Position and Force Control with Redundancy Resolution for Mobile Manipulators

Master Thesis

University of Toronto
Institute of Aerospace Studies
Dynamic Systems Lab

University of Stuttgart
Institute of Control Theorie of Machine Tools
and Manufacturing Units

Submitted by
Michael Jakob
Matriculation No. 3160251

Approved by
Prof. Dr. Angela P. Schoellig
Prof. Dr. Alexander Verl

Editing period
May 15, 2018 to November 15, 2018

Abstract

In contrast to traditional industrial automation, mobile manipulators combine the advantages of mobile platforms and robotic manipulator arms. This extends the workspace of the arm and, in general, enables the robot to do more complex tasks and interactions. However, the present technology of mobile manipulators is still in an early stage of its development process with many open research problems. This work particularly focuses on control approaches for mobile manipulators, considering how the redundancy between the mobile base and the arm can be exploited in order to improve the performance of the robot with respect to the desired task. In addition, a force and position control architecture is proposed which allows the mobile manipulator to interact with the environment and, at the same time, track a trajectory in unconstrained directions. Simulation and experimental results demonstrate the performance of the proposed control architecture.

Keywords: Robotics, Mobile Manipulator, Trajectory Tracking, Force Control, Redundancy Resolution

Contents

Acknowledgments

Most pieces of writing, when they exceed a certain size or complexity, can, though produced by a single person, not be attributed to the author alone. This master thesis is no exception. It would not exist in this form if I had not been able to rely on the help of many people, supporting me in both personal and professional matters. I want to thank all people who helped me, especially those mentioned hereafter.

To begin with, I am immensely grateful to Angela Schoellig, who gave me the opportunity of joining the Dynamic Systems Lab at the University of Toronto and in addition supervised my work. I deeply appreciated her remarks and suggestions during the course of the project and the writing of this work. My warmest thanks also go to Mohamed Helwa who supported me in many situations and helped to put me on the right path when I got lost, by supervising my work as well. Furthermore, I am very grateful to Adam Heins for his great support in doing the experiments of this work, as well as all the fruitful conversations we had. Many thanks also go to the remaining members of the Dynamic Systems Lab and STARS Lab for all the enjoyable conversations. I am very lucky to have worked with all in this inspiring facility.

Leaving the university environment, many special thanks go to my family and friends for their ongoing support and encouragement in all downs I had to experience during the course of this thesis. They were a comfort to me every time when things did not work out as planned.

Abbreviations and Symbols

Abbreviations

Symbol	Description
DH	Denavit-Hartenberg
Dof	Degree of freedom
EE	End-Effector
MM	Mobile Manipulator
TCP	Tool Center Point
TRL	Technology Readiness Level

Symbols

Symbol	Description
α	Vector of joint accelerations
$c(p_{ee}, \rho, q)$	Constraint function
$C(q, \dot{q})$	Vector of Coriolis and centrifugal torques
c_i	Distance between center of mass i and joint i of a manipulator link
d	Distance between mobile base center and manipulator base
E_{kin}	Kinetic energy

E_{pot}	Potential energy
Δf	Force error
f	Measured force
f_c	Contact force
Γ	Force compensation function
$g(p_{ee}, \rho, q)$	Objective function
$G(q)$	Vector of gravity torques
I	Identity matrix
I_i	Inertia of each robot component i
J	Jacobian
$J^{\dagger}$	Right pseudoinverse
$J_W^{\dagger}$	Weighted pseudoinverse
K	Gain matrix of type $diag\{K_1, ...K_n\}$
$L(q_i, \dot{q}_i)$	Lagrangian
l_i	Length of each manipulator link i
$M(q)$	Invertible inertia matrix
M	Number of dimensions in cartesian space
m_i	Mass of each component i of the mobile manipulator
$\mathcal{N}(J)$	Null space of J is the subspace of joint velocities that do not produce any ee velocity
N	Number of dimensions in joint space
o	Position vector of a point on the obstacle
ϕ_{ee}	End-effector orientation

Ψ	Position compensation function
p_{c_0}	Point of contact with the environment
p_c	Actual end-effector position
p_d	Desired end-effector position
p_{ee}	End-effector position
$\bar{q}_b$	Current posture of the mobile base
$\bar{q}_i$	Middle value of the joint range
$\dot{q}_i$	Velocity of joint i
q_i	Angle of joint i
q_{iM}	Maximum joint limit
q_{im}	Minimum joint limit
$\mathcal{R}(J)$	Range space of J is the subspace of ee velocities at a given manipulator posture can be generated by the joint velocities $\dot{q}$
ρ	Vector of redundancy parameters
ρ_{max}	Upper workspace boundary of the robot arm
ρ_{min}	Lower workspace boundary of the robot arm
R	Number of redundant degrees of freedom
S	Compliance selection vector
${}^{w}T_{ee}$	Forward transformation from the end-effector frame to the world frame
τ_i	Torque of joint i
θ	Mobile base orientation
v_{ee}	End-effector velocity
$w(q)$	Differentiable objective function

W	Symmetric positive defined weighting matrix
w_i	Weighting factor
x, y, z	Cartesian position parameters

Chapter 1

Introduction

Today's production lines are designed static, still based on the idea of Henry Ford in the early 20th century. The automation since then increased dramatically, however, the flexibility of the production only increased slightly and is still behind the goal of autonomous and independent configuration. Robots establish an important role as part of production line automation. Nevertheless, in order to drive the change towards a flexible manufacturing process, today's industrial robots suffer from a fundamental disadvantage, the absence of mobility. A fixed robot arm is limited in its range of motion that depends on the size of its workspace and the static placement of the robot in the production line. In contrast, a mobile robot would be able to travel throughout the manufacturing plant, to apply its skills wherever it is required. [22] This would also allow the application of robots in scenarios that need a higher flexibility and a large workspace such as in aerospace manufacturing, shipbuilding and wind turbine manufacturing.[18] [10] Thus, the combination of mobility and manipulation is considered a key technology for future automation, based on the idea of industry 4.0, to allow flexible manufacturing scenarios. However, in spite of its importance and vast potential, the present technology in this field is still in the early stages of its development process and not yet sufficiently mature. This is emphasized by the so-called Technology Readiness Level [1](TRL) of mobile manipulation which ranges between level 3 and 4, as well as the very limited range of commercially available mobile manipulator platforms suitable for automation and advanced manufacturing. [4]

[1]Level 1 of the TRL corresponds to basic research and level 9 to final state of technology and product development with products operating under mission condition.

Reasons, why the technology has not progressed further yet, are mainly caused by its advantage of mobility. The mobility leads to an operation of the robot in unstructured and more complex environments than the separated work cells of today's static industrial robots. This requires the mobile robot to orientate and interact with the environment, as well as be able to react to unknown events. These demands create challenges in different areas of engineering and control and offer a plentitude of open research problems. According to [4] unsolved topics in the field of research for mobile manipulators are the seamless integration of planning and control, the task effective choreography of motion, as well as the revision and adaptation of tasks and motion plans to avoid obstacles and enable safe human-robot interaction and collaboration. This requires sensing of the internal robot state, as well as the external environment by integrating 3D perception, mapping and modelling and resource management to enable self-perception, self-modelling and self-awareness of the robot. [4]

This thesis shall focus on control architectures for mobile manipulators to allow typical tasks like assembly and contact tooling, e.g. grinding, drilling, polishing and deburring. [23] The challenges, to enable the robot to track trajectories and simultaneously interact with the environment, are the design of position and force control methods that consider and exploit the redundancy of mobile manipulators. [25] addresses the problem of position and force control for holonomic constrained non-redundant mobile manipulators, by applying active disturbance rejection control. [12] and [15] propose a control method to apply motion and force control for non-holonomic constrained mobile manipulators. Redundancy Resolution in combination with robust or adaptive control for mobile manipulators is presented by [3] and [2].

It can be noticed that the control problem can be broken down into three subproblems - trajectory tracking, interaction with the environment and redundancy resolution. Thus, the mobile manipulator control method is the combination of the solution for each subproblem. However, a universal and general control approach for mobile manipulators with redundancy has not been presented yet.

1.1 Objectives

The goal of this work is to explore different options of control methods for trajectory tracking and interaction with the environment. Therefore, one objective is a literature review, comprising redundancy resolution, motion control and force control methods, that can be used as a foundation for the development of a mobile manipulator control architecture considering the following objectives:

(O1) Trajectory tracking The robot is able to track a trajectory assigned to the end-effector.

(O2) Performance The performance of a robot comprises many properties. Its purpose in this work is to focus on precision and accuracy in trajectory tracking of the end-effector position while providing dexterity to the robot.

(O3) Combined Motion Enable combined motion of all subsystems (mobile base and robotic arm) to execute the desired task assigned to the end-effector.

(O4) Redundancy Resolution Solve system redundancy in order to improve the performance of the mobile manipulator.

(O5) Interaction with the environment The robot end-effector is able to interact with the environment by controlling the contact force to the desired value in constraint directions.

(O6) Modular control structure The control architecture is based on a modular structure allowing to change parts of the architecture without the need to redesign the entire control approach.

To test and demonstrate the performance of the developed control architecture, one experiment aimed for is to show how the robot writes (O1)(O2) on a large whiteboard which requires the mobile base to move (O3)(O4) and in the same time maintain a constant contact force (O5) in the constraint direction, between the manipulator end-effector and the whiteboard.

1.2 Outline of the thesis

The thesis is structured as follows. After introducing the general topic and motivating this thesis in the context of open research questions, the overall objectives of this work are presented. The following chapter is listing the related fundamentals and background about mobile manipulator control. This comprises the fundamentals about manipulators and mobile platforms, as well as in particular the topics redundancy resolution, motion control and force control. In chapter 3 the kinematic and dynamic model of a mobile manipulator is presented. This model is used in chapter 4 to develop a control architecture based on the objectives listed above. Chapter 5 describes a Simulink model of a simplified 3-Dof MM to verify the control approach of the previous chapter and presents the simulation results followed by a brief evaluation. In chapter 6, the experiments at the real robot are described and results are evaluated. The last chapter summarizes the research and presents a brief outlook on future directions.

Chapter 2

Fundamentals and Background

This chapter summarizes a literature review about three major topics that are of interest for the further course of this thesis. The first topic provides an introduction into mobile manipulators by listing the fundamentals about robot manipulators and mobile platforms. Furthermore, different methods of redundancy resolution are investigated, as it is considered an important part of mobile manipulator modeling and control. The second topic deals with motion control methods for manipulators and mobile manipulators, where an overview about motion control, in general, and a procedure of obtaining the dynamic robot model, relevant for motion control, are provided. The third topic introduces the fundamentals of force control. Thereby, different approaches of force control methods are revealed.

2.1 Mobile Manipulators

The combination of a robot manipulator mounted on a mobile base is called mobile manipulator. This technology comprises a variety of mobile platforms and robot arms. First, the properties and mathematical relations of manipulators are presented, whereas a substantial part redundancy resolution methods are investigated. Second, mobile platforms, in general, are introduced and elaborated in context with the platform used in this work.

2.1.1 Robot Manipulator

A robot manipulator is a structure of links (rigid bodies) interconnected by joints. This kinematic structure can be serial or parallel, based on the required design and the area of application of the manipulator. One or multiple ends of this structure are fixed to the ground or base, the other end is attached to the end-effector (EE) that performs the required task. In the past different kind of robot kinematics have been developed, but especially for a wide range of industrial applications, a 6-Dof open-chain manipulator is a common solution. [20] The joints can be actuated or non-actuated and either be prismatic or revolute.

Definition 1. In this work, only serial kinematics with actuated revolute joints are considered.

Figure 2.1 gives an example of a serial open-chain kinematic scheme with 3-Dof and actuated revolute joints. A common method to model the kinematics and determine the forward transformation ${}^{w}T_{ee}$ of a manipulator is to use the Denavit-Hartenberg (DH) notation described in [23] and [20].

