Ph.D. DISSERTATION

Task-oriented whole-body motion generation strategy of mobile manipulator for considering kinematic and dynamic constraints

기구학적 및 동적 제한조건들을 고려한
모바일 매니퓰레이터의 작업 중심 전신 동작 생성 전략

BY

KEUNWOO JANG

FEBRUARY 2023

DEPARTMENT OF TRANSDISCIPLINARY STUDIES
THE GRADUATE SCHOOL OF CONVERGENCE
SCIENCE AND TECHNOLOGY
SEOUL NATIONAL UNIVERSITY

Ph.D. DISSERTATION

Task-oriented whole-body motion generation strategy of mobile manipulator for considering kinematic and dynamic constraints

기구학적 및 동적 제한조건들을 고려한
모바일 매니퓰레이터의 작업 중심 전신 동작 생성 전략

BY

KEUNWOO JANG

FEBRUARY 2023

DEPARTMENT OF TRANSDISCIPLINARY STUDIES
THE GRADUATE SCHOOL OF CONVERGENCE
SCIENCE AND TECHNOLOGY
SEOUL NATIONAL UNIVERSITY

Task-oriented whole-body motion generation strategy of mobile manipulator for considering kinematic and dynamic constraints

기구학적 및 동적 제한조건들을 고려한
모바일 매니퓰레이터의 작업 중심 전신 동작 생성 전략

지도교수 박 재 흥

이 논문을 공학박사 학위논문으로 제출함

2023년 1월

서울대학교 대학원

융합과학부

장 근 우

장근우의 공학박사 학위 논문을 인준함

2022년 12월

Abstract

A mobile manipulator is a manipulator mounted on a mobile robot. Compared to a fixed-base manipulator, the mobile manipulator can perform various and complex tasks because the mobility is offered by the mobile robot. However, combining two different systems causes several features to be considered when generating the whole-body motion of the mobile manipulator. The features include redundancy, inertia difference, and non-holonomic constraint. The purpose of this thesis is to propose the whole-body motion generation strategy of the mobile manipulator for considering kinematic and dynamic constraints.

First, a planning framework is proposed that computes a path for the whole-body configuration of the mobile manipulator to navigate from the initial position, traverse through the door, and arrive at the target position. The framework handles the kinematic constraint imposed by the closed-chain between the robot and door. The proposed framework obtains the path of the whole-body configuration in two steps. First, the path for the pose of the mobile robot and the path for the door angle are computed by using the graph search algorithm. In graph search, an integer variable called *area indicator* is introduced as an element of state, which indicates where the robot is located relative to the door. Especially, the area indicator expresses a process of door traversal. In the second step, the configuration of the manipulator is computed by the inverse kinematics (IK) solver from the path of the mobile robot and door angle. The proposed framework has a distinct advantage over the existing methods that manually determine several parameters such as which direction to approach the door and the angle of the door required for passage. The effectiveness of the proposed framework was validated through experiments with a nonholonomic mobile manipulator.

Second, a whole-body controller is presented based on the optimization method that can consider both equality and inequality constraints. The method computes the optimal solution of the weighted hierarchical optimization problem. The method is developed to resolve the redundancy of robots with a large number of Degrees of Freedom (DOFs), such as a mobile manipulator or a humanoid, so that they can execute multiple tasks with differently weighted joint motion for each task priority. The proposed method incorporates the weighting matrix into the first-order optimality condition of the optimization problem and leverages an active-set method to handle equality and inequality constraints. In addition, it is computationally efficient because the solution is calculated in a weighted joint space with symmetric null-space projection matrices for propagating recursively to a low priority task. Consequently, robots that utilize the proposed controller effectively show whole-body motions handling prioritized tasks with differently weighted joint spaces. The effectiveness of the proposed controller was validated through experiments with a nonholonomic mobile manipulator as well as a humanoid.

Lastly, as one of dynamic constraints for the mobile manipulator, a reactive self-collision avoidance algorithm is developed. The proposed method mainly focuses on self-collision between a manipulator and the mobile robot. We introduce the concept of a *distance buffer border* (DBB), which is a 3D curved surface enclosing a buffer region of the mobile robot. The region has the thickness equal to buffer distance. When the distance between the manipulator and mobile robot is less than the buffer distance, i.e. the manipulator lies inside the buffer region of the mobile robot, the proposed strategy is to move the mobile robot away from the manipulator in order for the manipulator to be placed outside the border of the region, the DBB. The strategy is achieved by exerting force on the mobile robot. Therefore, the manipulator can avoid self-collision with the mobile robot without modifying the predefined motion of the manipulator in a world Cartesian coordinate frame. In particular, the direction of the force is determined by considering the non-holonomic constraint of the differentially driven mobile robot. Additionally, the reachability of the manipulator is considered to arrive at a configuration in which the manipulator can be more maneuverable. To realize the desired force and resulting torque, an avoidance task is constructed by converting them into the accelerations of the mobile robot and smoothly inserted with a top priority into the controller. The proposed algorithm was implemented on a differentially driven mobile robot with a 7-DOFs robotic arm and its performance was demonstrated in various experimental scenarios.

keywords: Mobile Manipulator, Whole-body Control, Motion Planning

student number: 2016-26043

Contents

List of Tables

List of Figures

Chapter 1

INTRODUCTION

1.1 Motivation

A mobile manipulator, which is a manipulator mounted on a mobile robot, has infinite workspace offered by the mobile robot. Also, the degrees of freedom (DOFs) of the mobile robot usually provide the mobile manipulator with redundancy with respect to the task such as the trajectory tracking of the end-effector. By utilizing these properties, the mobile manipulator can perform complex and various tasks such as painting [1], polishing [2], and door opening [3]. To perform these tasks, the motion planning and control considering the whole-body of the mobile manipulator are necessary.

Among various tasks that require the whole-body of the mobile manipulator, door traversal still remains a challenging task because it requires tight coordinated motion of the whole-body and has to satisfy the constraint that the end-effector grasps the door handle. To plan the motion of robot, several approaches [4–8] have been developed. However, these approaches need to plan additional motion to traverse through the door because the goal of them is to open the door or articulated object to a predefined angle.

On the other hand, the whole-body control frameworks [9, 10] have been developed to perform various prioritized tasks to consider motion distribution with kinematic redundancy. However, these frameworks are difficult to generate natural whole-body behavior because they produce joint motion minimizing the same metric for all prioritized tasks. Recently, even though the controller [11] was developed to generate different motion patterns for each task hierarchy, it cannot calculate the solution for 3 or more tasks and cannot deal with inequality constraints.

Therefore, we propose the task-oriented whole-body motion generation strategy of the mobile manipulator for considering kinematic and dynamic constraints.

1.2 Contributions of thesis

The contributions of the thesis consist of three parts. First, we propose a framework that belongs to the motion planning approach and addresses the problem for the mobile manipulator to navigate from the start position to the goal position, including passing through the door. The framework is composed of two steps which plan separately for the mobile robot and manipulator. The framework first computes the path of the mobile robot and door angle by using the graph search algorithm. Based on the obtained paths of the first step, the framework finds a collision-free path of the joint position of the manipulator by utilizing the IK solver.

Second, the whole-body controller based on hierarchical quadratic programming (HQP) is proposed. The proposed controller can efficiently compute an analytic solution in a weighted least-squares norm manner for prioritized equality and inequality tasks. By reformulating HQP with complete orthogonal decomposition (COD) [12], the proposed controller assigns the weighting matrix to each task hierarchy and derives the first-order optimality conditions. Based on these conditions, the active-set method [12,13] is exploited to handle inequality tasks.

Third, as one of the constraints to be considered for the mobile manipulator, a new self-collision avoidance algorithm is developed. Our focus is on the avoidance of self-collision between the manipulator and the mobile robot. We propose a concept of *distance buffer border* (DBB), a border of the buffer region that the manipulator can reach around the mobile robot. The region has the thickness of the buffer distance. The strategy is to position the manipulator outside the DBB by the motion of the mobile robot. This is realized by generating the force exerted on the mobile robot because the DBB is attached to the mobile robot and moves with it. Therefore, the manipulator can avoid self-collision with the mobile robot without modifying reference motion of the manipulator.

1.3 Overview of thesis

Chapter 2 presents the whole-boy planning framework that addresses the navigation problem including door traversal. Next, Chapter 3 describes the whole-body controller that prioritizes equality and inequality tasks while assigning individual joint weights for each task priority. Chapter 4 shows the self-collision avoidance algorithm that handles the collision between the manipulator and mobile robot. Finally, the paper is concluded in Chapter. 5.

Chapter 2

WHOLE-BODY MOTION PLANNER : APPLICATION TO NAVIGATION INCLUDING DOOR TRAVERSAL

Mobile manipulators can manipulate objects in the extended workspace offered by the mobile robot. In this respect, mobile manipulators can help people by performing various tasks such as delivery, household chores, etc. Since the chances of successfully executing such tasks depend on whether the target object is reachable or not, mobile manipulators need the ability to open and traverse doors which connect spaces. However, the problem of generating the motion of mobile manipulator is not simple because the mobile manipulator is a coupled system and the door is an articulated object.

In this chapter, we propose a framework that belongs to the motion planning approach and addresses the problem for the mobile manipulator to navigate from the start position to the goal position, including passing through the door. The search space of the problem is high dimensional. Also, the problem has to

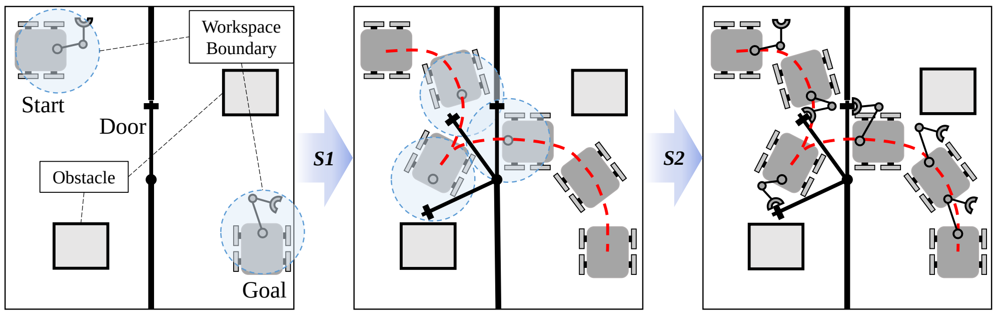


Figure 2.1: Overview of the proposed framework. Given the information of the environment, door, and start and goal positions, ***S1*** is the step that computes the pose path of the mobile robot and angle path of the door. The paths are computed with the constraint that the position of the door handle has to be located inside the workspace of the manipulator as shown in blue dotted circles. Red dotted line shows the computed path of the mobile robot. Then, ***S2*** is the step that computes the path of the joint configuration of the manipulator by using the inverse kinematics solver. Combining the computed paths, the motion of the mobile manipulator is generated that the mobile manipulator approaches, opens, traverses through, closes the door, and eventually arrives at the goal position.

satisfy the constraint that the end-effector of the mobile manipulator grasps the door handle. In these respects, the problem demands expensive computational cost. To alleviate the cost, the framework is composed of two steps which plan separately for the mobile robot and the manipulator, as shown in Fig. 2.1. The framework first computes the path of the mobile robot and door angle by using the graph search algorithm. Based on the obtained paths of the first step, the framework finds a collision-free path of the joint position of the manipulator by utilizing the IK solver. By merging these paths, the whole-body motion of the mobile manipulator is generated such that the robot navigates to the goal position after passing through the door.

The main contribution of the framework is the graph representation. The graph is designed to formulate the problem of planning the motion of the robot approaching, opening, traversing through, closing the door, and finally reaching the target position. Thus, the proposed framework can plan the path for all sub-problems by a single search, whereas the previous works [4–6, 14] can only plan the path for one sub-problem or a part of the sub-problems and thus require the additional path planning for the other sub-problems. Especially, an integer variable called *area indicator* is introduced as an element of the state, which represents where the robot is located relative to the door. Each value of the area indicator represents the process of the sub-tasks. Also, the area indicator allows to calculate the range of the door angle that the manipulator can reach, which eliminates the need to include the door angle directly in the state space.

The remainder of the chapter is organized as follows. Sec. 2.1 presents the background and related works for door traversal. In Sec. 2.2, the proposed framework consisting of two steps is explained. Sec. 2.3 describes the experimental validations and discussion of the proposed framework. Finally, the paper is concluded in Sec. 2.4.

2.1 Background & related works

Researches for door traversal have been actively conducted in recent decades. They can be categorized into two main approaches [15]: *sense-and-act* [16–19] and *plan-and-act* [4–8, 14]. The former estimates geometric information of the door and simultaneously generates the motion that opens the door on the basis of sensory feedback information in real-time. The latter plans the motion that opens and traverses through the door given prior information of the door. First, in the field of *sense-and-act* approach, Lee *et al.* [16] developed a control system combining impedance control and teleoperation in order for a humanoid to open and traverse through the door. Karayiannidis *et al.* [17] developed control strategy that estimates the position of joint axis from the measured force and simultaneously opens articulated objects such as door and drawer. However, since the strategy was only applied to the fixed-base manipulator, the workspace of the manipulator should be considered before opening such objects. Stuede *et al* [18] proposed a unified approach including impedance controller of mobile manipulator and robust door handle detection based on convolutional neural network. Although the method [18] was successfully applied to the push-type door, the method had limitations to opening the pull-type door.

On the other hand, the motion planning approaches have been developed. Chitta *et al.* [14] developed a planner that efficiently generates the whole-body motion of the mobile manipulator for door opening by combining the graph search algorithm and IK solver. The planner can handle both push and pull-type door. Furthermore, Gray *et al.* [4] extended the previous work to enable the mobile manipulator to open the spring-loaded door by using the contact for holding the door. Arduengo *et al.* [5] proposed a unified framework that includes detecting the door and handle, estimating the model of the door, and planning the motion for door opening. The framework utilizes a concept of task space region which represents the end-effector constraint. Jiao *et al.* [6] proposed a planner that computes the whole-body motion of the mobile manipulator to manipulate articulated objects by utilizing a virtual kinematic chain connecting the robot and object. However, these methods [4–6, 14] need to plan additional motion to traverse through the door because the goal of them is to open the door or articulated object to a predefined angle. More generally, Jorgensen *et al.* [7] proposed a method that plans a loco-manipulation motion of the humanoid considering the reachability of the arm and footstep given a predefined end-effector path. Also, Murooka *et al.* [8] developed a loco-manipulation planner that first computes the 2D pose path of the object and then obtains a sequence of footsteps and re-grasping poses. Based on the computed sequence, the whole-body configuration of the humanoid is computed by using the quadratic programming-based IK solver. Although these approaches [7, 8] can generate the whole-body loco-manipulation motion of the humanoid, they only cover the part of door operation that opens the door to a certain angle.

However, the *plan-and-act* approach is suited for well-defined and structured environment. To deal with uncertain and dynamic environment, many researchers developed methods belonging to the *sequential sense-plan-and* approach [15] which exploits the environmental feedback information and generates the modified motion in real-time. Lee *et al.* [20] proposed model predictive control (MPC) framework that generates the whole-body motion of aerial manipulator. The framework considers the aerial manipulator and the door as a whole articulated body and incorporates several constraints including self-collision and collision between the robot and door. Recently, Ito *et al.* [19] proposed a learning-based method that predicts the behavior of the robot from the sensory data and generates the motion to open and pass through the door in real-time.

2.2 Proposed framework

This section describes the proposed framework which computes the collision-free path of the mobile manipulator passing through the door and thereafter reaching the goal position. Rather than searching for the entire joint space of the mobile manipulator, the proposed framework is designed to sequentially execute two steps denoted as ***S1*** and ***S2*** which generate the motion for the mobile robot and manipulator, respectively. The first step ***S1*** generates the collision-free motion of the robot and door angle, assuming that the manipulator maintains its home configuration except during door traversal. Next, the second step ***S2*** computes the synchronized motion of the manipulator for the generated motion of the mobile robot and door in ***S1***. In ***S2***, the motion of the manipulator is computed by using the IK solver while considering the collision between the manipulator and environment including the door and wall. The schematic description of two steps is shown in Fig. 2.1.

2.2.1 Computing path for mobile robot and door angle - S1

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

2.2.1.1 State

The search space $\mathcal{S} \in \mathbb{R}^3 \times \mathbb{I}$ is four dimensional space. The state is expressed as $s = (x, y, \theta, a) \in \mathcal{S}$. (x, y, θ) is the pose of the mobile robot and a is called *area indicator*. The area indicator a expresses which area the current pose of the

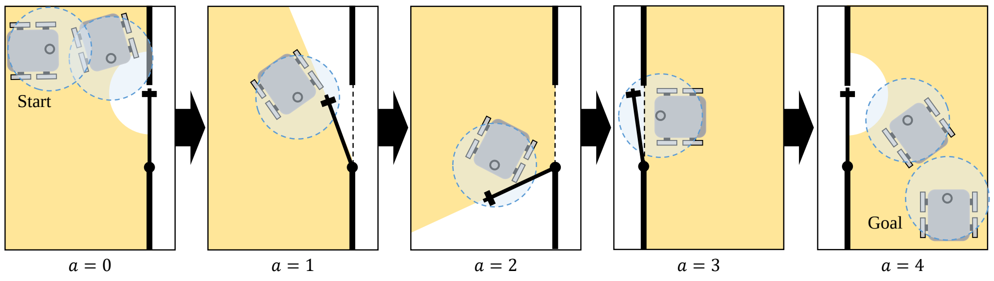


Figure 2.2: Illustration of the area indicator. The value of the area indicator is decided by where the robot is located with respect to the door. The yellow area shows the area corresponding to each value of the area indicator. As the indicator changes from 0 to 4, the robot moves towards the target position after passing through the door.

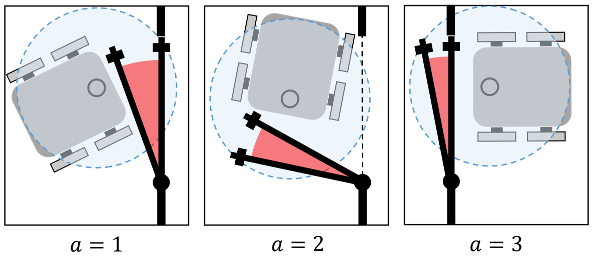


Figure 2.3: The area indicator a from 1 to 3 implicitly shows the range of the reachable door angles. Given the pose of the mobile base, the range shown in red can be calculated by checking whether the door handle is within the workspace boundary of the manipulator and the collision between the door and mobile robot.

mobile robot belongs to. The value of the area indicator is determined according to the pose of the mobile robot and the angle of the door, as shown in Fig. 2.2.

In Fig. 2.2, the value of the area indicator is defined as 0 when the robot is far from the door. At this position, the door handle is outside the workspace of the manipulator. This implies the process of the robot approaching the door. Next, when the robot is located outside the line of the door and can reach the door handle, a is 1. When the robot is located inside the line of the door but does not yet cross over the doorsill, a is 2. As the robot moves to open and pass through the door, the indicator changes from 1 to 2. At this point, two states can be generated that have the same pose of the mobile robot, but have two different indicators of 1 and 2. Then, when the robot is located beyond the doorsill but still can reach the door handle, a is 3. This indicates the process of the robot closing the door. Finally, when the robot is placed between the door and goal position, the value of a is defined as 4. This refers to the process of the robot reaching the goal position.

In fact, the indicator from 1 to 3 is an alternative to directly including the door angle as an element of the state. If the pose of the mobile robot is given, all possible angles of the door are checked and labeled as from 1 to 3. Then, the angles belonging to the same indicator are grouped into the range, as shown in Fig. 2.3. From a practical point of view, the range is computed by checking a finite set of the door angles obtained from discretizing the possible range of the door angle at a certain interval, e.g., 1 degree or 2 degrees.

The concept of the area indicator provides several advantages. First, it can handle both “pull” and “push”-type door. As the indicator changes from 1 to 3, it shows the process of the robot opening, passing through, and closing the “pull”-type door. On the contrary, as the indicator changes from 3 to 1, it expresses the task for “push”-type door. Second, all sub-tasks related to the door can be described. They include approaching the door, opening the door, traversing through the door, closing the door, and finally moving toward the goal position. Third, the state can be concisely expressed by including the area indicator without directly including the door angle as an element of the state.

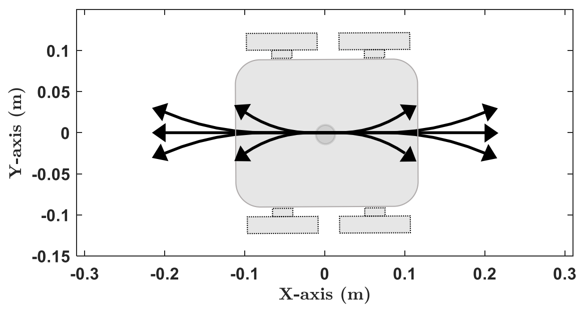


Figure 2.4: Example of a set of feasible actions for the given pose of the robot.

2.2.1.2 Action

To transit any state to its successor state, the action should be defined. Since the area indicator of the state is a variable automatically determined by the pose of the mobile robot, the action is defined only for the pose of the mobile robot. Thus, the concept of the *lattice* [21] is adopted in order to discretize the state space while considering non-holonomic constraints for the mobile robot, such as car-like or differentially-driven mobile robot. After defining a set of the feasible actions, as shown in Fig. 2.4, the lattice graph composed of the discretized states is constructed by executing the actions to the states.

Every transition between the two states is feasible when their area indicators are equal to 0 or 4 because the action of the robot is not related to the door. However, when the area indicator is 1 to 3, the reachable door angles of a state and its successor state should overlap, as shown in Fig. 2.5(a). This is because the door angle should not change discontinuously in the process of the robot opening and closing the door. From a practical point of view, since the reachable angle range is calculated as a set of discretized angles not as the inequality bounds of the angle, at least one discrete value of the door angle

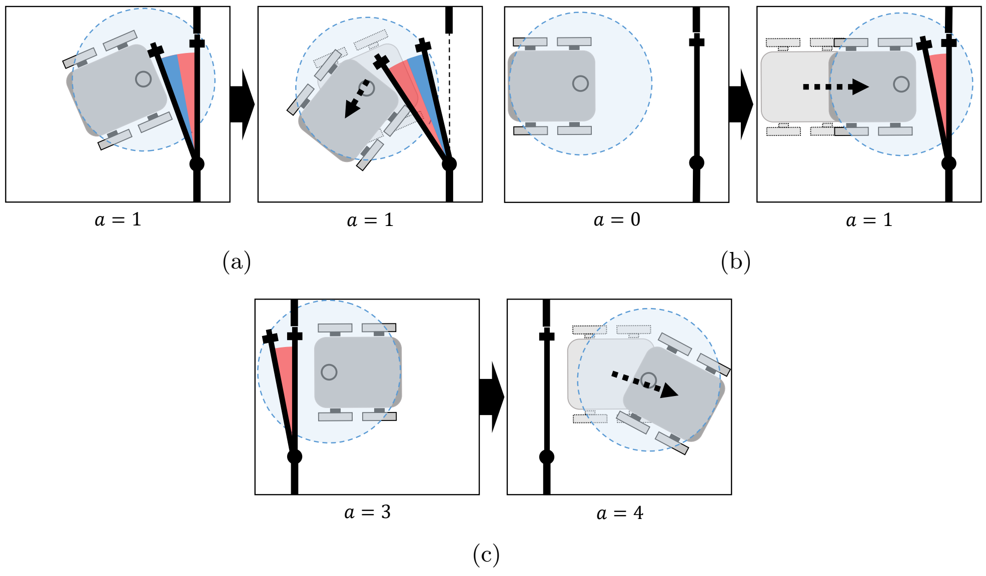


Figure 2.5: Illustration of the state transitions depending on the value of the area indicator: (a) In the process of the robot opening the door, the ranges of the reachable angles for two consecutive states should overlap as shown in blue; (b) When the robot approaches the door to open it, i.e., for the area indicator of the successor state to be 1, the range of the door angle should contain 0 degrees; (c) When the robot closes the door and moves towards the goal position, i.e., for the area indicator of the successor state to become 4, the range of the door angle for the predecessor state should contain 0 degrees.

should overlap to become a valid transition.

Furthermore, when the robot starts to open the door, i.e. the area indicator changes from 0 to 1, the successor state should contain 0 degrees in its set of the reachable door angles, as shown in Fig. 2.5(b). Likewise, when the robot finishes to close the door and starts to move towards the goal position, i.e. the area indicator changes from 3 to 4, the predecessor state whose area indicator is 3 should contain 0 degrees in its set of the feasible door angles, as shown in Fig. 2.5(c).

2.2.1.3 Cost

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

$$c(s, s_{succ}) = \begin{cases} c_{action} + c_{map} + c_{door} & \text{if } a = 1, 2, 3 \\ c_{action} + c_{map} & \text{otherwise} \end{cases} \tag{2.1}$$

where c_{action} is the cost to transit to the successor state by executing the predefined action of the mobile robot. This term is proportional to the weighted sum of the translational and rotational displacement [22]. Second, c_{map} is the cost based on the costmap which indicates how close the robot is to the obstacle [14]. Lastly, c_{door} is the cost to represent how properly the door handle is positioned from the base of the manipulator at the given pose of the mobile robot when the area indicator of the current state includes from 1 to 3. c_{door} is designed by utilizing the information of the reachability [23] which represents the quality of the pose of the end-effector based on the number of IK solutions. The reachability of our robot is illustrated in Fig. 2.6. Thus, c_{door} is defined as a second-order polynomial function of the distance from the base of the manipulator to the door handle. c_{door} has the minimum value at the region of high reachability as

$$c_{door} = K(d - d_{min})^2, \tag{2.2}$$

where K is the positive coefficient and d is the distance between the base of the manipulator and door handle at the current pose of the mobile robot. Based on the reachability data, d_{min} is set as 0.4 m. If the robot can reach the door

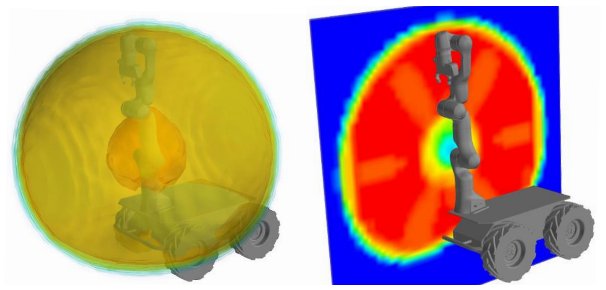


Figure 2.6: Visualization of the reachability of our robot in OPENRAVE [23]. Left figure shows the colored surfaces depending on the density levels of the reachability. Right figure shows the surface cut by a vertical plane at the base of the manipulator (red: high reachability, blue: low reachability).

handle at several door angles, all costs for the corresponding door angles are calculated and the minimum cost is selected among them.

