Safe and Coordinated Hierarchical Receding Horizon Control for Mobile Manipulators

Jessica Leu, Rachel Lim, and Masayoshi Tomizuka

***Abstract*— Mobile manipulators, constructed by mobile platforms and manipulators, have become a promising solution to future factories for introducing flexibility to manufacturing. This paper presents a method, hierarchical receding horizon control algorithm (HRHC), to assure safety and achieve higher time and space efficiency in robots surrounded by time-varying environments. HRHC contains an optimization based motion planning module that takes account of both the mobile platform and the manipulator to utilize the kinematic redundancy, and a low-level safety controller to deal with fast changes in the environment. With this method, we verify the performance through experiments. The result shows that space efficiency is increased and the HRHC can guarantee local safety in dynamic environments.**

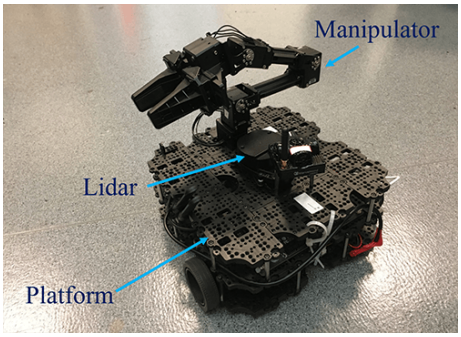


Fig. 1: Mobile manipulator.

I. INTRODUCTION

With the development in technology, robots are playing an important roll in factories nowadays [1]. While most of the industrial robots are either fixed base robotic arms or mobile platforms [2], [3], future factories are craving for robots that have agile manipulation and mobility. Mobile manipulator (MM) (Fig. 1), a kind of robot composed by a mobile platform and one or more than one manipulators, is a promising solution to level up industrial robot's performance and further assist human workers in factories [4], [5]. To achieve this goal, it is crucial to have real-time, safe, and stable motion planning for mobile manipulators in dynamic environments. For example, an autonomous industrial mobile manipulation in industrial human-robot interaction (HRI) system [5], needs to detect changes in the environment and re-plan in real time to bypass multiple human workers and a set of obstacles in order to approach its target efficiently and safely [6], [7].

Mobile manipulators have high degree of freedom and kinematic redundancy which makes it hard to do planning considering the platform and the manipulator at the same time, i.e., coordinated motion planning. However, motion planning in its different varieties has been developed extensively and in great details over the years. In [8], a general overview of motion planning methods is given. In literature, motion planning algorithms often fall into three categories: graph-search based algorithm [9], [10], sampling-based algorithm [8], [11], [12], and optimization-based algorithm [13], [14], [15]. In [9], a searching-base method, ARA* search, which is a variation of A* search, is used to solve manipulation in cluttered spaces by generating consistent, low-cost motion trajectories while providing guarantees on completeness and bounds on suboptimality. In [11], sampling-based methods, adaptive simulated annealing, combined with torque minimization is used to solve the motion planning problem globally. In [16], stochastic optimization method is used to solve motion planning problem for mobile manipulators in static cluttered environments. Although, these motion planning methods show promising results in their testing scenarios, there are still some challenges need to be faced to result in a reactive and safe motion planning method for mobile manipulator.

The first challenge is to deal with time-varying environment. Consider a factory where human workers and mobile robots share the work space, operating in the dynamic environment, i.e., the area where human workers and other robots are moving around, requires robots to re-plan and adapt its motion to the new environment. This is especially difficult for graph-search based algorithm and sampling-based algorithm due to the high dimensionality of the mobile manipulator, and thus, some prior knowledge such as motion primitives and limitations such as the environment needs to be time-invariant are often needed [17], [12]. On the other hand, optimization methods are less sensitive to the dimensionality, and can be solved within a relatively short period of time if formulated properly. Thus, these methods may have more potential in solving these time-varying motion planning problems.

However, optimization methods need to be properly formulated, e.g., formulated into a quadratic programming, in order to have smaller time complexity. The difficulty in problem formulation to describe the feasible set results in over conservative or over simplified problem formulations. This restricts the range of scenarios that optimization methods can solve effectively, i.e., getting a good solution. In [15], the proposed method using constrained sequential

All authors are with the Department of Mechanical Engineering, University of California, Berkeley, CA 94720 USAjess.leu24, rachellim, tomizuka@berkeley.edu

Manuscript 973 submitted to 2020 American Control Conference.
Received September 26, 2019.

linear quadratic optimal control in a receding horizon control framework is claimed to have planning rate up to 100Hz. However, the motion planning controller can only deal with convex optimization problems and relies on reference trajectory generated beforehand. In [14], a motion planning method for collaborative omnidirectional multi-robot manipulation is presented. While the update rate is shown to be in the order of 10Hz, and can deal with non-convex state constraints, the convexify method is relatively simple that makes the motion planning method conservative and a short time horizon assumption is also mentioned. Similar situation is also appearing in [18].