To represent the relation of a position regarding a coordinate frame, the notation ${}_{b}p_{ee}$ is used, were b is indicating the reference frame of the end-effector position p_{ee}. The vector $\mathbf{x}_{ee} = [p_{ee}\phi_{ee}]^T$ combines the end-effector position and orientation.

Definition 2. In this thesis, only the end-effector position will be considered, without the orientation ϕ_{ee}.

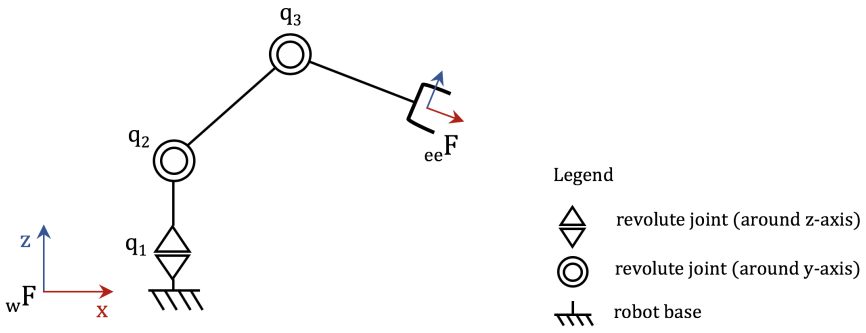

Figure 2.1: Model of a 3-Dof manipulator with revolute joints. The arrangement of joints allows the arm to reach any position in 3D-space within the workspace boundaries. Definition of the end-effector position $p_{ee} = (x, y, z)$ and the robot arm state $q = (q_1, q_2, q_3)$

The relation between the end-effector position p_{ee}, orientation ϕ_{ee}, and the robot state q is described on velocity level, by the Jacobian matrix, a function of the configuration q. This differential kinematics equation defines a mapping between $v_{ee} = [\dot{p}_{ee}\omega_{ee}]^T$, representing the end-effector velocity space, and $\dot{q}$, representing the joint velocity space, written as

$$v = J(q)\dot{q} \tag{2.1}$$

where the Jacobian matrix is the partial derivative of the vector $\mathbf{x}_{ee}$ by the joint vector q. [20] Accordingly, the inverse of equation (2.1.2) can be used to determine the joint variables corresponding to a given end-effector position and orientation, which is known as *inverse kinematics problem*

$$\dot{q} = J^{-1}(q)v \tag{2.2}$$

as described in [23] and [20]. The solution to this problem is essential in order to transform the desired end-effector motion in task space, into the corresponding joint space motions that allow the execution of the desired end-effector motion. However, the inverse kinematics problem is much more complex to solve. One reason for this behavior is that either multiple solutions or even an infinite amount of solutions for the inverse kinematics problem may exist, e.g., in the case of a kinematically redundant manipulator, as will be described more detailed in subsection 2.1.2. Another reason is that the equations to solve are in general nonlinear. Thus, it is not always

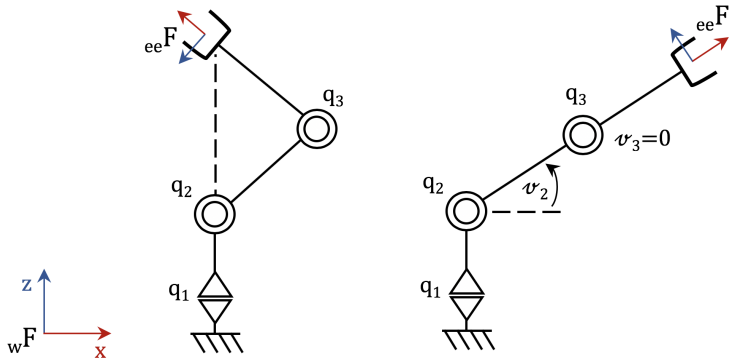

Figure 2.2: The left part reveals a *internal singularity*, where an infinite amount of joint configurations of q_1 yield the same end-effector position. The right part illustrates a *boundary singularity*, where the arm looses one degree of freedom [20]

possible to find a closed-form solution. The result of the inverse kinematics problem might be also a non-admissible solution in terms of the kinematic structure. [20] This requires the Jacobian to have the same amount of columns and rows and to be of full rank, in order to be able to calculate the inverse.

Joint configurations at which J is not of full rank, hence rank-deficient, are identified as kinematic singularities. Those configurations have to be figured out and handled in order to avoid them, due to the fact that singularities can cause an infinite amount of solutions to the inverse kinematics problem or they represent configurations at which the mobility of the kinematic structure is reduced. In other words, it is not possible to impose an arbitrary motion of the end-effector at this joint configuration. Another characteristic in the neighborhood of a singular joint configuration is that small end-effector velocities in the task space may cause large velocities in the joint space. This behavior can be observed in particular when the robot arm is either close to be outstretched or retracted. [20] In general, singularities can be classified into:

- **Boundary singularities** occur when the manipulator is either outstretched or retracted. These singularities can be avoided easily by defining operational workspace limits that the manipulator is not able to move to the boundaries of its reachable workspace. The right part of figure 2.2 visualizes a boundary singularity of a 3-link arm. [20]

- **Internal singularities** occur inside the reachable workspace and are generally caused by the alignment of two or more axes of motion or other particular end-effector configurations. These singularities are not as easy as the above to handle since they can occur anywhere in the reachable workspace for a planned path in the operational space. The left part of figure 2.2 shows an internal singularity of a 3-link arm, where the end-effector position is aligned with the axis of joint q_1. [20]

2.1.2 Redundancy Resolution

In case a robot is redundant, an infinite amount of joint configurations will result in the same end-effector position. Even when the end-effector is resting at a specified position the joints can still move and change the internal state of the robot. Compared to internal singularities, as introduced in subsection 2.1.1 above, this redundancy can be used to meet additional task requirements besides the execution of

the end-effector motion, e.g. by providing the manipulator with a higher moveability and versatility to execute a specified motion. In addition, the redundancy can be used to avoid singular joint configurations of the robotic arm. [21] [20]

A kinematic structure, in particular, a robot, can be defined as kinematically redundant if it has more degrees of freedom (dimensions in joint space N) than required to execute a task (number of dimensions in operational space M). This leads to the definition that the system is *intrinsically redundant* for $N > M$. However, this concept is relative to a specified task, with the required number of dimensions of the operational space M. The task could require to only consider the position of the robot end-effector or the position and orientation. Furthermore, the number of dimensions varies if the position is represented in the plane ($M = 3$) or in three-dimensional space ($M = 6$). [13] In case of a 6-Dof industrial manipulator kinematic in 3D-space, the arm is not intrinsically redundant ($M = N = 6$). But if the task requires to only consider the position $M = 3$, without the orientation, the system is *functionally redundant*, with a redundancy of $R = N - M = 6 - 3 = 3$ Dof. [21] [20]

A typical example of an intrinsically redundant kinematic structure is the human arm with seven Dof. Three Dof in the shoulder, one in the elbow and three in the wrist. This allows moving the elbow, even when the hand position and orientation are both fixed, which can be used, for instance, to avoid obstacles in the workspace. Since the base (body) of a human arm is mobile, the movement of the base allows executing the desired end-effector motion, even when the redundant manipulator reaches its mechanical limits. [20]

The challenge, however, is to solve the redundancy of the robot in order to increase its performance and to exploit the redundant degrees of freedom. In general, redundancy can be solved either on *position, velocity or acceleration level*. [7] Another common way to categorize the level of redundancy resolution methods is either on kinematic level, that is in the first stage of a kinematic control strategy or at dynamic level, which e.g. might be used to modify the inverse dynamics control law, which is of interest for task space control and will be shown in more detail later on. [21]

A very common way for robot manipulators is to solve the redundancy on **Velocity Level**. In order to understand the behavior of redundancy on the velocity level, the mapping between the joint velocity space and the end-effector velocity space [20] by the Jacobian 2.1.2 has to be investigated. This mapping is illustrated in figure 2.3, where the right side represents the end-effector velocities v_{ee}. The *range* space

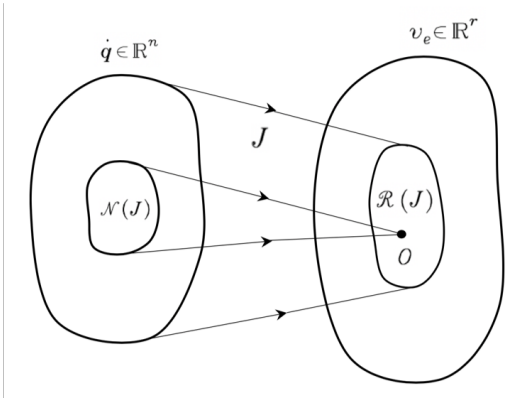

Figure 2.3: The mapping between the joint velocity space $\dot{q}$ and the end-effector velocity space v_{ee}. The subspace $\mathcal{R}(J)$ of the end-effector velocities that can be generated by the joint velocities, in the given manipulator posture. The subspace $\mathcal{N}(J)$ of joint velocities does not produce any end-effector velocity. [20]

$\mathcal{R}(J)$ of J is a subset of the end-effector velocities at a given manipulator posture that can be generated by the joint velocities $\dot{q}$. The left part in figure 2.3 covers the joint velocities $\dot{q}$. The *null* space $\mathcal{N}(J)$ of J is the subspace of joint velocities that do not produce any end-effector velocity. [20]

If the Jacobian has full rank, J spans the entire space of end-effector velocities. The existence of a subspace $\mathcal{N}(J) \neq 0$ for a redundant manipulator allows the determination of methods to handle the redundant degree of freedom R. [20] For the further investigations the differential kinematics equation is recapitulated.

$$v = J(q)\dot{q}$$

Assuming that the joint velocity vector $\dot{q}^\star$ is a solution to 2.1.2 for a given end-effector velocity of a redundant robot and P is a $(n \times n)$ matrix so that the joint velocity vector is

$$\dot{q} = \dot{q}^\star + P\dot{q}_0 \tag{2.3}$$

with arbitrary $\dot{q}_0$, the joint velocity vector 2.3 denotes a solution to equation (2.1.2) and yields

$$v_e = J\dot{q}^\star + JP\dot{q}_0 = J\dot{q} = J\dot{q}^\star \tag{2.4}$$

whereas $JP\dot{q}_0 = 0$ for any $\dot{q}_0$. [20] This result is key in terms of redundancy resolution since a solution of the kind 2.3 points out the possibility of choosing arbitrary joint velocities $\dot{q}_0$, in order to exploit the redundant degree of freedom R to increase the performance of the robot. The effect of $\dot{q}_0$ is to generate internal motions of the redundant manipulator that do not change the end-effector position and orientation. These internal motions might be used to reconfigure the manipulator into more dexterous postures for execution of a given task.[20]

The first methods considered, in order to solve redundancy on the velocity level are the *Jacobian-based* methods. According to [20] and [13] the inverse kinematics problem, shown in equation (2.2), can be rephrased with the weighting matrix W for redundant robots, where W is a suitable $(n \times n)$ symmetric positive definite weighting matrix so that the inverse kinematics is

$$\dot{q} = W^{-1}J^T(JW^{-1}J^T)^{-1}v_{ee} \tag{2.5}$$

where $W^{-1}J^T(JW^{-1}J^T)^{-1}$ is the weighted pseudoinverse $J^{\dagger}_W$. [13] It can be verified that this solution satisfies equation (2.1.2) by premultiplying both sides of the equation above with $J(q)$. A particular case occurs when the weighting matrix W is the identity matrix I and the solution simplifies to

$$\dot{q} = J^{\dagger}v_{ee} \tag{2.6}$$

with the definition of the matrix $J^{\dagger} = J^T(JJ^T)^{-1}$, which is called the right pseudo-inverse of J. [20] The limits of these Jacobian-based methods are, that they do not guarantee that singularities can be globally avoided during the execution of a given task. Furthermore, these methods typically lead to non-repeatable motion in the joint space. In other words, cyclic motion in the task space does not map to cyclic motion in joint space. [13]

