Towards Time-Optimal Trajectory Planning for Pick-and-Transport Operation with a Mobile Manipulator

Shantanu Thakar[1], Liwei Fang[1], Brual Shah[1], and Satyandra Gupta[1]

***Abstract*— Warehouses and factories are beginning to deploy mobile manipulators for transporting parts between machines and work stations. Minimizing the number of these robots on the shop floor requires that each robot should complete the assigned task in a time-optimal manner and therefore maximize the robot's capacity. In this paper we present a search-based algorithm for generating time-optimal trajectories for picking and transporting parts using a mobile manipulator. The approach is based on a hierarchical search that uses discrete search for planning of the mobile base. The time cost due to the motion of the manipulator for picking up the part is evaluated and integrated with the search state space at the appropriate times. The trajectories generated result in picking up of the part with the manipulator while the mobile base is in motion. The performance of the algorithm is studied for different factory sizes and obstacle densities. This approach enables the mobile manipulator to perform pick and transport operation in a smaller time duration compared to the operation where it stops to pick up the object.**

I. Introduction

Many material handling and inspection tasks require manipulating parts and tools over large distances. Representative examples include moving a part from one machine to another machine in a job shop, moving an ultrasound sensor over a large structure to perform inspection, or moving of products in warehouses. Mobile manipulators that integrate a robotic manipulator and a mobile robot in a single platform can be useful in such tasks [1], [2]. Recent works have shown usefulness of mobile manipulators in a variety of applications [3]–[7].

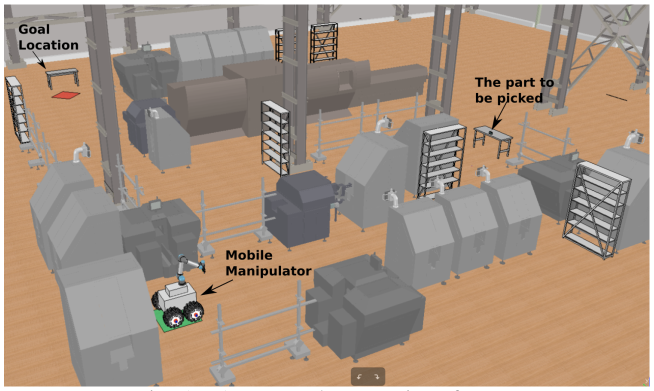

Fig. 1: An example scene in a factory

A potential way to utilize mobile manipulators is to decouple mobility and manipulation modes like in [8]. This means that the mobile manipulator first uses mobility mode to position itself at the task location. Then the manipulation mode is used to carry out the pick-up task. This decoupling of mobility and manipulation modes significantly simplifies the planning problem. The motion planning for the mobile base and the robot manipulator can then be solved independently, by using their respective existing methods. However, this approach has significant performance limitations as it does not guarantee the shortest time operation for the combined system. For example, for the scene in Fig. 1, the mobile manipulator takes about 60 seconds to reach the goal position from the current location without picking up the part. If the mobile base stops and then picks up the part with maximum joint rates of 15 degrees/second it adds a delay of about 14 seconds to the path, whereas picking up the part while moving adds only about 2 seconds delay to the path (Fig. 9).

In most factory and warehouse scenarios like in Fig. 1, increasing the number of deliveries made by a single mobile manipulator has significant performance benefits. Independent planning of the mobile base and the manipulator increases the execution time leading to reduction in throughput. This may increase the necessity to have more mobile manipulators resulting in congestion and further degrading the throughput. Therefore, minimization of the task execution time is a desired objective when performing pick-and-transport tasks. This motivates for the use of the mobility and the manipulation modes simultaneously (i.e the manipulator picks the object while the mobile base is moving).

Concurrent utilization of mobility and manipulation creates a need to solve a complex planning problem for the mobile manipulator. In this paper, we present a pick-and-transport operation that requires the mobile manipulator to travel from an initial location, pick up a part with the manipulator while the mobile base is still in motion, and finally move towards a goal location. In other words, the objective here is to find time-optimal trajectories for the mobile manipulator system that picks a part with a known pose and drops it off at a goal location.

We assume that the pose of the part that needs to be transported is known. Instead of stopping the mobile base, the manipulator picks up the part while the mobile base is still moving. The uncertainty in the pose of the part will make this process less robust as compared to grasping when the mobile base is stationary. However, by carefully choosing the grasping strategy (i.e., the way the part is to be picked up) we can make the process robust while saving on the time.

In this paper, we focus on the motion planning of the

[1]S. Thakar, B. Shah, L. Fang and S. K Gupta are with Aerospace and Mechanical Engineering Department, University of Southern California, 90089 CA, USA `sthakar, brualsha, liweifan, skgupta at usc.edu`

mobile base from an initial pose to a goal pose while passing through an intermediary location. The grasping of the part with the manipulator should happen around this intermediary location. The main challenge here is that the intermediary location which will give the fastest path is initially unknown. Hence, it is not possible to define an intermediary goal point for this location before the planning begins. However, we know an area surrounding the part within which if the mobile base is located, successful grasping of the part is possible. Hence, it is necessary to bias the search towards this area, grasp the part with the manipulator while the mobile base is still in the area and subsequently move towards the goal location. In most cases, shortest mobile base paths lead to fastest mobile base trajectories. However, if the manipulator joint rates are limited, then the mobile base may have to slow down to ensure that the manipulator has sufficient time to pick up the part. Hence, longer paths with no slow down may result in faster mobile base trajectories. On the other hand, there may be cases where stopping the mobile base to pick up the part is necessary in a time-optimal trajectory. The main contribution of this work is an algorithm that generates such time-optimal trajectories for the mobile manipulator that result in successful grasping of the part.