These two concerns, reacting time (sampling time of the planning module) and the range of scenarios that the method can deal with, give rise to the motivation of this work. To solve a non-convex motion planning problem in cluttered scenarios efficiently, the convex feasible set (CFS) algorithm [19] has been proposed to obtain a safe open-loop trajectory. In [20], a parallel planning-and-control architecture is introduced to solve motion planning problem in dynamic environments while assuring safety. Inspired by these methods, this work presents a effective control strategy for mobile manipulator.

The contribution of this work is to present and implement a hierarchical receding horizon control (HRHC) that solves motion planning problem for mobile manipulators in time-varying dynamic environments. A high level motion planning module takes in the environment information and solves the non-convex motion planning problem for the mobile manipulator accordingly. In addition, a low level safety controller running in a higher sampling rate can detect rapid changes in the environment and modify the commands to assure safety locally. Experiments are conducted to verify the performance of the proposed control method.

The remainder of the paper is organized as follows. Section II provides modeling of the mobile manipulator and the non-convex motion planning problem formulation. Section III introduces the hierarchical receding horizon control method. Section IV shows the experimental results (video is publicly available at `jessicaleu24.github.io/ACC2020.html`). Section V concludes the paper.

II. PROBLEM FORMULATION

A. Coordinate assignment

The mobile manipulator used in this work is composed by a 2-DoF mobile platform, TurtleBot3 (by ROBOTIS), and a 4-DoF open source design manipulator. As shown in Fig. 2, the world frame is defined as $F_w - X_wY_wZ_w$. Each link, $link_i$, has an associated body-fixed frame $F_i - X_iY_iZ_i$, $i \in \{1, 2, 3, 4, 5\}$. In addition to the original four joints of the manipulator, we added a virtual link, $link_1$, to the system for the convenience of constructing the motion planning optimization problem.

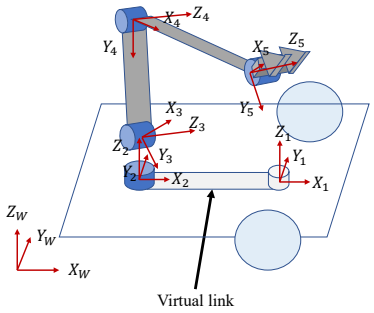


Fig. 2: Coordinate system of the mobile manipulator.

B. Kinematic system modeling

In order to establish the kinematic model of the mobile manipulator, we first examine the nonholonomic mobile platform and the manipulator separately.

1) Model of the mobile platform: Denote the states of the mobile platform as $z_{p,k} = [x_k, y_k, \theta_{p,k}, v_k, \omega_{p,k}]^\top$, where (x_k, y_k) and v_k are the location and speed of the origin of the body-fixed frame $F_1 - X_1Y_1Z_1$ relative to the world frame $F_w - X_wY_wZ_w$, respectively. $\theta_{p,k}$ and $\omega_{p,k}$ are the angle and angular velocity between X_w (the x-axis of the frame $F_w - X_wY_wZ_w$) and X_1 (the x-axis of the frame $F_1 - X_1Y_1Z_1$), respectively. The inputs in the model are the linear and angular acceleration denoted as $u_{p,k} = [a_{p,k}, \alpha_{p,k}]^\top$. k denotes time. The nonlinear kinematic model is:

$$f'_{p,k} = \begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{p,k+1} \\ v_{k+1} \\ \omega_{p,k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ \theta_{p,k} \\ v_k \\ \omega_{p,k} \end{bmatrix} + \begin{bmatrix} v_k T_s \cos(\theta_{p,k} + 0.5 T_s \omega_{p,k}) \\ v_k T_s \sin(\theta_{p,k} + 0.5 T_s \omega_{p,k}) \\ T_s \omega_{p,k} \\ T_s a_{p,k} \\ T_s \alpha_{p,k} \end{bmatrix}, \tag{1}$$

where T_s is the sampling time. By linearizing the model, we get:

$$f_{p,k} = z_{p,k+1} = \mathbf{A}_{p,k} z_{p,k} + \mathbf{B}_{p,k} u_{p,k}, \tag{2}$$

where

$$\mathbf{A}_{p,k} = \left[\frac{\partial f'_{p,k}}{\partial z_{p,k}}\right] \quad and \quad \mathbf{B}_{p,k} = \left[\frac{\partial f'_{p,k}}{\partial u_{p,k}}\right]. \tag{3}$$

2) Model of the manipulator: Denote the states of the manipulator as $z_{m,k} = [\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \omega_1, \omega_2, \omega_3, \omega_4, \omega_5]^\top_k$, where θ_i and ω_i, $i \in \{1, 2, 3, 4, 5\}$, are the position angle and the angular velocity of ith joint, respectively. The inputs are the angular acceleration at each joint, denoted as $u_{m,k} = [\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5]^\top_k$.