It was pointed out for the velocity vector 2.3 above, that if $\dot{q}^{\star}$ is a solution to 2.1.2, $\dot{q}^{\star} + P\dot{q}_0$ denotes a solution as well, where $\dot{q}_0$ is a vector of arbitrary joint velocities and P is a projector in the null space of J. The combination of the condition 2.4 and the pseudo-inverse of J yields

$$\dot{q} = J^{\dagger}v_{ee} + (I_n - J^{\dagger}J)\dot{q}_0 \tag{2.7}$$

where the matrix $(I_n - J^{\dagger}J)$ is one of those matrices P introduced in 2.3 above, which allows the projection of the vector $\dot{q}_0$ in the null space of J. It has to be noticed

that a direct consequence is that in the case $v_{ee} = 0$, it is possible to generate internal motions described by $(I_n - J^{\dagger}J)\dot{q}_0$ that reconfigure the internal state of the manipulator without changing the end-effector position and orientation. [20] This method is part of the *null-space methods* and is called *projected gradient* method. [13][7] In order to exploit the redundant degree of freedom, the vector $\dot{q}_0$ has to be specified. A typical choice is

$$\dot{q}_0 = k_0 \Big(\frac{\partial w(q)}{\partial q} \Big)^T = \nabla_q w \tag{2.8}$$

where $k_0 > 0$ and $w(q)$ is a differentiable (secondary) objective function of the joint variables. Since the solution moves along the direction of the gradient of the objective function, it attempts to maximize it locally, compatible to the primary objective (kinematic constraint). [20] Typical objective functions in order to exploit the redundancy of a manipulator are the following:

- **Manipulability** is a measure of quality of the internal configuration of a robot. Maximization of the manipulability index attains the maximum distance to singularities. The manipulability measure is introduced by [27] as

  $$\omega(q) = \sqrt{det(J(q)J^T(q))}$$

  for redundant kinematics and used by [13] and [20] for manipulator redundancy resolution, as well as by [1] for mobile manipulator redundancy resolution.

- **Joint Range** is a representation of the distance from mechanical joint limits. The objective is to minimize the 'distance' from the mid points of the joint ranges by

  $$\omega(q) = -\frac{1}{2n} \sum_{i=1}^{n} \left(\frac{q_i - \bar{q}_i}{q_{iM} - q_{im}} \right)^2$$

  where q_{iM} denotes the maximum and q_{im} the minimum joint limit and $\bar{q}_i$ the middle value of the joint range. The objective function is formulated negative in order to be able to minimize this distance by maximizing the cost function. Accordingly, the redundancy is used to keep the joint variables as close as possible to the center of their ranges. [13][20]

- **Obstacle Avoidance** , also called clearance, is maximizing the minimum distance to obstacles in cartesian space. This requires to detect obstacles and

measure the distance between the robot and the object with a sensor. A common way to do that for MM is to use the laser sensor reading of the mobile base. The objective function is defined as

$$\omega(q) = \min_{p,o} ||p(q) - o||$$

where o is the position vector of a suitable point on the obstacle, and p is the position vector of the closest point along the outer structure of the robot in the direction to the obstacle. Maximizing this distance implies that the redundancy can be used within certain limits to avoid collision of the robot with an obstacle while maintaining the desired end-effector position. [13][20]

Accordingly, the projected gradient method is formulated as

$$\dot{q} = J^{\dagger} v_{ee} + (I_n - J^{\dagger} J)\dot{\nabla}_q w \tag{2.9}$$

It has to be noted that null space methods, in particular, the projected gradient method, still require the pseudoinverse which is computationally intensive. Furthermore, the complexity of the redundancy resolution method should only depend on the redundant Dofs $R = N - M$.[13]

The *reduced gradient* as described in [7] for non-holonomic mobile manipulators uses a decomposition of the joint space into base q_b and arm q_a of the MM, such that $J_a(q)$ is non-singular. The variables q_b can be chosen independently and used for optimizing an objective function. The reduced gradient method is defined as

$$\dot{q} = \begin{pmatrix} q_a \\ q_b \end{pmatrix} = \begin{pmatrix} J_a^{-1} \\ 0 \end{pmatrix} v_{ee} + \begin{pmatrix} J_a^{-1} J_b \\ I \end{pmatrix} \Big(- \big(J_a^{-1} J_b\big)^T I \Big) \dot{\nabla}_q w \tag{2.10}$$

This method is analytically simpler and numerically faster than the projected gradient, but requires the search for a non-singular minor of the robot Jacobian $J_a(q)$. [7] [13]

Instead of solving the redundancy on velocity level it could be of interest to solve it on **Acceleration Level** also named dynamic level. [21] Therefore, the time derivative of equation (2.1.2) is used to represent the mapping between accelerations in task space and joint space, which can be formulated as

$$\alpha = J^{-1}(q)(a - \dot{J}(q, \dot{q})\dot{q}) \tag{2.11}$$

where a is representing the end-effector acceleration in cartesian space and $\alpha = \ddot{q}$ the joint acceleration. [21] Using the weighted pseudo inverse of the Jacobian $J_W^{\dagger}(q)$, the relation can be formulated as

$$\ddot{q} = J_W^{\dagger}(q)(a - \dot{J}(q, \dot{q})\dot{q}) + \ddot{q}_0 \tag{2.12}$$

where $\ddot{q}_0$ denotes a joint acceleration vector within the null space of J which might be exploited to meet additional objectives, similar to the null space method on velocity level. [21][13] Thus, $\ddot{q}_0$ can be chosen as

$$\ddot{q}_0 = \nabla_q w - K_D \dot{q} \tag{2.13}$$

where $K_D\dot{q}$ is necessary to damp and stabilize self-motions of the robot in null space $\mathcal{N}(J)$ and K_D is chosen as $K_D > 0$. A typical objective on acceleration level, for instance, is to minimize the torque norm.[13]

To cover all possible levels of redundancy resolution, the last one missing is solving redundancy on **Position Level**. One method is the decomposition of a redundant system in task space by defining the relation between subsystems. The relation between the defined subsystems can be described by suitable parameters that represent the geometrical meaning of the redundant degree of freedom R. These parameters are called redundancy parameters and introduced by [19] for mobile manipulators and generalized by [1] to model the redundancy of mobile manipulators

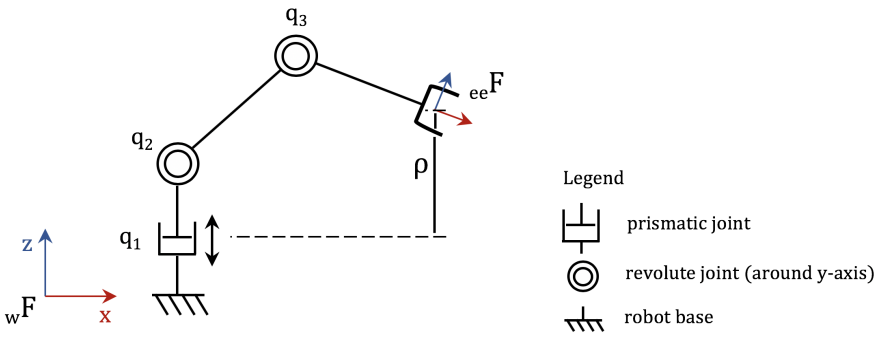

Figure 2.4: Defining the relationship between the height z of the prismatic joint q_1 and the end-effector by the redundancy parameter ρ. Assigning a value to ρ solves the redundancy of the robotic arm.

and to describe the relation between the mobile base and robot arm. By varying the redundancy parameters it is possible to change the internal configuration, i.e. the state of the robot, without any impact to the end-effector position and orientation. [1]

Figure 2.4 illustrates an example of defining redundancy parameters, where the kinematic structure is redundant in the z-direction, since the prismatic joint and the arm is able to move in the z-direction. Thus, the redundancy parameter ρ describes the geometrical relation between the height of q_1 and the end-effector. By knowing ρ and the desired end-effector position it is possible to determine $q_1 = p_{ee} - \rho$ and as a result the values of q_2 and q_3 by applying the method of inverse kinematics.

The robot redundancy can be formulated as a general optimization problem, where the optimal values of the redundancy parameters can be found by solving the problem based on the desired objectives. [1] Objectives for manipulator and mobile manipulator redundancy resolution on position level are the same, as mentioned above for solving redundancy on velocity level. An additional objective for mobile manipulators is

- **Reduction of the base movement** in order to improve the stability and precision of the end-effector placement during manipulation tasks [1], where the objective function is formulated as

$$\omega(q) = |\bar{q}_b - q_b|$$

with $\bar{q}_b$ as the current state of the base and q_b as the desired state of the base. This can be argued based on the position accuracy of omnidirectional wheel technology using macanum wheels e.g. in case of the kuka KMR quantec with +-5mm. [10] In comparison, the repeatability of a manipulator, e.g. the UR10 from Universal robotics is +-0.1mm. [24] Considering this fact, the objective is to execute as much as possible of the desired trajectory tracking within the workspace of the manipulator with the arm and minimize the base movement in order to achieve a higher accuracy in trajectory tracking

Another method to solve the redundancy on position level is to use machine learning methods to learn the inverse kinematics of a robot, taking the redundancy into account. The approach presented by [5] learns the direct inverse kinematics function on position level using structured output learning methods.

2.1.3 Mobile Base

A mobile base can be designed with legs or wheels. However, wheels are the most popular locomotion mechanism in mobile robotics due to its efficiency and simple mechanical implementation compared to legged locomotion. Another advantage is keeping the balance of the base using three or more wheels, that makes the base intrinsically stable. [22] A base can be equipped with wheels out of different types of wheel classes, as introduced in [22] and [20]. Wheels can be actuated or non-actuated, as well as steerable or non-steerable. [22] provides an overview of mobile base designs with different wheel configurations. Depending on the wheel configuration the mobile robot is non-holonomic constraint or omnidirectional. A non-holonomic mobile base is constrained on velocity level and doesn't allow direct position control. An example is a car with front-wheel drive, using two actuated and steered wheels in the front and two non-actuated wheels in the rear. [22] An omnidirectional mobile robot, instead, is able to move at any time in any cartesian direction along the ground plane and reorientate itself on the spot. A representative for this kind of mobile robot is a base with four menacum wheels that are actuated and non-steerable. Mecanum wheels, also called swedish wheels, have a set of free rollers along the wheel perimeter. These free rollers are usually mounted with an angle of 45° on the wheel, which allows the wheel to move in more than one direction. [22][20] A mecanum wheeled base, as illustrated in figure 2.5, is described and kinematically modeled in [18].

Definition 3. In this thesis, a wheeled mobile base shall be used. The mobile base is defined as omnidirectional and hence, the input parameters are represented by $q_b = (x, y, \theta)$.