II. Related Work

Sampling-based motion planning techniques like RRT, PRM and their variants, [9]–[13] are very useful in wide variety of motion planning problems with high dimensional configuration spaces. They are computationally fast. However, they have limitations when it comes to the problem discussed in this paper because of optimality requirement. Moreover, there is an intermediary location at which the part is to be picked up which is not known at the start, hence biasing the sampling towards this intermediary target is challenging.

Lattice-based planners [14]–[18] used with graph search algorithms are promising for motion planning of a mobile manipulators. These approaches compute the globally optimal paths but take significant computational time. Hence, using these approaches requires use of search space reduction method. Using these approaches independently for the mobile base and the manipulator for finding optimal paths for each, may not result in the global optimal path. Hence, it is necessary to plan coordinated base-manipulator motions, which respect the joint limits, avoidance of self-collisions as well as collisions with obstacles in the environment.

Primitive-based approaches [19]–[21] are used for motion planning of redundant manipulators. Here, the adaptive primitives are executed in the search-based on the distance from the target configuration. There may be viable paths of the mobile base where neither of the primitives of the manipulator is able to grasp the part. Hence, designing primitives for the application mentioned here becomes critical.

Optimization based algorithms like STOMP [22], CHOMP [23] can be used to generate smooth trajectories for high dimensional systems. CHOMP iteratively improves an initial seed trajectory by using functional gradient techniques. STOMP generates multiple candidate trajectories by sampling around an initial seed and iteratively improving them through stochastic optimization with an estimated gradient.

Multi-modal and hierarchical planners like in [24]–[26] can be used for high-dimensional systems like mobile manipulators where the motions for the mobile base and manipulator have very different properties. These algorithms are very effective in generating a feasible solution in complex systems, however the solution generated may not be optimal.

III. Problem Formulation

The planner needs to compute a time-optimal, collision-free trajectory for the mobile manipulator. It should be feasible with respect to joint rates of the manipulator and the forward velocity and steering rate of the mobile base, between the start and the goal states of the Mobile Manipulator. The grasping strategy given initially should be executed for picking up a part with a known pose. More formally, given:

(i.) The continuous state space $\chi = \chi_\eta \times \chi_\Theta$ consisting states $\mathbf{x} = [\eta^T, \Theta^T]$, where $\eta = [x, y, \phi]^T \in \chi_\eta \subset \mathbb{R}^2 \times \mathbb{S}^1$ is the pose of the mobile base and $\Theta = [\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6]^T \in \chi_\Theta \subset \mathbb{S}^6$ is the configuration of the 6 degrees of freedom (DoF) robotic manipulator fixed on the mobile base.

(ii.) The start state is $\mathbf{x}_{M,S}$ and the goal state is $\mathbf{x}_{M,G}$ of the mobile manipulator.

(iii.) Θ lies within the range $[\Theta_{min}, \Theta_{max}]$ which for the considered manipulator (UR5) is $(-\pi, \pi]$.

(iv.) The continuous, state-dependent control action space $\mathcal{U}(\mathbf{x}_M) \subset \mathbb{R} \times \mathbb{S}^1$ of the mobile base with each control action primitive $\mathbf{m}_c = [V, \psi_d]^T$ consists of a forward speed V, and a heading ψ_d. All control action primitives take a constant time T_p. V is constant for the entire planning process. Forward speed constraint $V < V_{max}$. The continuous, state dependent action space for the manipulator is $\dot{\Theta} = [\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4, \dot{\theta}_5, \dot{\theta}_6]^T$. The joint rate constraint $\dot{\Theta} \leq \dot{\Theta}_{max}$

(v.) $\mathcal{W}$: The world frame of reference; $\mathcal{P}$: Part frame; $\mathcal{G}$: Gripper frame; $\mathcal{R}$: Mobile base frame; $\mathcal{M}$: Manipulator base frame; $\mathcal{B}$: Table frame; ${}^iT_j$: Frame j with respect to frame i.

(vi.) ${}^{\mathcal{W}}\widetilde{T}_{\mathcal{P}}$ is the estimated pose of the part in the world frame. ${}^{\mathcal{P}}T_{\mathcal{G}}$ is the gripper pre-grasping(insert) pose for a part pose $\mathcal{P}$ for a particular grasping strategy $\mathcal{G}_s({}^{\mathcal{W}}\widetilde{T}_{\mathcal{P}})$.

(vii.) The geometric region $\mathcal{O}_s = \bigcup_{k=1}^{K} o_{s,k} \subset \mathbb{R}^2$, occupied by static known obstacles which may hinder the motion of the mobile base as well as the manipulator. $\mathcal{O}_T \in \mathcal{O}_s$ is the table as an obstacle. A representative scenario is shown in Fig. 1.