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

$$h(s) = \sum_{i=a}^{4} d_i, \tag{2.3}$$

where d_i is the estimated displacement for the area indicator i, shown in Fig. 2.7, and a is the current state's area indicator. Especially, d_i implies the distance from current state to reach the region of the next area indicator.

In order to obtain the optimal path, the heuristic $h(s)$ should satisfy an inequality as

$$h(s) \leq h(s_{succ}) + c(s, s_{succ}), \tag{2.4}$$

where s_{succ} is the successor of the state s and $c(s, s_{succ})$ is the transition cost.

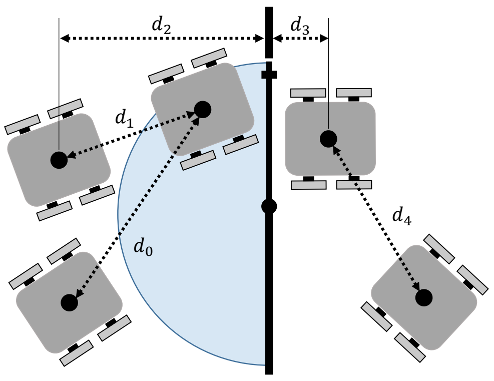


Figure 2.7: Illustration of each term of the heuristic. When the area indicator is 0, d_0 is computed as the distance from the current pose to the pose in front of the door handle. d_1 is the distance from the current pose to the pose outside the door at the current heading angle of the robot. d_2 is distance to the center of the door. d_3 is the distance to cross over the doorsill. d_4 is the distance to the target pose.

The inequality implies that *"heuristic difference never overestimates the transition cost"* and represents the condition called *consistency*. Additionally, the heuristic $h(s)$ should satisfy an inequality expressed as

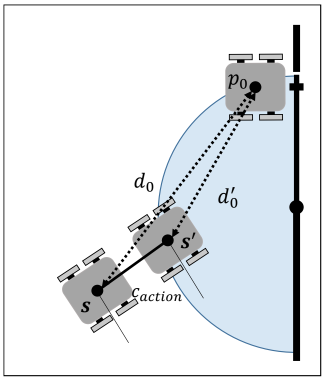
$$h(s) \leq c(s, s_1) + c(s_1, s_g). \tag{2.5}$$

The inequality implies that *"heuristic is never larger than the true cost of reaching the goal state"* and represents the property called *admissibility*. Based on the two inequalities, consistent heuristic is always admissible. Therefore, the consistency of the heuristic in (2.3) will be proven as follows. First, if the area indicator is 0, as shown in Fig. 2.8(a), the difference of the heuristic for two states is $d_0 - d_0^{'}$. d_0 is the distance between the current state and the state located in front of the door handle. The transition cost $c(s, s^{'})$ is $c_{action} + c_{map} + c_{door}$. Based on the triangle inequality, the heuristic is consistent. Next, if the area indicator is 1, as shown in Fig. 2.8(b), the heuristic difference is $d_1 - d_1^{'}$. d_1

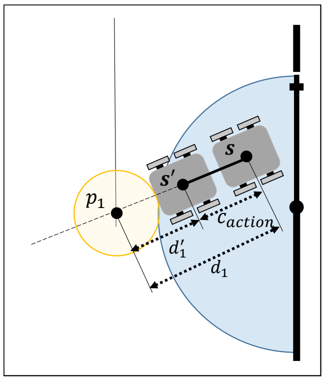
is the distance between the position p_1 between the current state. p_1 is the center of the circle located along the line between s and $s^{'}$ and the radius of the circle is same as the width of the robot. Since the value of $c(s, s^{'})$ is at least $c_{action} = d_1 - d_1^{'}$, the heuristic is proven to be consistent. When the area indicator is 2, the heuristic difference $d_2 - d_2^{'}$ is less than or equal to $c(s, s^{'})$. d_2 is the distance from the current state to the axis of the door. Furthermore, if the area indicator is 3, the difference of the heuristic is less than or equal to the $c(s, s^{'})$. d_3 is the distance between the current state to the axis l_3 in Fig. 2.8(d). l_3 is the axis located away from the axis of the door by the width of the robot. Lastly, the heuristic difference of the states with the area indicator of 4 is $d_4 - d_4^{'}$. d_4 is the distance between the current state and the goal state. Based on the triangle inequality, the heuristic is consistent.

2.2.1.4 Search

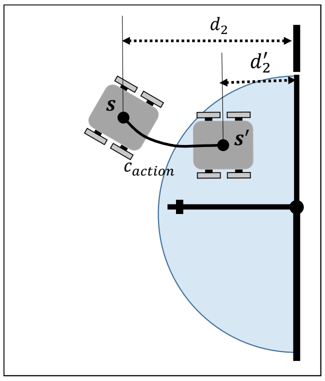
Based on the designed components of the graph including the state, action, and cost, the graph is constructed and searched in order to obtain the optimal path. The initial state s_0 is defined as $(x_0, y_0, \theta_0, a_0)$ where (x_0, y_0, θ_0) is the initial pose of the mobile robot and a_0 is its area indicator. The goal state s_g is defined as as $(x_g, y_g, \theta_g, a_g)$ where (x_g, y_g, θ_g) is the target pose of the mobile robot and a_g is its area indicator. The search algorithm used in this paper is Anytime Repairing A* (ARA*) [24]. The ARA* quickly finds the initial solution path which can be sub-optimal, then refines it towards the optimal path for the remaining planning time. The solution path can be obtained by using the other variants of A* algorithm, such as Anytime Dynamic A* [25] and Lifelong Planning A* [26].



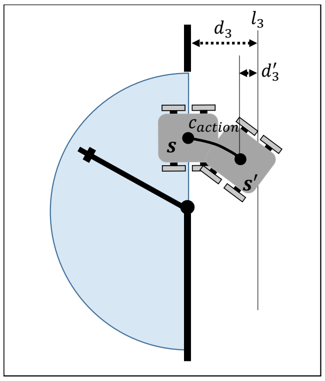
(a) $a = 0$



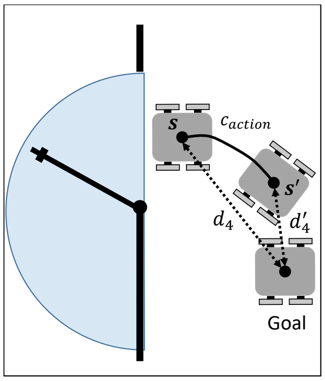
(b) $a = 1$



(c) $a = 2$



(d) $a = 3$



(e) $a = 4$

Figure 2.8: Illustration of the states depending on the value of the area indicator. From (a) to (e), the two states with the same area indicators are demonstrated in order to prove the consistency of the heuristic.

The output of the search is the path from the start pose to the target pose of the mobile robot with the corresponding area indicators. Since the area indicator represents a finite set of valid door angles, the door angle with the minimum cost is selected after they are evaluated by c_{door}. Consequently, the paths for the mobile robot and door angle can be obtained.

2.2.2 Computing path for arm configuration - S2

In order to compute the path for the joint position of the manipulator, the path for the position of the door handle is first computed from the path for the door angle obtained in ***S1*** with the given information of the door model. Then, the joint position of the manipulator grasping the door handle is computed by the IK solver at the corresponding pose of the mobile robot. Since the manipulator with more than 7 degrees of freedom (DOFs) has many IK solutions, the IK solution is selected such that it is furthest from the joint limits and closest to the previous IK solution among 5 IK solutions in this paper. At this point, we do not consider the case that the IK solution does not exist because we assume that the IK solution grasping the door handle always exists if the door handle is within the workspace boundary of the arm. It may fail to compute the IK solution even though the workspace boundary is computed conservatively based on the reachability data. However, this limitation, as also mentioned in [27], is not handled for now and will be dealt with in future work, including the analysis of the reachability. On the other hand, the end-effector of manipulator needs to move to regrasp the door handle on the other side of the door when the line of the door meets the mounted position of the manipulator, as shown in Fig. 2.9. The regrasping motion is planned by the sampling-based motion planners, such as Rapidly-exploring Random Trees.

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

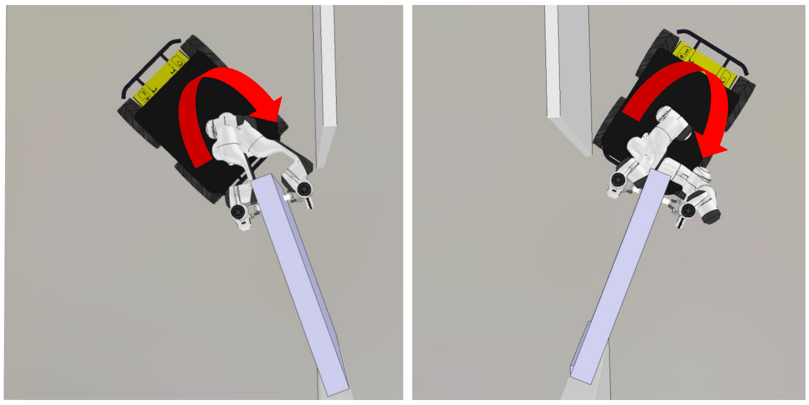

(a) (b)

Figure 2.9: Regrasping the door handle on the other side of the door is necessary in case of (a) pull-type door and (b) push-type door.

2.3 Results

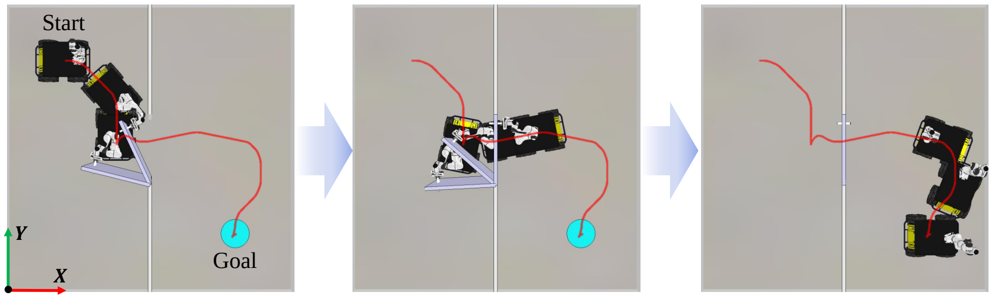
The proposed framework was validated through various simulations and real experiment using the differentially-driven mobile manipulator consisting of the four-wheel mobile base *Husky* (*Clearpath Robotics. Co.*) and the 7-DOFs manipulator *Panda* (*Franka Emika. Co.*). The specification of the computer is *i*7 4.2 GHz with 16 GB RAM. The external libraries were utilized, including SBPL [22] for the graph search algorithm in ***S1*** and TRAC-IK [28] for computing IK solution in ***S2***.

2.3.1 Application to pull and push-type door

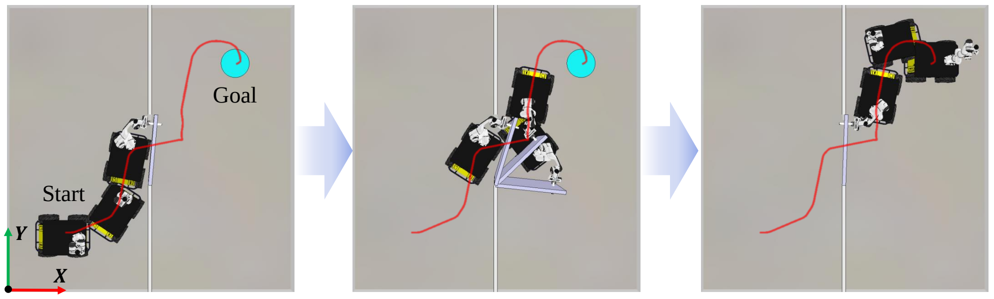
Motion planning for our mobile manipulator was performed by using the proposed framework in order to validate that it is possible to address the navigation problem including the tasks for both pull and push-type door. The snapshots for them are shown in Fig. 2.10. Figure 2.10(a) shows the result of the pull-type door. The inputs to the framework were the start state $s_0 = (1.0, 4.0, 0.0, 0)$ and goal state $s_g = (4.0, 1.0, 0.0, 4)$. The door length is set as 1.0 m. The computation time was recorded as 4.56 s for ***S1*** and 0.71 s for ***S2***. On the other hand, the snapshots of the result for the push-type door are depicted in Fig. 2.10(b). The start and goal state were defined as $s_0 = (1.0, 1.0, 0.0, 0)$ and $s_g = (4.0, 4.0, 0.0, 4)$, respectively. The planning time for ***S1*** was 4.40 s and the computation time for ***S2*** was 0.65 s. In the case of the push-type door, the path for the area indicator started at 0, changed to 3 when the robot was in front of the door, 2 when the robot is located inside the line of the door, 1 when the robot is outside the line of the door, and 4 when the robot moves toward the target position. Therefore, the results indicate that the motion of the mobile manipulator executing the tasks for both pull and push-type door can be planned by the proposed graph search using the concept of the area indicator.

2.3.2 Experiment in cluttered environment

As the experimental environment in the previous section did not have any obstacle except the wall, two experiments were conducted to show the effectiveness of the proposed algorithm. The obstacles were placed at the sides of the door in order to make it difficult for the robot to pass through the door. The snapshots of the results are shown in Fig. 2.11. The inputs to the algorithm were the start state $s_0 = (1.0, 4.0, 0.0, 0)$ and goal state $s_g = (4.0, 4.0, 0, 4)$. The length of the door and width of obstacle were set as 1.0 m. The computation time was recorded was 7.8 s for ***S1*** of Fig. 2.11(a), and 11.1 s for ***S1*** of Fig. 2.11(b). Even though the time for computing the path increased as the environment became



(a)



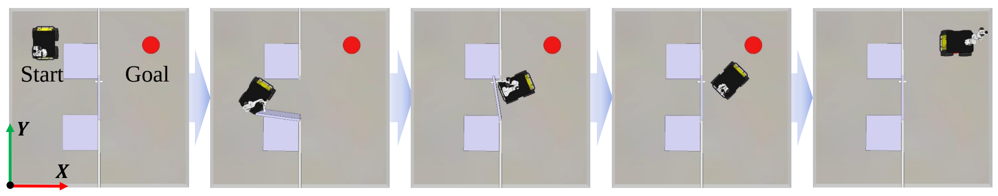
(b)

Figure 2.10: Snapshots of simulation from the proposed framework for (a) pull-type door and (b) push-type door. Red lines indicate the path of the mobile robot.

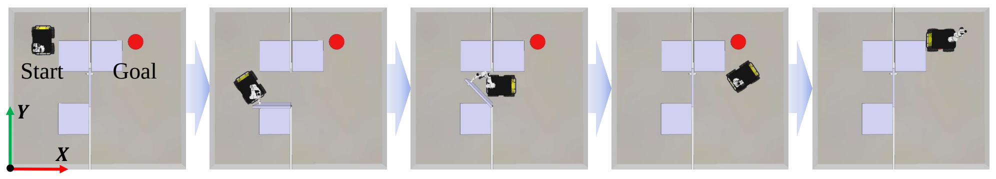
complex, the whole-body motion that the robot approaches, passes through the door, and reaches the goal position was successfully computed.

2.3.3 Experiment with different robot platform

The results in Sec. 2.3.1 and 2.3.2 were computed for the differentially-driven mobile manipulator. However, in order to show that the proposed algorithm is not dependent on the type of the robot, the additional experiment was conducted to apply for the different platform shown in Fig. 2.12(a). The robot in Fig. 2.12(a) has an omni-directional mobile robot called *KUKA Omnirob* and a 7-DOF manipulator called *KUKA LWR*. The snapshots of the result



(a)



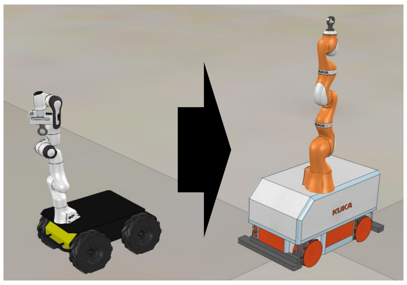
(b)

Figure 2.11: Snapshots of simulation from the proposed framework for the pull-type door with obstacles.

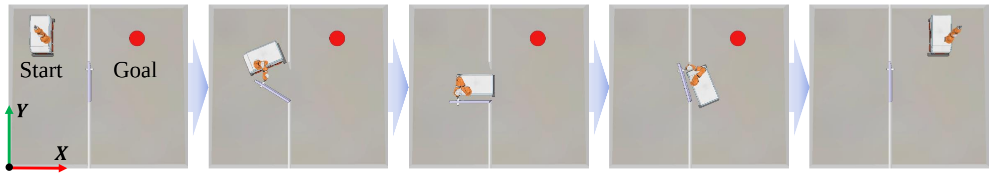
are shown in Fig. 2.12(b). The inputs to the algorithm were the start state $s_0 = (1.0, 4.0, 0.0, 0)$ and goal state $s_g = (4.0, 4.0, 0, 4)$. The length of the door and width of obstacle were set as 1.0 m. The computation time was recorded was 6.5 s for ***S1***. From the experimental result, the proposed algorithm can be applied for any type of mobile manipulator.

2.3.4 Comparison with separate planning by existing works

The previous works [4–6, 14] handled single or few sub-problems related to the door. As mentioned in Sec. 1.2, one can utilize the separate planning that addresses each sub-problem and combines each path of sub-problem into a unified one. Even though the separate planning consisting of the existing methods [14, 24] can solve the navigation problem addressed in the proposed framework, it has several limitations. To show this, an additional experiment was performed to solve the problem of ***S1*** by the separate planning based on the



(a)



(b)

Figure 2.12: Additional experiment was conducted with (a) omni-directional mobile manipulator and snapshots of simulation is shown in (b).

graph search. The problem of ***S1*** can be divided into 4 sub-problems denotes as *SP1*, *SP2*, *SP3*, and *SP4* : *SP1* - Depart from the initial position and approach the door, *SP2* - Open the door to the desired angle, *SP3* - Pass through the door and close it, and *SP4* - Arrive at the target position. To obtain the continuous path, the final position of the path for each sub-problem was set as the start position for the next sub-problem. The state spaces of *SP1* and *SP4* were three-dimensional which are the poses of the mobile robot since *SP1* and *SP4* were the navigation problem to reach the target pose of the mobile robot. The start pose of *SP1* was the same pose in Sec. 4.1 and the target pose of *SP1* was determined by randomly sampling the pose where the door handle is within the workspace boundary of the manipulator. The state spaces of *SP2* and *SP3* were the same as [14], which include the pose of the mobile robot and binary variable compactly representing the door angle. The desired door angle of *SP2* was defined as 135 degrees. For fair comparison, the action and cost designed

Table 2.1: Comparison with the separate planning

Door type	Pull-type door					
Approach	Separate planning					Ours
Success rate (%)	80					**100**
Planning time (s)	*SP1*	*SP2*	*SP3*	*SP4*	Total	**4.56**
	1.53	0.37	36.83	0.94	39.67	
Cost	31195	26393	47110	46227	150925	**99266**
Number of states	21.6	7.6	18.8	26.1	74.0	**35**
Path length (m)	7.68					**6.34**
Door type	Push-type door					
Approach	Separate planning					Ours
Success rate (%)	76					**100**
Planning time (s)	*SP1*	*SP2*	*SP3*	*SP4*	Total	**4.40**
	1.74	1.75	5.90	0.81	10.2	
Cost	35417	23036	50331	33673	142457	**88507**
Number of states	19.2	12.4	13.1	15.5	60.2	**47**
Path length (m)	7.96					**6.24**

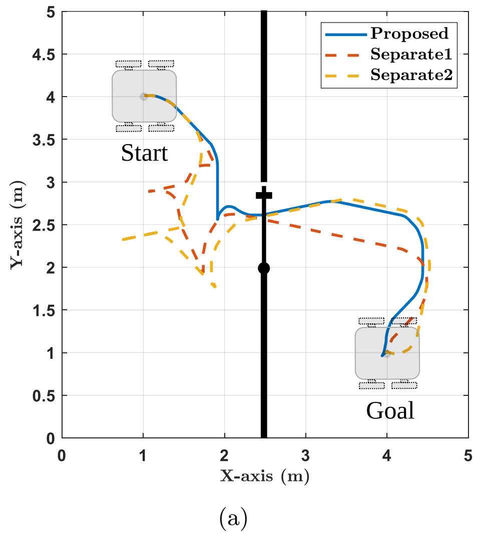
in Sec. 3.1 were utilized for constructing the graph. The ARA* was adopted for the graph search. The maximum computation time was limited as 100 s. The graph search of the separate planning for both pull and push-type door was conducted for 25 trials, respectively. The results of the trials are summarized in Table 2.1.

Regarding the result of the pull-type door in Table 2.1, the success rate of planning the problem of ***S1*** by the separate planning within 100 s was recorded as 80 %. The failure of the search occurred because the initial pose of *SP2* was randomly sampled. Specifically, the sampled pose might be possible to solve *SP2*, but could arrive at the unfavorable pose, which is the initial pose of *SP3*. Furthermore, *SP3* included narrow passage and considered the constraint that the door handle is within the workspace of the arm. These features made it difficult to search for the path of *SP3* within the predefined time period.

The proposed framework required the planning time of 4.56 s, whereas the separate planning required the average planning time of 39.67 s. The path cost computed from our framework was 99266, whereas the path cost obtained from separate planning was 150925 on average, which is 1.52 times more than the proposed framework. The number of states composed of the solution path from the proposed method is less than half that of the separate planning. The length of the path from our framework was obtained as 6.34 m, which 1.34 m shorter than the path from the separate planning.

On the other hand, the success rate of solving ***S1*** for the push-type door was recorded as 76 % in the separate planning. The failure cases were also due to the influence of the randomly sampled initial pose for *SP2*. The planning time of the proposed framework recorded 4.40 s, while that of the separate planning recorded 10.2 s on average. The path cost computed from the proposed framework was 88507, which is about 0.6 times less than the separate planning. The number of states consisting of the solution path from the proposed framework was 47, whereas the separate planning had 60.2 states on average. In terms of path length, the proposed framework computed the path of 6.24 m, while the path from the separate planning was measured as 7.96 m on average.

Additionally, to visualize the difference of the solution path, three paths including two example paths obtained by the separate planning and one path obtained by our framework are shown in Fig. 2.13. As shown in Fig. 2.13, the separate planning computed more complex path of the mobile robot to open and pass through the door than our method because the separate planning randomly sampled the initial pose of *SP2* and defined the desired door angle of *SP2* as a fixed value.



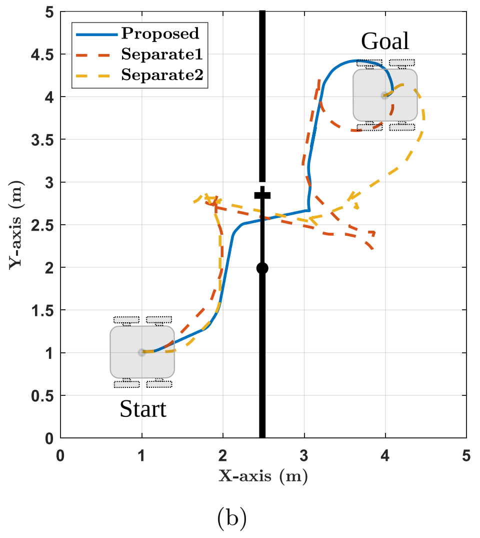
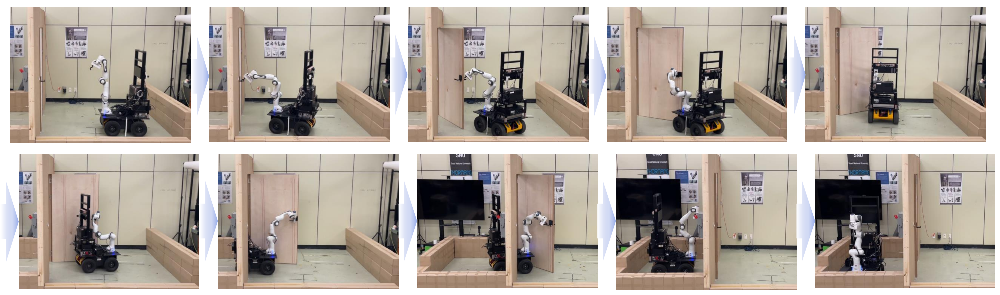


Figure 2.13: Paths of the mobile robot obtained by the proposed framework and separate planning for (a) pull-type door and (b) push-type door. The blue line indicates the path from the proposed framework. The dotted lines show the paths from the separate planning.

Consequently, it was validated through the results in Sec. 2.3.1 and 2.3.2 that the proposed framework computed the path reliably and efficiently due to the following characteristics. First, the complexity of the state was reduced by defining the area indicator expressing the range of the door angle rather than directly including the door angle as a component of the state. Second, the graph was searched efficiently because the heuristic in Eq. (2.3) properly describes the estimated displacement from the current pose to the goal pose. Especially, since the heuristic includes the estimated distance from the current pose to the region where the state has the next area indicator, the state can be easily propagated to the successor state with the next area indicator.



(a)

Figure 2.14: Snapshots of the real experiment results from the proposed framework for pull-type door

2.3.5 Experiment with real robot

The proposed framework was applied to the scenario in real indoor environment. Based on the computed path from the proposed framework, the trajectory was generated considering the velocity limits of the manipulator and the mobile robot. The robot was controlled by using the quadratic programming-based controller such as [29]. The snapshots of the experiment, shown in Fig. 2.14, indicate that the robot successfully executed the tasks including approaching the door, passing through the door, and reaching the target position.

2.4 Conclusion

This chapter proposes a framework to solve the navigation problem including door traversal for the mobile manipulator. The framework computes the whole-body motion of the mobile manipulator in two steps, instead of exploring the whole-body configuration space of the mobile manipulator. The first step is to formulate the navigation problem including the process of passing through the door as a graph search problem, and find the path of the mobile robot and the angular path of the door by the graph search algorithm. In particular, the search space is reduced by introducing a component of the state, called area indicator, which compactly expresses the range of the reachable door angles as an integer. Also, the area indicator implicitly shows the process of approaching, opening, traversing through, closing the door, and reaching the target position. In the second step, the path for the arm configuration grasping the door handle is computed by the IK solver. The effectiveness of the proposed framework was demonstrated through several simulations and real experiment with the differentially-driven mobile manipulator. Future work will involve the contact motion planning of the dual-arm mobile manipulator to compensate for the reaction force of the door when the door is heavy or has the stiffness.