The homogeneous transformation matrix of the frame $F_1 - X_1Y_1Z_1$ with respect to the frame $F_w - X_wY_wZ_w$ at a certain time step k can be describe as:

$$^{F_w}_{F_1}\mathbf{T}_k = \begin{bmatrix} ^{F_w}_{F_1}\mathbf{R}_x(\gamma_1) \, ^{F_w}_{F_1}\mathbf{R}_z(\theta_{1,k}) & o_{1,k} \\ 0_{1\times 3} & 1 \end{bmatrix}, \tag{4}$$

Manuscript 973 submitted to 2020 American Control Conference.
Received September 26, 2019.

where,

$$ {}^{F_w}_{F_1}\mathbf{R}_x(\gamma_1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\gamma_1) & -\sin(\gamma_1) \\ 0 & \sin(\gamma_1) & \cos(\gamma_1) \end{bmatrix}, $$

$$ {}^{F_w}_{F_1}\mathbf{R}_z(\theta_{1,k}) = \begin{bmatrix} \cos(\theta_{1,k}) & -\sin(\theta_{1,k}) & 0 \\ \sin(\theta_{1,k}) & \cos(\theta_{1,k}) & 0 \\ 0 & 0 & 1 \end{bmatrix}, $$

$\gamma_1 = 0$, and $o_{1,k} = [x_{1,k}, y_{1,k}, z_{1,k}]^\top$, which is the origin of the frame $F_1 - X_1Y_1Z_1$ with respect to the world frame. The homogeneous transformation matrix of $F_i - X_iY_iZ_i$ with respect to $F_{i-1} - X_{i-1}Y_{i-1}Z_{i-1}$ for $i \in \{2, 3, 4, 5\}$ is:

$$ {}^{F_{i-1}}_{F_i}\mathbf{T}_k = \begin{bmatrix} {}^{F_{i-1}}_{F_i}\mathbf{R}_x(\gamma_i) {}^{F_{i-1}}_{F_i}\mathbf{R}_z(\theta_{i,k}) & o_{i,k} \\ 0_{1\times 3} & 1 \end{bmatrix}. \quad (5) $$

where $[\gamma_2, \gamma_3, \gamma_4, \gamma_5] = [0, -0.5\pi, 0, 0]$, and $o_{i,k}$ is the origin of the coordinate $F_i - X_iY_iZ_i$ with respect to $F_{i-1} - X_{i-1}Y_{i-1}Z_{i-1}$.

The kinematic model of the manipulator is:

$$ f_{m,k} = z_{m,k+1} = \mathbf{A}_{m,k} z_{m,k} + \mathbf{B}_{m,k} u_{m,k}, \quad (6) $$

where,

$$ \mathbf{A}_{m,k} = \begin{bmatrix} \mathbf{I}_{5\times 5} & T_s\mathbf{I}_{5\times 5} \\ \mathbf{0}_{5\times 5} & \mathbf{I}_{5\times 5} \end{bmatrix}, $$

and,

$$ \mathbf{B}_{m,k} = \begin{bmatrix} 0.5T_s^2\mathbf{I}_{5\times 5} \\ T_s\mathbf{I}_{5\times 5} \end{bmatrix}. $$

3) Connection of the platform and the manipulator: The two systems, the mobile platform and the manipulator, are connected with three equality constraints, $\theta_{p,k} = \theta_{1,k}$, $\omega_{p,k} = \omega_{1,k}$, and $\alpha_{p,k} = \alpha_{1,k}$. The full state of the mobile manipulator is denoted as $z_k = [z_{p,k}^\top, z_{m,k}^\top]^\top$, and the input is denoted as $u_k = [u_{p,k}^\top, u_{m,k}^\top]^\top$.

C. Formulation of the motion planning optimization problem

In this work, motion planning is done by solving an optimization problem. According to the model described previously, an optimization problem can be formulated to plan for both the platform and the manipulator, thus, utilizing the kinematic redundancy in the coordinated motion planning. The decision variables for each time step is u_k and the input vector that the problem optimizes over is denoted as $\mathbf{u}_k := [u_k^\top, u_{k+1}^\top, u_{k+2}^\top, \cdots, u_{k+H-1}^\top]^\top$, where$H$ is the prediction horizon. Similarly, the resulting state vector is $\mathbf{z}_k := [z_k^\top, z_{k+1}^\top, z_{k+2}^\top, \cdots, z_{k+H}^\top]^\top$. The current states are recorded in $z(k)$, and $z_k = z(k)$. Denote the kinematic relation of $\mathbf{u}_k$ and $\mathbf{z}_k$ to be $\mathbf{z}_k = f_k(\mathbf{u}_k, z(k))$. In order to obtain the optimal solution $\mathbf{u}_k$, the following optimization needs to be solved.

Problem 1 *(Motion planning problem):*

$$ \min_{\mathbf{u}_k} \quad J(\mathbf{u}_k, z(k)), \quad (7) $$

$$ s.t. \quad f_k(\mathbf{u}_k, z(k)) \in \Gamma_k, \quad (8) $$

$$ g_1(\mathbf{u}_k) = 0, \quad (9) $$

where $g_1(\mathbf{u}_k) = 0$ represents the equality constraints.