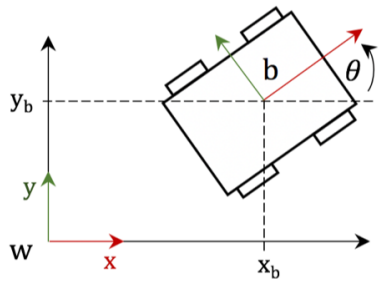

Figure 2.5: Definition of the pose $q_b = (x, y, \theta)$ of an omnidirectional mobile base equipped with mecanum wheels [18]

2.2 Motion Control

Motion control of a robot describes the problem of tracking a trajectory prescribed to the end-effector. This robot motion can either be unconstrained or constrained. A unconstrainted motion describes the movement in space without any physical interaction between the end-effector and the environment. The motion is considered as constrained when the end-effector is in contact with an object and a contact force arises. The execution of a robot task requires tracking of a trajectory with respect to the end-effector and hence, a specific motion of each actuated joint of the robot. The task of the motion controller is to allow the robot system to track trajectories, by providing the joint actuators the corresponding commands. The joint actuator commands are formulated as joint torques ensuring that the end-effector attains the desired position and orientation. [21]

2.2.1 Dynamic Modeling

The kinematic and dynamic modeling of the robot is the foundation of the chosen control strategy. [21] The modeling of a kinematic structure, using DH-parameters, is already introduced in subsection 2.1.1 above. An introduction to the dynamic model shall follow now, which describes the relationship between the forces and torques, exerted on the structure of the robot, as well as the joint positions, velocities, and accelerations. A common method to model the dynamics of a robot is to use the Lagrangian formulation, described in [20] and [21]. The *Lagrangian* is given by the difference of the kinetic energy E_{kin} and potential energy E_{pot} of the robot, as represented by

$$L(q_i, \dot{q}_i) = E_{kin} - E_{pot} \tag{2.14}$$

The *Lagrangian equation* is calculating the generalized forces τ_i of each joint q_i by partially deriving the Lagrangian $L(q_i, \dot{q}_i)$ and is defined as

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i \tag{2.15}$$

Substituting the relevant equations of the potential and kinematic energy of the robot into equation (2.14) and applying the Lagrangian equation (2.15) for each joint, leads to the equation of motion that describes a dynamic model of the robot and the relation between the joint position q, velocity $\dot{q}$, acceleration $\ddot{q}$ and the generalized forces and torques τ [21]

$$M(q)\ddot{q} + C(q, \dot{q}) + G(q) = \tau \tag{2.16}$$

where $M(q)$ is an invertible inertia matrix, $C(q, \dot{q})$ a vector of Coriolis and centrifugal torques and $G(q)$ a vector of gravity torques. By adding the torque τ_e due to external forces to the equation of robot motion, it can be written as

$$M(q)\ddot{q} + C(q, \dot{q}) + G(q) = \tau - \tau_e \tag{2.17}$$

where τ_e is formulated as $\tau_e = J^T(q) f_{ext}$, by applying the principle of virtual work. [21][23]

2.2.2 Task Space vs. Joint Space

Having modeled the robot entirely, the actual control scheme for motion can be designed. Motion control can be divided into two major control methods. Task specifications, e.g tracking predefined trajectories, are usually described in task space. Thus, control actions can be performed in task space as well, which leads to the definition of *task space control*. The inverse kinematics, specified in equation (2.2) above, allows transforming the task space commands into joint space commands. By performing the control actions in joint space the method is called *joint space control*. [20]

Joint space control is divided into two steps. As a first step, the manipulator inverse kinematics is solved in order to transform the motion commands $\dot{p_d}$, defined in task space, into the corresponding desired motion of each joint $\dot{q_d}$ in joint space. The second step is a joint space control scheme, allowing the actual motion of $\dot{q_c}$ to track the desired motion $\dot{q_d}$. It has to be pointed out, that this method has a weighty

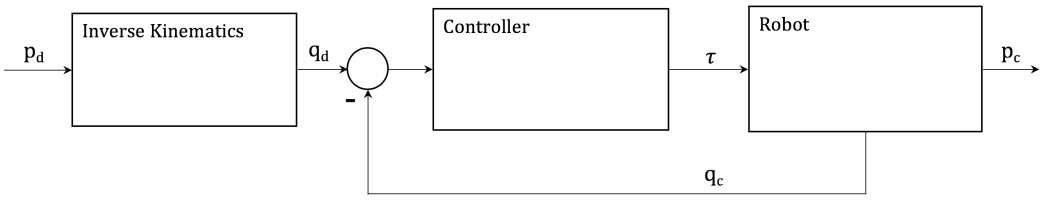

Figure 2.6: A general scheme of joint space control where the control action is applied to the actual joint positions q_c of the robot [20]

drawback. The control scheme in joint space does not influence the end-effector position p_{ee} in operational space, which is controlled in an open-loop fashion based on the mechanical structure of the robot. Accordingly, any uncertainty of the mechanical structure or lack of knowledge about the robot model leads to a loss of accuracy of the actual end-effector position in cartesian space. [20] The basic principle of joint space control is illustrated in figure 2.6.

As already indicated, the *operational space control* method applies the control action on the task space variables where the inverse kinematics is now embedded in the feedback control loop. The concept of task space control is shown in figure 2.7, where it can be noticed that the control action is applied to the actual position of the robot. However, the conceptual advantage in terms of the possibility of acting directly on task space variables is only a potential advantage, since the measurement of the end-effector task space position is very often not performed directly. Instead, the joint space variables are measured and transformed into task space by applying the direct kinematics functions. [20]

According to [21], task space control methods are suitable to extend the motion control scheme with force control approaches, which is of particular interest in this work. The reason for this is that measured forces, exerted on the robot end-effector, are values expressed in task space directions. Accordingly, the application of both, the position control action and the force control action to the robot has to be done in the operational space. Thus, only operational space control methods will be considered in the further course of this chapter.

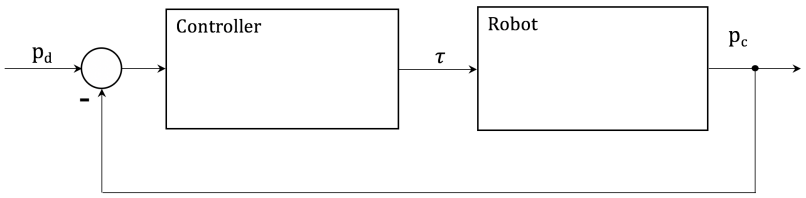

Figure 2.7: A general scheme of task space control where the control action is applied to the actual position p_c of the robot [20]

2.2.3 Inverse Dynamics Control

A common method to solve the motion control problem of a robot system in task space is *Inverse Dynamics Control*, that uses the dynamic equation (2.17), which represents a complex non-linear system. The idea of the inverse dynamics control method is to find a control vector u in order to realize a linear input/output relationship. Due to the form of equation (2.17) and the fact that $M(q)$ is invertible, it is guaranteed to find such a linearized controller based on the non-linear inverse dynamics control law

$$u = M(q)\alpha + C(q, \dot{q}) + G(q) + J^T(q) f_{ext} \tag{2.18}$$

where the system reduces to the linear relation

$$\ddot{q} = \alpha \tag{2.19}$$

where α represents the input to the system as an acceleration of each joint. [20][23] The motion control problem is now reduced to find a stabilizing control law for α as an outer loop control action. In joint space, this can be expressed as

$$\alpha = K_P(q - q_c) + K_D(\dot{q} - \dot{q_c}) + \ddot{q} \tag{2.20}$$

where K_P and K_D are diagonal gain matrices of the type $diag\{K_1, ...K_n\}$ and q_c represents the vector of the actual joint position and q the desired joint position with the corresponding derivatives. However, in order to be able to apply a task space control method, the control input a in task space has to be converted into the input α in joint space of the inverse dynamics control law. The relation between accelerations in joint space $\ddot{q}$ and task space a is defined by 2.11 as $\alpha = J^{-1}(q)(a - \dot{J}(q, \dot{q})\dot{q})$. Thus, the stabilizing control law in task space is formulated as

$$a = -K_P(p - p_c) - K_D(\dot{p} - \dot{p_c}) + \ddot{p} \tag{2.21}$$

representing a PD-controller and can be used as input for the inverse dynamics, without the need to compute a joint space trajectory or modify the non-linear inverse dynamics control law. The variables K_P and K_D are again the diagonal gain matrices and p the desired as well as p_c the actual end-effector position. [21] [20] [23]

2.3 Force Control

One of the fundamental requirements to enable an interaction between the manipulator and the environment is to control the contact force that arises at the manipulator end-effector. During the interaction, the environment sets constraints on the trajectory that can be followed by the end-effector. Thus, this is considered as constrained motion, as introduced in section 2.2. [20][23] Using motion control to enable the interaction between the manipulator and the environment would require to have an accurate model of the robot (kinematics and dynamics) and the environment (geometry and mechanical features). As soon as the model deviates from the real robot and the geometry of the environment a planning error arises. This planning error would either cause the end-effector to leave contact with the manipulated object or rise the contact force since the control system would react to reduce the deviation of the desired trajectory. In the case of the latter, this would create high values of contact force that stresses both the robot and the manipulated object and even could cause damage. Therefore, an appropriate force control strategy has to be applied, which modifies the end-effector trajectory based on the sensed contact force. [23][20]

In general, there are three types of sensors to measure the force, wrist force sensors, joint torque sensors, and tactile or hand sensors. For controlling the interaction between the robot end-effector and the environment, the six-axis wrist sensor usually gives the best results according to [23]. However, the quality of a force sensor depends on the quality of its signal what usually correlates with its cost. [11]

Definition 4. In order to control the interaction between the robot and environment, the manipulator shall be equipped with a six-axis wrist sensor to measure the forces and torques acting on the end-effector.

Force control strategies can be grouped into *direct force control* and *indirect force control*. A representative control method for indirect force control is *Compliance control* and *impedance control*, where the force control is achieved via motion control without an explicit force feedback loop. Instead, direct force control allows controlling the contact force to a desired value, due to the closure of a force feedback loop. Representative control methods for direct force control are *hybrid position/-force control* and *motion and force control*, which is based on an inner/outer loop control scheme. [21][20] In the following, the three different force control methods shall be investigated more detailed.

2.3.1 Compliance and Impedance Control

Compliance control is designed to achieve a desired static behavior of the interaction. Instead, impedance control performs a dynamic model-based compensation, where the position error is related to the contact force through an impedance of adjustable parameters. The impedance control law describes a mass-spring-damper system with the contact force as input. [21] Therefore, a modified inverse dynamics control law is considered where the outer loop stabilizing control law 2.21 is formulated as

$$a = \ddot{p_d} + K_{Mp}^{-1}\Big(K_{Dp}(\dot{p_d} - \dot{p_c}) + K_{Pp}(p_d - p_c) - f\Big) \tag{2.22}$$

with K_{Mp} as a positive defined gain matrix and f the measured contact force. This impedance control law can be rewritten to the relation of a mass-spring-damper system, defined as

$$f = K_{Mp}(\ddot{p_d} - a) + K_{Dp}(\dot{p_d} - \dot{p_c}) + K_{Pp}(p_d - p_c) \tag{2.23}$$

where in this particular case a represents the actual end-effector acceleration and f is the measured force with the assumption of a measurement without error. It shall be noted that neither compliance nor impedance control does allow to control the contact force to the desired value. [21]

2.3.2 Hybrid Position/Force Control

As part of the direct force control methods, hybrid position/force control can be applied if a detailed model of the environment is available. This method is controlling the end-effector position along the unconstrained directions in task space and the contact force along the constrained directions. [21] The basic idea of hybrid position/force control is an architecture that associates the constraints of the desired task to the controller design. The key component is a compliance selection vector S that specifies which degrees of freedom in cartesian space are under force control ($S_i = 1$), and which are under position control ($S_i = 0$). Assuming the compliance selection vector as $S = [0, 0, 1, 0, 1, 1]$, the respective joint torque for each joint of the robot can be calculated as

$$\tau_i = \Psi_{i1}(\Delta x_1) + \Psi_{i2}(\Delta x_2) + \Gamma_{i3}(\Delta f_3) + \Psi_{i4}(\Delta x_4) + \Gamma_{i5}(\Delta f_5) + \Gamma_{i6}(\Delta f_6) \tag{2.24}$$

where Γ is representing a force compensation function and Ψ a position compensation function, as well as Δx_i and Δf_i the position and force error in the respective direction and orientation in cartesian space. [16] The actuator control signal τ_i has in this example six components, one for each force controlled degree of freedom in cartesian space, and one for each position controlled degree of freedom. It has to be pointed out, that in this case the position and orientation in task space are considered.[16] This method can be used typically for planar contact surfaces, instead, for curved contact surfaces the constraint equations have to be considered. [21]