Compute:

A time-optimal, collision free trajectory $\tau : [0, T] \to \chi$, satisfying velocity and joint rate constraints, such that $\tau(0) = \mathbf{x}_{M,S}$, $\tau(T) = \mathbf{x}_{M,G}$ and T is minimized, and the grasping strategy $\mathcal{G}_s({}^{\mathcal{W}}T_{\mathcal{P}})$ is executed at time T_g, where $T_g \in [0, T]$. Each state $\mathbf{x}_M(t)$ along τ belongs to the free space $\chi_{free} = \chi \setminus \chi_{obs} =$

$\{\mathbf{x}_M(t) | M(\eta_M(t), \Theta_M(t)) \cap \mathcal{O}_s = \emptyset\}$ for $t \in [0, T]$, where $M(\eta_M(t), \Theta_M(t)) \subseteq \mathbb{R}^3$ is the region in 3D space occupied by the mobile manipulator.

IV. Grasping Strategy and Grasping Area

For a part, there can be different ways in which it can be picked up with a two fingered gripper. Each of these ways is called a grasping strategy. Fig. 2 shows a part and the three grasping strategies associated with it. It can be observed that all the three grasping strategies are robust to uncertainty in the pose of the part as they have a large overlap with a wide two fingered gripper. Hence, we consider only these grasping strategies. A grasping strategy consists of insertion, grasping, and post-grasping (or retract) poses of the end effector for any particular part pose, given by ${}^{\mathcal{W}}T_{\mathcal{G}}(t)$. The grasping strategy to be used is given before the planning begins. Although grasping strategies shown in Fig. 2(b) and Fig. 2(c) seem to be mirroring each other, they depend on the placement of the part with respect to the table and may differ significantly in terms of collisions and inverse kinematics (IK). For a part, these grasping strategies can be generated by performing grasp planning [27], [28].

For a given pose of the part, the grasping strategy decides the pose of the gripper called the grasp-pose with which it is supposed to approach the part. For a given grasping strategy ${}^{\mathcal{W}}T_{\mathcal{G}}(t)$, the pose of the part ${}^{\mathcal{W}}T_{\mathcal{P}}$, and the table orientation ${}^{\mathcal{W}}T_{\mathcal{B}}$, we can compute a region consisting of valid mobile base poses from which the entire grasping strategy is executable with respect to IK, and without collision between the table and mobile manipulator. We call this region the Grasping-Area $\mathcal{A}_g(\phi) \subset \mathcal{A}_G$. Where, $\mathcal{A}_g(\phi)$ is the grasping area dependent on the orientation of the mobile base ϕ (i.e., yaw). $\mathcal{A}_G$ is the grasping area for the entire range of ϕ i.e., $(-\pi, \pi]$. For each grasping strategy and the location of the part along with its table, there is a grasping area $\mathcal{A}_{G,a}$, where $\mathcal{A}_{G,a}$ is denotes the grasping area for the entire range of ϕ for the grasping strategy a. It can be observed from Fig. 3, that the grasping area is dependent on the pose of the part, the table, and the orientation of the base.

The grasping area for a corresponding grasping strategy is computed by sampling collision free points for the pose of the mobile base. Further, checking if a valid and collision free inverse kinematic solution is available for the manipulator. This generates exhaustive grasping area for each orientation ϕ of the mobile base.

V. Planning Algorithm

The planning of the mobile base is based on the lattice-based planning for dynamically feasible trajectories [15]. The lattice-based representation results in the discretization of the configuration space into a set of states and connections between these states.

A. Graph Representation

Let $G = (S, E)$ represent the lattice-based graph, where S is the set of all discrete states, and E represents the transitions between any two states. Each state is the discretized 3 DoF state of the mobile base represented as a tuple (x, y, ϕ). And, each edge E is the from the set of predefined feasible

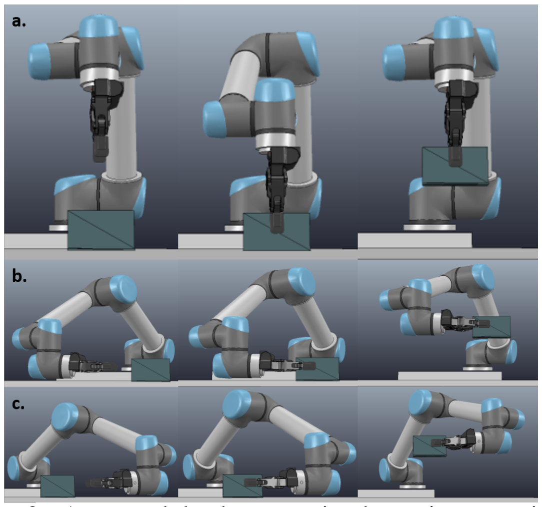

Fig. 2: A part and the three associated grasping strategies

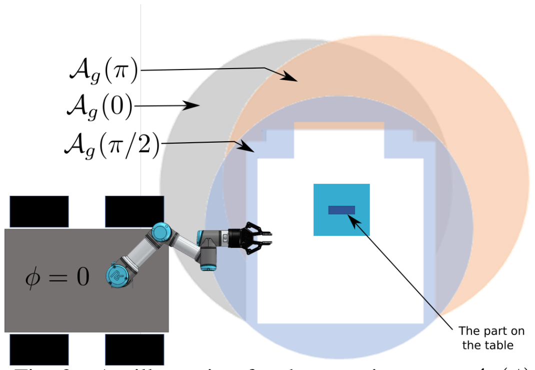

Fig. 3: An illustration for the grasping areas $\mathcal{A}_g(\phi)$

paths or motion primitives from one such state to another. For each state, there is a set of n motion primitives to chose from to move to the successive state. The mobile base is a differentially drive robot and the primitives considered for its motion are constant time primitives, i.e the time to go from a state to any of its successor states is constant (T). The turning time and the forward motion time will be different for each primitive such that the total primitive time is T. The following are the rules for marking colors to nodes during graph construction as shown in Fig. 4. They are formalized in algorithm 3 in lines 13 to 23.