Chapter 3

WHOLE-BODY CONTROLLER : WEIGHTED HIERARCHICAL QUADRATIC PROGRAMMING

Robots with high Degrees of Freedom (DOFs) such as mobile manipulators and humanoids are designed for human-centered environments. To control these robots, whole-body control frameworks [9, 10] have been used to perform various prioritized tasks to consider motion distribution with kinematic redundancy. However, in practice, depending on the scenarios (e.g., locomotion, manipulation, and loco-manipulation), it is difficult to generate natural whole-body behavior only with whole-body controllers. This is because typical whole-body controllers produce joint motion minimizing the same metric for all prioritized tasks. For example, when the mobile manipulator tracks the desired trajectory by using a whole-body, it often comes into singular configuration or reaches the boundary of the workspace due to dynamic and kinematic difference of mobile base and manipulator [30, 31]. Also, during locomotion phase of the humanoid,

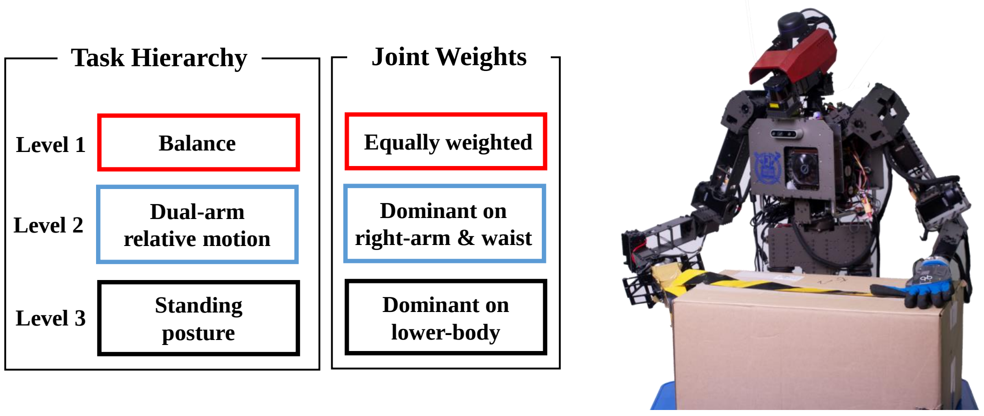


Figure 3.1: An example application of the proposed method with individual weighting matrix for each task priority: a box-taping scenario of humanoid.

the movement of the upper body by whole-body control may adversely affect walking performance [32].

In this chapter, we propose a novel Weighted Hierarchical Quadratic Programming (WHQP) framework to characterize joint movement for each task priority. By combining two concepts of HQP [12] and weighted least-squares norm [33], it can handle various inequality and equality tasks with differently weighted joint motion for each priority effectively. Consequently, the proposed controller can generate natural whole-body behavior without additional subtasks which restrict undesirable movements, as shown in Fig. 3.1.

3.1 Related works

HQP has been actively studied in that it can handle both equality and inequality tasks while ensuring strict priorities of tasks. Kim *et al.* [34] proposed an HQP-based task transition method that can insert, remove, swap the priorities of the tasks while ensuring continuous control inputs. Tassi *et al.* [35] extended HQP in order to produce an impedance-like motion under external disturbance by augmenting the variable for Cartesian velocity. To enhance the computational efficiency, Lee *et al.* [36] utilizes operational space formulation [37] so that the size of the decision variable in QP is reduced when controlling the whole-body of the humanoid.

On the other hand, the weighted least-squares norm which is based on the weighted pseudo inverse can treat the redundancy of the robots without additional constraints. Dariush *et al.* [38] penalized the motion of joints by designing a weighting matrix that determines contribution to the main task according to the extent of the proximity to collision in order to avoid self-collision or obstacle. Similarly, Farelo *et al.* [39] generated optimized joint motion for wheelchair-mounted arm by using weighted pseudo-inverse that considers not only joint limit of arm but also motion limit of wheelchair. Tsuichihara *et al.* [40] utilized weighted pseudo-inverse that restricts chest motion of humanoid to improve stability in manipulation. Park *et al.* [41] and Choi *et al.* [42] combined the task-priority method and the weighted pseudo-inverse method in order to generate the joint motion minimizing the residual error caused by singularity-robust framework. However, since these methods treat the same joint weights for all tasks, they cannot assign individual joint weights for each task priority.

To tackle this issue, Wu *et al.* [11] recently proposed a two-level prioritized whole-body Cartesian impedance controller with individual weighting matrices. In contrast to aforementioned methods [38–42] with the same weighted joint distribution for all tasks, the proposed controller used individual weighting matrices to generate different motion patterns for each task. However, it cannot calculate the solution when extending to multiple tasks since the main task and its null-space are only considered. Also, [11] cannot deal with inequality constraints.

On the other hand, in terms of generating the whole-body motion, a concept called *inverse reachability map* (IRM) [43–45] can be utilized. The IRM is an inverse kinematic solver that gives the whole-body configuration of the robot. Burget and Bennewitz [43] proposed a method that generates a whole-body motion of humanoid robot by using the IRM for calculating the optimal footstep of the humanoid given the grasp pose. Yang *et al.* [44] calculated the whole-body configuration of the humanoid for considering several ground situations including obstacle and inclined surface. Moreover, Chen *et al.* [45] calculated optimal whole-body configuration of the mobile manipulator for the given grasp pose while maximizing the extended manipulability which incorporates joint limit and collision. However, the IRM cannot produce the whole-body motion in real-time as fast as HQP.

3.2 Problem statement

This section provides the modification of the weighted pseudo-inverse to handle individual weights of joints for each task hierarchy and its limitations.

3.2.1 Pseudo-inverse with weighted least-squares norm for each task

Let us assume that there are p prioritized tasks and p weighting matrices for each task of n-DOFs redundant robot:

$$\dot{x}_k = J_k \dot{q}, \quad (k = 1, \cdots, p) \tag{3.1}$$

where $\dot{x}_k \in \mathbb{R}^{m_k}$, $J_k \in \mathbb{R}^{m_k \times n}$, and $q \in \mathbb{R}^n$ are the task space velocity, the Jacobian matrix, and the joint position of the robot, respectively. The priorities are indicated by the number of subscripts: the smaller the subscript number, the higher the task priority. Also, the weighting matrix $W_k \in \mathbb{R}^{n\times n}$ is assumed to be symmetric and positive definite.

To execute the first task $\dot{x}_1$ while assigning dominant joints or optimizing a performance criterion through the weighting matrix W_1, an optimization problem is formulated as

$$\begin{aligned} \min_{\dot{q}_1} \quad & \frac{1}{2}\dot{q}_1^T W_1 \dot{q}_1, \\ \text{s. t.} \quad & J_1 \dot{q}_1 = \dot{x}_1. \end{aligned} \tag{3.2}$$

The optimal solution $\dot{q}_1^* \in \mathbb{R}^n$ is calculated analytically as

$$\dot{q}_1^* = J_1^{W_1+}\dot{x}_1 = W_1^{-1}J_1^T(J_1 W_1^{-1} J_1^T)^{-1}\dot{x}_1, \tag{3.3}$$

where $J_1^{W_1+} \in \mathbb{R}^{n\times m_1}$ is the weighted pseudo-inverse of J_1.

By utilizing the weighted pseudo-inverse solution in (3.3), a solution that executes multiple tasks with differently weighted joint solution for each task priority can be computed. Considering the secondary task $\dot{x}_2$ with the corresponding weighting matrix W_2, the total solution $\dot{q}_2^*$ for two prioritized tasks would be

$$\dot{q}_2^* = \dot{q}_1^* + \dot{\tilde{q}}_2^*, \tag{3.4}$$

where $\dot{\tilde{q}}_2^*$ denotes the contribution to the secondary task without modifying the first task. This can be obtained by solving the following optimization problem as

$$\begin{aligned} \min_{\dot{\tilde{q}}_2} \quad & \frac{1}{2}\dot{\tilde{q}}_2^T W_2 \dot{\tilde{q}}_2, \\ \text{s. t.} \quad & J_1 \dot{\tilde{q}}_2 = \dot{x}_1 - J_1 \dot{q}_1^* = 0, \\ & J_2 \dot{\tilde{q}}_2 = \dot{x}_2 - J_2 \dot{q}_1^*, \end{aligned} \tag{3.5}$$

The solution for (3.5) is calculated as

$$\dot{\tilde{q}}_2^* = (J_2 N_1^{W_2})^{W_2+}(\dot{x}_2 - J_2 \dot{q}_1^*). \tag{3.6}$$

where $N_1^{W_2} = I - W_2^{-1} J_1^T (J_1 W_2^{-1} J_1^T)^{-1} J_1 \in \mathbb{R}^{n\times n}$ is the projector onto the null-space of J_1 weighted by W_2. Note that the projector $N_1^{W_2}$ is idempotent, but not symmetric.

Thus, it is straightforward to obtain a general solution for p prioritized tasks:

$$\dot{q}_p^* = \sum_{k=1}^{p} (J_k N_{k-1}^{W_k})^{W_k+}(\dot{x}_k - J_k \dot{q}_{k-1}^*), \tag{3.7}$$

where $N_0^{W_1} = I$, $\dot{q}_0^* = 0$, and $N_{k-1}^{W_k}$ is the projector onto the null-space of the augmented Jacobian $\underline{J}_{k-1} = \left[J_1^T, J_2^T, \quad \dots \quad, J_{k-1}^T \right]^T \in \mathbb{R}^{\sum_{i=1}^{k-1} m_i \times n}$ weighted by W_k. At this point, all tasks are assumed to be full rank. The projector $N_{k-1}^{W_k}$

can be computed in two ways as

$$N_{k-1}^{W_k} = N_{[1]}^{W_k} N_{[2]}^{W_k} \cdots N_{[k-1]}^{W_k} = \prod_{j=1}^{k-1} N_{[j]}^{W_k}, \tag{3.8a}$$

$$N_{k-1}^{W_k} = I - \underline{J}_{k-1}^{W_k+} \underline{J}_{k-1}, \tag{3.8b}$$

where $N_{[j]}^{W_k} = I - (J_j \prod_{i=0}^{j-1} N_{[i]}^{W_k})^{W_k+}(J_j \prod_{i=0}^{j-1} N_{[i]}^{W_k})$, $N_{[0]}^{W_k} = I$, and $N_{[1]}^{W_k} = I - W_k^{-1} J_1^T (J_1 W_k^{-1} J_1^T)^{-1} J_1$.

3.2.2 Problem statement

When computing the null-space projection matrix in (3.8), the matrix is asymmetric and its size remains constant. Owing to these properties, the computational cost increases exponentially when propagating to multiple tasks. To improve computational efficiency, [46] proposed a scheme to accelerate the computation by decomposing a symmetric null-space projection matrix. Therefore, our goal is to efficiently compute weighted least-squares norm solution by transforming the asymmetric null-space projection matrix in (3.8) into a symmetric matrix. Furthermore, we extend the pseudo-inverse with weighted least-squares norm for each task priority of Sec. 3.1 to deal with inequality as well as equality tasks.

3.3 WHQP with equality constraints

This section formulates the WHQP that minimizes the violation of the equality task in a weighted least-squares norm manner. When formulating the WHQP, Jacobian matrix and joint velocity are transformed into a weighted Jacobian matrix and joint velocity. This makes the null-space projection matrix symmetric, which enables the proposed solution to be computed more efficiently than the traditional method using weighted pseudo-inverse.

For the first task $\dot{x}_1$, the weighted Jacobian matrix J_{1,W_1} and weighted joint velocity $\dot{q}_{W_1}$ are defined as

$$\begin{aligned} J_{1,W_1} &= J_1 W_1^{-\frac{1}{2}}, \\ \dot{q}_{W_1} &= W_1^{\frac{1}{2}} \dot{q}. \end{aligned} \tag{3.9}$$

Then, the WHQP is formulated as

$$\begin{aligned} \min_{\dot{q}_{W_1}, s_1} \quad & \frac{1}{2}\|s_1\|_2^2, \\ \text{s. t.} \quad & J_{1,W_1}\dot{q}_{W_1} = \dot{x}_1 + s_1, \end{aligned} \tag{3.10}$$

where $s_1 \in \mathbb{R}^{m_1}$ is the slack variable that alleviates the task $\dot{x}_1$. Note that the variable to optimize is not $\dot{q}$, but $\dot{q}_{W_1}$.

To solve this problem, the Lagrangian $\mathcal{L}_1$ is computed as

$$\mathcal{L}_1 = \frac{1}{2} s_1^T s_1 + \lambda_1^T (J_{1,W_1}\dot{q}_{W_1} - \dot{x}_1 - s_1), \tag{3.11}$$

where $\lambda_1 \in \mathbb{R}^{m_1}$ is the Lagrange multiplier. From the first-order optimality conditions, the Lagrangian differentiated by λ_1, s_1, and $\dot{q}_{W_1}$ should be zero:

$$\begin{aligned} \frac{\partial \mathcal{L}_1}{\partial \lambda_1} &= J_{1,W_1}\dot{q}_{W_1} - \dot{x}_1 - s_1 = 0, \\ \frac{\partial \mathcal{L}_1}{\partial s_1} &= s_1 - \lambda_1 = 0, \\ \frac{\partial \mathcal{L}_1}{\partial \dot{q}_{W_1}} &= J_{1,W_1}^T \lambda_1 = 0. \end{aligned} \tag{3.12}$$

To obtain the weighted solution $\dot{q}^*_{W_1}$, the pseudo-inverse of the weighted Jacobian matrix J_{1,W_1} is computed using COD [12, 47] which is cheaper than computing the singular value decomposition as

$$\begin{aligned} J_{1,W_1} &= \begin{bmatrix} V_{1,W_1} & U_{1,W_1} \end{bmatrix} \begin{bmatrix} 0 & 0 \\ L_{1,W_1} & 0 \end{bmatrix} \begin{bmatrix} Y_{1,W_1} & Z_{1,W_1} \end{bmatrix}^T \\ &= U_{1,W_1} L_{1,W_1} Y^T_{1,W_1}, \end{aligned} \tag{3.13}$$

where $U_{1,W_1} \in \mathbb{R}^{m_1 \times r_1}$ and $V_{1,W_1} \in \mathbb{R}^{m_1 \times (m_1 - r_1)}$ are the orthonormal bases for the column space of J_{1,W_1}. r_1 is the rank of J_{1,W_1}. $Y_{1,W_1} \in \mathbb{R}^{n \times r_1}$ and $Z_{1,W_1} \in \mathbb{R}^{n \times (n - r_1)}$ are the orthonormal bases for the row space of J_{1,W_1}. $L_{1,W_1} \in \mathbb{R}^{r_1 \times r_1}$ is a lower triangular matrix. With this decomposition, the weighted solution and slack variable satisfying (3.12) are obtained as

$$\dot{q}^*_{W_1} = J^+_{1,W_1} \dot{x}_1 = Y_{1,W_1} L^{-1}_{1,W_1} U^T_{1,W_1} \dot{x}_1, \tag{3.14}$$

$$s^*_1 = U_{1,W_1} U^T_{1,W_1} \dot{x}_1 - \dot{x}_1 = -V_{1,W_1} V^T_{1,W_1} \dot{x}_1, \tag{3.15}$$

where J^+_{1,W_1} denotes the pseudo-inverse of J_{1,W_1}. Based on (3.13) and (3.15), the last optimality condition of (3.12) is satisfied. In the end, the complete solution $\dot{q}^*_1$ for the first task is obtained by transforming the weighted solution to the original joint space as

$$\dot{q}^*_1 = W_1^{-\frac{1}{2}} \dot{q}^*_{W_1}. \tag{3.16}$$

Similarly, the WHQP for the secondary task $\dot{x}_2$ is formulated as

$$\begin{aligned} \min_{\dot{q}_{W_2}, s_2} \quad & \frac{1}{2}\|s_2\|_2^2. \\ \text{s. t.} \quad & J_{1,W_2}\dot{q}_{W_2} = 0, \\ & J_{2,W_2}\dot{q}_{W_2} = \dot{x}_2 - J_2\dot{q}_1^* + s_2. \end{aligned} \tag{3.17}$$

Similar to (3.6), the weighted solution $\dot{q}^*_{W_2}$ lies in the null-space of J_{1,W_2} and at the same time executes the residual task after subtracting the effect of $\dot{q}_1^*$ on the secondary task space. It is obtained as

$$\dot{q}^*_{W_2} = (J_{2,W_2}N_{1,W_2})^+(\dot{x}_2 - J_2\dot{q}_1^*), \tag{3.18}$$

where $N_{1,W_2} = I - J^+_{1,W_2}J_{1,W_2}$ is the null-space projection matrix of J_{1,W_2}.

Unlike (3.6), N_{1,W_2} is both idempotent and symmetric. Thus, N_{1,W_2} can be represented as

$$N_{1,W_2} = Z_{1,W_2}Z^T_{1,W_2}, \tag{3.19}$$

where $Z_{1,W_2} \in \mathbb{R}^{n\times(n-r_1)}$ is the null-space bases of J_{1,W_2} and obtained by decomposing J_{1,W_2}. Then, the solution in (3.18) can be represented in a more efficient form [46] as

$$\begin{aligned} \dot{q}^*_{W_2} &= Z_{1,W_2}(J_{2,W_2}Z_{1,W_2})^+(\dot{x}_2 - J_2\dot{q}_1^*) \\ &= Y_{2,W_2}L^{-1}_{2,W_2}U^T_{2,W_2}(\dot{x}_2 - J_2\dot{q}_1^*), \end{aligned} \tag{3.20}$$

where $Y_{2,W_2} = Z_{1,W_2}\tilde{Y}_{2,W_2} \in \mathbb{R}^{n\times r_2}$. $\tilde{Y}_{2,W_2} \in \mathbb{R}^{(n-r_1)\times r_2}$, $L_{2,W_2} \in \mathbb{R}^{r_2\times r_2}$, and

$U_{2,W_2} \in \mathbb{R}^{m_2 \times r_2}$ are obtained by decomposing $J_{2,W_2} Z_{1,W_2} \in \mathbb{R}^{m_2 \times (n-r_1)}$ as

$$J_{2,W_2} Z_{1,W_2} = \begin{bmatrix} V_{2,W_2} & U_{2,W_2} \end{bmatrix} \begin{bmatrix} 0 & 0 \\ L_{2,W_2} & 0 \end{bmatrix} \begin{bmatrix} \tilde{Y}_{2,W_2} & \tilde{Z}_{2,W_2} \end{bmatrix}^T. \tag{3.21}$$

Using (3.20), the optimal slack variable s_2^* is computed as

$$\begin{aligned} s_2^* &= J_{2,W_2} \dot{q}_{W_2}^* - \dot{x}_2 + J_2 \dot{q}_1^*, \\ &= U_{2,W_2} U_{2,W_2}^T (\dot{x}_2 - J_2 \dot{q}_1^*) - \dot{x}_2 + J_2 \dot{q}_1^* \\ &= V_{2,W_2} V_{2,W_2}^T (J_2 \dot{q}_1^* - \dot{x}_2). \end{aligned} \tag{3.22}$$

Therefore, the total solution for two prioritized tasks $\dot{x}_1$, $\dot{x}_2$ is

$$\dot{q}_2^* = W_1^{-\frac{1}{2}} \dot{q}_{W_1}^* + W_2^{-\frac{1}{2}} \dot{q}_{W_2}^*. \tag{3.23}$$

More generally, WHQP for the k-th task $\dot{x}_k$ is formulated as

$$\begin{aligned} \min_{\dot{q}_{W_k}, s_k} \quad & \frac{1}{2} \|s_k\|_2^2. \\ \text{s. t.} \quad & J_{k,W_k} \dot{q}_{W_k} = \dot{x}_k - J_k \dot{q}_{k-1}^* + s_k \\ & \underline{J}_{k-1,W_k} \dot{q}_{W_k} = 0 \end{aligned} \tag{3.24}$$

where $\underline{J}_{k-1,W_k} = \begin{bmatrix} J_{1,W_k}^T, J_{2,W_k}^T, \cdots, J_{k-1,W_k}^T \end{bmatrix}^T$ is the augmented Jacobian matrix weighted by $W_k^{-\frac{1}{2}}$.

To solve (3.24), the Lagrangian $\mathcal{L}_k$ is constructed as

$$\begin{aligned} \mathcal{L}_k = & \frac{1}{2} s_k^T s_k + \lambda_k^T (J_{k,W_k} \dot{q}_{W_k} - \dot{x}_k + J_k \dot{q}_{k-1}^* - s_k) \\ & + \underline{\lambda}_{k-1}^T (\underline{J}_{k-1,W_k} \dot{q}_{W_k}). \end{aligned} \tag{3.25}$$

Then, the optimality conditions for the k-th task are directly obtained as

$$\begin{aligned}\frac{\partial \mathcal{L}_k}{\partial \lambda_k} &= J_{k,W_k}\dot{q}_{W_k} - \dot{x}_k + J_k \dot{q}^*_{k-1} - s_k = 0,\\ \frac{\partial \mathcal{L}_k}{\partial \underline{\lambda}_{k-1}} &= \underline{J}_{k-1,W_k}\dot{q}_{W_k} = 0,\\ \frac{\partial \mathcal{L}_k}{\partial s_k} &= s_k - \lambda_k = 0,\\ \frac{\partial \mathcal{L}_k}{\partial \dot{q}_{W_k}} &= J^T_{k,W_k}\lambda_k + \underline{J}^T_{k-1,W_k}\underline{\lambda}_{k-1} = 0.\end{aligned} \tag{3.26}$$

To calculate the solution, the Jacobian matrices J_{k,W_k}, $\underline{J}_{k-1,W_k}$ are decomposed as in [12]:

$$\begin{aligned}J_{k,W_k} &= \begin{bmatrix} V_{k,W_k} & U_{k,W_k}\end{bmatrix}\begin{bmatrix} N_{k,W_k} & 0 \\ M_{k,W_k} & L_{k,W_k}\end{bmatrix}\begin{bmatrix}\underline{Y}_{k-1,W_k} & Y_{k,W_k}\end{bmatrix}^T\\ &= E_{k,W_k} H_{k,W_k} \underline{Y}^T_{k,W_k},\end{aligned} \tag{3.27}$$

where

$$\begin{aligned} N_{k,W_k} &= V^T_{k,W_k} J_{k,W_k} \underline{Y}_{k-1,W_k} \in \mathbb{R}^{(m_k - r_k)\times \sum_{i=1}^{k-1} r_i},\\ M_{k,W_k} &= U^T_{k,W_k} J_{k,W_k} \underline{Y}_{k-1,W_k} \in \mathbb{R}^{r_k \times \sum_{i=1}^{k-1} r_i},\\ \underline{Y}_{k-1,W_k} &= \begin{bmatrix} Y_{1,W_k}, \cdots, Y_{k-1,W_k}\end{bmatrix} \in \mathbb{R}^{n\times \sum_{i=1}^{k-1} r_i},\\ Y_{k,W_k} &= Z_{k-1,W_k}\tilde{Y}_{k,W_k} \in \mathbb{R}^{n\times r_k}.\end{aligned} \tag{3.28}$$

V_{k,W_k}, U_{k,W_k}, L_{k,W_k}, and $\tilde{Y}_{k,W_k}$ are obtained from decomposing $J_{k,W_k}Z_{k-1,W_k}$. The basis $\underline{Y}_{k-1,W_k}$ is obtained by recursively decomposing $k-1$ times from J_{1,W_k} to J_{k-1,W_k}. Using the representation of (3.27), the stacked matrix $\underline{J}_{k-1,W_k}$ is described as

$$\underline{J}_{k-1,W_k} = \begin{bmatrix} E_{1,W_k} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & E_{k-1,W_k} \end{bmatrix} \begin{bmatrix} H_{1,W_k} & 0 \\ \vdots & \vdots \\ N_{k-1,W_k} & 0 \\ M_{k-1,W_k} & L_{k-1,W_k} \end{bmatrix} \begin{bmatrix} Y^T_{1,W_k} \\ \vdots \\ Y^T_{k-1,W_k} \end{bmatrix} \tag{3.29}$$
$$= \underline{E}_{k-1,W_k} \underline{H}_{k-1,W_k} \underline{Y}^T_{k-1,W_k}.$$

Then, the weighted solution $\dot{q}^*_{W_k}$ satisfying (3.26) is computed as

$$\dot{q}^*_{W_k} = Y_{k,W_k} L^{-1}_{k,W_k} U^T_{k,W_k} (\dot{x}_k - J_k \dot{q}^*_{k-1}). \tag{3.30}$$

Next, the optimal slack variable s^*_k is obtained as

$$s^*_k = V_{k,W_k} V^T_{k,W_k} (J_k \dot{q}^*_{k-1} - \dot{x}_k). \tag{3.31}$$

The Lagrange multipliers, λ^*_k and $\underline{\lambda}^*_{k-1}$, satisfying (3.26) are directly computed as

$$\lambda^*_k = s^*_k, \tag{3.32}$$

$$\underline{\lambda}^*_{k-1} = -\underline{J}^{\ddagger T}_{k-1,W_k} J^T_{k,W_k} s^*_k, \tag{3.33}$$

where $\underline{J}^{\ddagger}_{k-1,W_k}$ is the pseudo-inverse matrix of $\underline{J}_{k-1,W_k}$ which fulfills Moore-Penrose conditions except that $\underline{J}_{k-1,W_k}\ \underline{J}^{\ddagger}_{k-1,W_k}$ is symmetric. Each component of the Lagrange multipliers $\underline{\lambda}^*_{k-1}$ can be obtained recursively in a descending order as follows:

$$\underline{\lambda}^*_{k-1} = \begin{bmatrix} \underline{\lambda}^*_{k-2} \\ \lambda^*_{k-1} \end{bmatrix} = \begin{bmatrix} -\underline{J}^{\ddagger T}_{k-2,W_k} (J^T_{k,W_k} s^*_k + J^T_{k-1,W_k} \lambda^*_{k-1}) \\ -U_{k-1,W_k} L^{-T}_{k-1,W_k} Y^T_{k-1,W_k} J^T_{k,W_k} s^*_k \end{bmatrix} \tag{3.34}$$

The obtained Lagrange multipliers are used as an indicator to determine which task to be deactivated depending on whether they are negative or not in the following section.