There are two assumptions in this formulation:

Assumption *1 (Cost): The cost function is convex and regular, and has the following form:*

$$ J(\mathbf{u}_k, z(k)) = C_1\|\mathbf{D}\mathbf{u}_k - \mathbf{d}\|_2^2 + C_2\|\mathbf{V}\mathbf{u}_k - \mathbf{v}_{ref}\|_2^2 + C_3\|\mathbf{A}\mathbf{u}_k\|_2^2. \quad (10) $$

Here, C_1 is the coefficient of the first term, which penalizes the robot's deviation from a desired path and a desired manipulator pose, so that the robot output trajectory is not too irregular. Matrix $\mathbf{D}$ is a transformation matrix that converts the decision variables, $\mathbf{u}_k$, to the states $\mathbf{z}_k$. Vector b contains the desired states of the mobile manipulator. C_2 is the coefficient of the second term, which penalizes the speed profile of the planned trajectory with regard to a constant speed so that the robot will reach the goal close to a desired timing. $\mathbf{V}\mathbf{u}_k$ is the velocity vector. Here, the speed reference, $\mathbf{v}_{ref}$, is set to be a constant speed. C_3 is the coefficient of the third term, which penalizes the acceleration and angular acceleration of the mobile platform output trajectory, and the joints' angular acceleration so that the motion will be smooth. $\mathbf{A}\mathbf{u}_k$ is the acceleration/angular acceleration vector.

Assumption *2 (Constraint): The state constraint Γ_k is non-convex and its complement is a collection of disjoint convex sets, i.e., each of the obstacle-region is itself convex.*

From the previous problem formulation and assumptions, it is clear that the motion planning problem for mobile manipulators is non-convex. In order to solve this kind of problem fast enough for real-time implementation, the CFS algorithm [19] will be used.

D. Convex Feasible Set Algorithm

An example of a non-convex optimization problem is shown in the following:

$$ \mathbf{x}^* = \underset{\mathbf{x}\in\Gamma\subset\mathbb{R}^n}{\arg\max} \quad J(\mathbf{x}). \quad (11) $$

Here $\mathbf{x}$ is a vector contains n decision variables. J is the cost function which is convex and smooth. Γ is the constrained state space, which is a non-convex subset of $\mathbb{R}^n$. CFS solves the non-convex problem iteratively given the following information:

1) *initialization*: An initial value of the state variable $\mathbf{x}^{(0)}$, which does not necessarily satisfy $\mathbf{x}^{(0)} \in \Gamma$.

2) *Safety index and disjoint convex obstacles* : Assuming there are m disjoint obstacles, safety index, $\phi_j(\mathbf{x})$, is a measure of the robot to the jth disjoint convex-obstacle-region. Note that, $\phi_j(\mathbf{x})$ is a convex function. With the safety index, $\Gamma_j = \{\phi_j(\mathbf{x}) \geq 0\}$, the state constraint, is the complement of the jth obstacle-region in $\mathbb{R}^n$.

3) *Convex feasible set*: With Γ_j, the constrained state space, Γ, is the intersection of m constraints: $\Gamma = \cap_j\Gamma_j$. Given the states from the last iteration, $\mathbf{x}^{(r)}$, the convex feasible set is constructed corresponding to $\mathbf{x}^{(r)}$, $\mathcal{F}(\mathbf{x}^{(r)}) \in \Gamma$.

With CFS, a sub-optimization problem is formulated and solved for the optimal value of $\mathbf{x}^{(r+1)}$:

$$ \mathbf{x}^{(r+1)} = \underset{\mathbf{x}\in\mathcal{F}(\mathbf{x}^{(r)})}{\arg\max} \quad J(\mathbf{x}). \quad (12) $$

Manuscript 973 submitted to 2020 American Control Conference.
Received September 26, 2019.

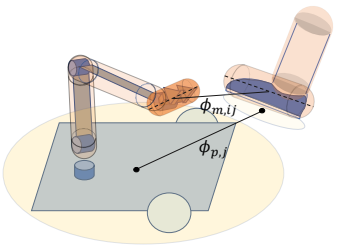


Fig. 3: Distance function $\phi_{m,ij}(z_k)$ and $\phi_{p,j}(z_k)$.

The algorithm solves the problem iteratively and results in a sequence of $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(r)}, \ldots$. It is shown in [19] that this sequence will converge to a local optimal, $\mathbf{x}^*$, in (11).

E. Constraints formulation

In the following section, formulation for both equality and inequality constraints for the optimization will be discussed.

1) Equality constraints: As mention in the previous section, three equality constraints, $\theta_{p,k} = \theta_{1,k}$, $\omega_{p,k} = \omega_{1,k}$, and $\alpha_{p,k} = \alpha_{1,k}$ are used to constrain the motions of the mobile platform and the manipulator.