- The start node and the consecutive nodes are marked red if they are not in the Grasping Area $\mathcal{A}_g(\phi)$, where(ϕ) is the mobile base orientation at the particular node).
- Once a node is expanded in $\mathcal{A}_g(\phi)$, it will be marked as yellow. All children of a yellow node are blue.
- All children of a blue or a green node are marked green.

B. Cost Function

Since the objective is to minimize the time taken to go from the initial configuration to the target configuration, the cost function is in units of time. The cost function is $F(S) = G(S) + H(S)$, where $G(S)$ is the cost-to-come to node S from the start node and $H(S)$ is the cost-to-go from node S to the goal. As mentioned in the Sec. V-A, the time taken to traverse between any two neighboring states is a constant T. The G cost of any state S' whose parent is S is $G(S') =$

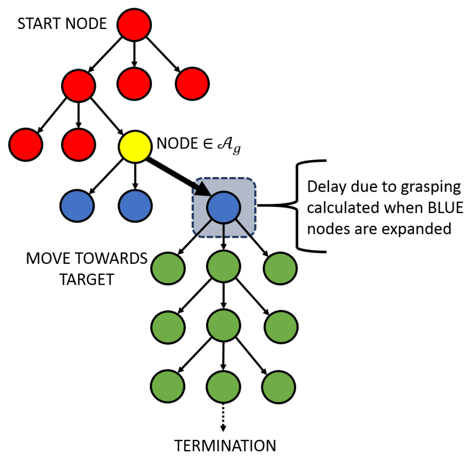

Fig. 4: Color scheme in the search

$G(S) + T$. The H cost is explained in Sec. V-C.

C. Heuristic

We are interested in constructing a computationally efficient and admissible heuristic. In this case, the mobile base must reach the Grasping-area $\mathcal{A}_G$ before moving towards the goal point. Initially, the heuristic should guide the search towards the grasping area. Once, the search reaches the grasping area, the heuristic should then guide it towards the final goal point. Fig. 5 explains the two different heuristics.

The red nodes are the ones where the search hasn't reached the grasping area. Hence for them we use the heuristics H_1 as described in Fig. 5a. For nodes which are in the grasping area (i.e yellow) or have an ancestor in the grasping area (i.e blue or green nodes), the search uses heuristic H_2 as described in Fig. 5b. The algorithm 1 describes how these heuristics are generated. The Fast-Marching method (FMM) [29] is used and generates an array with shortest distance values for each discrete point in the workspace to the point about which it is calculated. It must be noted that this is a pre-computation operation and takes about 3-5 seconds in MATLAB to compute for a given scene, part and table poses. Here using $\mathcal{A}_G$, the union of all grasping areas makes sure that the heuristic is admissible.

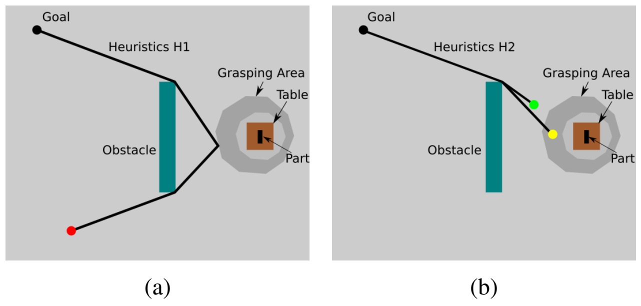

Fig. 5: Description of the heuristic before (a) and after (b) the search reaches the grasping area $\mathcal{A}_G$

D. The Algorithm

The algorithm 3 describes the search procedure for the motion planning of the mobile base. The differences from the traditional A^* search are the color scheme and a delayed evaluation of the path when particular types of nodes are expanded. Every successive node of the current node being

Algorithm 1 $COMPUTE\_HEURISTIC(T, \mathcal{A}_G, \mathcal{O}_s)$

```
Require: $T(Goal), \mathcal{A}_G$
  Consider all $n$ points $(P_1, P_2 \dots P_n)$ on the boundary of $\mathcal{A}_G$
1: for Each $P_i$ do
2:     Compute $FMM_{P_i}$ given $\mathcal{O}_s$
3: end for
4: Compute $FMM_T$ for the Goal given $\mathcal{O}_s$
5: for every (X,Y) location in the workspace do
6:     $H_1(X,Y) = min(FMM_{P_i}(X,Y) + FMM_T(X_{P_i}, Y_{P_i}))/V$
7: end for
8: $H_2(X,Y) = FMM_T(X,Y)/V$
9: return  $H = [H_1, H_2]$
```

Algorithm 2 $HEURISTIC(S, T, H)$

```
Require: $T(Goal)$
1: $X$, $Y$, $\phi$ = $S.state$
2: if $S.color$ is $RED$ then
3:     return  $H_1(X,Y)$
4: else
5:     return  $H_2(X,Y)$
6: end if
```

expanded has a prescribed color based on the rules mentioned in algorithm 3. A typical search tree is as shown in Fig. 4. Any prospective path consists of red nodes, followed by one yellow node and one blue node, and then green nodes. If the current node being expanded is a blue node, the $grasping\_delay$ function is called (see Sec. V-E). This function computes the delay if any caused due the grasping of the part with the given grasping strategy. Further, this delay is added to the g-cost of the blue node.