Finally, the total solution for k prioritized tasks is

$$\begin{aligned} \dot{q}_k^* &= \dot{q}_{k-1}^* + W_k^{-\frac{1}{2}} \dot{q}_{W_k}^* \\ &= \dot{q}_{k-1}^* + W_k^{-\frac{1}{2}} Y_{k,W_k} L_{k,W_k}^{-1} U_{k,W_k}^T (\dot{x}_k - J_k \dot{q}_{k-1}^*). \end{aligned} \tag{3.35}$$

Additionally, the solution can be computed in a recursive form as follows:

$$\dot{q}_k^* = \underline{J}_{k,W_k}^{\ddagger} \underline{\dot{x}}_k, \tag{3.36}$$

where

$$\begin{aligned} \underline{J}_{k,W_k}^{\ddagger} &= \Big[(I - W_k^{-\frac{1}{2}} \quad Y_{k,W_k} L_{k,W_k}^{-1} U_{k,W_k}^T \quad J_k) \quad \underline{J}_{k-1,W_{k-1}}^{\ddagger}, \\ & \quad W_k^{-\frac{1}{2}} \quad Y_{k,W_k} \quad L_{k,W_k}^{-1} \quad U_{k,W_k}^T \Big] \\ \underline{\dot{x}}_k &= \Big[\underline{\dot{x}}_{k-1}^T, \quad \dot{x}_k^T \Big]^T . \end{aligned} \tag{3.37}$$

Since the solution minimizing weighted least-squares norm is the generalized solution to cover Euclidean least-squares norm solution, the formulas (3.36) and (3.37) can be utilized to compute the solution of the HQP if the weighting matrix is set to the identity matrix.

3.4 WHQP with inequality constraints

To obtain the optimal solution for p inequality tasks, the WHQP for the k-th inequality task is formulated by rewriting (3.24) as

$$\begin{aligned} \min_{\dot{q}_{W_k}, s_k} \quad & \frac{1}{2}\|s_k\|_2^2, \\ \text{s. t.} \quad & J_{k,W_k}\dot{q}_{W_k} \leq \dot{x}_k - J_k\dot{q}^*_{k-1} + s_k, \\ & \underline{J}_{k-1,W_k}\dot{q}_{W_k} \leq \underline{\dot{x}}_{k-1} - \underline{J}_{k-1}\dot{q}^*_{k-1}. \end{aligned} \tag{3.38}$$

The optimality conditions for (3.38) are additionally considered in (3.26) as follows:

$$\begin{aligned} \lambda_k^T(J_{k,W_k}\dot{q}_{W_k} - \dot{x}_k + J_k\dot{q}^*_{k-1} - s_k) &= 0, \\ \underline{\lambda}_{k-1}^T(\underline{J}_{k-1,W_k}\dot{q}_{W_k} - \underline{\dot{x}}_{k-1} + \underline{J}_{k-1}\dot{q}^*_{k-1}) &= 0, \\ \lambda_k &\geq 0, \\ \underline{\lambda}_{k-1} &\geq 0. \end{aligned} \tag{3.39}$$

The conditions of (3.39) denote complementary conditions that if an inequality task becomes active, that is, it becomes an equality task, its corresponding Lagrange multiplier must be greater than or equal to zero.

Based on the above conditions, the active-set method [13] is adopted. The active-set method iterates a loop until the optimal solution and optimal working set are determined. The working set denotes a set of equality tasks. The optimal solution indicates that it does not activate or violate any other task except the working set. The optimal working set indicates that the corresponding Lagrangian multipliers are non-negative.

The detailed procedure is described in Algorithm 1. At first, the algorithm estimates an initial working set $\mathcal{W}_0$ for warm-start. Typically, $\mathcal{W}_0$ contains only

Algorithm 1 WHQP

```
Input: $\mathcal{W}_0$ : an initial working set
Output: $\dot{q}^*$ : an optimal solution
 1: $\mathcal{W} = \mathcal{W}_0,\ \mathit{iter} = 1,\ \dot{q} = 0$
 2: while $\mathit{iter} \leq p - 1$ do
    // Compute the optimal solution in (3.36)
 3:     $\dot{q}^* = \mathit{WHQP\_equality}(\underline{J}^{\ddagger}_{p,W_p}, \underline{\dot{x}}_p, \mathcal{W})$
    // Compute the step length in (3.40)
 4:     $\alpha, k, r = \mathit{ComputeStepLength}(\underline{J}_p, \underline{\dot{x}}_p, \dot{q}^*, \dot{q})$
 5:     $\dot{q} := \dot{q} + \alpha(\dot{q}^* - \dot{q})$
    // Update the working set
 6:     if $\alpha < 1$ then
 7:         $\mathcal{W} \bigcup (J_k[r], \dot{x}_k[r])$
 8:         continue
 9:     else
    // Check Lagrange multipliers
10:         for $i = \mathit{iter}$ to $p$ do
11:             $\underline{\lambda}^*_{i-1}, s^*_i = \mathit{ComputeLambda}(i)$
12:             $\mu, k, r = \min(\underline{\lambda}^*_{i-1}, s^*_i)$
13:             if $\mu < 0$ then
14:                 $\mathcal{W} \setminus (J_k[r], \dot{x}_k[r])$
15:                 break
16:             end if
17:             $\mathit{iter} = i$
18:         end for
19:     end if
20: end while
        return $\dot{q}^* := \dot{q}$
```

equality tasks if there is no initial guess for the inequality tasks. Given the working set $\mathcal{W}$, *WHQP_equality* computes the optimal solution by using (3.36) (see Line 3).

Then, *ComputeStepLength* finds a step length α in order for the current solution to step toward the optimal solution without violating the tasks as (see Line 4)

$$\alpha = \min(1, \min_{k,r}(\alpha_{k,r})), \tag{3.40}$$

where

$$\alpha_{k,r} = \begin{cases} \frac{\dot{x}_k[r] - J_k[r]\dot{q}^{(j)}}{J_k[r](\dot{q}^{*(j)} - \dot{q}^{(j)})} & if \quad J_k[r]\dot{q}^{(j)} \leq \dot{x}_k[r] \\ 1, & otherwise. \end{cases} \tag{3.41}$$

In (3.40) and (3.41), k and r denote the indices of the task priority and row of the Jacobian matrix and task vector, respectively. In addition, $\dot{q}^{*(j)}$ and $\dot{q}^{(j)}$ represent the optimal solution and current solution at iteration j, respectively. Calculation of (3.41) is performed for each row of $\underline{J}_p$ that is not in the working set $\mathcal{W}$.

Starting at the current solution $\dot{q}^{(j)}$, the solution that is translated toward the optimal solution $\dot{q}^{*(j)}$ with a step length α is calculated as follows (see Line 5):

$$\dot{q}^{(j+1)} = \dot{q}^{(j)} + \alpha(\dot{q}^{*(j)} - \dot{q}^{(j)}) \tag{3.42}$$

If the step length is less than 1, the corresponding row of the Jacobian matrix, $J_k[r]$ and component of the task vector, $\dot{x}_k[r]$, are added to the working set, and a new iteration begins with the new working set (see Line 6-8).

Once the obtained step length is equal to 1, which indicates that the optimal solution does not activate the remaining inequality tasks, *ComputeLambda* computes the Lagrangian multipliers using (3.32) and (3.33) (see Line 11). Next, to ensure that the optimality conditions of (3.39) are satisfied, it is checked whether or not the minimum Lagrangian multiplier is negative (see Line 12-13). If there exist any negative Lagrangian multiplier, the corresponding row of the Jacobian matrix and component of the task vector are removed from the working set, and the next iteration begins (see Line 14-15).

3.5 Experimental results

The WHQP was validated through various experiments using the differentially-driven mobile base with the 7-DOFs manipulator [10] and the humanoid, *DYROS-JET* [48], with 28-DOFs.

3.5.1 Simulation experiment with nonholonomic mobile manipulator

To maximize the dexterity and mobility of the mobile manipulator, the controller should provide different motion patterns depending on the locomotion, manipulation, and loco-manipulation phases. In particular, because the mobile manipulator comprises two independent systems (mobile base and manipulator), assigning dominant subsystem for each task priority is effective.

3.5.1.1 Scenario description

To validate this, we designed the following scenario with an 11-DOFs nonholonomic mobile manipulator on simulator, *CoppeliaSim*. The scenario has two phases and each phase is performed for 5 sec. In the first phase, the mobile manipulator reaches the target point above a laptop by tracking the desired trajectory of the end-effector. Then, the robot begins to keep the end-effector focused on the moving target point above the laptop while avoiding an obstacle in the second phase.

3.5.1.2 Task and weighting matrix description

During the first phase, the robot has a single task of tracking the end-effector position trajectory as

$$J_1\dot{q} = \dot{x}_1 \tag{3.43}$$

where $J_1 \in \mathbb{R}^{3\times 11}$ is the whole-body translational Jacobian of the end-effector [10] and $\dot{x}_1 \in \mathbb{R}^3$ is the desired linear velocity of the end-effector.

In the second phase, the robot executes three prioritized tasks. As the highest priority task, an 11-DOFs joint limit avoidance task [49] is assigned as

$$\begin{aligned} &\underline{\dot{x}}_1 \leq J_1\dot{q} \leq \overline{\dot{x}}_1, \\ &J_1 = I_{11}, \\ &\underline{\dot{x}}_1 = \epsilon\frac{\underline{q}-q}{\Delta t} \quad \mathit{and} \quad \overline{\dot{x}}_1 = \epsilon\frac{\overline{q}-q}{\Delta t}. \end{aligned} \tag{3.44}$$

$I_{11} \in \mathbb{R}^{11\times 11}$ denotes an identity matrix and $\underline{\dot{x}}_1$ and $\overline{\dot{x}}_1 \in \mathbb{R}^{11}$ are the lower and upper bound velocity, respectively. $\underline{q}$ and $\overline{q} \in \mathbb{R}^{11}$ are the lower and upper limit of joint position. $q \in \mathbb{R}^{11}$ is the current joint position, ϵ is the tuning parameter and Δt is the control loop period. Then, the obstacle avoidance is formed as an 1-DOF inequality task [50] with the second priority as

$$\begin{aligned} J_2\dot{q} &\leq \dot{x}_2, \\ J_2 &= n^T J_{obs}, \\ \dot{x}_2 &= \epsilon\frac{d-d_{thre}}{\Delta t}, \end{aligned} \tag{3.45}$$

where $n \in \mathbb{R}^3$ is the direction vector from the closest point on the robot to the obstacle and $J_{obs} \in \mathbb{R}^{3\times 11}$ is the Jacobian matrix of the closest point. d is the distance between the robot and obstacle and d_{thre} denotes the threshold distance. Finally, the lowest priority task is designed as the 3-DOFs gaze task [51] which aligns the target point with the line of sight from the end-effector. The gaze task is defined as

$$\begin{aligned} J_3\dot{q} &= \dot{x}_3, \\ J_3 &= p_{ee,t} \times (p_{ee,l} \times J_w) - p_{ee,l} \times J_v, \\ \dot{x}_3 &= -\lambda(p_{ee,l} \times p_{ee,t}), \end{aligned} \tag{3.46}$$

where $p_{ee,l} = p_l - p_{ee}$ and $p_{ee,t} = p_t - p_{ee}$. $p_l \in \mathbb{R}^3$ is the position of arbitrary point on the line of sight, $p_{ee} \in \mathbb{R}^3$ is the position of the end-effector, $p_t \in \mathbb{R}^3$ is the position of the target point, $J_w \in \mathbb{R}^{3\times 11}$ is the rotational Jacobian matrix, $J_v \in \mathbb{R}^{3\times 11}$ is the translational Jacobian matrix, and λ is gain constant.

For analyzing the effectiveness of the WHQP, the combinations of the weighting matrices of the three comparative cases are presented in Fig. 3.2. For Phase 1, to check the contribution of the mobile base to the tracking task depending on the weighting matrix, the weighting matrix is set from the identity matrix (Case 1) to the mobile-dominant matrix (Case 3). In the case of Phase 2, to demonstrate the effectiveness of the individual weighting matrix, the weighting matrix of Case 1 is designed as the manipulator-dominant matrix for all tasks (one weighting matrix for all prioritized tasks). Case 2 assigns the identity matrix for all tasks (equivalent to pseudo-inverse method [12]). Case 3 uses individual weighting matrices for each task priority.

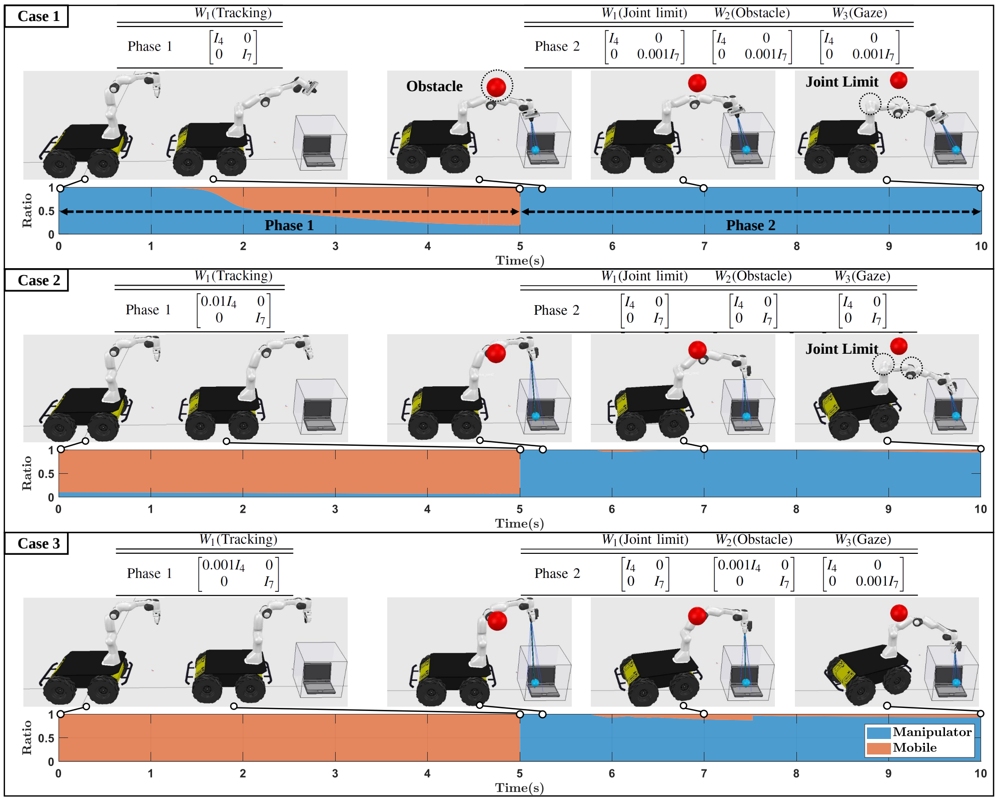


Figure 3.2: Experimental results with a nonholonomic mobile manipulator in simulation. A scenario composed of two phases is designed: The first phase is for the robot to approach a laptop. Then, the robot begins to inspect the laptop while avoiding the obstacle during the second phase.

3.5.1.3 Results

Figure 3.2 illustrates the snapshots of the simulation results and the ratio of the joint velocity norm between the mobile base, $\dot{q}_b \in \mathbb{R}^4$, and manipulator, $\dot{q}_m \in \mathbb{R}^7$, over time. The ratio is calculated as $\frac{\|\dot{q}_b\|_2^2}{\|\dot{q}\|_2^2}$ and $\frac{\|\dot{q}_m\|_2^2}{\|\dot{q}\|_2^2}$, where $\dot{q} = \left[\dot{q}_b^T, \dot{q}_m^T\right]^T$.

In Phase 1 of Case 1, the mobile manipulator began to track the trajectory only by using the manipulator and used both the mobile robot and manipulator together after the arm was stretched around 1.5 sec, which comes into the singular configuration. In practice, this phenomenon occurs frequently due to the difference in dynamics and performance between the mobile base and the manipulator [11, 52]. However, since the robot tracked the trajectory with the mobile base moving relatively more than the manipulator in Case 2 and only the mobile base moving in Case 3, the robot did not reach the singular posture.

After the end of Phase 1, the robot began to align the line from the end-effector with the blue ball over the laptop with the desired trajectory of the target point while avoiding the obstacle around 6 sec. In Case 1, the resultant posture reached the joint limit and singularity because all the tasks were executed by the manipulator. In Case 2, the configuration of the manipulator reached the joint limit and singularity even though the mobile robot helped to avoid the obstacle. This implies that the weighting matrices given in Case 2 did not properly distribute the movement between the mobile base and manipulator. In contrast, the gaze task was executed by the manipulator and the obstacle was avoided by the mobile robot in Case 3. Since the whole-body motion was properly distributed by assigning individual weighting matrices for each task priority, the robot could execute the tasks successfully without additional consideration of the necessary constraints.

Therefore, setting individually proper weighting matrices for given tasks is very effective for performing various and complex scenarios for the redundant robot.

3.5.2 Real experiment with nonholonomic mobile manipulator

3.5.2.1 Scenario description

In this section, a typical delivery scenario consisting of locomotion, manipulation, and loco-manipulation phases was designed with a real robot. The scenario has three phases. The first phase is for the robot to execute locomotion task for 15 sec. The second phase is to perform manipulation task of picking up a box from 15 sec to 42 sec. In the last phase of loco-manipulation, the robot delivers the box in front of the door from 42 sec to 50 sec.

3.5.2.2 Task and weighting matrix description

Each phase includes the same task hierarchies: a joint limit avoidance task (the first priority) and tracking task for the trajectory of the end-effector (the second priority). The joint limit avoidance task ($J_1 \in \mathbb{R}^{11\times 11}$ and $[\underline{\dot{x}}_1, \bar{\dot{x}}_1] \in \mathbb{R}^{11}$) is same as (3.44). The second priority task is to track the desired trajectory of the end-effector as

$$\begin{aligned} J_2\dot{q} &= \dot{x}_2, \\ J_2 &= \begin{bmatrix} J_v \\ J_w \end{bmatrix} \quad and \quad \dot{x}_3 = \begin{bmatrix} \dot{p}_{ee,d} \\ \delta\Phi \end{bmatrix}, \end{aligned} \tag{3.47}$$

where $J_2 \in \mathbb{R}^{6\times 11}$ is the whole-body Jacobian of the end-effector, $\dot{p}_{ee,d} \in \mathbb{R}^3$ is the desired linear velocity of the end-effector, and $\delta\Phi \in \mathbb{R}^3$ is the orientation error.

To generate various types of motion patterns, different weights are assigned

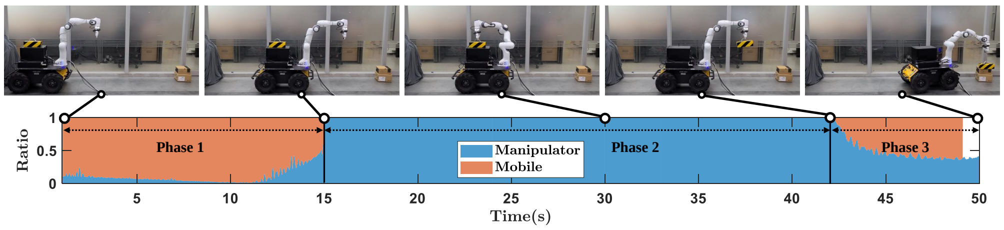


Figure 3.3: Experimental results with a nonholonomic mobile manipulator in delivery scenario.During Phase 1, the robot dominantly used the mobile robot to reach the target position. Then, the robot only exploited the manipulator to approach and pick up the box. For the last phase, the robot stacked the box on the other boxes by using the whole-body coordinately.

to the tracking task (J_2, $\dot{x}_2$) for each phase:

$$W_2 = \begin{bmatrix} 0.001I_4 & 0 \\ 0 & I_7 \end{bmatrix} \rightarrow \begin{bmatrix} I_4 & 0 \\ 0 & 0.001I_7 \end{bmatrix} \rightarrow \begin{bmatrix} I_4 & 0 \\ 0 & I_7 \end{bmatrix} \tag{3.48}$$

In the first phase, the weighting matrix is set such that the trajectory is dominantly tracked by the mobile robot. Then, the robot mainly exploits the manipulator to track the trajectory in the second phase. In the last phase, the weights are equally distributed to the manipulator and mobile base.

3.5.2.3 Results

As depicted in Fig. 3.3, from 0 to 15 sec, the robot tracked the desired locomotion trajectory through mobile dominant behavior[1]. Then, our WHQP-based controller generated the motions of the manipulator to accurately pick up a box during the manipulation phase. Finally, the robot succeeded in placing a box

[1]Although we set a weighting matrix for the movement of the mobile robot alone during the locomotion phase, the manipulator moved slightly. This is due to the low tracking performance of a mobile robot caused by the difference in control frequency between a mobile robot (10 Hz) and manipulator (1000 Hz).

on the stacked boxes using whole-body motion from 42 to 50 sec.

3.5.3 Real experiment with humanoid

3.5.3.1 Scenario description

Similar to Sec. 3.5.1 and 3.5.2, we validated the performance of the WHQP by implementing the inverse kinematics controller for a humanoid. A box-taping scenario was designed by using the 16-DOFs upper body of the humanoid for 15 sec.

3.5.3.2 Task and weighting matrix description

The humanoid executes two prioritized tasks. The first task is to maintain the initial position of the left hand as

$$\begin{aligned} &J_1\dot{q} = \dot{x}_1, \\ &J_1 = \begin{bmatrix} J_{left}, 0_{6\times 7}, J_{waist} \end{bmatrix} \quad and \quad \dot{x}_1 = \begin{bmatrix} \dot{p}_{left,d} \\ \delta\Phi_{left} \end{bmatrix}, \end{aligned} \tag{3.49}$$

where $J_1 \in \mathbb{R}^{6\times 16}$ is the Jacobian matrix for the left hand, $J_{left} \in \mathbb{R}^{6\times 7}$ is the Jacobian matrix from the left shoulder to the left hand, and $J_{waist} \in \mathbb{R}^{6\times 2}$ is the Jacobian matrix for the waist roll and yaw joint. $\dot{p}_{left,d} \in \mathbb{R}^3$ and $\delta\Phi_{left} \in \mathbb{R}^3$ are the desired linear velocity and orientation error for the left hand, respectively. Then, the relative motion task [53] between the right and left hand is assigned

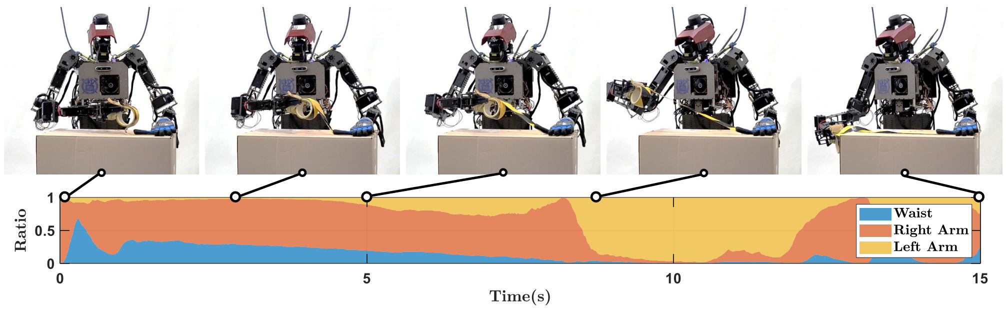


Figure 3.4: Experimental results with a humanoid in box-taping scenario

as the second priority:

$$J_2\dot{q} = \dot{x}_2,$$

$$J_2 = \begin{bmatrix} -J_{left}, J_{right}, J_{waist} \end{bmatrix} \quad and \quad \dot{x}_1 = \begin{bmatrix} \dot{p}_{rel,d} \\ \delta\Phi_{rel} \end{bmatrix}, \tag{3.50}$$

where $J_2 \in \mathbb{R}^{6\times16}$ is the Jacobian matrix for the relative motion, $J_{right} \in \mathbb{R}^{6\times7}$ is the Jacobian matrix from the right shoulder to the right hand. $\dot{p}_{rel,d} \in \mathbb{R}^3$ and $\delta\Phi_{rel} \in \mathbb{R}^3$ are the desired relative linear velocity and relative orientation error, respectively.

Since the joints for executing two prioritized tasks share waist and left arm joints, individual weighting matrices are assigned for each task, as follows:

$$W_1 = \begin{bmatrix} 0.01I_7 & & \\ & I_7 & \\ & & I_2 \end{bmatrix} and\ W_2 = \begin{bmatrix} I_7 & & \\ & 0.01I_7 & \\ & & 0.01I_2 \end{bmatrix} \tag{3.51}$$

The weighting matrix is set such that the left arm is predominantly used for the first priority task (W_1) and the waist and right arm are predominantly used for the relative motion task of the second priority (W_2).

3.5.3.3 Results

Figure 3.4 depicts the snapshots of box-taping scenarios and a motion ratio for the left-arm, right-arm, and waist. Thanks to the WHQP formulation with weighting matrices, the humanoid robot achieved dexterous manipulation through whole-body behavior. Consequently, the WHQP for a nonholonomic mobile manipulator and a humanoid presented in Sec. 3.5.1 and 3.5.2, respectively, can deal with complex real-world scenarios without any additional constraints or planners.

3.6 Discussions and implementation details

To validate the computational efficiency of the WHQP, the computation time for the equality tasks was compared with those of the two methods in Sec. 3.2.1: weighted pseudo-inverse (3.7) with (3.8a) and (3.7) with (3.8b). The total number of DOFs was set to 30, and the number of task hierarchies p ranged from 1 to 10. Also, the task dimension m_k was uniformly distributed depending on the number of task hierarchies, $\sum_{k=1}^{p} m_k \approx 30$. We randomly generated Jacobian matrices with full rank and task vectors and performed 500 calculations for each number of task priorities. Finally, an Intel Core i7 with a 16 GB RAM computer was used.