2.3.3 Motion and Force Control

In case, a detailed model of the environment is not available, motion and force control is another approach as part of the direct force control methods, where control of both, force and motion, is executed in all task space directions. The basic idea is to have an outer force control loop closed around an inner motion control loop, to allow motion control along the unconstrained directions and adapt the motion along the constraint directions to perform force control. The resulting *parallel control* is designed to dominate the inner motion control loop to ensure force control along the constrained directions. [21]

The idea of parallel control is the composition of the compliant position p_f with the desired position p_d to obtain the reference position with

$$p_r = p_f + p_d \tag{2.25}$$

referred to as parallel composition. The reference position p_r can then be used as input of the outer loop stabilizing control law of the inverse dynamics control method [cf. Eq.(2.21)] and formulated as

$$a = \ddot{p_d} + K_D(\dot{p_d} - \dot{p_c}) + K_P(p_r - p_c) \tag{2.26}$$

The parallel composition also can be modified with the velocity $\dot{p}_r = \dot{p}_f + \dot{p}_d$ and acceleration $\ddot{p}_r = \ddot{p}_f + \ddot{p}_d$ to obtain

$$a = \ddot{p_r} + K_D(\dot{p_r} - \dot{p_c}) + K_P(p_r - p_c) \tag{2.27}$$

The compliant position p_f comprises the position offset in constrained directions to adapt the movement of the robot end-effector in order to obtain the desired con-

tact force f_d. Therefor p_f and the respective derivatives are determined to minimize the force error

$$\Delta f = f_d - f \tag{2.28}$$

This task is done by the outer loop force controller, where [21] presents different approaches to minimize the force error, e.g. by using a basic PI-controller or obtain p_f by solving the differential equation

$$K_A \ddot{p_f} + K_V \dot{p_f} = \Delta f \tag{2.29}$$

where K_A and K_V are diagonal gain matrices. The drawbacks of these approaches are practical issues like a not optimal model-based compensation or to slow measurements of the contact force. Therefore, [21] introduces advanced force and position control schemes with the idea of parameter adaptation and output feedback. Similar to that, [11] presents a controller that concurrently adapts feedforward force and impedance parameters to compensate for the interaction with the environment and adapts the desired trajectory when interacting to maintain the contact force at the desired level.

Chapter 3

Modeling

In order to develop a control approach for a mobile manipulator with the objectives described in section 1.1 the physical behavior of the system itself has to be mathematically described and modeled. This model consists of the kinematics, describing the motion of the robot and the dynamics, considering the occurring forces and torques. First, a simplified mobile manipulator with 3-Dof in the 2D-plane is described, suitable for Matlab simulations of the controller to verify the concept. Second, a mobile manipulator with 6-Dof in 3D-space, as schematically shown in figure 3.1, is modeled. The modeling in this chapter only considers the end-effector position in task space without the orientation. Since the selected redundancy resolution approach might be part of the kinematic modeling of the robot, the outcome of using redundancy parameters, as introduced in 2.1.2, is anticipated. Details about

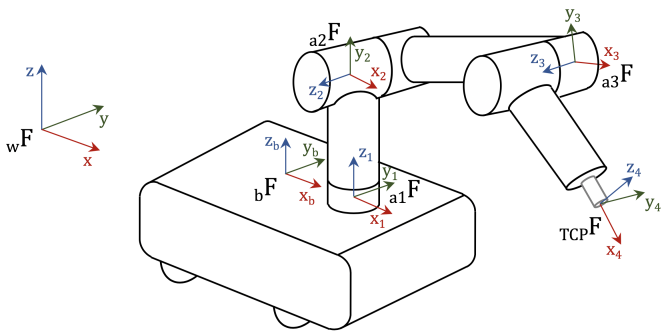

Figure 3.1: A generalized model of a mobile manipulator with a three-link arm and mecanum wheeled base. The world frame F_w is the reference for the mobile manipulator position. The base position is represented by F_b and the end-effector or TCP position by F_{TCP}

the selection and design of this particular redundancy resolution approach will be explained as part of the proposed control architecture in 4.1.

This chapter is structured as follows. First, the simplified 3-Dof MM is kinematically and dynamically modeled. This includes the definition of redundancy parameters, introduced in 2.1.2. In addition, a model of the environment is presented, appropriate to test the behavior of the interaction between manipulator and objects. Thereafter, the kinematics of a 6-Dof mobile manipulator, as well as the redundancy parameters are described.

3.1 3-DoF Model in 2D-Plane

The 3-Dof Model in 2D-plane is representing a simplified mobile manipulator system, used for simulations in order to test control approaches during the development phase. The model consists of a two-link arm and a mobile base. The arm, as shown in figure 3.2, consists of two revolute joints, able to reach every position in the 2D-plane, within its workspace. The base is able to move in x-direction without any physical limits. The relation between both components, in particular, the end-effector position and the base position is described by the redundancy parameters.

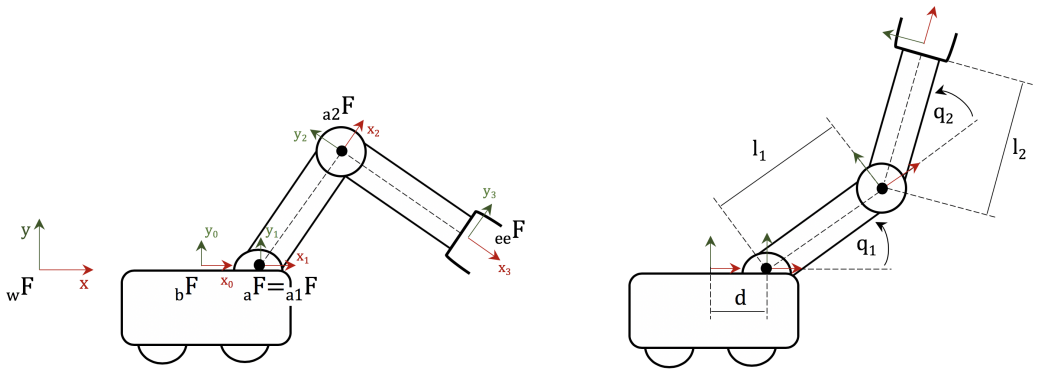

Figure 3.2: Model of the mobile manipulator with two link arm. F_b represents the base frame associated with the base position and $F_e e$ the end-effector frame of the end-effector position. d represents the distance between F_b and the base of the two link arm F_a. The link length of the arm is represented by l_i.

3.1.1 Kinematic Model

Before modeling the mobile manipulator, a closer look should be taken at the kinematic redundancy. A 3-Dof mobile manipulator has $N = 3$ joints, considering the arm as two revolute joints and the base as a prismatic joint without limitation. Thus, the number of dimensions in joint space is $N = 3$. As defined, the model doesn't consider the end-effector orientation, which leads to an operational space $M = 2$. Accordingly, the system has a redundancy of $R = 3 - 2 = 1$ degree of freedom. The end-effector position in the operational space is defined as $p_{ee} = (x_{ee}, y_{ee})^T$. A schematical model of the redundant robot is illustrated in figure 3.2. First, the end-effector position relative to the base position ${}_bp_{ee}$ is described, which is defined as ${}_bp_{ee} = [{}_bx_{ee}, {}_by_{ee}]^T$. The DH-notation is used to determine the forward transformation ${}^bT_{ee}$ of the robot. The frames, beginning with the base frame over each joint up to the end-effector frame, are shown in figure 3.2. Further information about the DH-parameter is listed in the appendix A.0.1. The state of the robot is represented by the joint configuration $q = [q_a, q_b]^T$ where $q_b = [x_b]$ represents the base position and $q_a = [q_1, q_2]^T$ the joint configuration of the arm. Based on the forward transformation ${}^bT_{ee}$ the kinematic equations of the 2-link arm are

$$\begin{aligned} {}_bx_{ee} &= l_1 cos(q_1) + l_2 cos(q_1 + q_2) \\ {}_by_{ee} &= l_1 sin(q_1) + l_2 sin(q_1 + q_2) \end{aligned} \tag{3.1}$$

As introduced in section 2.1.1 by equation (2.1.2), the Jacobian describes the relation between the end-effector velocity space and the joint velocity space and is the partial derivative of the end-effector position ${}_bp_{ee}$ by the joint vector q_a, as

$$J(q) = \begin{bmatrix} \frac{\partial_b x_{ee}}{\partial q_1} & \frac{\partial_b x_{ee}}{\partial q_2} \\ \frac{\partial_b y_{ee}}{\partial q_1} & \frac{\partial_b y_{ee}}{\partial q_2} \end{bmatrix} = \begin{bmatrix} -l_1 sin(q_1) - l_2 sin(q_1 + q_2) & -l_2 sin(q_1 + q_2) \\ l_1 cos(q_1) + l_2 cos(q_1 + q_2) & l_2 cos(q_1 + q_2) \end{bmatrix} \tag{3.2}$$

After the end-effector position regarding the base is kinematically described, the relation between the end-effector position and the world frame has to be characterized. Accordingly, the equations representing the end-effector position with respect to the world frame are

$$\begin{aligned} {}_wx_{ee} &= x_b + l_1 cos(q_1) + l_2 cos(q_1 + q_2) \\ {}_wy_{ee} &= l_1 sin(q_1) + l_2 sin(q_1 + q_2) \end{aligned} \tag{3.3}$$

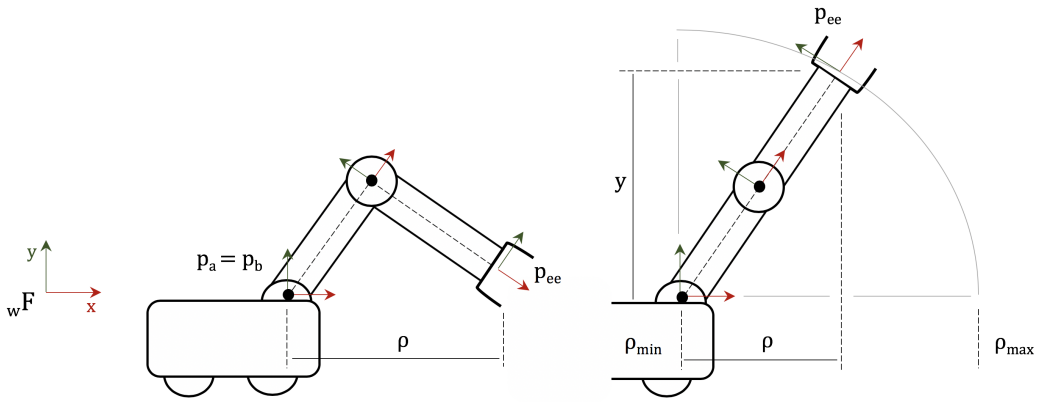

Figure 3.3: Introducing the redundancy parameter ρ on the left side of the figure, which describes the distance between x_{ee} and x_b. The value of ρ is constraint by the maximum stretched length of the manipulator and the desired height y_{ee}, as illustrated on the right side.