The algorithm terminates when a green node is chosen from the open list as the current node and it satisfies goal tolerance. This makes sure that the path passes through the grasping area, and has an evaluation of the delay caused due to grasping.

E. Manipulator Planning and Grasping Delay

For the planning of the manipulator, we have used a simple inverse kinematics solver to provide a collision free motion from the home position (Θ_s) to the insert pose. For picking up of the part, another inverse kinematics based solver is used which determines if the mobile base needs to reduce its velocity for grasping the part, given the maximum joint rates of the manipulator. The $grasping\_delay$ function returns this delay.

The $grasping\_delay$ function is called when any blue node is expanded. The parent of a blue node is a yellow node which lies inside the grasping area denoted by $\mathcal{A}_g(\phi)$ (orientation of the mobile base at the yellow node is ϕ) as shown in Fig. 6. The orientation associated with the blue node is ψ. The mobile base arrives at the yellow node with orientation ϕ, rotates by an angle $\psi - \phi$ at the yellow node, and then moves towards the blue node with an orientation ψ.

The grasping strategy as described in Sec. IV is implemented over the segment from points 1 to 2. Point 2 is at the edge of $\mathcal{A}_g(\psi)$ along the line segment from the yellow node to the blue node. When the mobile base is at point 1 (yellow node) with orientation ϕ, the manipulator is in the insert pose. And at point 2 it is in the post-grasping (retract)

Algorithm 3 Search algorithm for mobile base motion

```
Require: $S_{start}, T(Goal), \Theta_s, \Theta_g, {}^{\mathcal{W}}T_{\mathcal{P}}, \mathcal{A}_G, \mathcal{O}_s$
    $H = COMPUTE\_HEURISTIC(T, \mathcal{A}_G, \mathcal{O}_s)$
 1: $g(S_{start}) = 0$; $S_{start}.color$ is $RED$; $OPEN = \emptyset$; $CLOSED = \emptyset$
 2: insert $S_{start}$ into $OPEN$ with
    $F(S_{start}) \leftarrow G(S_{start}) + HEURISTIC(S_{start}, T, H)$
 3: while $OPEN \neq \emptyset$ do
 4:     pop $S$ with the smallest $F$ value from $OPEN$
 5:     if $S.color = GREEN$ and $S$ within goal tolerance then
 6:         return PATH
 7:     end if
 8:     if $S.color$ is $BLUE$ then
 9:         $G(S) \leftarrow G(S) + grasping\_delay(S, S.parent, {}^{\mathcal{W}}T_{\mathcal{P}}, \mathcal{A}_G)$
10:     end if
11:     for each successor $S'$ of $S$ do
12:         $G(S') \leftarrow G(S) + T$
13:         if $S.color$ is $RED$ and $S' \notin \mathcal{A}_G$ then
14:             $S'.color \leftarrow S.color = RED$
15:         else if $S.color$ is $RED$ and $S' \in \mathcal{A}_G$ then
16:             $S'.color \leftarrow YELLOW$
17:         else if $S.color$ is $YELLOW$ then
18:             $S'.color \leftarrow BLUE$
19:         else if $S.color$ is $BLUE$ then
20:             $S'.color \leftarrow GREEN$
21:         else
22:             $S'.color \leftarrow GREEN$
23:         end if
24:         $F(S') \leftarrow G(S') + HEURISTIC(S', T, H)$
25:         if a node with the same state and color as $S' \in OPEN$ and has
            lower $F$ value than $S'$ then
26:             continue
27:         else if a node with the same state and color as $S' \in CLOSED$
            and has lower $F$ value than $S'$ then
28:             continue
29:         else
30:             insert $S'$ into $OPEN$
31:         end if
32:     end for
33:     add $S$ to $CLOSED$
34: end while
```

pose. Point 2 is the last point on the edge from yellow to blue nodes, where the post-grasping pose is feasible.

The line segment from 1 to 2 is divided into m points. As the mobile base moves from 1 to 2, the manipulator goes from the insert pose to the grasping pose and then to the post-grasping pose. These motions are determined by interpolating between the IK of the manipulator over the m points. It may be possible that an intermediate IK (at one of the m points) of the manipulator may not be achievable if the maximum joint rate(s) is low and/or the velocity(translational and rotational) of the mobile base is high. In such a case, reduction of the velocity of the mobile base is the only solution for the intermediate configuration to be achievable. This results in a delay in the original trajectory of the mobile base. Such delays along the m intermediate points are added to give the total delay due to slowing down of the mobile base. It should be noted that having the post-grasping pose at the point 2 makes sure that the delay due to the limited joint rates of the manipulator is minimized. Also, since in the grasping strategies discussed before there are no large joint angle differences, we assume that the configurations of the manipulator when interpolating between the IKs are valid.