2) Inequality constraints: According to the CFS algorithm, a safety index function is needed. Here, distance from the robot to the obstacle is chosen as the function to formulate constraints. As shown in Fig. 3, each link of the manipulator, as well as the links of the obstacle, can be captured with a capsule. Distance in between two capsules can be calculated by obtaining the distance between the center lines of the capsules, $\phi_{m,ij}(z_k)$, where i represent the ith manipulator link and j represent the jth obstacle link. The mobile platform is captured by a disk, and so does the area of the obstacle projection (starting from 10 cm away from the floor) to the floor. Therefore the distance, $\phi_{p,j}(z_k)$, can be measured by calculating the distance between the center of the robot to the center of the jth obstacle projection. In this work, two cases are considered, collision avoidance and end-effector position keeping.

In collision avoidance, the convex feasible set corresponding to the jth obstacle-region during time step k at the rth iteration is:

$$\mathcal{F}_j(\mathbf{u}_k^{(r)}) = \left\{ \mathbf{u} : \phi_j(\mathbf{u}_k^{(r)}) + \nabla\phi_j(\mathbf{u}_k^{(r)})(\mathbf{u} - \mathbf{u}_k^{(r)}) \geq M_c \right\}. \tag{13}$$

The inequality constraints not only prevent collision but also require the mobile manipulator to keep a margin, M_c, away from the obstacle.

On the other hand, for end-effector position keeping, obstacle-region is instead named as "lingering-target" and set corresponding to the lingering-target is:

$$\mathcal{F}(\mathbf{u}_k^{(r)}) = \left\{ \mathbf{u} : 0 \leq \phi(\mathbf{u}_k^{(r)}) + \nabla\phi(\mathbf{u}_k^{(r)})(\mathbf{u} - \mathbf{u}_k^{(r)}) \leq M_l \right\}. \tag{14}$$

The inequality constraints keeps the end-effector in the lingering-target within a margin M_l.

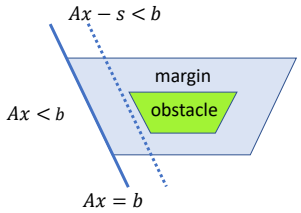


Fig. 4: Illustration of the slack variable.

III. Hierarchical receding horizon control

In the following section, two modifications, adding soft constraints and a low-level safety controller, are introduced to realize closed-loop control for the mobile manipulator.

A. Soft constraints for implementation

To implement the motion planing result, the optimization problem is solved in a receding horizon control (RHC) framework. Although the planned trajectory from RHC is feasible, tracking error will occur in real world experiment, making the states end up to be infeasible, i.e., moving into the margin boundary of the obstacles (Fig. 4). This results in a violent motion because the control command according to the new planning result will immediately pull the robot out of the infeasible area. In order to avoid such problem, we introduce $\mathbf{S}^k = [s_{k+1}, s_{k+2}, \ldots, s_{k+H}]$, the slack variable vector. Introducing slack variables allows the states to violate the original constraint, which is the margin boundary. However these slack variables are also added to the cost function so that the violation is penalized [21]. The new problem is shown in the following:

Problem 2 *(Optimization problem with soft constraints):*

$$\min_{\mathbf{u}_k, \mathbf{S}^k} \quad J(\mathbf{u}_k, z(k)) + \|\mathbf{S}^k\|_2^2, \tag{15}$$

$$s.t. \quad f_k(\mathbf{u}_k, z(k)) \in \Gamma_k(\mathbf{S}^k), \tag{16}$$

$$g_1(\mathbf{u}_k) = 0. \tag{17}$$

B. Low-level safety controller

As mentioned in the introduction section, it is important to allow the motion planning algorithm to be able to cope with large range of situations while maintaining sufficient sampling rate. Here we introduce a low-level safety controller, which runs in a higher sampling rate, in the proposed HRHC.

The overall control system is as shown in Fig. 5. The low-level safety controller takes in the reference command given by the motion planning module every T_s. Environment detection, i.e., obstacle detection and prediction, as well as current states, $z(t)$, is updated every Δt. $T_s = c\Delta t$ where c is an even number. We consider the case where there is only one obstacle and it is moving at constant speed. Prediction of a future collision will be made for the future $1.5T_s$ time step by checking the distances, $\phi_p(t)$, between future mobile manipulator positions and future obstacle positions. At a specific time step, $t = t'$ and $k' \leq t' \leq k' + T_s$, the inputs during t' to $t' + 1.5T_s$ is denoted as $\mathbf{u}_{t'}$ (Fig 6).

Manuscript 973 submitted to 2020 American Control Conference.
Received September 26, 2019.

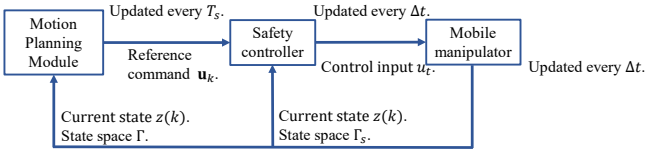


Fig. 5: The overall control system.