3.6.1 Computation cost

As shown in Fig. 3.5, the WHQP exhibits a low computation time even when the number of task hierarchies increases. This is because the null-space projec-

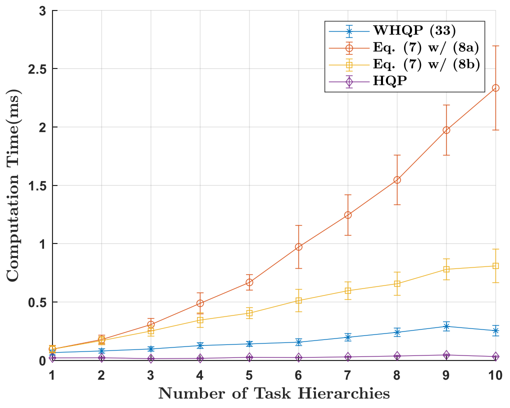


Figure 3.5: Computation time for the equality tasks with respect to the number of task hierarchies

tion matrix is decomposed into the product of the orthogonal bases. Thus, the size of the inverse matrix is reduced when recursively propagating to compute the solution of a low priority task. In contrast, the other two approaches have a higher computation time than that of the WHQP because the null-space projection matrix of the weighted pseudo-inverse is inherently asymmetric. Therefore, the size of the inverse matrix cannot be reduced.

On the other hand, the proposed WHQP has inherently higher computation cost than the HQP as shown in Fig. 3.5. When computing the total solution for k-th task in (3.35), the operation of the COD is performed $k(k+1)/2$ times, whereas the HQP performs k operations of the COD. This is because the null-space matrix Z_{k-1,W_k} is obtained by recursively decomposing from J_{1,W_k} to J_{k-1,W_k}.

3.6.2 Composite weighting matrix in same hierarchy

Since our formulation handles one weighting matrix at each level, a composite weighting matrix is required if there are two or more tasks at the same task level. In practice, we present a simple construction method using a linear combination [38].

$$W = a_1 W_1 + a_2 W_2 + \cdots + a_n W_n, \tag{3.52}$$

where $\sum_{i=1}^{n} a_i = 1$ and $0 \leq a_i \leq 1$. The magnitude of a_i indicates which weighting matrix is more dominant at the same level.

3.6.3 Nullity of WHQP

If the remaining nullity of the WHQP is 0[2], the result of the WHQP becomes the same as that of the original HQP solver. It is evident that a weighted Euclidean norm is equivalent to a normal Euclidean norm when the matrix is not redundant [54]. Therefore, to enhance the effectiveness of the weighting matrices of tasks in the WHQP, it is recommended to configure a set of the tasks where the nullity of the controller exists.

3.7 Conclusion

Whole-body controllers with hierarchical optimization have great advantages in controlling redundant robots, including mobile manipulators and humanoids. However, the motion generated by these controllers is often undesirable, without additional constraints. In this study, a novel weighted hierarchical quadratic

[2]This implies $\sum_{i=1}^{p} r_i = n$ when the number of DOFs of the robot is n.

programming method to assign joint weights for each task priority is proposed. The proposed method can be summarized as follows. First, because our algorithm treats a weighting matrix for each task, it can generate various whole-body behaviors depending on the scenarios, in comparison with previous studies. Next, our algorithm using the active-set method can efficiently handle inequality tasks as well as equality tasks. Finally, we demonstrated the effectiveness of the proposed controller through several experiments with a real mobile manipulator and humanoid. With these excellent results, we expect that our method can be applied to other highly redundant systems, such as aerial manipulators. Our future work will involve the extension of the proposed framework for automatically generating suitable weighing matrices for each task.

Chapter 4

WHOLE-BODY CONSTRAINT : SELF-COLLISION AVOIDANCE

In this chapter, we present a new self-collision avoidance algorithm for differentially-driven mobile manipulator. Our focus is on the avoidance of self-collision between the manipulator and the mobile robot. The goal is to generate a motion that the manipulator can avoid self-collision without modifying reference motion of the manipulator. To this end, we propose a concept of *distance buffer border* (DBB), a border of the buffer region that the manipulator can reach around the mobile robot. The region has the thickness of the buffer distance (See Fig. 4.1(a)). When the manipulator and the mobile robot are close to each other, in other words, their distance is less than the buffer distance (See Fig. 4.1(b)), the strategy is to position the manipulator outside the DBB by the motion of the mobile robot. This is realized by generating the force exerted on the mobile robot because the DBB is attached to the mobile robot and moves with it (See Fig. 4.1(c)). Therefore, the manipulator can avoid self-collision with the

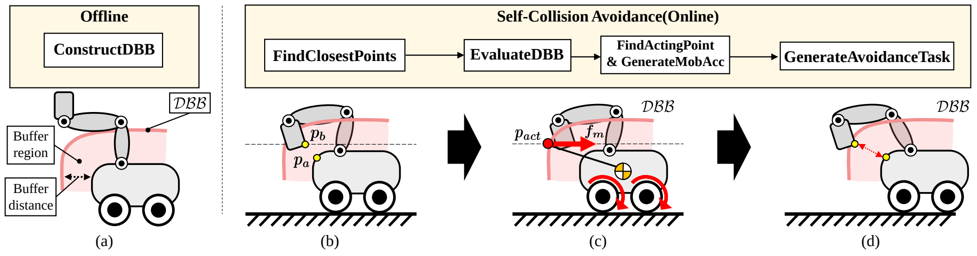


Figure 4.1: Overview of the proposed algorithm

mobile robot without modifying reference motion of the manipulator (See Fig. 4.1(d)).

Especially, the direction of the force is determined by considering the following two factors. First, the singularity of the differentially-driven mobile robot due to non-holonomic constraint is considered in order not to lose the controllability of the mobile robot. Second, we consider the reachability of the manipulator, a representation of the robot's workspace with the information of the pose quality. As the direction of the force is determined towards enhancing the reachability, the resultant configuration of the robot can secure the larger workspace of the manipulator.

To implement the proposed algorithm on the robot, the avoidance task is constructed by combining two types of motions depending on whether the link collides with the mobile robot or not. First, for the link pair including the mobile robot, the force and the resulting torque are generated and converted to the accelerations of the mobile robot. Second, for the link pair not including the mobile robot, 1-DOF acceleration is generated in a direction that the distance between the closest points of the link pair increases. Then, the task is constructed by stacking the two types of accelerations and their Jacobian

Table 4.1: Notation and symbols

Symbol	Description
$< l_a, l_b >$	a link pair of the link a and the link b
$\mathcal{L} = \mathcal{L}_m \cup \mathcal{L}_m^c$	a set of potentially colliding link pairs
$\mathcal{L}_m$	a subset of $\mathcal{L}$ including the mobile robot
$\mathcal{L}_m^c$	a complement set of $\mathcal{L}_m$
$\mathcal{L}(i) =< l_a, l_b >$	i-th link pair of $\mathcal{L}$
$\mathcal{DBB}_i$	distance buffer border of i-th link pair in $\mathcal{L}_m$
n	DOFs of the mobile manipulator
n_m	DOFs of the manipulator
$\mathcal{T}_j$	j-th equality or inequality task
$\mathcal{T}_j \prec \mathcal{T}_{j+1}$	$\mathcal{T}_j$ has higher priority than $\mathcal{T}_{j+1}$

matrix. The task is inserted continuously with the highest priority depending on the distances between the link pairs by using the controller based on Hierarchical Quadratic Programming (HQP) with the continuous task transition algorithm [55, 56].

The remainder of this chapter is organized as follows. First, Sec. 4.2 introduces the DBB and the computation of its score to design the force. Second, Sec. 4.3 explains our overall strategy for self-collision avoidance. Next, Sec. 4.4 describes the experimental validations of the proposed strategy. Finally, the paper is concluded in Sec. 4.5. To enhance readability of this paper, Table 4.1 denotes the symbols and their corresponding meanings. Also, bold roman letters denote vectors and matrices while normal roman letters denote real numbers.

4.1 Background & related Works

Self-collision can be avoided by an offline or online motion generation. Planning collision-free motion is normally implemented offline, whereas online motion generation is mainly embedded with the controllers. First, in the field of motion planning, Kuffner *et al.* [57] presented a motion planning algorithm to compute dynamically stable and collision-free trajectory for humanoid robots based on Rapidly-exploring Random Trees (RRT). Oriolo and Mongillo [58] proposed a randomized planner that resolves the redundancy of non-holonomic mobile manipulator. The planner allows the mobile robot to be located within a compatible circle for a given end-effector position so that the inverse kinematics solution of the manipulator exists. Regarding pose constraints on the end-effector, Berenson *et al.* [59] developed Constrained Bi-directional RRT (CBiRRT) which plans the trajectory by projecting sampled position onto Task Space Region (TSR). Burget *et al.* [60] proposed planning framework called Bi-directional Informed RRT*(BI^{2}RRT*) which can efficiently obtain optimal paths for mobile manipulation with the task space constraints. However, these methods are difficult to be implemented in the unstructured and dynamic environments because the trajectory might have to be regenerated in real time. Particularly, their computational cost increases for robots with a large number of DOFs such as humanoids or mobile manipulators.

To overcome these limitations, many reactive methods have been proposed in order to detect and avoid self-collision in real time. Our proposed method belongs to this category. Seto *et al.* [61, 62] designed the outer parts of links as elastic elements so that the reaction force is generated between elastic elements when the links are close to each other. In [63] and [64], the motion for self-collision avoidance was generated by the gradient of cost function related to the distances between links. Dariush *et al.* [65] penalized the motion of joints using the inverse matrix of weighted Jacobian based on the gradient of collision function. Fang *et al.* [50] presented the method to generate relative motion between the links using the inequality task. However, these methods are not applicable to non-holonomic mobile manipulators because the methods are developed for holonomic systems. Specifically, the methods may not instantaneously generate motion of the mobile robot in a certain direction because the non-holonomic constraint is not considered [66,67]. On the other hand, Dietrich *et al.* [68, 69] proposed the repulsive force-based approach with efficient damping design and extended continuous null space projection method. Sugiura *et al.* [70] proposed the method using only 1-DOF repulsive force while dynamically swapping priority of the tasks. While the repulsive force-based methods are conservative and effective solution for avoiding collisions, the methods repel two proximate links and thus the modified motion may be farther away from the reference motion more than necessary.

4.2 Distance buffer border and its score computation

This section introduces the concept of the DBB for generating the force which enables to avoid self-collision between the manipulator and the mobile robot. Also, we explain how to compute the score of the DBB in order to determine the direction and magnitude of the force. In Sec. 4.2.1, all link pairs which can potentially collide with each other are identified based on the collision model

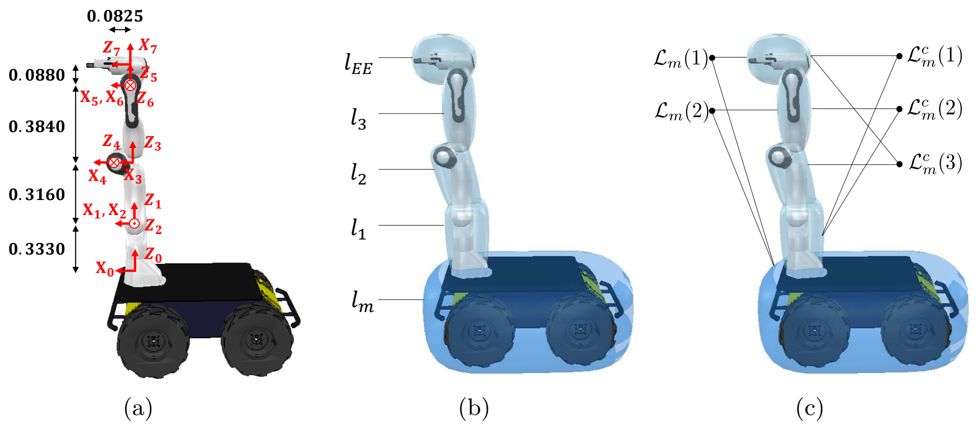


Figure 4.2: Our mobile manipulator system consists of a four-wheel differentially driven mobile robot called *Husky* (*Clearpath Robotics. Co.*) and 7-DOFs manipulator called *Panda* (*Franka Emika. Co.*). (a) Kinematic structure of the manipulator is shown with the scale of meter ; (b) The simplified collision models of the robot consist of five links; (c) Based on the collision models and joint range of the manipulator, all link pairs that potentially collide with each other are identified. $\mathcal{L}_m(i)$ denotes the link pair including the mobile robot, whereas $\mathcal{L}_m^c(j)$ denotes the link pair not including the mobile robot.

of the robot. Next, for the link pairs including the mobile robot, we define the DBB in Sec. 4.2.2. Finally, two factors are introduced to calculate the scores of the points on the DBB in Sec. 4.2.3.

4.2.1 Identification of potentially colliding link pairs

To decrease the computational cost for checking self-collision, simplified collision models are designed by using the convex shapes based on the kinematic structure of the manipulator. Fig. 4.2(a) and Fig. 4.2(b) show the kinematic structure of our robot and collision models of the robot, respectively. Utilizing these models and the joint position ranges of the manipulator, the link pairs which never collide with each other can be precomputed by randomly sampling the joint position of the manipulator. From this analysis, the link pairs potentially colliding with each other are identified. The set of the link pairs is defined as follows.

$$\mathcal{L} = \{< l_m, l_{EE} >, < l_m, l_3 >, < l_1, l_{EE} >, < l_1, l_3 >, < l_2, l_{EE} >\}, \tag{4.1}$$

where $l_\bullet$ denotes an individual link in Fig. 4.2(b). Because the avoidance motion is generated differently depending on whether or not the link pair includes the mobile robot l_m, the set $\mathcal{L}$ is divided into two subsets, namely $\mathcal{L}_m$ and its complement $\mathcal{L}_m^c$, and shown in Fig. 4.2(c) as follows.

$$\begin{aligned} \mathcal{L}_m &= \{< l_m, l_{EE} >, < l_m, l_3 >\} \\ \mathcal{L}_m^c &= \{< l_1, l_{EE} >, < l_1, l_3 >, < l_2, l_{EE} >\}. \end{aligned} \tag{4.2}$$

In (4.2), we denote each element of subset as $\mathcal{L}_m(i)$ and $\mathcal{L}_m^c(j)$ respectively. Therefore, to avoid self-collision, we generate a force on the mobile robot for $\mathcal{L}_m$ as discussed in Sec. 4.3.1 and 1-DOF repulsive acceleration for $\mathcal{L}_m^c$ in as discussed in Sec. 4.3.2.

4.2.2 Distance buffer border

Our avoidance strategy is to move the mobile robot in order to place the manipulator outside a region surrounding the mobile robot with a thickness equal to the buffer distance. To this end, we define a border of the region as the *distance buffer border*(DBB) of $\mathcal{L}_m$. Geometrically, the DBB represents a group of 3D points located away from the mobile robot l_m by the buffer distance.

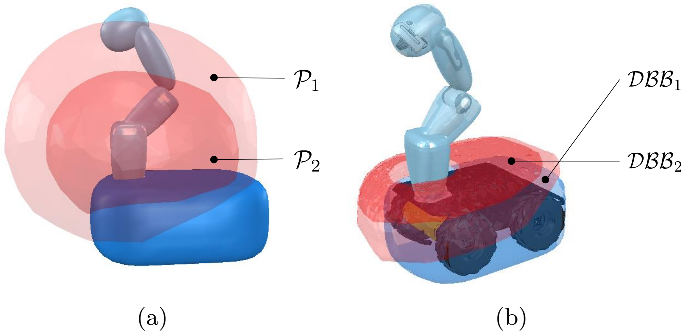


Figure 4.3: Visualization of $\mathcal{P}_i$ and $\mathcal{DBB}_i$ for $\mathcal{L}_m(i)$. (a) Red volume represents $\mathcal{P}_i$ which is a point set around the mobile robot; (b) Red hyperplanes represent the distance buffer borders of $\mathcal{L}_m$. The buffer distance d_b is set to 0.15m and the tolerance ϵ is set to 0.01m.

Algorithm 2 ConstructDBB

Input: $\mathcal{P}_i$: a set of points on the manipulator's link for $\mathcal{L}_m(i)$
Output: $\mathcal{DBB}_i$: distance buffer border of $\mathcal{L}_m(i)$

```
1: for each $\boldsymbol{p}_i$ in $\mathcal{P}_i$ do
2:     $d_m \leftarrow$ DistanceToMobile($\boldsymbol{p_i}, l_m$)
3:     if $\|d_m - d_b\| \leq \epsilon$ then
4:         Store $\boldsymbol{p_i}$ in $\mathcal{DBB}_i$
5:     end if
6: end for
```

Algorithm 2 describes the construction of the DBB in detail and is implemented offline. The input for the algorithm is the set of stored points on the manipulator's link for $\mathcal{L}_m$. For each link pair in $\mathcal{L}_m$, the two closest points are calculated after randomly sampling the joint positions of the manipulator. The point on the link of the manipulator is then stored to the set. This process repeats until the set contains a sufficient number of points. For the i-th link pair in $\mathcal{L}_m$, each set is denoted by $\mathcal{P}_i$ as shown in Fig. 4.3. The algorithm operates as follows.

First, for each point $\boldsymbol{p_i}$ of $\mathcal{P}_i$, the DistanceToMobile function calculates the distance between $\boldsymbol{p_i}$ and the mobile robot (see Line 2). Second, if the distance is within a bounded range, then the point $\boldsymbol{p_i}$ is stored in the DBB (see Line 3-4). After these procedures are repeated, the DBB is constructed as shown in Fig. 4.3(b) and defined as follows :

$$\begin{aligned} &\mathcal{DBB}_i \ni {}^{\forall}\boldsymbol{p_i} \\ \text{s. t.} \ \ &\|d_m - d_b\| \leq \epsilon, \boldsymbol{p_i} \in \mathcal{P}_i \end{aligned} \tag{4.3}$$

where $\mathcal{DBB}_i$ denotes the DBB for $\mathcal{L}_m(i)$, $\boldsymbol{p_i} \in \mathbb{R}^3$ is the position on the link of the manipulator, d_m is the minimum distance between $\boldsymbol{p_i}$ and l_m, d_b is buffer distance, and ϵ is tolerance value. To accomplish our strategy, we generate a force on the mobile robot based on the DBB. The direction of the force is defined to begin at a point on the DBB and head toward a point on the manipulator. The point on the DBB becomes the acting point of the force as shown in Fig. 4.1(c). In the following subsection, we propose a score for evaluating every point on the DBB to select the acting point.

4.2.3 Evaluation of distance buffer border

To select the point on the DBB that satisfies the desired capabilities of the force, the DBB is evaluated based on a score consisting of two factors: the singularity of the differentially driven mobile robot and the reachability of the manipulator.

4.2.3.1 Singularity of the differentially driven mobile robot

First, the singularity of the differentially driven mobile robot is considered to prevent the force from generating the motion of the non-holonomic mobile robot

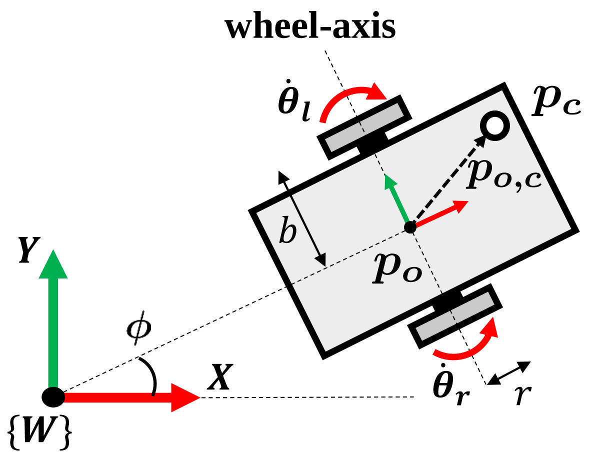


Figure 4.4: Schematic drawing of the differentially driven mobile robot. $\boldsymbol{p_c}$ is the control point of the mobile robot in the world frame $\boldsymbol{\{W\}}$ and $\boldsymbol{p_{o,c}}$ is the planar vector from the center of the mobile robot, $\boldsymbol{p_o}$, to $\boldsymbol{p_c}$ in $\boldsymbol{\{W\}}$. b and r are the distance between the wheel and center of the mobile robot and the radius of the wheel, respectively. $\dot{\boldsymbol{\theta}}_{\boldsymbol{r}}$ and $\dot{\boldsymbol{\theta}}_{\boldsymbol{l}}$ are the spinning velocities of the right and left wheel, respectively.

along the singular direction. Fig. 4.4 shows the kinematic modeling of two-wheel differentially driven mobile robot which simplifies that of four-wheel differentially driven mobile robot [71]. The differentially driven mobile robot is subject to a constraint in terms of the velocity as follows.

$$-\dot{x}_o sin(\phi) + \dot{y}_o cos(\phi) = 0 \tag{4.4}$$

where $\dot{x}_o$ and $\dot{y}_o$ are planar velocity of the center of the mobile robot and ϕ is the heading angle of the robot from the X-axis in the world frame as shown in Fig 4.4. Physically, (1) means that there is no velocity component parallel to the wheel-axis at the center of the differentially-driven mobile robot. The constraint is non-integrable, thus termed as non-holonomic constraint [72, 73].

The velocity relationship between the control point and the configuration of the differentially driven mobile robot is given by

$$\dot{\boldsymbol{p}}_{\boldsymbol{c}} = \boldsymbol{J}_{\boldsymbol{c}}(\boldsymbol{q}_{\boldsymbol{c}})\dot{\boldsymbol{q}}_{\boldsymbol{b}}, \tag{4.5}$$

where $\dot{\boldsymbol{p}}_{\boldsymbol{c}} = \begin{bmatrix} \dot{x}_c & \dot{y}_c \end{bmatrix}^T \in \mathbb{R}^2$ is the planar velocity of the control point of the mobile robot. $\dot{\boldsymbol{q}}_{\boldsymbol{b}} = \begin{bmatrix} \dot{\theta}_r & \dot{\theta}_l \end{bmatrix}^T \in \mathbb{R}^2$ is spinning velocity of the wheels and the subscripts r and l of $\dot{\theta}$ denote the right and left wheel, respectively. $\boldsymbol{J}_{\boldsymbol{c}}(\boldsymbol{q}_{\boldsymbol{c}}) \in \mathbb{R}^{2\times 2}$ is Jacobian matrix given by

$$\boldsymbol{J}_{\boldsymbol{c}}(\boldsymbol{q}_{\boldsymbol{c}}) = \begin{bmatrix} c(b\cos\phi - y_{o,c}) & c(b\cos\phi + y_{o,c}) \\ c(b\sin\phi + x_{o,c}) & c(b\sin\phi - x_{o,c}) \end{bmatrix}, \tag{4.6}$$

where $c = r/2b$, r is the radius of the wheel, b is the distance between the wheel and the center of the mobile robot, and $\boldsymbol{q}_{\boldsymbol{c}} = \begin{bmatrix} \boldsymbol{p}_{\boldsymbol{o,c}} & \phi \end{bmatrix}^T$. $\boldsymbol{p}_{\boldsymbol{o,c}} = \begin{bmatrix} x_{o,c} & y_{o,c} \end{bmatrix}$ are the coordinates of the control point from the center of the mobile robot in the global frame and ϕ is the orientation of the mobile robot.

To identify the singularity, we derive the determinant of the product of the Jacobian matrix $\boldsymbol{J}_{\boldsymbol{c}}(\boldsymbol{q}_{\boldsymbol{c}})$ as

$$det(\boldsymbol{J}_{\boldsymbol{c}}\boldsymbol{J}_{\boldsymbol{c}}^T) = 4b^2c^4(x_{o,c}\cos\phi + y_{o,c}\sin\phi)^2. \tag{4.7}$$

According to (4.7), the Jacobian matrix $\boldsymbol{J}_{\boldsymbol{c}}(\boldsymbol{q}_{\boldsymbol{c}})$ loses rank when

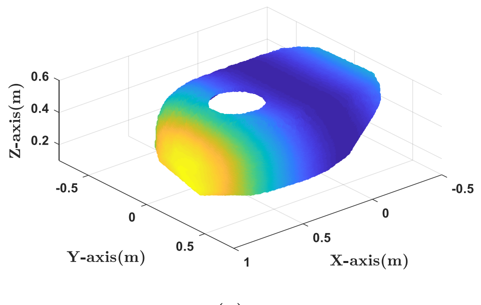
$$x_{o,c}\cos\phi + y_{o,c}\sin\phi = 0. \tag{4.8}$$

Geometrically, the left side of (4.8) represents the distance between the control point and the line of the wheel-axis. As the value of (4.8) tends to zero, meaning the control point is located on the wheel-axis, the control point of the differential-driven mobile robot cannot instantaneously move along the wheel-axis [74].

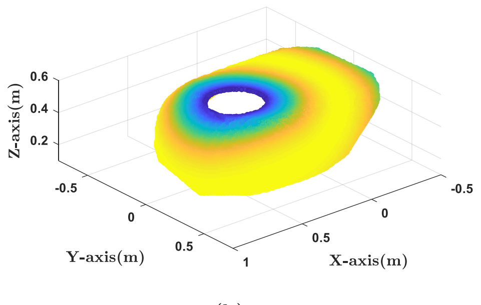
Thus, assuming that each point $\boldsymbol{p_i}$ of $\mathcal{DBB}_i$ is set to the control point of the mobile robot, we can measure how close it is to the singularity by setting $\boldsymbol{q_c} = \begin{bmatrix} x_i & y_i & 0 \end{bmatrix}^T$ where x_i and y_i are the coordinates along X-axis and Y-axis, respectively. For our robot, r is set to 0.165m and b is set to 0.51m. In Fig. 4.5(a), the points of $\mathcal{DBB}_1$ are evaluated by using (4.7). As shown in Fig. 4.5(a), the determinant value is symmetric about X-axis.

4.2.3.2 Reachability of the manipulator

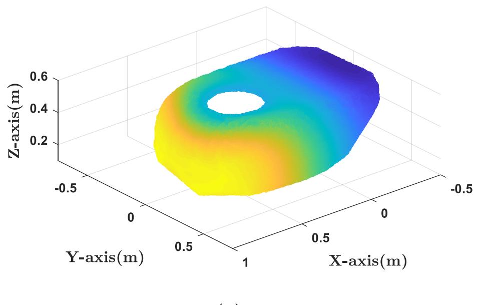
Second, the reachability of the manipulator is considered in order for the force to place the manipulator in the suitable workspace. The reachability is defined as the density of Inverse Kinematics(IK) solutions for the pose of the end-effector [23]. Reachability is computed by uniformly sampling the pose of the end-effector over the entire workspace and recording the number of IK solutions for each pose. The reachability of our robot is illustrated in Fig 4.6. One can see that the value of reachability increases and then decreases as the pose of the end-effector moves outwards from the base of the manipulator. Based on this observation, reachability can be expressed as a scalar concave function of the distance from the base of the manipulator. Among the various types of concave functions, the following second-order polynomial function was selected in this paper.