There may be cases where point 1 lies at the edge of the grasping area $\mathcal{A}_g(\phi)$ and point 2 coincides with it. In such cases, the grasping of the part will happen when the mobile base is rotating instead of translating. The same principles are used to calculate the delay in motion in such cases as well. After the post-grasping pose is reached, the planning to the Θ_g is done again using the inverse kinematic solver. As mentioned before, the total time delay associated with a blue node is added to its g-cost.

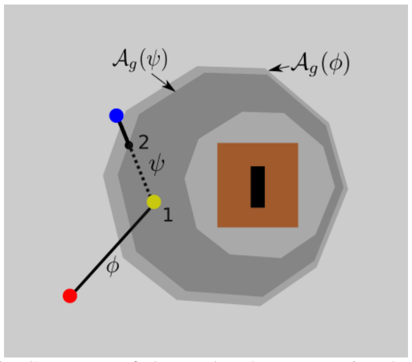

Fig. 6: Segment of the path where grasping happens

VI. Results and Discussion

We ran the planner with 11 motion primitives for the mobile base, with each primitive of a constant time T of 1 second. The maximum turning velocity of the mobile base was taken to be 1 rad/sec and the maximum forward velocity to be 1 m/s. The maximum joint rates of the for the manipulator (UR5) was taken to be 15 degrees/second. Experiments were carried out on varying scene layouts ranging from 10 m $\times$ 10 m to 50 m $\times$ 50 m. The discretization of 20 cm is considered in each case. Several simulation scenarios are generated by varying the percentage free space (i.e., obstacle density) in the scene. The computation time, the number of expansions, and the path cost are recorded for each of the scenarios. To decrease the percentage free area, obstacles are progressively added to an initially random scene. As the size of the scene increases, the obstacle size is also proportionally increased for the same scene. The results presented in Fig. 7 are obtained for 75 trials. The pre-computation of heuristics is included in the computation time of the search. The grasping strategy a (from Fig. 2) is used in each case with the corresponding grasping area. This area is marked in magenta in Fig. 8. For the following experiments, planning was performed off-board on a single core of a standard desktop CPU (Intel Xeon, 3.5 GHz) in MATLAB. The maximum computation time is about 110 seconds. This can be reduced considerably by having a multi-core implementation and parallelization in C++.

For each scene size, it can be observed that the computation time, and the number of expansions are similar in values from high to low percentage free area. However, the computation time is lower for lower percentage of free area. The reason for this is that the heuristic as obtained from the algorithm 1, guides the search through narrow corridors effectively as a result of FMM. This results in lesser node expansions due to lesser control actions feasible at each pose and hence less computation time. It can also be observed that the number of expansions and the computation time increases for a value of percentage free area as the scene size increases.