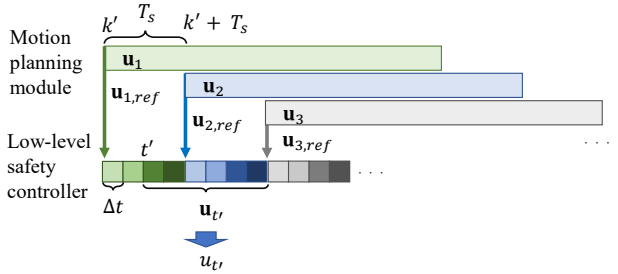


Fig. 6: The hierarchical structure.

While the motion planning module aims for both safety and efficiency, the low-level controller mainly focuses on safety. Therefore, to lower time complexity, the low-level controller simplifies the problem and treats the mobile manipulator as a single capsule that covers the whole robot, thus, the whole system can have a higher sampling rate of the environment. According to the environment information updated every Δt, if the original control command from the motion planning module is not feasible due to the new change in the environment, the low-level safety controller modifies the future trajectory of the mobile platform by solving an optimization problem as stated below:

Problem 3 *(Optimization problem for the safety controller):*

$$\arg\min_{\mathbf{u}_{t'}} \quad \|\mathbf{u}_{t'} - \mathbf{u}_{t',ref}\|_2^2, \tag{18}$$

$$s.t. \quad f_{s,t'}(\mathbf{u}_{t'}) \in \Gamma_{s,t'}, \tag{19}$$

where $\mathbf{u}_{t',ref}$ is the original command, $f_{s,t'}$ is the kinematic function, and $\Gamma_{s,t'} = \{u_i : C_i f_{p,i}(u_i) - d_i \geq M_s \forall i = 0, \ldots, 1.5c-1\}$ is the collision-free set. Here, M_s is a margin and $C_i f_{p,i}(u_i) = d_i$ is the hyperplane tangent to the obstacle that is the closest to the mobile manipulator at time $t'+i$. The optimization problem solves for a series of local command to push the mobile manipulator into the collision-free set. With HRHC, as long as the collision-free set is reachable to the robot hardware, the mobile manipulator can react to dynamic changes in the environment locally at all times, thus, safety is guaranteed.

IV. RESULTS

A. Experimental set up

To verify the performance, the HRHC (with $H = 15$) controller is tested on TurtleBot3 with OpenManipulator (an open source design manipulator), which were developed by ROBOTIS. TurtleBot3 is a ROS standard platform robot. The HRHC controller is running in MATLAB and python on a separate laptop with an 2.8GHz Intel Core i7-7700HQ. An iterative LQR (ILQR) controller [22] is used for better tracking performance.

B. Experimental results

Four experimental setups are selected to verify the performance of the proposed HRHC controller.

In the first scenario, the mobile manipulator is expected to move along a line, $y = 0$, toward the positive x-axis direction, while maintaining a neutral pose and constant speed which also points along the positive x-axis. Initial reference trajectory is a line segment in the state space. In Fig. 10a and Fig. 10b, it is shown that the robot avoids the obstacles successfully. In Fig. 7a, the gray lines are the planned open-loop trajectories at each time steps (colored from light to dark throughout time). It can be seen that although the open-loop trajectory has variations in between each other, the overall trajectory is still smooth. In the second scenario (Fig. 7c), the robot is supposed to move along $y = 0$ while avoiding obstacles both on the ground and hanging from above. Output angles are shown in Fig. 7b and Fig. 7d, respectively.

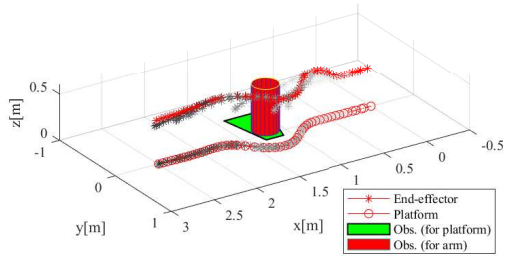
In the third scenario, a mobile manipulator is holding an object in place. However, a human worker, having higher priority, is pushing a chair and is heading towards a direction where the platform part of the robot is blocking the way. Here, the mobile manipulator is expected to maintain its end-effector to stay close to a certain location. In Fig. 10c, it is shown that the robot can keep the end-effector within a small range. Fig. 8 shows the performance of end-effector position keeping.