(a)



(b)



(c)

Figure 4.5: All the points in $\mathcal{DBB}_1$ are colored depending on (a) the determinant value in (4.7), (b) the reachability in (4.9), and the score in (4.10) (yellow: high, blue: low)

$$\mathcal{R}(\boldsymbol{p_i}) = -A(\|\boldsymbol{p_i} - \boldsymbol{p_{base}}\|_2 - B)^2 + C, \tag{4.9}$$

where $\mathcal{R}(\boldsymbol{p_i}) : \mathbb{R}^3 \to \mathbb{R}^+$ maps points on the DBB to reachability values, $\boldsymbol{p_{base}} \in \mathbb{R}^3$ is the position of the base of the manipulator, and A, B, C are positive coefficients of the polynomial. Based on the reachability data in Fig. 4.6, we set A to 525.9 /m^2, B to 0.575 m, and C to 100. Fig. 4.5(b) presents the reachability value for each point of $\mathcal{DBB}_1$. Although reachability is originally defined for the pose of the end-effector, the reachability of other link of the manipulator can also be obtained using kinematics information.

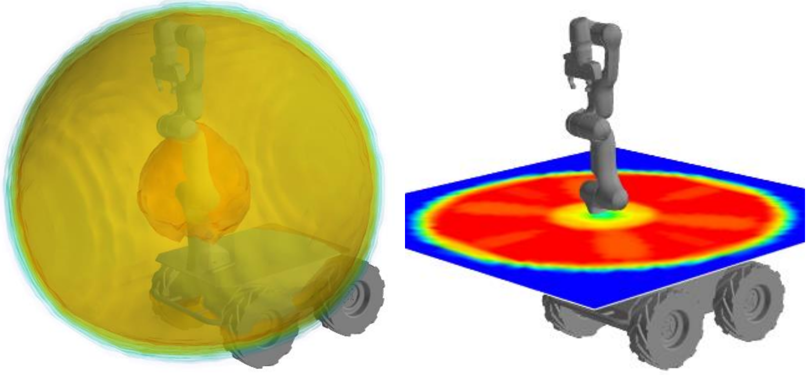

(a) (b)

Figure 4.6: Visualization of reachability shown in OPENRAVE [75]. (a) the contour of reachability of the end-effector; (b) the reachability cut by a horizontal plane at the base of the manipulator is colored (right, red: high, blue: low).

4.2.3.3 Score of the DBB

We compute a score for every point on the DBB denoted as $\mathcal{S}(\boldsymbol{p_i}) \in \mathbb{R}$. A score is expressed as

$$\mathcal{S}(\boldsymbol{p_i}) = sign(x_i)det(\boldsymbol{J_c}\boldsymbol{J_c}^T)\mathcal{R}(\boldsymbol{p_i}) \tag{4.10}$$

where

$$sign(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0. \end{cases} \tag{4.11}$$

Note that the function *sign* ensures that the DBB has a point of global maximum score as shown in Fig. 4.5(c).

Algorithm 3 Self-Collision Avoidance

```
Input: A set of link pairs $\mathcal{L} = \mathcal{L}_m \cup \mathcal{L}_m^c$; DBBs of the links $\mathcal{DBB}$
 1: while IsControl() do
 2:     UpdateKinematics($\boldsymbol{q}$)
    // Avoidance between mobile robot and manipulator
 3:     for each $\mathcal{L}_m(i)$ do
 4:         $(\boldsymbol{p_{a,i}}, \boldsymbol{p_{b,i}}) \leftarrow$ FindClosestPoints($\mathcal{L}_m(i)$)
    // $\boldsymbol{p_{a,i}}$ on mobile robot, $\boldsymbol{p_{b,i}}$ on manipulator
 5:         $\boldsymbol{p_{act,i}} \leftarrow$ FindActingPoint($\boldsymbol{q}, \boldsymbol{p_{b,i}}, \mathcal{DBB}_i$)
 6:         $\underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{m}} \leftarrow$ GenerateMobAcc($\boldsymbol{p_{act,i}}, \boldsymbol{p_{b,i}}$)
 7:     end for
    // Avoidance between links of manipulator
 8:     for each $\mathcal{L}_m^c(j)$ do
 9:         $(\boldsymbol{p_{a,j}}, \boldsymbol{p_{b,j}}) \leftarrow$ FindClosestPoints($\mathcal{L}_m^c(j)$)
10:         $\underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{r}} \leftarrow$ GenerateRepAcc($\boldsymbol{p_{a,j}}, \boldsymbol{p_{b,j}}$)
11:     end for
    // Insert the task continuously to the controller
12:     if $\|\underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{m}}\|_2 > 0$ or $\|\underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{r}}\|_2 > 0$ then
13:         $\mathcal{T}_{sca} \leftarrow$ GenerateAvoidanceTask($\underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{m}}, \underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{r}}$)
14:         $SoT \leftarrow$ UpdateSoT($\mathcal{T}_{sca}$)
15:         $\boldsymbol{u} \leftarrow$ HQPSolver($SoT$) // See (4.29)
16:     else
17:         $\boldsymbol{u} \leftarrow$ HQPSolver($SoT$) // See (4.26)
18:     end if
19: end while
```

4.3 Self-collision avoidance algorithm

In this section, we explain how to avoid self-collision for the differentially driven mobile manipulator. Algorithm 3 details the procedure. First, the FindClosest-Points function calculates the closest pair of points for each link pair. The link pair in $\mathcal{L}_m$ then generates the acceleration of the mobile robot, whereas the acceleration of the manipulator is generated for $\mathcal{L}_m^c$. For the subset $\mathcal{L}_m$, the FindActingPoint function determines the acting point of the force based on the computation for the score of the DBB. Next, the GenerateMobAcc function generates the force and the resulting torque exerted on the mobile robot and converts them to the linear and angular acceleration of the mobile robot as $\underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{m}} \in \mathbb{R}^2$ (see Line 5-6 and Sec. 4.3.1). On the other hand, for the subset $\mathcal{L}_m^c$, the GenerateRepAcc function generates a 1-DOF repulsive acceleration which pushes the two proximal links of the manipulator away from each other and stacks the accelerations for k link pairs of $\mathcal{L}_m^c$ as $\underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{r}} \in \mathbb{R}^k$ (see Line 10 and Sec. 4.3.2). Then, the GenerateAvoidanceTask function combines these accelerations and constructs the task for avoiding self-collision, $\mathcal{T}_{sca}$, as an equality task (see Line 13 and Sec. 4.3.3). Finally, the UpdateSoT function inserts the task $\mathcal{T}_{sca}$ as a top priority in the original SoT by using the continuous task transition scheme, as summarized in Line 14-15 and Sec. 4.3.4. In the following subsections, each function in the Algorithm 3 is described in detail.

4.3.1 Generation of the acceleration for the mobile robot

This subsection describes the generation of the force. The direction of the force is designed to start from the selected point on the DBB and heads to the closest point on the manipulator. Thus, we focus on how to select the acting point of the force that satisfies the following requirements.

First, the acting point should be located with the same height of the closest point on the manipulator because the force should act on a horizontal plane to move the mobile robot. Second, the acting point should be selected so that the force has two orthogonal components that play different roles. As shown in Fig. 4.7(a), the direction of the force can be decomposed into two orthogonal directions. One is the direction of the line connecting the closest point on the manipulator and the DBB. The other is its normal direction. The former increases the distance between the mobile robot and the manipulator as the

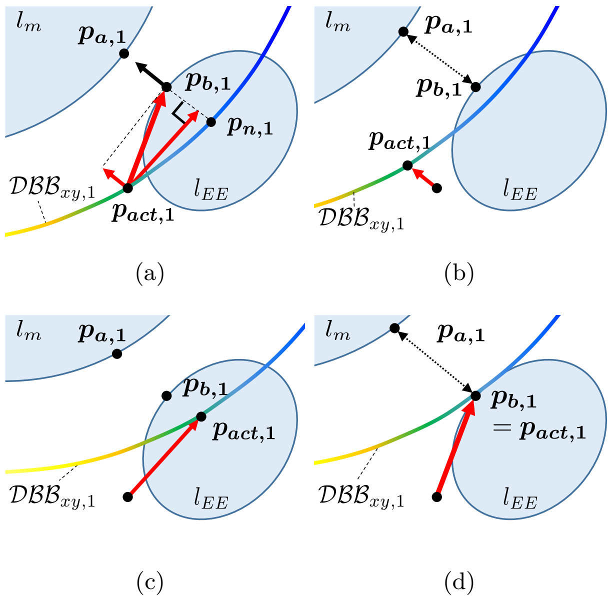


Figure 4.7: Illustration of the direction of force considering collision between the end-effector l_{EE} and mobile robot l_m. $\mathcal{DBB}_{xy,1}$ denotes the points whose heights are same as that of the closest point $\boldsymbol{p_{b,1}}$ on the end-effector. The points are colored by the score(yellow: high, blue: low). (a) The direction of the force starts from the acting point $\boldsymbol{p_{act,1}}$ and points toward $\boldsymbol{p_{b,1}}$. The force can be decomposed into two orthogonal components; (b) One of them moves the mobile robot away from the end-effector; (c) The other direction places the high-score part of DBB closer to the end-effector; (d) By combining these components, self-collision between the end-effector and mobile robot can be avoided.

DBB moves closer to the manipulator as shown in Fig. 4.7(b). On the other hand, as shown in Fig. 4.7(c), the latter places the manipulator closer to the DBB with high score in order to avoid selecting the acting point i.e. the control point of the mobile robot near the singularity and enhance the reachability of the manipulator. Combining two orthogonal components, self-collision between the manipulator and the mobile robot can be avoided in Fig. 4.7(d).

The FindActingPoint function (see Algorithm 4 and Fig. 4.8) finds the acting point that satisfies these requirements. Algorithm 4 operates as follows.

First, the TransformToWorld function transforms the points of $\mathcal{DBB}_i$ to be expressed in the world frame. Next, the FindSameHeight function finds the points in $\mathcal{DBB}_i$ whose height are same as that of the closest point $\boldsymbol{p_{b,i}}$ on the manipulator, which satisfies the first requirement (see Line 1-3). The obtained points are denoted as $\mathcal{DBB}_{xy,i}$. Then, for the second requirement, we calculate the acting point on $\mathcal{DBB}_{xy,i}$ that the generated force can have two orthogonal components. Among the points in $\mathcal{DBB}_{xy,i}$, the point $\boldsymbol{p_{n,i}}$ closest to the point $\boldsymbol{p_{b,i}}$ is identified (see Line 4). Starting at $\boldsymbol{p_{n,i}}$, the position of point $\boldsymbol{p}_{t,i}$ translated along the tangential direction $\boldsymbol{t_i}$ with a step size α_i as follows (see Line 5 and Fig. 4.8(a)):

$$\begin{gathered}\boldsymbol{p_{t,i}} = \boldsymbol{p_{n,i}} + \alpha_i \left(\frac{\nabla \mathcal{S}(\boldsymbol{p_{n,i}}) \cdot \boldsymbol{t_i}}{\|\nabla \mathcal{S}(\boldsymbol{p_{n,i}})\|} \right) \boldsymbol{t_i}, \\ \boldsymbol{t_i} \perp (\boldsymbol{p_{n,i}} - \boldsymbol{p_{b,i}}), \|\boldsymbol{t_i}\|_2 = 1,\end{gathered} \tag{4.12}$$

where the inner product of $\nabla \mathcal{S}(\boldsymbol{p_{n,i}})$ and $\boldsymbol{t_i}$ determines the direction of $\boldsymbol{t_i}$ to the higher score of the DBB. The step size α_i is calculated as

$$\alpha_i \propto \frac{d_i}{|\mathcal{S}(\boldsymbol{p_{n,i}})|}. \tag{4.13}$$

where d_i is the distance of the i-th link pair of $\mathcal{L}_m$. Therefore, the acting point $\boldsymbol{p_{act,i}}$ is calculated as that with the shortest distance from $\boldsymbol{p_{t,i}}$ to $\mathcal{DBB}_{xy,i}$ (see Line 6). In (4.13), as the distance between the manipulator and mobile robot decreases, the step size decreases to generate both orthogonal directions of the force. However, a larger step size can be used to proceed more rapidly toward a higher score of the DBB when the distance increases. Additionally, the step size increases as the absolute value of the score of point $\boldsymbol{p_{n,i}}$ decreases to zero,

```
Algorithm 4 FindActingPoint($\boldsymbol{q}, \boldsymbol{p_{b,i}}, \mathcal{DBB}_i$)
1: $z_{b,i} \leftarrow$ height of $\boldsymbol{p_{b,i}}$
2: $\mathcal{DBB}_i \leftarrow$ TransformToWorld($\boldsymbol{q}, \mathcal{DBB}_i$)
3: $\mathcal{DBB}_{xy,i} \leftarrow$ FindSameHeight($z_{b,i}$, $\mathcal{DBB}_i$)
4: $\boldsymbol{p_{n,i}} \leftarrow$ FindMinDistancePoint($\mathcal{DBB}_{xy,i}$, $\boldsymbol{p_{b,i}}$)
5: $\boldsymbol{p_{t,i}} \leftarrow$ Equation (4.12) and (4.13)
6: $\boldsymbol{p_{act,i}} \leftarrow$ FindMinDistancePoint($\mathcal{DBB}_{xy,i}$, $\boldsymbol{p_{t,i}}$)
7: return $\boldsymbol{p_{act,i}}$
```

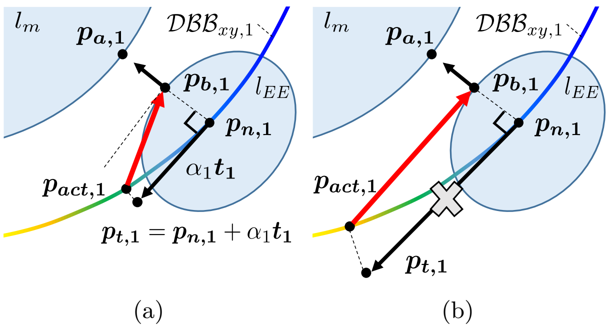


Figure 4.8: Illustration of finding the acting point. (a) The acting point on the DBB is selected as the closest to the point $\boldsymbol{p_{n,1}}$ which is translated along the tangential direction $\boldsymbol{t_1}$ with a step size α_1; (b) With a large step size, the generated force may not have a component along the direction connecting the closest point on the manipulator and DBB.

indicating that the point $\boldsymbol{p_{n,i}}$ is near the singularity. To prevent obtaining a step size that is too small or too large as shown in Fig. 4.8(b), the step size is bounded by lower and upper limits.

After finding the acting point, the GenereateMobAcc function first computes the force as follows:

$$\boldsymbol{f_{m,i}} = f_{max}(1 - \frac{d_i}{d_b}) \frac{\boldsymbol{p_{b,i}} - \boldsymbol{p_{act,i}}}{\|\boldsymbol{p_{b,i}} - \boldsymbol{p_{act,i}}\|_2}, \tag{4.14}$$

where $\boldsymbol{f_{m,i}} \in \mathbb{R}^3$ is the force for the i-th link pair and f_{max} is the maximum force. Fig. 4.9 presents the variables in (4.14) when the link pair $\mathcal{L}_m(1)$ is

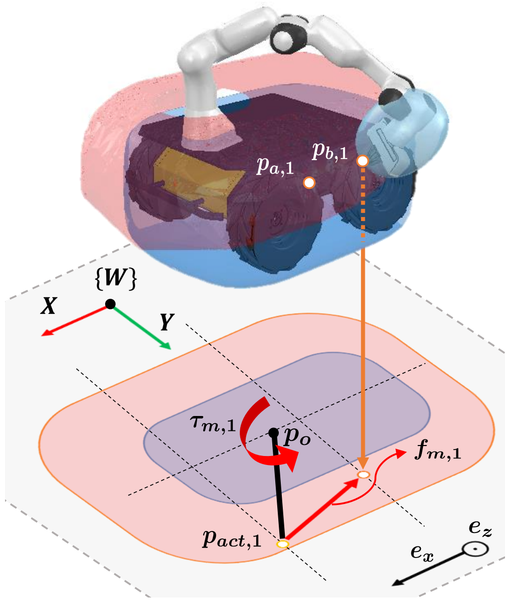


Figure 4.9: Illustration of generating the desired force and resulting torque for the link pair $\mathcal{L}_m(1)$. When the end-effector and mobile robot are close to each other, the force $\boldsymbol{f_{m,1}}$ and resulting torque $\boldsymbol{\tau_{m,1}}$ with respect to the center of the mobile robot, $\boldsymbol{p_o}$, are generated. The generated force and resulting torque are projected onto the acceleration directions of the mobile robot under the non-holonomic constraint. The directions are expressed as the unit vectors, $\boldsymbol{e_x}$ and $\boldsymbol{e_z}$.

considered.

The resultant force for all link pairs in $\mathcal{L}_m$ is calculated by adding each force as follows:

$$\boldsymbol{f_m} = \sum_{i=1}^{N(\mathcal{L}_m)} \boldsymbol{f_{m,i}}, \tag{4.15}$$

where $\boldsymbol{f_m} \in \mathbb{R}^3$ and $N(\mathcal{L}_m)$ are the resultant force and number of link pairs in $\mathcal{L}_m$, respectively.

To realize the resultant force, the corresponding accelerations and corresponding Jacobian matrices are derived as follows:

$$\underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{m}} = \begin{bmatrix} \dot{v}_d & \dot{w}_d \end{bmatrix}^T, \tag{4.16}$$

where

$$m\dot{v}_d = \boldsymbol{f_m} \cdot \boldsymbol{e_x}, \tag{4.17}$$

$$I\dot{w}_d = (\sum_{i=1}^{N(\mathcal{L}_m)} (\boldsymbol{p_{act,i}} - \boldsymbol{p_o}) \times \boldsymbol{f_{m,i}}) \cdot \boldsymbol{e_z}. \tag{4.18}$$

In (4.17) and (4.18), $m \in \mathbb{R}$, $\dot{v}_d \in \mathbb{R}$, and $\boldsymbol{e_x} \in \mathbb{R}^3$ are the mass of the mobile robot, desired linear acceleration, and a unit vector perpendicular to the rolling axis of the wheel and pointing forward, respectively. In addition, $I \in \mathbb{R}$, $\dot{w}_d \in \mathbb{R}$, and $\boldsymbol{e_z} \in \mathbb{R}^3$ are the moment of inertia, desired angular acceleration, and a unit vector perpendicular to the ground and pointing upward, respectively. By (4.17) and (4.18), the resultant force can be converted into the desired linear and angular accelerations. The Jacobian matrix of the differentially driven mobile robot can be expressed as

$$\begin{aligned} \underline{\boldsymbol{J}}_{\boldsymbol{m}} &= \left[\begin{array}{c|c} \boldsymbol{J_m} & \boldsymbol{O_{2\times n_m}} \end{array}\right], \\ \boldsymbol{J_m} &= \begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ \frac{r}{2b} & -\frac{r}{2b} \end{bmatrix}, \end{aligned} \tag{4.19}$$

where $\boldsymbol{J_m} \in \mathbb{R}^{2\times2}$ and $\boldsymbol{O_{2\times n_m}} \in \mathbb{R}^{2\times n_m}$ are the Jacobian matrix of the mobile robot and the zero matrix, respectively.

4.3.2 Generation of the repulsive acceleration for the other link pair

To avoid the self-collision of $\mathcal{L}^c_m$, we design a 1-DOF repulsive acceleration to push the link pair away from each other.

Let's consider that the distance of the j-th link pair in $\mathcal{L}^c_m$ is less than the buffer distance. The task for avoiding self-collision with the repulsive acceleration $\ddot{\boldsymbol{x}}_{\boldsymbol{r},\boldsymbol{j}} \in \mathbb{R}^1$ and Jacobian $\boldsymbol{J}_{\boldsymbol{r},\boldsymbol{j}} \in \mathbb{R}^{1\times n}$ is designed as follows.

$$\ddot{\boldsymbol{x}}_{\boldsymbol{r},\boldsymbol{j}} = \boldsymbol{u}_{\boldsymbol{j}}^T (k_p \frac{\boldsymbol{p}_{\boldsymbol{b},\boldsymbol{j}} - \boldsymbol{p}_{\boldsymbol{a},\boldsymbol{j}}}{\|\boldsymbol{p}_{\boldsymbol{b},\boldsymbol{j}} - \boldsymbol{p}_{\boldsymbol{a},\boldsymbol{j}}\|_2} - k_v(\dot{\boldsymbol{p}}_{\boldsymbol{b},\boldsymbol{j}} - \dot{\boldsymbol{p}}_{\boldsymbol{a},\boldsymbol{j}}), \tag{4.20}$$

$$\begin{aligned} \boldsymbol{J}_{\boldsymbol{r},\boldsymbol{j}} &= \boldsymbol{u}_{\boldsymbol{j}}^T (\boldsymbol{J}_{\boldsymbol{b},\boldsymbol{j}} - \boldsymbol{J}_{\boldsymbol{a},\boldsymbol{j}}), \\ \boldsymbol{u}_{\boldsymbol{j}} &= \frac{\boldsymbol{p}_{\boldsymbol{b},\boldsymbol{j}} - \boldsymbol{p}_{\boldsymbol{a},\boldsymbol{j}}}{\|\boldsymbol{p}_{\boldsymbol{b},\boldsymbol{j}} - \boldsymbol{p}_{\boldsymbol{a},\boldsymbol{j}}\|_2} \end{aligned} \tag{4.21}$$

where k_p and k_v are gains, $\boldsymbol{u}_{\boldsymbol{j}} \in \mathbb{R}^3$ is the unit vector from $\boldsymbol{p}_{\boldsymbol{a},\boldsymbol{j}}$ to $\boldsymbol{p}_{\boldsymbol{b},\boldsymbol{j}}$, and $\boldsymbol{J}_{\boldsymbol{a},\boldsymbol{j}}$ and $\boldsymbol{J}_{\boldsymbol{b},\boldsymbol{j}} \in \mathbb{R}^{3\times n}$ are translation Jacobian matrices for points $\boldsymbol{p}_{\boldsymbol{a},\boldsymbol{j}}$ and $\boldsymbol{p}_{\boldsymbol{b},\boldsymbol{j}}$, respectively. For convenience, we define the link to which $\boldsymbol{p}_{\boldsymbol{b},\boldsymbol{j}}$ belongs as being farther from the base of the manipulator than the link to which $\boldsymbol{p}_{\boldsymbol{a},\boldsymbol{j}}$ belongs.

When k link pairs in $\mathcal{L}^c_m$ are considered, the repulsive acceleration and Jacobian matrix are stacked as

$$\underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{r}} = \left[\ddot{\boldsymbol{x}}_{\boldsymbol{r},\boldsymbol{1}}, \cdots, \ddot{\boldsymbol{x}}_{\boldsymbol{r},\boldsymbol{j}}, \cdots, \ddot{\boldsymbol{x}}_{\boldsymbol{r},\boldsymbol{k}},\right]^T \tag{4.22}$$

$$\underline{\boldsymbol{J}}_{\boldsymbol{r}} = \left[\boldsymbol{J}^T_{\boldsymbol{r},\boldsymbol{1}}, \cdots, \boldsymbol{J}^T_{\boldsymbol{r},\boldsymbol{j}}, \cdots, \boldsymbol{J}^T_{\boldsymbol{r},\boldsymbol{k}}\right]^T, \tag{4.23}$$

where $\underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{r}} \in \mathbb{R}^k$ and $\underline{\boldsymbol{J}}_{\boldsymbol{r}} \in \mathbb{R}^{k\times n}$ are the stacked accelerations and Jacobian, respectively.

4.3.3 Construction of an acceleration-based task for self-collision avoidance

Based on the obtained accelerations and Jacobians in 4.3.1 and 4.3.2, we construct a task, $\mathcal{T}_{sca}$, for avoiding self-collision of all link pairs by stacking them as follows.

$$\ddot{\boldsymbol{x}}_{\boldsymbol{sca}} = \begin{bmatrix} \underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{m}} \\ \underline{\ddot{\boldsymbol{x}}}_{\boldsymbol{r}} \end{bmatrix}, \tag{4.24}$$

$$\boldsymbol{J}_{\boldsymbol{sca}} = \begin{bmatrix} \underline{\boldsymbol{J}}_{\boldsymbol{m}} \\ \underline{\boldsymbol{J}}_{\boldsymbol{r}} \end{bmatrix}, \tag{4.25}$$

where $\ddot{\boldsymbol{x}}_{\boldsymbol{sca}} \in \mathbb{R}^{(2+k)}$ is the desired acceleration for the avoidance task and $\boldsymbol{J}_{\boldsymbol{sca}} \in \mathbb{R}^{(2+k)\times n}$ is the corresponding Jacobian matrix.

4.3.4 Insertion of the task in HQP-based controller

To insert the designed task, $\mathcal{T}_{sca}$, a controller is designed based on the HQP with the continuous task transition approach developed in our previous work [55], [56]. HQP is a cascade of QP formulation for dealing with prioritized SoT [12, 76]. The main characteristic of the controller with the continuous task transition method is that the continuity of control inputs is always guaranteed even if arbitrary tasks are inserted and removed from the existing SoT. In particular, by using an activation parameter that interpolates feasible solution

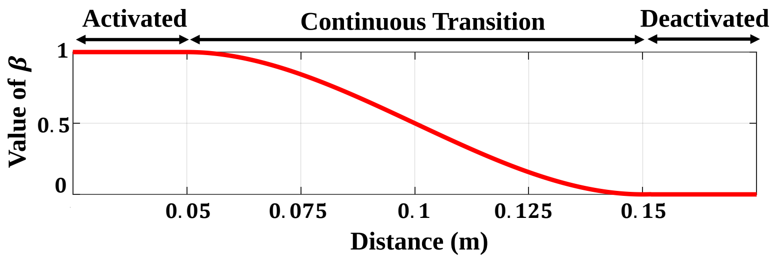


Figure 4.10: Value of the activation parameter depending on the distance of the link pair.

between existing SoT and new SoT, the method can generate continuous control inputs without modifying the control structure.