In order to solve the inverse kinematics of the mobile manipulator, as described in subsection 2.1.1, and determine the joint configuration q based on a desired end-effector position ${}_wp_{ee}$ the redundancy between q_a and q_b has to be solved, since a change of the manipulator state q_a, as well as a change of the base position x_b will result in a change of the end-effector position x_{ee}. According to subsection 2.1.2, one way is to use a redundancy parameter ρ, where ρ defines the relationship between the end-effector position ${}_bx_{ee}$ and base position ${}_wx_b$, to solve the inverse kinematics for each component independent. Figure 3.3 illustrates the defined redundancy parameter ρ. Hence, ρ is mathematically defined as

$$\rho = {}_bx_{ee} = a_1 cos(q_1) + a_2 cos(q_1 + q_2) \tag{3.4}$$

Accordingly, the base position regarding the world frame can be determined with

$${}_wx_b = {}_wx_{ee} - \rho \tag{3.5}$$

However, ρ can't be chosen arbitrarily, since it is reliant on the physical limitations of the robot manipulator. Thus, the value of ρ has to be within the workspace of the manipulator, which is a constraint for the described redundancy resolution, as shall be seen in section 4.1. This limitation is set by two parameters. First, the maximum stretched lenght of the manipulator (l_1+l_2) and second, the desired height y_{ee} as

$$0 > \rho < \|\sqrt{(l_1 + l_2)^2 - y^2}\| \tag{3.6}$$

The result is a circular limitation, corresponding to the physical limitations of the robot arm workspace, as illustrated on the right side in figure 3.3. In order to avoid a boundary singularity, this limit has to be set slightly smaller than the actual workspace limitation.

3.1.2 Dynamic Model

After the kinematic structure is modeled, the focus lies on the required torques and forces in order to obtain the desired motion. This dynamic behavior requires an additional model of the robot. As introduced in 2.2.1, the Lagrangian formulation is a common method, in order to determine the dynamics of the mobile manipulator model. Therefore, the respective equations are recapitulated. The *Lagrangian* is expressed by the difference of the potential and kinetic energy function, as represented by equation [cf. Eq.(2.14)]

$$L(q_i, \dot{q}_i) = E_{kin} - E_{pot}$$

The *Lagrangian equation* is calculating the generalized forces τ_i, associated with the generalized coordinate q_i, as [cf. Eq.(2.15)]

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i$$

Applying equation (2.14) to (2.15), yields to the following relation

$$\frac{d}{dt}\frac{\partial E_{kin} - E_{pot}}{\partial \dot{q}_i} - \frac{\partial E_{kin} - E_{pot}}{\partial q_i} = \tau_i \tag{3.7}$$

Since $\frac{\partial E_{pot}}{\partial \dot{q}_i} = 0$, equation (3.7) can be simplified to

$$\frac{d}{dt}\frac{\partial E_{kin}}{\partial \dot{q}_i} - \frac{\partial E_{kin}}{\partial q_i} + \frac{\partial E_{pot}}{\partial q_i} = \tau_i \tag{3.8}$$

where $\frac{\partial E_{pot}}{\partial q_i}$ is representing the generalized forces caused by gravity. Accordingly, this can be summarized in a gravity vector $G(q)$. Calculating equation (3.8) for each joint yields to the generalized form of the dynamic model [cf. Eq.(2.17)], where $M(q)$ is representing the inertia of the system and $C(q, \dot{q})$ the Coriolis and centrifugal forces.

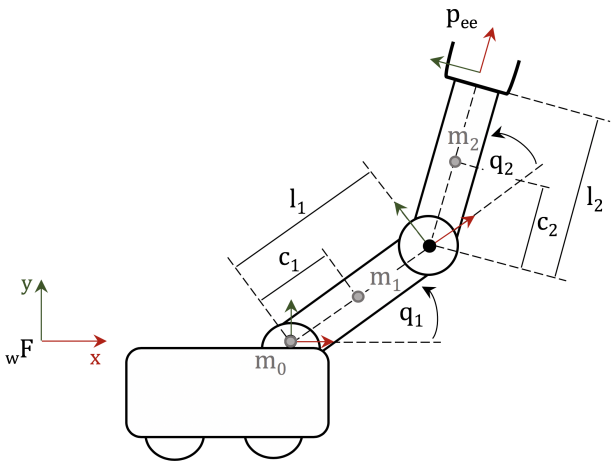

Figure 3.4: Model of the mobile manipulator with a two-link arm, where the center of mass m_0 is aligned with the axis of joint q_1. The mass m_1 and m_2 are placed in the center of the respective link. The distance between the joint axis and the center of mass is described by c_i

$$M(q)\ddot{q} + C(q,\dot{q}) + G(q) = \tau$$

with $q = [q_a, q_b]^T$ and $\tau = [\tau_{q_a}, \tau_{q_b}]^T$. In case of the 3-Dof mobile manipulator, figure 3.4 illustrates a schematical model including the mass m_i of the base and each link. This allows determining the kinematical equations of each center of gravity of each rigid structure, required to determine the *potential energy* and *kinetic energy* of each component.

The *potential energy* considers the mass of link one and two since the base has no contribution to the potential energy. Thus, the equation is defined as

$$E_{pot} = m_1 g y_{m_1} + m_2 g y_{m_2} \tag{3.9}$$

where the height y_i of each center of gravity of m_1 and m_2 is calculated with

$$\begin{aligned} y_{m_1} &= c_1 sin(q_1) \\ y_{m_2} &= l_1 sin(q_1) + c_2 sin(q_1 + q_2) \end{aligned}$$

The *kinetic energy* contains the mass m_i of each rigid body, as well as the inertia I_i. The kinetic energy of the 3-Dof mobile manipulator can be summarized by

$$E_{kin} = \frac{1}{2}m_0\dot{x_b}^2 + \frac{1}{2}m_1(\dot{x_1}^2 + \dot{y_1}^2) + \frac{1}{2}I_1\dot{q_1}^2 + \frac{1}{2}m_2(\dot{x_2}^2 + \dot{y_2}^2) + \frac{1}{2}I_2(\dot{q_1} + \dot{q_2})^2 \quad (3.10)$$

with the velocities of each center of gravity

$$\begin{aligned}
\dot{x}_{m_1} =& \dot{x_b} - c_1 sin(q_1)\dot{q_1} \\
\dot{y}_{m_1} =& c_1 cos(q_1)\dot{q_1} \\
\dot{x}_{m_2} =& \dot{x_b} - l_1 sin(q_1)\dot{q_1} - c_2 sin(q_1 + q_2)\dot{q_1} - c_2 sin(q_1 + q_2)\dot{q_2} \\
\dot{y}_{m_2} =& - l_1 cos(q_1)\dot{q_1} - c_2 cos(q_1 + q_2)\dot{q_1} - c_2 cos(q_1 + q_2)\dot{q_2}
\end{aligned}$$

where $\dot{x_b}$ represents the velocity of the mobile base and its mass respectively. Applying equation (3.8) with (3.9) and (3.10), under consideration of external forces, yields to the following equation of the generalized forces for each joint

$$M_3(q)\ddot{q} + C_3(q, \dot{q}) + G_3(q) + J^T(q)f_{ext} = \tau \quad (3.11)$$

where $q = [q_1, q_2, x_b]^T$ and $\tau = [\tau_{q_1}, \tau_{q_2}, \tau_{x_b}]^T$, as well as the inertia matrix is represented by

$$M_3(q) = \begin{bmatrix} M_{q_1q_1} & M_{q_1q_2} & M_{q_1x} \\ M_{q_2q_1} & M_{q_2q_2} & M_{q_2x} \\ M_{xq_1} & M_{xq_2} & M_{xx} \end{bmatrix}$$

$$\begin{aligned}
M_{q_1q_1} &= I_1 + I_2 + l_1^2 m_2 + c_1^2 m_1 + c_2^2 m_2 + 2l_1 c_2 m_2 cos(q_2) \\
M_{q_1q_2} &= I_2 + l_1 c_2 m_2 cos(q_2) + c_2^2 m_2 \\
M_{q_1x} &= -c_1 m_1 sin(q_1) - c_2 m_2 sin(q_1 + q_2) - l_1 m_2 sin(q_1) \\
M_{q_2q_1} &= I_2 + c_2^2 m_2 + l_1 c_2 m_2 cos(q_2) \\
M_{q_2q_2} &= I_2 + c_2^2 m_2 \\
M_{q_2x} &= -c_2 m_2 sin(q_1 + q_2) \\
M_{xq_1} &= -m_1 c_1 sin(q_1) - m_2 c_2 sin(q_1 + q_2) - m_2 l_1 sin(q_1) \\
M_{xq_1} &= -m_2 c_2 sin(q_1 + q_2) \\
M_{xx} &= m_0 + m_1 + m_2
\end{aligned}$$

the centrifugal and Coriolis vector by

$$C_3(q,\dot{q}) = \begin{bmatrix} C_{q_1} \\ C_{q_2} \\ C_x \end{bmatrix}$$

$$\begin{aligned} C_{q_1} &= -l_1 c_2 m_2 sin(q_2) \dot{q_2}^2 - 2 l_1 c_2 m_2 sin(q_2) \dot{q_2} \dot{q_1} \\ C_{q_2} &= l_1 c_2 m_2 sin(q_2) \dot{q_1}^2 \\ C_x &= -m_1 c_1 cos(q_1) \dot{q_1}^2 - m_2 l_1 cos(q_1) \dot{q_1}^2 - m_2 c_2 cos(q_1 + q_2)(\dot{q_1} + \dot{q_2})^2 \end{aligned}$$

as well as the gravity vector by

$$G_3(q) = \begin{bmatrix} G_{q_1} \\ G_{q_2} \\ G_x \end{bmatrix}$$

$$\begin{aligned} G_{q_1} &= g m_2 (c_2 cos(q_1 + q_2) + l_1 cos(q_1)) + g m_1 c_1 cos(q_1) \\ G_{q_2} &= g m_2 c_2 cos(q_1 + q_2) \\ G_x &= 0 \end{aligned}$$

3.1.3 Environment and Contact Forces

There are two different methods to model the dynamic interaction with the environment, in particular, the contact forces arising between the robot end-effector and the manipulated object in the environment. On the one hand, the *hard contact method* is treating the interaction with the environment, i.e. the event of contact, as a kinematic constraint without any compliant behavior. On the other hand, the *soft contact method* represents the interaction between the robot and the environment by force elements, in particular, a spring-damper system. [17][20]

Definition 5. In the further course of this thesis and modeling of the environment, the soft contact method is considered.

In order to model the contact and calculate the related force, the point of contact p_{c_0} with the environment has to be identified first. A spring-damper system considers the position offset $p_c - p_{c_0}$ and velocity $\dot{p_c}$ in the situation of contact and defines the contact force as

$$f_c = k_s(p_c - p_{c_0}) + k_d \dot{p_c} \tag{3.12}$$

considering that the event of contact only happens if $p_c - p_{c_0} > 0$ otherwise $f_c = 0$. In order to represent the compliance of the environment an appropriate spring and damping parameter has to be chosen. [11] proposes to represent a rigid environment with $k_s = -1000N/m$. To simplify the model, the damping shall be defined $k_d = 0$. It is considered that representing the interaction with force elements is not very precise in order to represent the environment physically correct. However, it enables to validate in simulation a basic behavior of the interaction between robot and environment in constraint directions. [17] It has to be pointed out that the model also doesn't consider any kind of contact friction along the unconstrained directions.

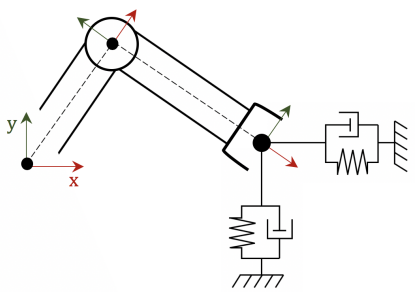

Figure 3.5: Model of the compliant environment using the soft contact method to obtain contact forces represented by a spring-damper system in x- and y-direction.

3.2 6-Dof Model in 3D-Space

The 6-Dof Model in 3D-Space represents a real mobile manipulator system, used for experiments in order to validate control methods. The mobile base is a mecanum wheeled omnidirectional platform, allowing to arbitrarily position and orientate the platform on the ground plane in cartesian space. The manipulator consists of three links with actuated revolute joints. It shall be mentioned that only the first three joints of a classical 6-Dof industrial robotic arm are considered, the wrist joints are defined as mechanically fixed and are not considered in the further modeling. A schematical model of this particular mobile manipulator system has been already presented at the beginning of this chapter in figure 3.1.