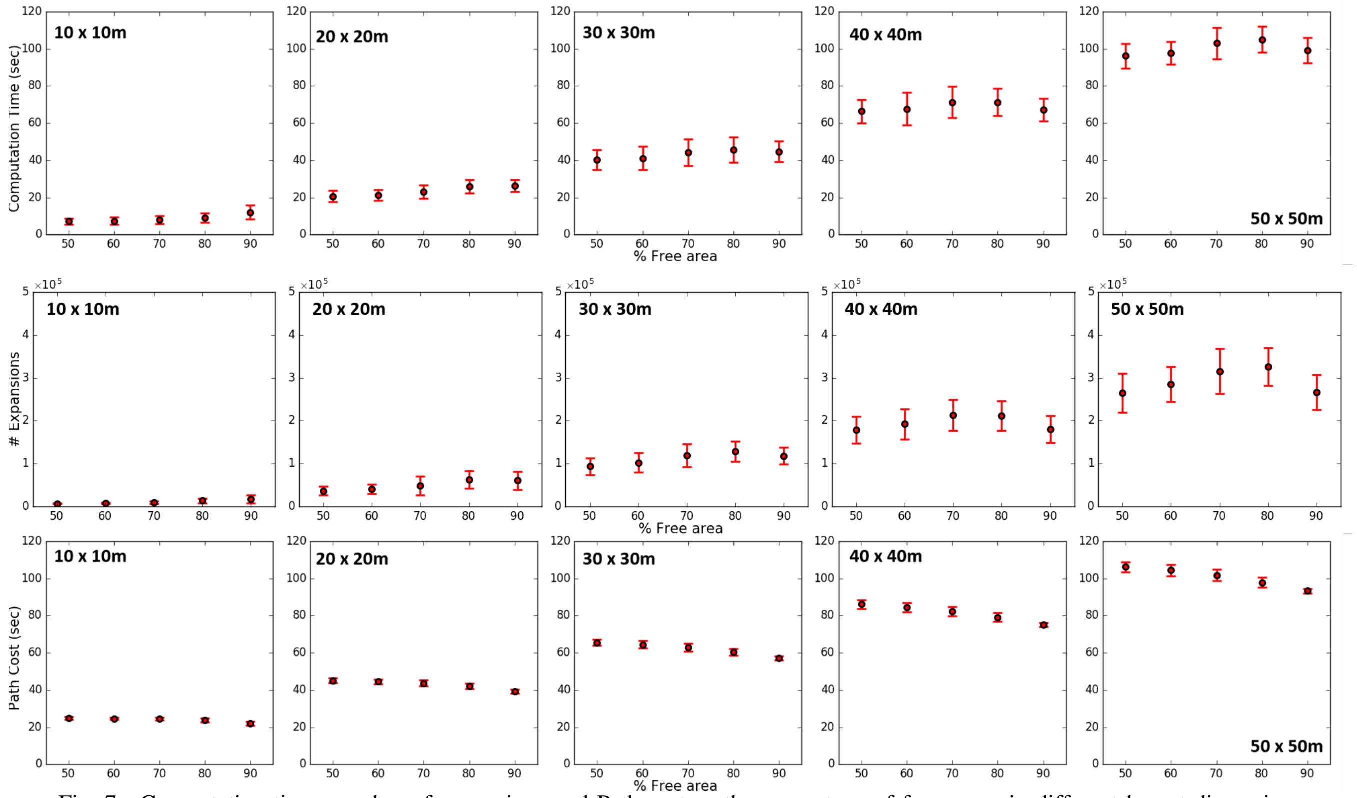

Fig. 7: Computation time, number of expansions and Path cost vs the percentage of free space in different layout dimensions

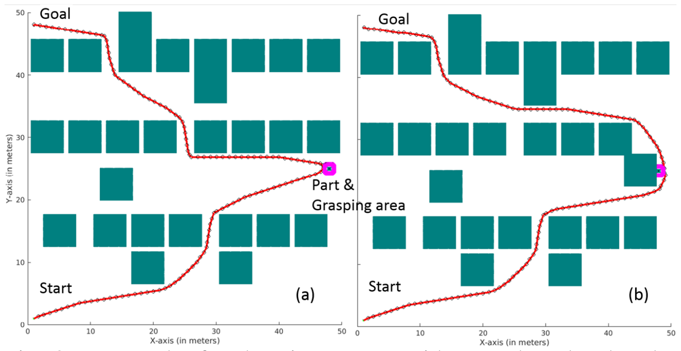

Fig. 8: Two paths for the given scene with one obstacle placed differently and 70% free area

Fig. 8 shows paths for the composite search from a starting location to the goal location via the grasping area for a different placement of a single obstacle. It can be observed that the path changes and passes through the grasping area on the other side. Please see the trajectories generated for the illustrative example in the video at https://youtu.be/B4SumiabVus We have used the robot simulator V-REP for our simulations. For the trajectory in Fig. 8(a), the grasping of the part happens when the mobile base is rotating. Whereas for Fig. 8(b), it happens when the mobile base is translating. Fig. 9 shows the time delay added to the path in Fig. 8(b) as the maximum joint rates of the manipulator increases. This also includes the gripper opening and closing times. For very high joint rates the delay drops to zero. The time delays if the same path is taken and the grasping happens when the mobile base has completely

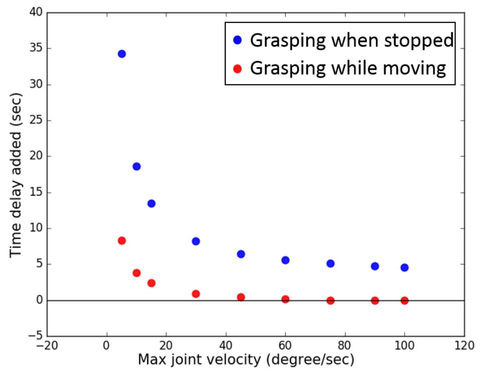

Fig. 9: The time delay for the path in Fig. 8(b) vs the maximum joint rate for the manipulator

stopped at the node inside the grasping area is shown in blue. Even for very high joint rates, there is a considerable time delay in that case.

VII. Conclusions

We have developed a planner which generates time-optimal trajectories for the selected action discretization level. The resulting trajectory results in simultaneous motion of the mobile base and the manipulator. It takes into account the joint constraints for the manipulator. This results in grasping of the part more naturally and without stopping the mobile base motion, giving time-optimal trajectories with respect to the resolution of the search space.

The focus of this paper has been the planning of the mobile base while using the manipulator motion as constraints.

The joint limits of the manipulator result in added costs of the path of the mobile base. The manipulator planner presented in this paper for moving into the grasping pose and implementing grasping is simplistic and purely based on inverse kinematics. Furthermore, there may be a time delay due to the motion of the manipulator from the initial pose to the grasping pose at the grasping node, if the scene near the part is cluttered. Also, it may happen that the joint configurations of the manipulator for some grasp strategies for a part may not be valid when interpolating between IK solutions. Hence, in the future we plan to develop a more sophisticated planner for the manipulator. In order to plan for increased speeds of the mobile base and the manipulator, we must integrate the dynamics of the system into the planner. We can further improve trajectories by doing optimization over continuous action space. This will also result in smoother mobile base paths. Also, currently we are grasping the part with an initially given grasping strategy. It may happen that for a different grasping strategy the overall time for the path is lesser. Hence, in the future we plan to implement co-optimization for the grasping strategy into the planner.

The grasping strategies we have considered inherently reduce the effect of uncertainty in the part pose. However, there may be parts for which no grasping strategy reduces the uncertainty. We plan to analyze this uncertainty for the robustness of grasping.

VIII. Acknowledgements

This work is supported in part by National Science Foundation Grants #1634431 and The Provost Fellowship from the University of Southern California. Opinions expressed are those of the authors and do not necessarily reflect opinions of the sponsors.