We consider the HQP formulation of a single equality task, $\mathcal{T}_2$, with $\ddot{\boldsymbol{x}}_{\boldsymbol{d_2}} \in \mathbb{R}^{m_2}$ and $\boldsymbol{J_2} \in \mathbb{R}^{m_2 \times n}$, as follow:

$$\begin{aligned} \min_{\ddot{\boldsymbol{q}},\boldsymbol{u},\boldsymbol{w_2}} \quad & \|\boldsymbol{w_2}\|_2, \\ \text{s. t.} \quad & \boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{g} = \boldsymbol{u} \\ & \boldsymbol{J_2}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_{\boldsymbol{2}}\dot{\boldsymbol{q}} + \boldsymbol{w_2} = \ddot{\boldsymbol{x}}_{\boldsymbol{d_2}} \end{aligned} \tag{4.26}$$

where $\boldsymbol{M} \in \mathbb{R}^{n\times n}$, $\boldsymbol{C} \in \mathbb{R}^{n\times n}$, $\boldsymbol{g} \in \mathbb{R}^{n}$, and $\dot{\boldsymbol{q}} = \begin{bmatrix} \dot{\boldsymbol{q}}_{\boldsymbol{b}}^T & \dot{\boldsymbol{q}}_{\boldsymbol{m}}^T \end{bmatrix}^T \in \mathbb{R}^n$ are the inertia matrix, Coriolis and centrifugal matrix, gravity vector, and joint velocity vector of the non-holonomic mobile manipulator, respectively [77]. In addition, $\boldsymbol{w_2} \in \mathbb{R}^{m_2}$ is a slack variable for $\mathcal{T}_2$ and $\boldsymbol{u} \in \mathbb{R}^n$ is the control torque vector for the robot.

The activation parameter, β, is determined based on the distance between each link pair. Fig. 4.10 presents the activation trajectory when using a cubic spline to insert $\mathcal{T}_{sca}$ smoothly. When the distance is less than the buffer distance of 0.15m, β gradually increases, and the avoidance task begins to be inserted.

In addition, if the distance is less than 0.05m, β is set to 1 so that the task for avoiding self-collision is fully considered. Because the avoidance tasks for $\mathcal{L}_m$ and $\mathcal{L}_m^c$ are stacked according to (4.24), we construct a diagonal matrix $\boldsymbol{B}$ from the activation parameters as follows.

$$\boldsymbol{B} = \begin{bmatrix} \beta_m & 0 & 0 & \cdots & 0 \\ 0 & \beta_m & 0 & \cdots & 0 \\ 0 & 0 & \beta_{r,1} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 0 & \beta_{r,k} \end{bmatrix}, \tag{4.27}$$

where $\boldsymbol{B} \in \mathbb{R}^{(2+k)\times(2+k)}$ is the diagonal matrix of the activation parameters, β_m is the activation parameter for the link pairs of $\mathcal{L}_m$, and $\beta_{r,j}$ is the activation parameter for the j-th link pair in $\mathcal{L}_m^c$. When considering multiple link pairs of $\mathcal{L}_m$, we choose the maximum value among the activation parameters as

$$\beta_m = \max\left(\beta_1, \cdots, \beta_{N(\mathcal{L}_m)}\right). \tag{4.28}$$

Based on the activation parameter matrix $\boldsymbol{B}$, the HQP formulation for inserting the self-collision avoidance task as the higher-priority task than $\mathcal{T}_2$ ($\mathcal{T}_{sca} \prec \mathcal{T}_2$), can be represented as

$$\begin{aligned}
\min_{\ddot{\boldsymbol{q}},\boldsymbol{u},\boldsymbol{w_2}} \quad & \|\boldsymbol{w_2}\|_2, \\
\text{s. t.} \quad & \boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{g} = \boldsymbol{u} \\
& \boldsymbol{J_2}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_{\boldsymbol{2}}\dot{\boldsymbol{q}} + \boldsymbol{w_2} = \ddot{\boldsymbol{x}}_{\boldsymbol{d_2}} \\
& \boldsymbol{J_{sca}}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_{\boldsymbol{sca}}\dot{\boldsymbol{q}} + (\boldsymbol{I_{2+k}} - \boldsymbol{B})\boldsymbol{J_{sca}}\ddot{\boldsymbol{q}}^*_{\boldsymbol{2}} + \boldsymbol{w}^*_{\boldsymbol{sca}} = \boldsymbol{B}\ddot{\boldsymbol{x}}_{\boldsymbol{sca}}
\end{aligned} \tag{4.29}$$

where $\boldsymbol{w}^*_{\boldsymbol{sca}} \in \mathbb{R}^{2+k}$ is the optimal slack variable vector for the self-collision avoidance task $\mathcal{T}_{sca}$, $\boldsymbol{I_{2\times k}} \in \mathbb{R}^{(2+k)\times(2+k)}$ is an identity matrix, and $\ddot{\boldsymbol{q}}^*_{\boldsymbol{2}}$ is the optimal solution of (4.26). Thus, if $\boldsymbol{B}$ is a zero matrix, then the feasible solution of (4.29) is the same as that of (4.26). When $\boldsymbol{B}$ is the identity matrix, the solution of (4.29) is strictly satisfied with the priority order, $\mathcal{T}_{sca} \prec \mathcal{T}_2$. In addition, when β gradually increases 0 to 1, the feasible solution of (4.29) can be derived by interpolating the solution of the HQP of $\mathcal{T}_2$ and the HQP with $\mathcal{T}_{sca} \prec \mathcal{T}_2$. Consequently, the HQP-based controller with the continuous task transition method can insert a self-collision avoidance task without a discontinuous control input.

4.4 Experimental results

The self-collision avoidance algorithm was verified through various experiments using a differentially driven mobile robot with a 7-DOFs robotic manipulator. The subsections below describe the details of our system configuration and the experimental results for the robot. It is worthwhile to note that the video clips of the experiments described in this paper are available in [78].

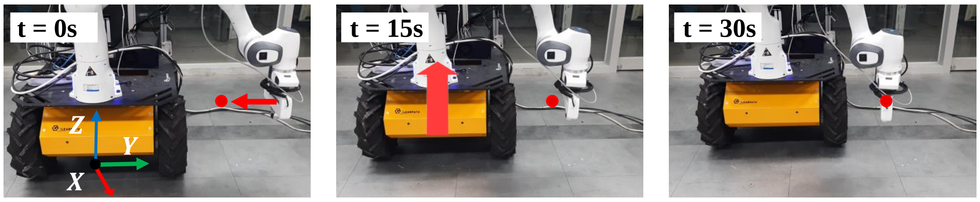
4.4.1 System overview

Our mobile manipulator consists of the velocity-controlled four-wheel differentially driven mobile robot called *Husky* (*Clearpath Robotics. Co.*) and a 7-DOFs robot arm manipulator called *Panda* (*Franka Emika. Co.*). The specification of the computer for the controller is an Intel *i*7 4.2 GHz CPU with 16 GB of RAM and the control frequencies of the manipulator and mobile robot are 1 kHz and 10 Hz, respectively. The desired velocity command for the mobile robot is computed from the desired torque calculated in (4.29) by applying the admittance control law [79].

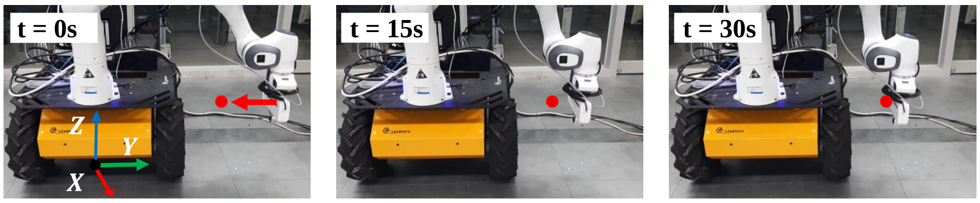
4.4.2 Experimental results

4.4.2.1 Self-collision avoidance while tracking the predefined trajectory

To validate the effectiveness of the proposed method, we conducted a comparative experiment using the repulsive force-based method [68, 69]. This experiment was designed for the end-effector to track a predefined trajectory that approaches the mobile robot. The task for trajectory tracking of the end-effector is denoted as $\mathcal{T}_{ee} \in \mathbb{R}^6$ and the task for the repulsive force-based method is denoted as $\mathcal{T}_{rep} \in \mathbb{R}^3$. The target position is -0.2 m along the Y-axis from the end-effector. The left snapshots in Fig. 4.11(a) and (b) show the initial positions. The red dots and arrows depict the target position and desired trajectory, respectively. The trajectory was generated for a time period of 30 s using a cubic spline function.



(a)



(b)

Figure 4.11: Snapshots during experiments in which the end-effector tracks a predefined trajectory: (a) The proposed method generates force to move the mobile robot back, enabling the manipulator to not only avoid self-collision but also reach the target position; (b) The repulsive-force based method pushes the manipulator from the mobile robot so that the manipulator cannot reach the target position.

The experimental results are presented in Fig. 4.11 and 4.12. In Fig. 4.11(a), as the end-effector moves close to the mobile robot, force is exerted to move the mobile robot back. As a result of the force, the end-effector reaches the target position while avoiding self-collision. In contrast, in Fig. 4.11(b), because repulsive force is generated to push the end-effector away from the mobile robot, self-collision is avoided, but the end-effector can not reach the target position. The distances between the link pairs are shown in Fig. 4.12(a). Because the distances are less than 0.15 m, the self-collision avoidance tasks ($\mathcal{T}_{sca}$ and $\mathcal{T}_{rep}$) are inserted continuously with top priority. In Fig. 4.12(b), the repulsive force-based method has a position error, while the proposed method does not.

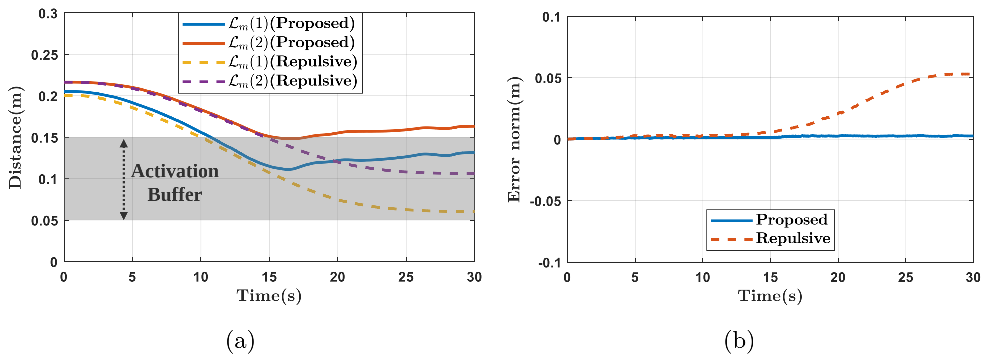
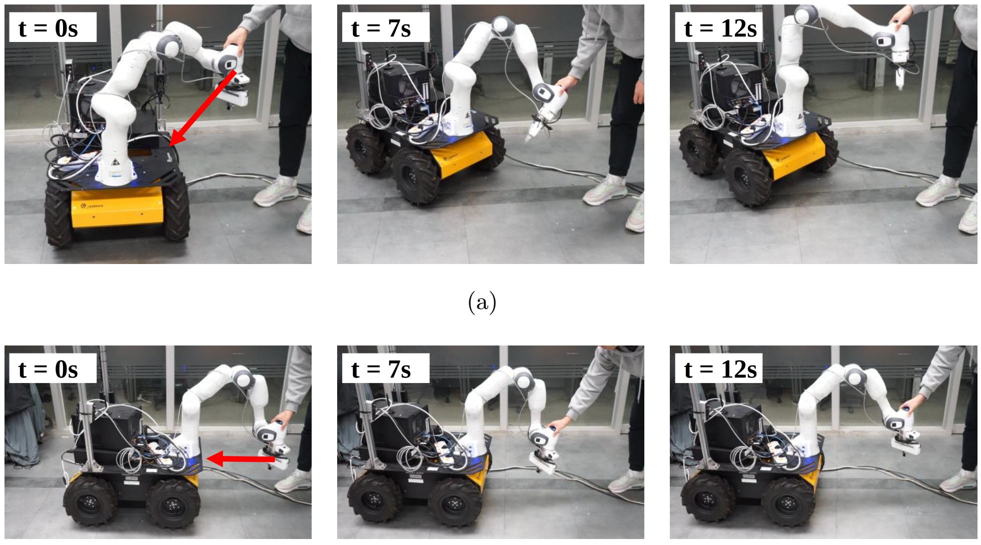


Figure 4.12: Experimental results of self-collision avoidance while tracking the predefined trajectory. (a) Distances of the link pairs $(\mathcal{L}_m(1), \mathcal{L}_m(2))$; (b) The norm of the position error

4.4.2.2 Self-collision avoidance while manually guiding the end-effector

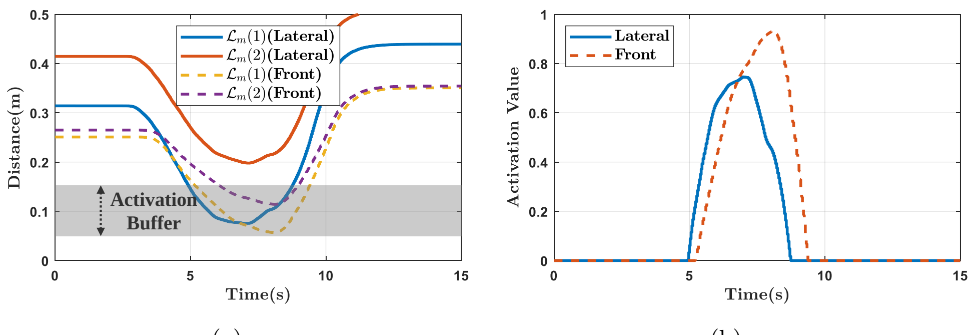
In this experiment, the end-effector was manually guided by an operator to approach the mobile robot to validate reactive self-collision avoidance during human–robot interaction. In the initial state, no tasks are executed other than the gravity compensation of the manipulator. Two directions are considered: the lateral direction and the front direction. The left snapshots in Fig. 4.13 show the initial positions of the mobile manipulator and the guiding directions are depicted by red arrows.

As shown in Fig. 4.13(a), self-collision between the manipulator and mobile robot are avoided by generating a force exerted on the mobile robot. As shown in Fig. 4.13(b), as the manipulator approaches the mobile robot, the mobile robot moves back to avoid self-collision. Fig. 4.14 presents the distances between the links of the manipulator and mobile robot and the values of the activation parameter. As the distance decreases below the buffer distance of 0.15 m, the value of the activation parameter increases accordingly and the self-



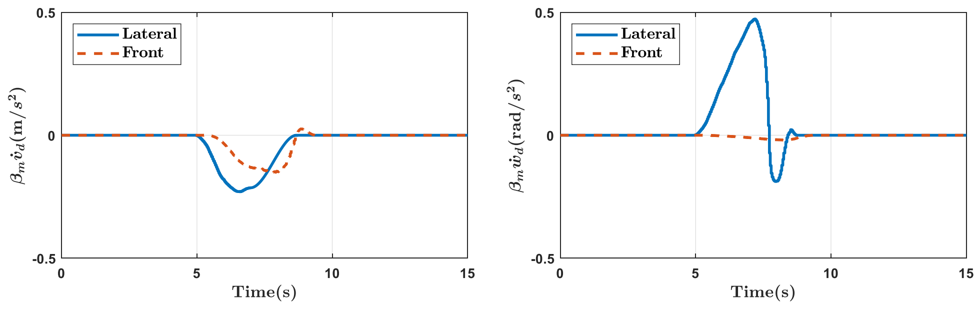
(b)

Figure 4.13: (a) the snapshots during the experiment that the manipulator approaches the mobile from the lateral direction and (b) the front direction. Red arrows show the guiding directions



(b)

Figure 4.14: (a) The distances between the link pairs ($\mathcal{L}_m(1)$, $\mathcal{L}_m(2)$, and $\mathcal{L}(3)$); (b) The value of the activation parameter



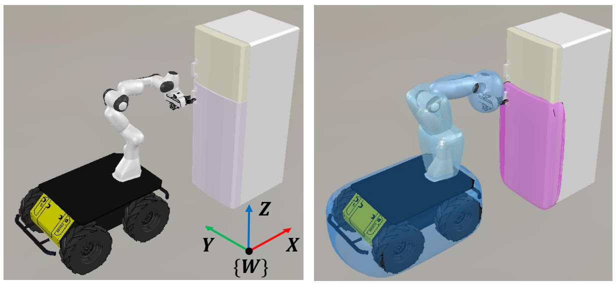
(a) (b)

Figure 4.15: Experimental results of self-collision avoidance while manually guiding the end-effector. (a) The desired linear accelerations multiplied by activation parameter; (b) The desired angular accelerations multiplied by activation parameter.

collision avoidance task, $\mathcal{T}_{sca}$, is inserted continuously as shown in Fig. 4.14(b). In Fig. 4.15, the command values of the linear and angular accelerations of the HQP-based controller are plotted. Therefore, self-collision can be avoided regardless of the approach direction of the manipulator, which is an advantage over existing methods [62,69] that do not consider the non-holonomic constraint of the differentially driven mobile robot.

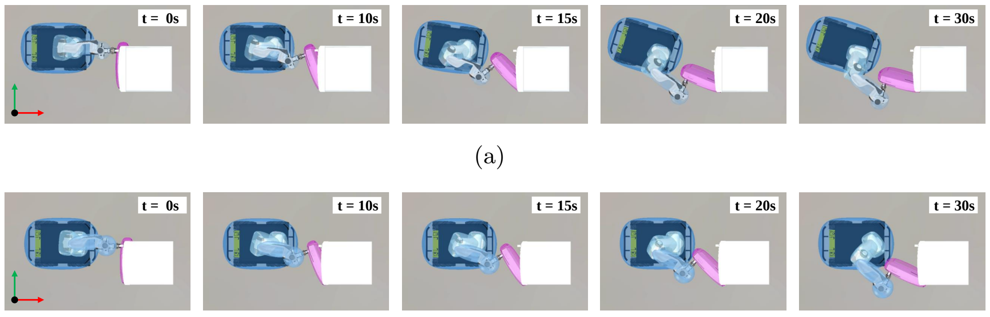
4.4.2.3 Extension to obstacle avoidance when opening the refrigerator

In this subsection, we extend our method to obstacle avoidance. The proposed method was tested in a reactive scenario representing a typical example of mobile manipulation. We consider the scenario of opening the refrigerator as shown in Fig .4.16(a). We assume that the end-effector achieves a fixed grasp on the door of the refrigerator, meaning there is no relative motion between them. In this respect, collision between the door and mobile robot is considered. We



(a) (b)

Figure 4.16: (a) Illustration of the scenario of the mobile manipulator opening a refrigerator; (b) Collision models including the door of the refrigerator are shown. The collision model for the door is colored with magenta.



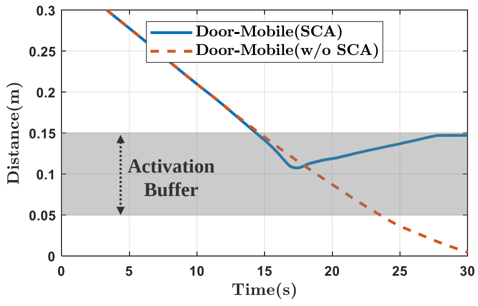
(a)

(b)

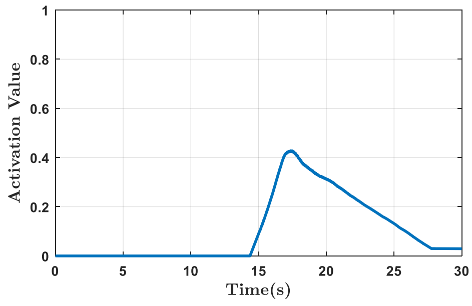
Figure 4.17: Simulation results of opening a refrigerator. (a) Snapshots of opening a refrigerator with obstacle avoidance; (b) Snapshots of opening a refrigerator without obstacle avoidance

used a hyper-ellipsoid to design a collision model for the door as shown in Fig. 4.16(b).

To open the refrigerator, a control strategy based on adaptive control [80,81] was utilized. The strategy estimates the radial direction of the door based on the force measured at the end-effector so that the end-effector can open the door even with the incomplete knowledge regarding door models. The strategy uses only the manipulator to open the door, meaning the robot may collide with

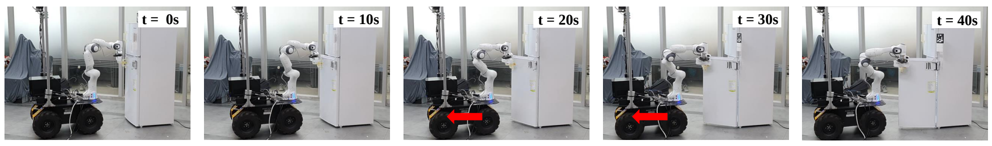


(a)

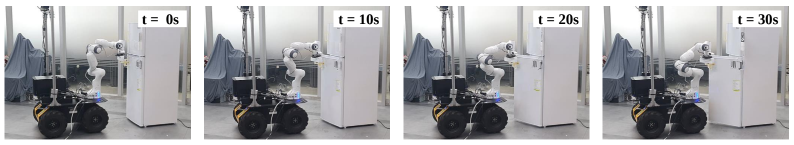


(b)

Figure 4.18: Simulation results of opening the refrigerator. (a) The distance between the door and the mobile robot; (b) The value of the activation parameter



(a)



(b)

Figure 4.19: Experimental results of opening a refrigerator. (a) Snapshots of opening a refrigerator with obstacle avoidance; (b) Snapshots of opening a refrigerator without obstacle avoidance

the door depending on the initial pose of the mobile robot. We validated our extension to obstacle avoidance by comparing the results of experiment with and without obstacle avoidance. The scenario of opening the refrigerator was validated in both simulation and experiment.

The simulation results are presented in Fig 4.17 and 4.18. In Fig.4.17(a), the mobile robot moves back and turns clockwise as the door moves closer to the mobile robot. In contrast, the door collides with the mobile robot at 30 s in Fig. 4.17(b). As the distance between the door and mobile robot is less than the buffer distance in Fig. 4.18(a), the obstacle avoidance task is inserted continuously as shown in Fig. 4.18(b).

The experimental results are presented in Fig. 4.19. As shown in Fig. 4.19(a), as the distance between the door and robot decreases, the mobile robot begins to move back at 20 s and the manipulator opens the door completely at 40 s, while avoiding collision. In contrast, in Fig. 4.19(b), the manipulator stops opening the door at 30 s because the robot is about to collide with the door.

4.4.3 Discussion

The experimental results in Sec. 4.4.2 demonstrate that the proposed method can place the manipulator outside the DBB. Specifically, the proposed method has the following advantages. First, the force can generate motion for the differentially driven mobile robot with non-holonomic constraint as shown in Sec. 4.4.2.2. This is because the acting point is selected such that it is away from the singularity of the mobile robot. Second, the proposed method can be applied to holonomic mobile manipulators if the score of the DBB is designed to include only the reachability of the manipulator. Finally, command values are free from chattering and vibration problems caused by the mobile robot unlike repulsive force-based method [82]. This is because the continuous task transition of (4.26) can calculate continuous control input. Therefore, the desired accelerations of the mobile robot are smooth, as shown in Fig. 4.15.

From a practical perspective, a trade-off relationship exists between the density of the DBB and the discontinuous position of the acting point. The denser the DBB, the more computational cost increases. However, with a denser DBB, the position of the acting point can be obtained more continuously. According to our practical experience, the proper number of points in the DBB is approximately 50,000 for running the algorithm at a control frequency of 1 kHz.

4.5 Conclusion

We presented a reactive self-collision avoidance algorithm for differentially driven mobile manipulators. The proposed algorithm generates a force exerted on a mobile robot so that a manipulator can not only avoid self-collision with the mobile robot, but also can track the desired trajectory. The force is designed based on the concept of the DBBs and their score measurement. The two factors for evaluating the score of the DBB are the determinant value of the Jacobian matrix of the non-holonomic mobile robot and the reachability of the end-effector. Based on these two factors, the force can generate the desired motion of a non-holonomic mobile robot without considering the singularity and can secure the workspace. Based on the force and resulting torque, an avoidance task is formulated and inserted into the HQP-based controller with a continuous task transition algorithm. The results of several experiments validated the proposed self-collision avoidance algorithm with a continuous task transition algorithm. Our future work will involve extending the proposed method to other mobile platforms like car-like robots. Additionally, we will apply the proposed algorithm to a wider range of mobile manipulation tasks that require the consideration of both self-collision and obstacles.

Chapter 5

CONCLUSIONS

This thesis proposes a strategy for generating the task-oriented whole-body motion of the mobile manipulator while considering kinematic and dynamic constraints. Since the mobile manipulators have both mobility and dexterity, they have the extended workspace and can execute numerous kinds of tasks that are difficult for the fixed-base manipulator. However, due to several features including high DOFs and different inertia characteristics, it is time-consuming and challenging to generate the whole-body motion while considering numerous constraints. Thus, it is effective to apply the task-oriented approach depending on the goal and enforced constraint of the assigned task. To this end, there are three strategies for generating the whole-body motion of mobile manipulator as follows.

First, the motion planner is proposed that addresses the problem of the navigation including passing through the door. The planner computes the whole-body path of the mobile manipulator in two consecutive steps which plan separately for the mobile robot and the manipulator. Especially, in the first step, the planner reduces the search space by defining a component of the state that compactly represents the range of the reachable door angles as an integer value. In the second step, the planner utilizes the IK solver to obtain the joint configuration which grasps the door handle. The effectiveness of the proposed framework was demonstrated through several simulations and real experiment with the differentially-driven mobile manipulator.

Next, the motion generation method is proposed based on hierarchical quadratic programming to assign individual joint weights for each task priority. The method formulates the optimization problem in weighted joint space. This reduces the size of the inverse matrix so that computational cost is decreased compared to traditional approach. Also, the method can deal with both equality and inequality tasks by utilizing the active-set method. The effectiveness of the method was demonstrated through several experiments with the mobile manipulator and humanoid.

Finally, the method to generate the motion to avoid self-collision is presented for the differentially-driven mobile manipulators. The method exerts the force on the mobile robot in order to place the manipulator outside the region which is a 3D curved surface enclosing the mobile robot. The direction and amplitude of the force is determined by considering the non-holonomic constraint of the mobile robot and reachability of the manipulator. The results of several scenarios show that the mobile manipulator can avoid self-collision without modifying the predefined motion of the manipulator.