In the fourth scenario, the set up is similar to the first scenario while the obstacle is changed to a human worker standing on a ladder. At the beginning, the robot is only able to see the ladder and plans to avoid it. When the worker comes down to the floor and walks away, it appears to the robot that there is another obstacle appearing close to itself suddenly. The low-level safety controller will react to the change while bypassing the worker. In Fig. 10d, it is shown that the robot performed successfully and can deal with dynamic environment. The performance of the safety controller can be seen from Fig. 9b. Around time step $k = 20$, it is shown that the platform angles changes ($k = 21$) before the manipulator starts to react ($k = 23$) to the new obstacle. This is because the safety controller has kicked in ($k = 21$) to make the platform to avoid the worker. And after the planning model solved the new plan ($k = 23$), the manipulator will also retract to avoid the worker. With the safety controller, the overall system is sampling and reacting at 10Hz.

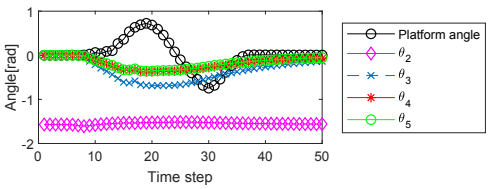
V. CONCLUSION

This paper presented a motion planning method, HRHC (hierarchical receding horizon control), for mobile manipulators to handle time-varying scenarios in industrial HRI

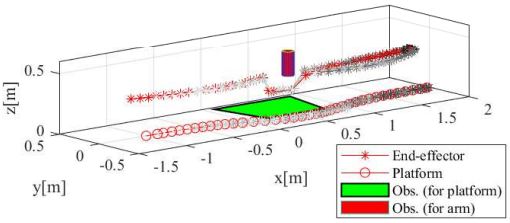
Manuscript 973 submitted to 2020 American Control Conference.
Received September 26, 2019.



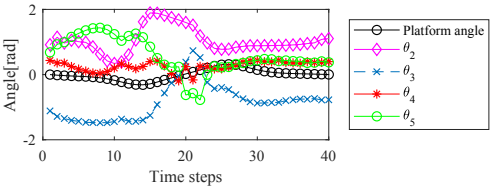
(a) Open-loop plans and the closed-loop result (scenario I).



(b) Output angles of the platform and each joint (scenario I).

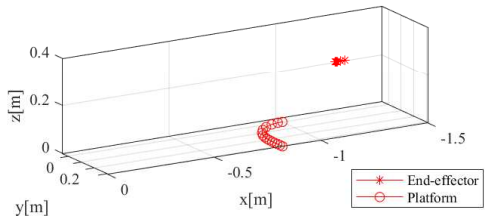


(c) Open-loop plans and the closed-loop result (scenario II).

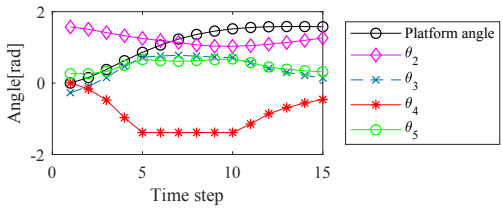


(d) Output angles of the platform and each joint (scenario II).

Fig. 7: Results of collision avoidance in two scenarios.

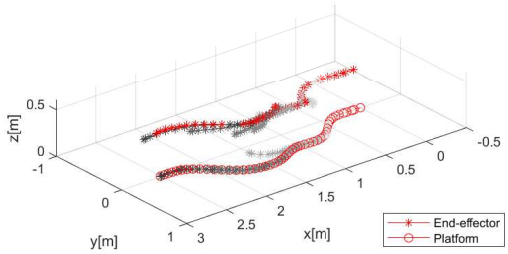


(a) Closed-loop result.

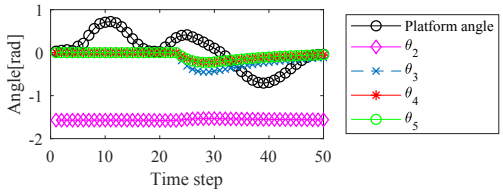


(b) Output angles of the platform and each joint.

Fig. 8: Result of end-effector position keeping.



(a) Open-loop plans and the closed-loop result.



(b) Output angles of the platform and each joint.

Fig. 9: Result of avoiding moving human worker.

systems. To better utilize the shared-space and achieve higher efficiency, a high level motion planning module considered the environment information and the mobile manipulator kinematic redundancy. Combined with a low-level safety controller that modified commands locally, HRHC was developed to perform coordinated motion planning and guarantee safety maneuvers for mobile manipulators in dynamic environments. To evaluate HRHC, experiments were conducted and the results showed that the HRHC enabled the mobile manipulator to perform collision avoidance while completing its tasks successfully in both static and dynamic environments.

ACKNOWLEDGEMENT

The authors thank Weiye Zhao, Liting Sun, and Xili Yi for the help in editing the paper and in conducting the experiments. This work was supported by the National Science Foundation under Grant No.1734109. Any opinion, finding, and conclusion expressed in this paper are those of the authors and do not necessarily reflect those of the National Science Foundation.

REFERENCES

[1] J. N. Pires, *Industrial robots programming: building applications for the factories of the future*. Springer Science & Business Media, 2007.