References

[1] O. Khatib, "Mobile manipulation: The robotic assistant," *Robotics and Autonomous Systems*, vol. 26, no. 2-3, pp. 175–183, 1999.

[2] M. T. Mason, D. K. Pai, D. Rus, L. R. Taylor, and M. A. Erdmann, "A mobile manipulator," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3. IEEE, 1999, pp. 2322–2327.

[3] J. Alonso-Mora, R. Knepper, R. Siegwart, and D. Rus, "Local motion planning for collaborative multi-robot manipulation of deformable objects," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 5495–5502.

[4] R. A. Knepper, T. Layton, J. Romanishin, and D. Rus, "Ikeabot: An autonomous multi-robot coordinated furniture assembly system," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2013, pp. 855–862.

[5] S. Chitta, B. Cohen, and M. Likhachev, "Planning for autonomous door opening with a mobile manipulator," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2010, pp. 1799–1806.

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

[7] A. Pratkanis, A. E. Leeper, and K. Salisbury, "Replacing the office intern: An autonomous coffee run with a mobile manipulator," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2013, pp. 1248–1253.

[8] R. Holmberg and O. Khatib, "Development and control of a holonomic mobile robot for mobile manipulation tasks," *The International Journal of Robotics Research*, vol. 19, no. 11, pp. 1066–1074, 2000.

[9] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation (ICRA)*, vol. 12, no. 4, pp. 566–580, 1996.

[10] B. Akgun and M. Stilman, "Sampling heuristics for optimal motion planning in high dimensions," in *International Conference on Intelligent Robots and Systems*, 2011, pp. 2640–2645.

[11] S. M. Lavalle, "Planning Algorithms," *Journal of Chemical Information and Modeling*, vol. 53, no. 9, pp. 1689–1699, 2013.

[12] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2. IEEE, 2000, pp. 995–1001.

[13] F. Burget, M. Bennewitz, and W. Burgard, "Bi 2 rrt*: An efficient sampling-based path planning framework for task-constrained mobile manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 3714–3721.

[14] M. Pivtoraiko, R. A. Knepper, and A. Kelly, "Differentially constrained mobile robot motion planning in state lattices," *Journal of Field Robotics*, vol. 26, no. 3, pp. 308–333, 2009.

[15] M. Likhachev and D. Ferguson, "Planning long dynamically feasible maneuvers for autonomous vehicles," *The International Journal of Robotics Research*, vol. 28, no. 8, pp. 933–945, 2009.

[16] P. Švec, B. C. Shah, I. R. Bertaska, J. Alvarez, A. J. Sinisterra, K. v. Ellenrieder, M. Dhanak, and S. K. Gupta, "Dynamics-aware target following for an autonomous surface vehicle operating under COLREGs in civilian traffic," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'13)*, 2013.

[17] B. C. Shah, P. Švec, I. R. Bertaska, A. J. Sinisterra, W. Klinger, K. von Ellenrieder, M. Dhanak, and S. K. Gupta, "Resolution-adaptive risk-aware trajectory planning for surface vehicles operating in congested civilian traffic," *Autonomous Robots*, vol. 40, no. 7, pp. 1139–1163, Oct 2016. [Online]. Available: https://doi.org/10.1007/s10514-015-9529-x

[18] A. M. Kabir, B. C. Shah, and S. K. Gupta, "Trajectory planning for manipulators operating in confined workspaces," in *IEEE International Conference on Automation Science and Engineering (CASE)*, Munich, Germany, Aug 2018.

[19] A. Menon, B. Cohen, and M. Likhachev, "Motion planning for smooth pickup of moving objects," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 453–460.

[20] B. J. Cohen, G. Subramania, S. Chitta, and M. Likhachev, "Planning for manipulation with adaptive motion primitives," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2011, pp. 5478–5485.

[21] K. Gochev, V. Narayanan, B. Cohen, A. Safonova, and M. Likhachev, "Motion planning for robotic manipulators with independent wrist joints," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 461–468.

[22] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2011, p. 4569.

[23] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[24] V. Pilania and K. Gupta, "A hierarchical and adaptive mobile manipulator planner with base pose uncertainty," *Autonomous Robots*, vol. 39, no. 1, pp. 65–85, 2015.

[25] K. Hauser and J.-C. Latombe, "Multi-modal motion planning in non-expansive spaces," *The International Journal of Robotics Research*, vol. 29, no. 7, pp. 897–915, 2010.

[26] K. Hauser and V. Ng-Thow-Hing, "Randomized multi-modal motion planning for a humanoid robot manipulation task," *The International Journal of Robotics Research*, vol. 30, no. 6, pp. 678–698, 2011.

[27] J. L. Jones and T. Lozano-Perez, "Planning two-fingered grasps for pick-and-place operations on polyhedra," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1990, pp. 683–688.

[28] N. B. Kumbla, S. Thakar, K. N. Kaipa, J. Marvel, and S. K. Gupta, "Handling perception uncertainty in simulation-based singulation planning for robotic bin picking," *Journal of Computing and Information Science in Engineering*, vol. 18, no. 2, p. 021004, 2018.

[29] J. A. Sethian, "Fast marching methods," *SIAM review*, vol. 41, no. 2, pp. 199–235, 1999.