2006.

[7] A. Y. N. Ellen Klingbeil, Ashutosh Saxena, "Learning to Open New Doors," in *Robotics Science and Systems (RSS) workshop on Robot Manipulation*, 2008.

[8] C. Ott, B. Baeuml, C. Borst, and G. Hirzinger, "Autonomous Opening of a Door with a Mobile Manipulator: A Case Study," in *6th IFAC Symposium on Intelligent Autonomous Vehicles (IAV)*, 2007.

[9] R. B. Rusu, W. Meeussen, S. Chitta, and M. Beetz, "Laser-based perception for door and handle identification," in *International Conference on Advanced Robotics*, 2009.

[10] K. Nagatani and S. Yuta, "An experiment on opening-door-behavior by an autonomous mobile robot with a manipulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1995, pp. 45–50.

[11] G. Niemeyer and J. Slotine, "A simple strategy for opening an unknown door," in *IEEE International conference on Robotics and Automation*, vol. 2, 1997.

[12] A. Jain and C. C. Kemp, "Behaviors for Robust Door Opening and Doorway Traversal with a Force-Sensing Mobile Manipulator," in *RSS Manipulation Workshop: Intelligence in Human Environments*, 2008.

[13] ——, "Behavior-Based Door Opening with Equilibrium Point Control," in *RSS Workshop on Mobile Manipulation*, 2009.

[14] L. Petersson, D. Austin, and D. Kragic, "High-level control of a mobile manipulator for door opening," in *IEEE International Conference on Intelligent Robots and Systems*, 2000.

[15] C. Rhee, W. Chung, M. Kim, Y. Shim, and H. Lee, "Door opening control using the multi-fingered robotic hand for the indoor service robot," in *IEEE International Conference on Robotics and Automation*, 2004.

[16] E. Klingbeil, A. Saxena, and A. Y. Ng, "Learning to Open New Doors," in *RSS Workshop on Robot Manipulation: Intelligence in Human Environments*, 2008.

[17] W. Meeussen, M. Wise, S. Glaser, S. Chitta, C. McGann, P. Mihelich, E. Marder-Eppstein, M. Muja, V. Eruhimov, T. Foote, J. Hsu, R. B. Rusu, B. Marthi, G. Bradski, K. Konolige, B. Gerkey, and E. Berger, "Autonomous Door Opening and Plugging In using a Personal Robot," in *International Conference on Robotics and Automation*, 2010.

[18] R. Diankov, S. Srinivasa, D. Ferguson, and J. Kuffner, "Manipulation Planning with Caging Grasps," in *IEEE International Conference on Humanoid Robots*, December 2008.

[19] M. Pivtoraiko and A. Kelly, "Generating near minimal spanning control sets for constrained motion planning in discrete state spaces," in *IEEE International Conference on Intelligent Robots and Systems*, 2005, pp. 3231–3237.

[20] M. Likhachev and D. Ferguson, "Planning long dynamically-feasible maneuvers for autonomous vehicles," *International Journal of Robotics Research (IJRR)*, 2009.

[21] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, "The office marathon: Robust navigation in an indoor office environment," in *International Conference on Robotics and Automation*, 2010.

[22] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems, Science, and Cybernetics*, vol. SSC-4, no. 2, pp. 100–107, 1968.

[23] I. Sucan, M. Kalakrishnan, and S. Chitta, "Combining Planning Techniques for Manipulation Using Real-time Perception," in *International Conference on Robotics and Automation*, 2010.