3.2.1 Kinematic Model

Based on the description above, extending the kinematic model from the 2D-plane to 3D-space is adding three additional degrees of freedom to the mobile manipulator. The base has 3-Dof, the manipulator is extended with an additional revolute joint and accordingly has 3-Dof either. Thus, the number of dimension of the joint space is $N = 6$. As defined for the model in the 2D-plane, the model in 3D-space also doesn't consider the end-effector orientation, which leads to a number of dimensions

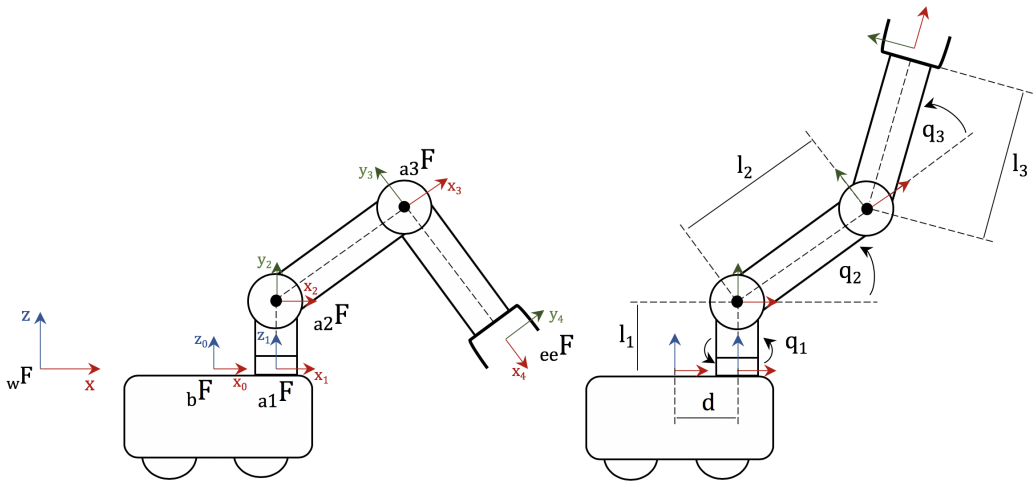

Figure 3.6: Kinematic model of a 6-Dof mobile manipulator in the xz-plane. The distance between the center of the mobile platform F_b and the base of the arm F_a is represented by the parameter d. The length of each link is represented by l_i and the joint angles by q_i

in the operational space of $M = 3$. Accordingly, the system has a redundancy of $R = 6 - 3 = 3$. The end-effector position in the operational space is defined as ${}_wp_{ee} = [{}_wx_{ee}, {}_wy_{ee}, {}_wz_{ee}]^T$. A schematical model is illustrated in figure 3.6.

Again, as a first step, the end-effector position relative to the arm base position ${}_ap_{ee}$ is described, which is defined as ${}_ap_{ee} = [{}_ax_{ee}, {}_ay_{ee}, {}_az_{ee}]^T$. The DH-notation is used to determine the forward transformation ${}^aT_{ee}$ of the robot. The frames, beginning with the base frame over each joint up to the end-effector frame, are shown in figure 3.6. Further information about the DH-parameter is listed in the appendix A.0.2. The state of the robot arm is represented by the joint configuration $q_a = [q_1, q_2, q_3]^T$ the joint configuration of the arm. Based on the forward transformation ${}^aT_{ee}$ the kinematic equations of the 3 link arm, representing the end-effector position regarding the arm base position, are

$$\begin{aligned} {}_ax_{ee} =& l_2cos(q_2)cos(q_1) + l_3cos(q_2+q_3)cos(q_1) \\ {}_ay_{ee} =& l_2cos(q_2)sin(q_1) + l_3cos(q_2+q_3)sin(q_1) \\ {}_az_{ee} =& l_1 + l_2sin(q_2) + l_3sin(q_2+q_3) \end{aligned} \tag{3.13}$$

The Jacobian of the 3-Dof manipulator is defined as described for the 2-Dof manipulator with the partial derivative of the end-effector position by the joint vector

$$J(q) = \begin{bmatrix} \frac{\partial_b x_{ee}}{\partial q_1} & \frac{\partial_b x_{ee}}{\partial q_2} & \frac{\partial_b x_{ee}}{\partial q_3} \\ \frac{\partial_b y_{ee}}{\partial q_1} & \frac{\partial_b y_{ee}}{\partial q_2} & \frac{\partial_b y_{ee}}{\partial q_3} \\ \frac{\partial_b z_{ee}}{\partial q_1} & \frac{\partial_b z_{ee}}{\partial q_2} & \frac{\partial_b z_{ee}}{\partial q_3} \end{bmatrix} = \begin{bmatrix} J_{xq_1} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix} \tag{3.14}$$

$$\begin{aligned} J_{xq_1} &= -l_2cos(q_2)sin(q_1) - l_3cos(q_2+q_3)sin(q_1) \\ J_{12} &= -l_2sin(q_2)cos(q_1) - l_3sin(q_2+q_3)cos(q_1) \\ J_{13} &= -l_3sin(q_2+q_3)cos(q_1) \\ J_{21} &= l_2cos(q_2))cos(q_1) + l_3cos(q_2+q_3)cos(q_1) \\ J_{22} &= -l_2sin(q_2)sin(q_1) - l_3sin(q_2+q_3)sin(q_1) \\ J_{23} &= -l_3sin(q_2+q_3)sin(q_1) \\ J_{31} &= 0 \\ J_{32} &= l_2cos(q_2) + l_3cos(q_2+q_3) \\ J_{33} &= l_3cos(q_2+q_3) \end{aligned}$$

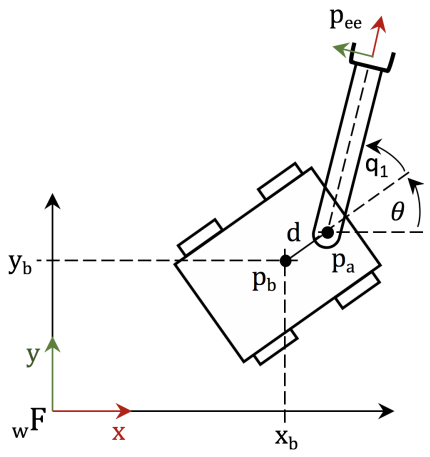

Figure 3.7: Kinematic model of a 6-Dof mobile manipulator in the xy-plane. The angle θ represents the orientation of the mobile base and the parameter x_b and y_b represent the position of the mobile base p_b with respect to the world frame. Again, the distance between the center of the mobile platform p_b and the base of the arm p_a is represented by the parameter d.

Taking the mobile base into consideration, the end-effector position ${}_wp_{ee}$ with respect to the world frame consists of the sum of the end-effector position ${}_ap_{ee}$, the offset d and the position of the mobile platform ${}_wp_b$. Figure 3.7 illustrates this relationship.

Based on the joint configuration $q = [q_a, q_b]^T$ where $q_b = [x_b, y_b, \theta_b]^T$ represents the base position and $q_a = [q_1, q_2, q_3]^T$ the joint configuration of the arm, the following equations define the end-effector position regarding the world frame ${}_wp_{ee}$ with

$$\begin{aligned} {}_wx_{ee} =& x_a + l_2 cos(q_2) cos(\theta_b + q_1) + l_3 cos(q_2 + q_3) cos(\theta_b + q_1) \\ {}_wy_{ee} =& y_a + l_2 cos(q_2) sin(\theta_b + q_1) + l_3 cos(q_2 + q_3) sin(\theta_b + q_1) \\ {}_wz_{ee} =& l_1 + l_2 sin(q_2) + l_3 sin(q_2 + q_3) \end{aligned} \tag{3.15}$$

where $x_a = x_b + dcos(\theta_b)$ and $y_a = y_b + dsin(\theta_b)$. As introduced for the 3-Dof model, to solve the inverse kinematics problem of the mobile manipulator and determine the joint configuration q based on a desired end-effector position ${}_wp_{ee}$, the redundancy between q_a and q_b has to be solved. Using a redundancy parameter vector $\rho = [\rho_1, \rho_2, \rho_3]$, where ρ defines the relation between the end-effector position ${}_ax_{ee}$ and base position ${}_wx_b$, to solve the inverse kinematics for each component independet. The defined redundancy parameters are visualized in figure 3.8 and

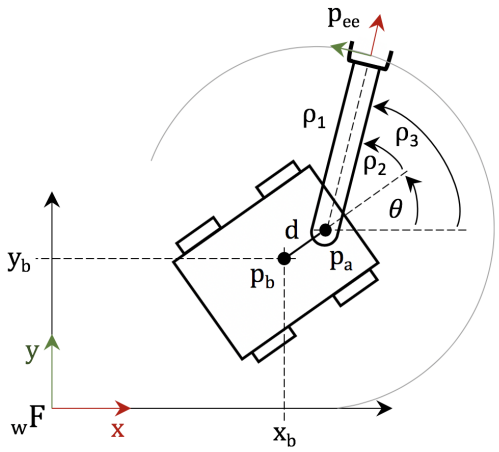

Figure 3.8: Illustration of the defined redundancy parameters. ρ_1 represents the distance between the end-effector p_{ee} and the base of the arm p_a. ρ_2 describes the angle between the heading of the base and the ee-orientation and ρ_3 the angle between the world frame and the ee-orientation with respect to the z-axis.

represented by the following equations

$$\begin{aligned} \rho_1 &= \sqrt{(x_{ee} - x_a)^2 + (y_{ee} - y_a)^2} \\ \rho_2 &= tan^{-1}\Big(\frac{y_{ee} - y_a}{x_{ee} - x_a}\Big) - \theta_b \\ \rho_3 &= tan^{-1}\Big(\frac{y_{ee} - y_a}{x_{ee} - x_a}\Big) \end{aligned} \tag{3.16}$$

Considering the physical limits of the manipulator, the values of the redundancy parameters can't be chosen arbitrarily. ρ_1 is describing the distance between the end-effector and the manipulator base. Accordingly, the maximum distance between these two points is the sum of link length l_2 and l_3, as well as constraint based on the desired end-effector height. Thus, the limit of ρ_1 is the same as defined for the 3-Dof MM parameter ρ in the 2D-plane. ρ_2 describes the end-effector orientation around the z-axis with respect to the base orientation or in other words the heading of the base.

$$\begin{aligned} 0 > \rho_1 < \|\sqrt{(l_2 + l_3)^2 - z^2}\| \\ -\pi > \rho_2 \leq \pi \end{aligned} \tag{3.17}$$

Since the end-effector orientation is not considered in this work, the base orientation θ is defined as $\theta = \rho_3 - \rho_2$. Thus, the end-effector position with respect to the base of the arm ${}_ap_{ee}$ can be calculated with

$$ {}_ap_{ee} = \begin{pmatrix} x_{ee} - \rho_1 cos(\rho_3) \\ y_{ee} - \rho_1 sin(\rho_3) \end{pmatrix} \tag{3.18} $$

where x_{ee} and y_{ee} are the coordinates of ${}_wp_{ee}$. ${}_wp_a$ allows then to calculate the position of the base with respect to the world frame as

$$ \begin{aligned} {}_wp_a &= \begin{pmatrix} x_a \\ y_a \end{pmatrix} = \begin{pmatrix} x_{ee} - \rho_1 cos(\rho_3) \\ y_{ee} - \rho_1 sin(\rho_3) \end{pmatrix} \\ {}_wp_b &= \begin{pmatrix} x_b \\ y_b \end{pmatrix} = \begin{pmatrix} x_a - d cos(\theta) \\ y_a - d sin(\theta) \end{pmatrix} \end{aligned} \tag{3.19} $$

In this particular case, where only the position in 3D-space is considered, the base orientation is only useful if the redundancy is exploited in a certain way. An example could be to change the internal state of the robot to avoid collisions with moving obstacles, as shall be seen later. For now, the base orientation is not considered and defined as $\theta = 0$. Accordingly, the degrees of freedom of the mobile manipulator are reduced to $N = 5$, by defining $\rho_3 = \rho_2$.