Even though the proposed strategies show the effectiveness for generating the task-oriented whole-body motion, there still remain further researches to enhance the performance in the future. For the motion planner related to door opening, re-planning framework should be developed in order to respond to the changes and uncertainty of the environment. Next, for the HQP-based motion generation method, the method for determining suitable weights is required depending on the assigned tasks. At last, for the self-collision avoidance method, learning-based method for calculating the distance between the links because the calculation by the conventional library using FCL is still expensive.

Bibliography

[1] E. Asadi, B. Li, and I.-M. Chen, “Pictobot: A cooperative painting robot for interior finishing of industrial developments,” *IEEE Robotics & Automation Magazine*, vol. 25, no. 2, pp. 82–94, 2018.

[2] R. Jamisola, M. J. Ang, D. Oetomo, O. Khatib, T. M. Lim, and S. Y. Lim, “The operational space formulation implementation to aircraft canopy polishing using a mobile manipulator,” in *Robotics and Automation (ICRA), 2002 IEEE International Conference on.* IEEE, 2002, pp. 400–405.

[3] H. Gao, C. Ma, L. Ding, H. Yu, K. Xia, H. Xing, and Z. Deng, “Dynamic modeling and experimental validation of door-opening process by a mobile manipulator,” *IEEE Access*, vol. 7, pp. 80 916–80 927, 2019.

[4] S. Gray, S. Chitta, V. Kumar, and M. Likhachev, “A single planner for a composite task of approaching, opening and navigating through non-spring and spring-loaded doors,” in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 3839–3846.

[5] M. Arduengo, C. Torras, and L. Sentis, “Robust and adaptive door operation with a mobile robot,” *Intelligent Service Robotics*, vol. 14, pp. 409–425, 2021.

[6] Z. Jiao, Z. Zhang, X. Jiang, D. Han, S. Zhu, Y. Zhu, and H. Liu, "Consolidating kinematic models to promote coordinated mobile manipulations," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021, pp. 979–985.

[7] S. J. Jorgensen, M. Vedantam, R. Gupta, H. Cappel, and L. Sentis, "Finding locomanipulation plans quickly in the locomotion constrained manifold," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 6611–6617.

[8] M. Murooka, I. Kumagai, M. Morisawa, F. Kanehiro, and A. Kheddar, "Humanoid loco-manipulation planning based on graph search and reachability maps," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1840–1847, 2021.

[9] Y. Lee, S. Kim, J. Park, N. Tsagarakis, and J. Lee, "A whole-body control framework based on the operational space formulation under inequality constraints via task-oriented optimization," *IEEE Access*, vol. 9, pp. 39 813–39 826, 2021.

[10] S. Kim, K. Jang, S. Park, Y. Lee, S. Y. Lee, and J. Park, "Whole-body control of non-holonomic mobile manipulator based on hierarchical quadratic programming and continuous task transition," in *IEEE International Conference on Advanced Robotics and Mechatronics.* IEEE, 2019, pp. 414–419.

[11] Y. Wu, E. Lamon, F. Zhao, W. Kim, and A. Ajoudani, "Unified approach for hybrid motion control of moca based on weighted whole-body cartesian impedance formulation," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3505–3512, 2021.

[12] A. Escande, N. Mansard, and P.-B. Wieber, “Hierarchical quadratic programming: Fast online humanoid-robot motion generation,” *The International Journal of Robotics Research*, vol. 33, no. 7, pp. 1006–1028, 2014.

[13] J. Nocedal and W. Stephen J., *Numerical Optimization.* Springer, 2006.

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

[15] D. Kappler, F. Meier, J. Issac, J. Mainprice, C. G. Cifuentes, M. Wüthrich, V. Berenz, S. Schaal, N. Ratliff, and J. Bohg, “Real-time perception meets reactive motion generation,” *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1864–1871, 2018.

[16] J. Lee, A. Ajoudani, E. M. Hoffman, A. Rocchi, A. Settimi, M. Ferrati, A. Bicchi, N. G. Tsagarakis, and D. G. Caldwell, “Upper-body impedance control with variable stiffness for a door opening task,” in *2014 IEEE-RAS International Conference on Humanoid Robots*, 2014, pp. 713–719.

[17] Y. Karayiannidis, C. Smith, F. E. V. Barrientos, P. Ögren, and D. Kragic, “An adaptive control approach for opening doors and drawers under uncertainties,” *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 161–175, 2016.

[18] M. Stuede, K. Nuelle, S. Tappe, and T. Ortmaier, “Door opening and traversal with an industrial cartesian impedance controlled mobile robot,” in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 966–972.

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

[20] D. Lee, H. Seo, D. Kim, and H. J. Kim, "Aerial manipulation using model predictive control for opening a hinged door," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 1237–1242.

[21] 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.

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

[23] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. dissertation, Carnegie Mellon University, Pittburgh, PA, September 2010.

[24] M. Likhachev, G. J. Gordon, and S. Thrun, "Ara$*$: Anytime a$*$ with provable bounds on sub-optimality," in *Advances in Neural Information Processing Systems*, S. Thrun, L. Saul, and B. Schölkopf, Eds., vol. 16. MIT Press, 2003.

[25] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime dynamic a*: An anytime, replanning algorithm," in *Proceedings of the Fifteenth International Conference on International Conference on Automated Planning and Scheduling*, ser. ICAPS'05. AAAI Press, 2005, p. 262–271.

[26] S. Koenig, M. Likhachev, and D. Furcy, “Lifelong planning a*,” *Artificial Intelligence*, vol. 155, no. 1, pp. 93–146, 2004.

[27] S. Tonneau, N. Mansard, C. Park, D. Manocha, F. Multon, and J. Pettré, *Robotics Research: Volume 2.* Springer Ibnternational Publishing, 2018, ch. A Reachability-Based Planner for Sequences of Acyclic Contacts in Cluttered Environments, pp. 287–303.

[28] P. Beeson and B. Ames, “Trac-ik: An open-source library for improved solving of generic inverse kinematics,” in *2015 IEEE-RAS 15th International Conference on Humanoid Robots*, 2015, pp. 928–935.

[29] K. Jang, S. Kim, S. Park, J. Kim, and J. Park, “Weighted hierarchical quadratic programming: assigning individual joint weights for each task priority,” *Intelligent Service Robotics*, vol. 15, p. 475–486, 2022.

[30] D. Omrčen, L. Žlajpah, and B. Nemec, “Autonomous motion of a mobile manipulator using a combined torque and velocity control,” *Robotica*, vol. 22, no. 6, p. 623–632, 2004.

[31] G. Antonelli and S. Chiaverini, “Fuzzy redundancy resolution and motion coordination for underwater vehicle-manipulator systems,” *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 1, pp. 109–120, 2003.

[32] J. Wang, S. Kim, S. Vijayakumar, and S. Tonneau, “Multi-fidelity receding horizon planning for multi-contact locomotion,” in *20th IEEE-RAS International Conference on Humanoid Robots*, 2021.

[33] T. F. Chan and R. V. Dubey, “A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators,” *IEEE*

Transactions on Robotics and Automation, vol. 11, no. 2, pp. 286–292, 1995.

[34] S. Kim, K. Jang, S. Park, Y. Lee, S. Y. Lee, and J. Park, "Continuous task transition approach for robot controller based on hierarchical quadratic programming," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1603–1610, 2019.

[35] F. Tassi, E. De Momi, and A. Ajoudani, "Augmented hierarchical quadratic programming for adaptive compliance robot control," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 3568–3574.

[36] Y. Lee, J. Ahn, J. Lee, and J. Park, "Computationally efficient hqp-based whole-body control exploiting the operational-space formulation," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 5197–5202.

[37] J. Park and O. Khatib, "Contact consistent control framework for humanoid robots," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, 2006, pp. 1963–1969.

[38] B. Dariush, Y. Zhu, A. Arumbakkam, and K. Fujimura, "Constrained closed loop inverse kinematics," in *IEEE International Conference on Robotics and Automation.* IEEE, 2010, pp. 2499–2506.

[39] F. Farelo, R. Alqasemi, and R. Dubey, "Optimized dual-trajectory tracking control of a 9-dof wmra system for adl tasks," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 1786–1791.

[40] S. Tsuichihara, A. Yamaguchi, J. Takamatsu, and T. Ogasawara, “Using a weighted pseudo-inverse matrix to generate upper body motion for a humanoid robot doing household tasks,” in *IEEE International Conference on Robotics and Biomimetics*, 2015, pp. 333–338.

[41] J. Park, Y. Choi, W. K. Chung, and Y. Youm, “Multiple tasks kinematics using weighted pseudo-inverse for kinematically redundant manipulators,” in *IEEE International Conference on Robotics and Automation*, vol. 4, 2001, pp. 4041–4047 vol.4.

[42] Y. Choi, Y. Oh, S. R. Oh, J. Park, and W. K. Chung, “Multiple tasks manipulation for a robotic manipulator,” *Advanced Robotics*, vol. 18, no. 6, pp. 637–653, 2004.

[43] N. Vahrenkamp, T. Asfour, and R. Dillmann, “Robot placement based on reachability inversion,” in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 1970–1975.

[44] F. Zacharias, C. Borst, and G. Hirzinger, “Capturing robot workspace structure: representing robot capabilities,” in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, pp. 3229–3236.

[45] F. Chen, M. Selvaggio, and D. G. Caldwell, “Dexterous grasping by manipulability selection for mobile manipulator with visual guidance,” *IEEE Transactions on Industrial Informatics*, vol. 15, no. 2, pp. 1202–1210, 2019.

[46] A. Escande, N. Mansard, and P. Wieber, “Fast resolution of hierarchized inverse kinematics with inequality constraints,” in *IEEE International Conference on Robotics and Automation.* IEEE, 2010, pp. 3733–3738.

[47] G. Golub and C. Van Loan, *Matrix Computations.* Baltimore, MD: John Hopkins University Press, 1996.

[48] J. Sim, S. Kim, S. Park, S. Kim, M. Kim, and J. Park, "Design of jet humanoid robot with compliant modular actuators for industrial and service applications," *Applied Sciences*, vol. 11, no. 13, 2021.

[49] A. Rocchi, E. M. Hoffman, D. G. Caldwell, and N. G. Tsagarakis, "Opensot: A whole-body control library for the compliant humanoid robot coman," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 6248–6253.

[50] C. Fang, A. Rocchi, E. M. Hoffman, N. G. Tsagarakis, and D. G. Caldwell, "Efficient self-collision avoidance based on focus of interest for humanoid robots," in *IEEE-RAS International Conference on Humanoid Robots.* IEEE, 2015, pp. 1060–1066.

[51] O. Kanoun, F. Lamiraux, and P.-B. Wieber, "Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 785–792, 2011.

[52] Y. Jia, N. Xi, Y. Cheng, and S. Liang, "Coordinated motion control of a nonholonomic mobile manipulator for accurate motion tracking," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 1635–1640.

[53] H. A. Park and C. G. Lee, "Dual-arm coordinated-motion task specification and performance evaluation," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 929–936.

[54] J. Park, W. Chung, and Y. Youm, “Weighted decomposition of kinematics and dynamics of kinematically redundant manipulators,” in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 1, 1996, pp. 480–486 vol.1.

[55] S. Kim, K. Jang, S. Park, Y. Lee, S. Y. Lee, and J. Park, “Continuous task transition approach for robot controller based on hierarchical quadratic programming,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1603–1610, 2019.

[56] S. Kim, K. Jang, S. Park, Y. Lee, S. Y. Lee, and J. Park, “Whole-body controller for non-holonomic mobile manipulator based on hqp with continuous task transition,” in *Advanced Robotics and Mechatronics (ARM), 2019 IEEE International Conference on.* IEEE, 2019, pp. 414–419.

[57] J. J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue, “Dynamically-stable motion planning for humanoid robots,” *Autonomous Robots*, vol. 12, pp. 105–118, 2002.

[58] G. Oriolo and C. Mongillo, “Motion planning for mobile manipulators along given end-effector paths,” in *Robotics and Automation (ICRA), 2005 IEEE International Conference on.* IEEE, 2005, pp. 2154–2160.

[59] D. Berenson, J. Chestnutt, S. S. S., J. J. Kuffner, and S. Kagami, “Pose-constrained whole-body planning using task space region chains,” in *Humanoid Robots (Humanoids), 2009 9th IEEE-RAS International Conference on.* IEEE, 2009, pp. 181–187.

[60] F. Burget, M. Bennewitz, and W. Burgard, “Bi2rrt*: An efficient sampling-based path planning framework for task-constrained mobile manipulation,”

in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on.* IEEE, 2016, pp. 3714–3721.

[61] F. Seto, K. Kosuge, and Y. Hirata, “Real-time self-collision avoidance system for robots using robe,” *The International Journal of Humanoid Robotics*, vol. 1, no. 3, pp. 533–550, 2004.

[62] F. Seto, K. Kosuge, and Y. Hirata, “Self-collision avoidance motion control for human robot cooperation system using robe,” in *Intelligent Robots and Systems (IROS), 2005 IEEE/RSJ International Conference on.* IEEE, 2005, pp. 50–55.

[63] O. Stasse, A. Escande, N. Mansard, S. Miossec, P. Evrard, and A. Kheddar, “Real-time (self)-collision avoidance task on a hrp-2 humanoid robot,” in *Robotics and Automation (ICRA), 2008 IEEE International Conference on.* IEEE, 2008, pp. 3200–3205.

[64] M. Schwienbacher, T. Buschmann, S. Lohmeier, V. Favot, and H. Ulbrich, “Self-collision avoidance and angular momentum compensation for a biped humanoid robot,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on.* IEEE, 2011, pp. 581–586.

[65] B. Dariush, G. Bin Hammam, and D. Orin, “Constrained resolved acceleration control for humanoids,” in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on.* IEEE, 2010, pp. 710–717.

[66] A. De Luca, G. Oriolo, and P. Robuffo Giordano, “Kinematic modeling and redundancy resolution for nonholonomic mobile manipulators,” in *Robotics and Automation (ICRA), 2002 IEEE International Conference on.* IEEE, 2006, pp. 1867–1873.

[67] A. De Luca, G. Oriolo, and P. Robuffo Giordano, “Kinematic control of nonholonomic mobile manipulators in the presence of steering wheels,” in *Robotics and Automation (ICRA), 2002 IEEE International Conference on.* IEEE, 2010, pp. 1792–1798.

[68] A. Dietrich, T. Wimböck, H. Täubig, A. Albu-Schäffer, and G. Hirzinger, “Extensions to reactive self-collision avoidance for torque and position controlled humanoids,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on.* IEEE, 2011, pp. 3455–3462.

[69] A. Dietrich, T. Wimböck, A. Albu-Schäffer, and G. Hirzinger, “Integration of reactive, torque-based self-collision avoidance into a task hierarchy,” *IEEE Transactions on Robotics*, vol. 28, no. 6, pp. 1278–1293, 2012.

[70] H. Sugiura, M. Gienger, H. Janssen, and C. Goerick, “Reactive self collision avoidance with dynamic task prioritization for humanoid robots,” *The International Journal of Humanoid Robotics*, vol. 7, no. 1, pp. 31–54, 2010.

[71] G. Campion, G. Bastin, and B. Dandrea-Novel, “Structural properties and classification of kinematic and dynamic models of wheeled mobile robots,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 1, pp. 47–62, 1996.

[72] A. M. Bloch, M. Reyhanoglu, and N. H. McClamroch, “Control and stabilization of nonholonomic dynamic systems,” *IEEE Transactions on Automatic Control*, vol. 37, no. 11, pp. 1746–1757, 1992.

[73] G. Campion, B. d’Andrea Novel, and G. Bastin, “Modelling and state feedback control of nonholonomic mechanical systems,” in *IEEE Conference on Decision and Control.* IEEE, 1991, pp. 1184–1189.

[74] Y. Yamamoto and Y. Xiaoping, “Coordinating locomotion and manipulation of a mobile manipulator,” *IEEE Transactions on Automatic Control*, vol. 39, no. 6, pp. 1326–1332, 1994.

[75] R. Diankov and J. Kuffner, “Openrave : A planning architecture for autonomous robotic,” Robotics Institute, Carnegie Mellon University, Tech. Rep. CMU-RI-TR-10-29, 2008.

[76] S. Hong, K. Jang, S. Kim, and J. Park, “Regularized hierarchical quadratic program for real-time whole-body motion generation,” *IEEE/ASME Transactions on Mechatronics*, 2020.

[77] G. D. White, R. M. Bhatt, C. P. Tang, and V. N. Krovi, “Experimental evaluation of dynamic redundancy resolution in a nonholonomic wheeled mobile manipulator,” *IEEE/ASME Transactions on Mechatronics*, vol. 14, no. 3, pp. 349–357, 2009.

[78] S. Kim, https://github.com/ggory15/weightedhqp, 2021.

[79] A. Dietrich, K. Bussmann, F. Petit, P. Kotyczka, C. Ott, B. Lohmann, and A. Albu-Schäffer, “Whole-body impedance control of wheeled mobile manipulators,” *Autonomous Robots*, vol. 40, no. 3, pp. 505–517, 2016.

[80] Y. Karayiannidis, C. Smith, F. E. Viña, P. Ogren, and D. Kragic, ““open sesame!” adaptive force/velocity control for opening unknown doors,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on.* IEEE, 2012, pp. 4040–4047.

[81] Y. Karayiannidis, C. Smith, F. E. V. Barrientos, P. Ögren, and D. Kragic, “An adaptive control approach for opening doors and drawers under un-

certainties," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 161–175, 2016.

[82] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Autonomous robot vehicles.* Springer, 1986, pp. 396–404.

초 록

모바일 매니퓰레이터는 모바일 로봇에 장착된 매니퓰레이터입니다. 모바일 매니퓰레이터는 고정형 매니퓰레이터에 비해 모바일 로봇의 이동성을 제공받기 때문에 다양하고 복잡한 작업을 수행할 수 있습니다. 그러나 두 개의 서로 다른 시스템을 결합함으로써 모바일 매니퓰레이터의 전신을 계획하고 제어할 때 여러 특징을 고려해야 합니다. 이러한 특징들은 여자유도, 두 시스템의 관성 차이 및 모바일 로봇의 비홀로노믹 제한 조건 등이 있습니다. 본 논문의 목적은 기구학적 및 동적 제한조건들을 고려하여 모바일 매니퓰레이터의 전신 동작 생성 전략을 제안하는 것입니다.

먼저, 모바일 매니퓰레이터가 초기 위치에서 문을 통과하여 목표 위치에 도달하기 위한 전신 경로를 계산하는 프레임워크를 제안합니다. 이 프레임워크는 로봇과 문에 의해 생기는 기구학적 제한조건을 고려합니다. 제안하는 프레임워크는 두 단계를 거쳐 전신의 경로를 얻습니다. 첫 번째 단계에서는 그래프 탐색 알고리즘을 이용하여 모바일 로봇의 자세 경로와 문의 각도 경로를 계산합니다. 특히, 그래프 탐색에서 *area indicator*라는 정수 변수를 상태의 구성 요소로서 정의하는데, 이는 문에 대한 모바일 로봇의 상대적 위치를 나타냅니다. 두 번째 단계에서는 모바일 로봇의 경로와 문의 각도를 통해 문의 손잡이 위치를 계산하고 역기구학을 활용하여 매니퓰레이터의 관절 위치를 계산합니다. 제안된 프레임워크의 효율성

은 비홀로노믹 모바일 매니퓰레이터를 사용한 시뮬레이션 및 실제 실험을 통해 검증되었습니다.

둘 째, 최적화 방법을 기반으로한 전신 제어기를 제안합니다. 이 방법은 등식 및 부등식 제한조건 모두에 대해 가중 행렬을 반영한 계층적 최적화 문제의 해를 계산합니다. 이 방법은 모바일 매니퓰레이터 또는 휴머노이드와 같이 자유도가 많은 로봇의 여자유도를 해결하기 위해 개발되어 작업 우선 순위에 따라 가중치가 다른 관절 동작으로 여러 작업을 수행할 수 있습니다. 제안된 방법은 가중 행렬을 최적화 문제의 1차 최적 조건을 만족하도록 하며, Active-set 방법을 활용하여 등식 및 부등식 작업을 처리합니다. 또한, 대칭적인 영공간 사영 행렬을 사용하여 계산상 효율적입니다. 결과적으로, 제안된 제어기를 활용하는 로봇은 우선 순위에 따라 개별적인 관절 가중치를 반영하여 전신 움직임을 효과적으로 보여줍니다. 제안된 제어기의 효용성은 모바일 매니퓰레이터와 휴머노이드를 이용한 실험을 통해 검증하였습니다.

마지막으로, 모바일 매니퓰레이터의 동적 제한조건들 중 하나로서 자가 충돌 회피 알고리즘을 제안합니다. 제안된 방법은 매니퓰레이터와 모바일 로봇 간의 자가 충돌에 중점을 둡니다. 모바일 로봇의 버퍼 영역을 둘러싸는 3차원 곡면인 *distance buffer border*의 개념을 정의합니다. 버퍼 영역의 두께는 버퍼 거리입니다. 매니퓰레이터와 모바일 로봇 사이의 거리가 버퍼 거리보다 작은 경우, 즉 매니퓰레이터가 모바일 로봇의 버퍼 영역 내부에 있는 경우 제안된 전략은 매니퓰레이터를 버퍼 영역 밖으로 내보내기 위해 모바일 로봇의 움직임을 생성합니다. 따라서 매니퓰레이터는 미리 정의된 매니퓰레이터의 움직임을 수정하지 않고도 모바일 로봇과의 자가 충돌을 피할 수 있습니다. 모바일 로봇의 움직임은 가상의 힘을 가함으로써 생성됩니다. 특히, 힘의 방향은 차동 구동 이동 로봇의 비홀로노믹 제약 및 조작기의 도달 가능성을 고려하여 결정됩니다. 제안된 알고리즘은 7자유도 로봇팔을 가진 차동 구동 모바일 로봇에 적용하여 다양한 실험 시나리오에서

입증되었습니다.

주요어: 모바일 매니퓰레이터, 전신 제어, 경로 계획

학번: 2016-26043

ACKNOWLEGEMENT

7년이라는 석박사 학위과정을 수행하면서 많은 분들의 도움과 응원이 있었기에 끝까지 마무리할 수 있었습니다. 특히, 입학할 당시에 로봇에 대한 부족한 지식으로 인해 연구 주제에 대해 고민하고 헤매고 있을 때 지켜봐 주시고 지도해주신 박재흥 교수님께 감사의 말씀드립니다. 그런 시간을 보내지 않았다면 지금의 모습으로 성장할 수 없었다고 생각합니다.

같이 연구실 생활했던 선배분들에게 먼저 감사함을 표시하고 싶습니다. 소셜로봇 팀을 만들어 같이 밤을 지새우며 연구에 몰두하고 연구을 포함하여 생활에서도 멘토 역할을 해주신 상현형, 같이 술 한잔 하면서 노래방을 항상 가고싶게 만드는 순욱형, 술먹고 싶다고 하면 바로 달려와 주셔서 1차에서만 끝낼 수가 없게 만드는 투자의 귀재 민곤형, 연구 상담뿐만 아니라 취업에서도 많은 도움을 주신 이수형, 수학적인 질문을 해도 항상 진지하게 고민해주신 호상형, 긴말 필요없이 모든 걸 다 갖추신 범영형, 성수동 핫플레이스 카페 CTO로 활약중이신 현준형, 육아로 바쁘셨지만 가끔 고민 상담도 해주신 수민누나, 연구실 신입생 시절 졸업하시면서 많은 걸 알려주신 세호형, 항상 뚝심있게 MPC를 진지하게 연구하신 민성형, 장난은 많이 치시지만 다방면으로 많은 걸 알려주신 원제형 감사합니다.

다음으로 동기를 포함하여 동기같이 지내준 분들에게 감사함을 전하고 싶습니다. 나의 영원한 술메이트이자 하드웨어와 소프트웨어를 모두 잘하는 재훈, 7년

내내 자리 한번 안 바꾸고 내 옆자리에서 동고동락한 승연, 연구와 육아로 바쁘지만 광교에 오면 누구보다 반가운 준우, 독일에서 연구원으로 활약중인 재석이 감사합니다.

소셜팀 후배들에게도 감사한 마음을 표현하고 싶습니다. 연구실에서 모두가 필요로 해서 도울 일이 있으면 항상 도맡아서 해주는 수한, 가끔 술친구가 되주면서 모션 플래닝으로 이끌어준 지영, 3층의 분위기 메이커로 활약해준 형철, 힘든 내색 없이 궂은 일을 도맡아서 해준 명수, 뚝심있게 자기 연구를 잘 수행하는 해성, 말수가 적지만 연구 앞에서는 누구보다 진지하고 열정적인 재현, 바빠서 먼저 챙겨주진 못하지만 항상 먼저 안부를 물어주는 상엽에게도 감사함을 전하고 싶습니다.

진지한 모습이 매력적인 RRT 달인 민수, 운동도 잘하고 술도 잘먹는 양우, 미국에서 나의 룸메이크로 고생한 동현, 성격이 화끈한 부산사나이 명주, 후배들을 잘 챙겨주시고 품어주시는 승훈형, 재활로봇팀의 허리를 맡고 계신 주완이형, 체격이 좋지만 누구보다 관심을 갈구하는 준형, 연구실의 미래 랩장으로 도약하고 있는 경재, 운동도 열심히하고 누구보다 햅틱 장비를 잘 만드는 은호, 의외의 모습을 많이 가지고 있는 재용, 토카비의 뇌를 담당하고 있는 준휘, 뚝딱뚝딱 잘 만들고 추진력이 좋은 승빈형, 연구실의 막내지만 마음이 넓은 호균, 연구실의 딥러닝 전문가 대규, 3층의 터줏대감 역할을 해주시는 현범이형, CPR 로봇의 대가 성문이형, 아바타 대회에서 통역가로 활약한 준혁, 3층에서 항상 열심히하는 관우와 민수, 박사과정 마무리 실험을 많이 도와준 준헌이를 포함하여 모두 감사합니다. 앞으로 좋은 연구성과 기대하겠습니다.

마지막으로, 대학원 생활에만 몰두할 수 있게 다방면으로 도와주시고 항상 지지해주신 부모님, 독일에서 대학원 다니느라 고생하고 있는 남동생 근한, 행복한 신혼 생활을 하고 있는 여동생 혜린, 매부이자 친구이자 술친구가 되어준 형진에게도 감사함을 전하고 싶습니다.