[2] J. J. Craig, *Introduction to robotics: mechanics and control*. Pearson/Prentice Hall Upper Saddle River, NJ, USA:, 2005, vol. 3.

[3] M. Hvilshøj, S. Bøgh, O. Madsen, and M. Kristiansen, "Calibration techniques for industrial mobile manipulators: Theoretical configurations and best practices," in *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)*. VDE, 2010, pp. 1–7.

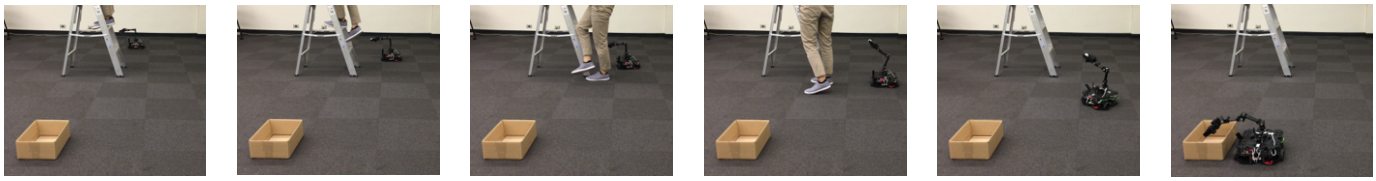
[4] M. Hvilshøj and S. Bøgh, ""little helper"—an autonomous industrial mobile manipulator concept," *International Journal of Advanced Robotic Systems*, vol. 8, no. 2, p. 15, 2011.

[5] M. Hvilshøj, S. Bøgh, O. Skov Nielsen, and O. Madsen, "Autonomous industrial mobile manipulation (aimm): past, present and future," *Industrial Robot: An International Journal*, vol. 39, no. 2, pp. 120–135, 2012.

(a) Experimental result with mobile manipulator performing collision avoidance with an obstacle on the floor in the front.

(b) Experimental result with mobile manipulator performing collision avoidance with one obstacle on the floor and one hanging from above.

(c) Experimental result with mobile manipulator performing collision avoidance while keeping the end-effector position in the Cartesian space.



(d) Experimental results with mobile manipulator avoiding a moving human worker.

Fig. 10: Experimental result of mobile manipulator avoiding obstacle under different conditions.

[6] T. Chettibi, H. Lehtihet, M. Haddad, and S. Hanchi, "Minimum cost trajectory planning for industrial robots," *European Journal of Mechanics-A/Solids*, vol. 23, no. 4, pp. 703–715, 2004.

[7] M. R. Pedersen, L. Nalpantidis, R. S. Andersen, C. Schou, S. Bøgh, V. Krüger, and O. Madsen, "Robot skills for manufacturing: From concept to industrial deployment," *Robotics and Computer-Integrated Manufacturing*, vol. 37, pp. 282–291, 2016.

[8] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[9] B. J. Cohen, S. Chitta, and M. Likhachev, "Search-based planning for manipulation with motion primitives," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 2902–2908.

[10] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling, "Backward-forward search for manipulation planning," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 6366–6373.

[11] D. P. Garg and M. Kumar, "Optimization techniques applied to multiple manipulators for path planning and torque minimization," *Engineering applications of artificial intelligence*, vol. 15, no. 3-4, pp. 241–252, 2002.

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

[13] Q. Huang, K. Tanie, and S. Sugano, "Coordinated motion planning for a mobile manipulator considering stability and manipulation," *The International Journal of Robotics Research*, vol. 19, no. 8, pp. 732–742, 2000.

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

[15] M. Giftthaler, F. Farshidian, T. Sandy, L. Stadelmann, and J. Buchli, "Efficient kinematic planning for mobile manipulators with non-holonomic constraints using optimal control," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3411–3417.

[16] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 4569–4574.

[17] B. Cohen, S. Chitta, and M. Likhachev, "Single-and dual-arm motion planning with heuristic search," *The International Journal of Robotics Research*, vol. 33, no. 2, pp. 305–320, 2014.

[18] G. B. Avanzini, A. M. Zanchettin, and P. Rocco, "Constraint-based model predictive control for holonomic mobile manipulators," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 1473–1479.

[19] C. Liu, C.-Y. Lin, and M. Tomizuka, "The convex feasible set algorithm for real time optimization in motion planning," *SIAM Journal on Control and Optimization*, vol. 56, no. 4, pp. 2712–2733, 2018.

[20] C. Liu, T. Tang, H.-C. Lin, Y. Cheng, and M. Tomizuka, "Serocs: Safe and efficient robot collaborative systems for next generation intelligent industrial co-robots," *arXiv preprint arXiv:1809.08215*, 2018.

[21] J. Chen, C. Liu, and M. Tomizuka, "Foad: Fast optimization-based autonomous driving motion planner," in *2018 Annual American Control Conference (ACC)*. IEEE, 2018, pp. 4725–4732.

[22] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 4906–4913.

Manuscript 973 submitted to 2020 American Control Conference.
Received September 26, 2019.