Definition 6. The base orientation is not considered and defined as $\theta = 0$

Chapter 4

Proposed Control Architecture

The goal of the proposed control architecture is to combine all objectives described in section 1.1. The control architecture can be integrated into a scheme, as shown in figure 4.1, where the input is defined as a trajectory of desired end-effector positions ${}_wp_{ee}$ and the output as the corresponding joint torques τ in order to obtain the desired robot motion. A wrist force sensor is measuring the contact force f between the end-effector and the environment. In addition, the controller receives the position feedback of the current robot posture $q = [q_a, q_b]^T$.

In order to be able to do trajectory tracking (O1) and simultaneously interact with the environment (O5) the controller requires a force and position control architecture (introduced in 2.3) as a basic concept. This control architecture has to

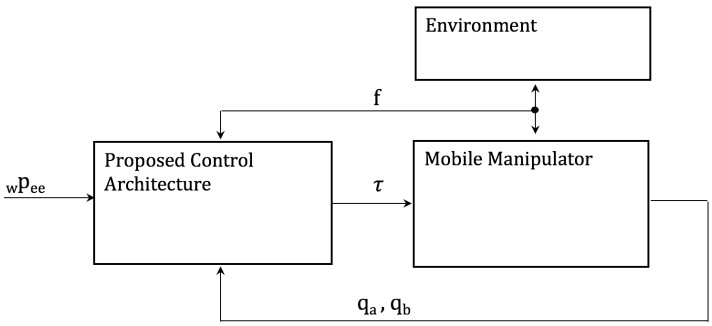

Figure 4.1: Integration of the proposed control architecture in the control scheme of the mobile manipulator. The input of the architecture is the desired end-effector positions p_{ee}, as well as the feedback of the current contact force f and the robot posture q. The outputs are the corresponding joint torques τ.

include a redundancy resolution scheme (O4), in order to exploit the redundancy between the robot arm and base and allow combined motion control (O3). Taking this into account, as well as the modularity (O5) of the desired control structure, the selection of the motion control approach can be argued as follows.

Motion control can be implemented as a *task space* or *joint space control method*, as introduced in chapter 2.2. However, in this particular case, the motion control approach has to be suitable for an additional force control method (O5). According to [21] and as introduced in 2.2.2, operational space control methods are convenient for the extension and integration of further control approaches. *Inverse dynamics control* (presented in 2.2.3) is a representative of operational control methods that comprises all requirements to integrate a force control method and shall be chosen as motion control method of the proposed control architecture.

Force control methods, as introduced in 2.3, can be divided into *direct* and *indirect force control*. Since it is aimed to be able to control contact forces to the desired value (O5), only direct force control methods shall be considered. The two main representatives are *hybrid position/force control* and *motion and force control*. The first method requires a detailed model of the environment, which can't be guaranteed in this work and accordingly, is not suitable. Motion and force control methods, instead, extend the inverse dynamics control law with an outer loop force feedback controller and are more robust to uncertainties in the environment. Thus, the method of motion and force control is chosen, comprising a PI-controller to control the contact force in constraint directions to the desired value. The principle of motion and force control also allows implementing more advanced force control methods with e.g. the idea of parameter adaptation. However, a PI-controller should be considered as the first step in order to test the overall concept of the proposed control architecture and can be replaced by another method afterward.

As introduced in 2.1.2, redundancy can be solved on different levels and is partially related on the used motion control method. The common approach of solving the redundancy on velocity level enables the MM to exploit the redundant Dofs, by meeting additional objectives to increase the manipulator performance (O1). The projected gradient or reduced gradient methods are suitable representatives of this redundancy resolution category. However, these two methods require a differentiable objective function and generally do not grant the convergence of the solution [1]. Furthermore, they only can be applied as part of the inverse kinematics problem to transform cartesian velocities into joint velocities. The inverse dynamics control method in operational space, instead, only considers the relation between cartesian

and joint accelerations. Thus, a redundancy resolution method on acceleration level would be required. These methods, however, only allow objectives regarding the dynamics of the robot. Instead, solving the redundancy on position level, by applying the method of redundancy parameters would consider the geometrical meaning of the redundant degrees of freedom between the arm and the base. Therefore, the mobile manipulator system is decomposed into the arm and the base, where the redundancy parameters describe the relation between both subsystems. Accordingly, the redundancy problem is reduced to a structured optimization to find the optimal values of the defined redundancy parameters. Solving the redundancy on position level comprises also independence of the motion and force control approach and offers a certain modularity to the control architecture (O5). As a result, redundancy parameters are used as part of the proposed control architecture.

The remainder of this chapter is structured as follows. First, the selected redundancy resolution method, customized for the control architecture, is described. Afterwards, the inverse dynamics control method, followed by the developed force control approach is presented. The chapter is concluded with a description of the overall proposed control architecture, comprising the three subsystems.

4.1 Redundancy Resolution and Optimization

The used redundancy resolution allows calculating, with respect to the desired end-effector trajectory, a separate trajectory for the arm ${}_{b}p_{ee}$ and the base ${}_{w}p_{b}$ of the mobile manipulator. This is done by using redundancy parameters and applying the kinematic equations in 3.1 and 3.2.

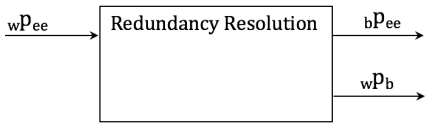

Figure 4.2: Redundancy Resolution subsystem, calculating the end-effector position with respect to the base p_{ee} and the base position p_b with respect to the world, regarding the desired end-effector position p_{ee} with respect to the world.

As introduced in subsection 2.1.2 based on the approach of [1], the redundancy parameters can be determined by solving an optimization problem, describing ob-

jectives the redundancy can be used for. The optimization problem is formulated with the general form

$$\begin{aligned} \underset{\rho}{\text{minimize}} \quad & g(p_{ee}, \rho, q) \\ \text{subject to} \quad & c(p_{ee}, \rho, q) \leq b_i \; i = 1 \ldots, m \end{aligned} \tag{4.1}$$

where $g(p_{ee}, \rho, q)$ represents the cost function and function $c(p_{ee}, \rho, q)$ limits ρ as defined in 3.1 and 3.2, describing the physical constraints of the robot that are equal to the workspace boundaries in order to avoid *boundary singularities* and a subset of *internal singularities* (subsection 2.1.1). The following objectives are considered in order to define the cost function $g(p_{ee}, \rho, q)$ and exploit the redundancy of the robot.

- **Reduction of base movement:** The position precision and repeatability of an omnidirectional base are less accurate than of a manipulator, as explained in 2.1.2. Therefore, the difference between the desired q_b and current base position $\bar{q}_b$ has to be minimized in order to reduce the movement of the base.

$$g_1(p_{ee}, \rho, q) = |\bar{q}_b - q_b| \tag{4.2}$$

- **Manipulability:** The manipulability index, as introduced in subsection 2.1.2 and specified for mobile manipulators by [1], is describing the dexterity of the current state of a robotic arm. This index is high if the current joint configuration is far from a singularity. Accordingly, maximizing this value allows using the redundancy of the mobile manipulator to avoid singularities of the arm. Since the optimization problem is defined to minimize the overall cost function, the manipulability index is defined as negative and represented by the following equation.

$$g_2(p_{ee}, \rho, q) = -\sqrt{det\left(\frac{\partial p_{ee}}{\partial I(p_{ee}, \rho)} \frac{\partial p_{ee}}{\partial I(p_{ee}, \rho)}^T\right)} \tag{4.3}$$

Combining $g_1(p_{ee}, \rho, q)$ and $g_2(p_{ee}, \rho, q)$ leads to a multi-objective function and the approach of linear scalarization can be used. The weights w_i of the objectives are the parameters of the scalarization. [26] For the case of the present optimization problem with two objective functions the parameter n is defined as $n = 2$.

$$g(p_{ee}, \rho, q) = \sum_{i=1}^{n} w_i g_i(p_{ee}, \rho, q) \tag{4.4}$$

However, having a closer look at these two objectives allows another simplified solution without solving a multi-objective cost function. This requires to rewrite the optimization problem and therefore, a deeper understanding of the two objectives mentioned above. Minimizing the base movement leads to the behavior that the arm is doing the entire movement within the defined limits of ρ. Only if the arm reaches these limits the base starts to move. Maximizing the manipulability of a non-redundant robotic arm leads to an opposite behaviour. For this investigation the two link manipulator in 2D-plane shall be used. According to [27] the analytical solution of the manipulability index w of a non-redundant manipulator is

$$w = |det(J(q))| = l_1 l_2 |sin(q_2)| \tag{4.5}$$

Thus, the value of joint q_1 has no impact to the measure of manipulability w. Instead, only q_2 affects the index and the best posture is given for $q_2 = \pm 90°$. The link length is defined as $l_1 = l_2$. Accordingly, the optimal distance d_{opt} between the end-effector and manipulator base is $d_{opt} = \sqrt{(l_1^2 + l_2^2)}$ in order to maximize w. Optimizing the manipulability index with respect to the redundancy parameter ρ results in keeping the joint q_2 at its best posture. The result is a line of the possible end-effector position in the manipulator workspace, as shown in figure 4.3. Since it is not convenient for the robot arm to move only on this line of ρ_{opt} while keeping the maximum index w, an upper and lower boundary can be defined, where within these boundaries the manipulability is considered as high and the constraint of the posture $q_2 = -90°$ is loosened.

As mentioned above, minimizing the base movement generates the behavior of maximizing the arm movement within the defined limits of ρ. These limits represent the workspace boundaries of the arm. However, instead of defining these limits as workspace boundaries, the upper and lower boundary of the manipulability index can be used. The result is the movement of the arm, within a part of the actual workspace where the manipulability index is considered as high and accordingly, the arm delivers good performance.

As postponed so far, the offset of ρ_{opt}, within the area where the manipulability is considered as high, has to be defined. This parameter Δ_{opt} can be chosen arbitrarily

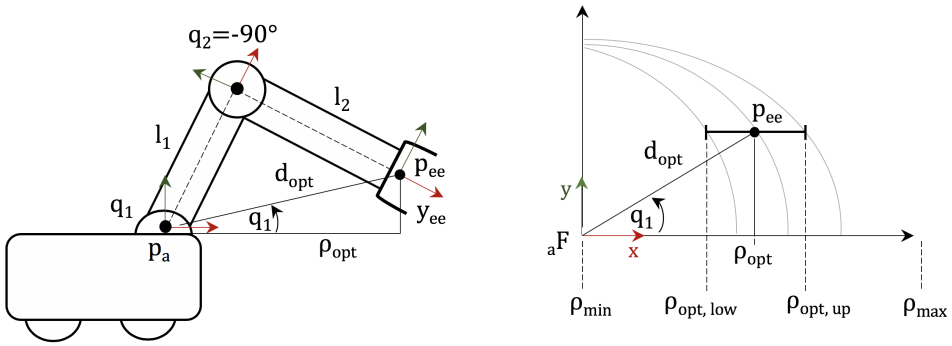

Figure 4.3: Maximum manipulability index of a 2-Dof robotic arm with the distance d_{opt} between p_{ee} and p_a. With respect to y_{ee} the redundancy parameter ρ_{opt} with a maximized manipulability is obtained. The right side of the figure illustrates ρ_{opt} and the defined boundaries, within the manipulability index is considered as high.

and considers the weighting between a high manipulability and minimized base movement. The manipulability boundaries are calculated with

$$\begin{aligned} \rho_{opt,up} &= \left|\sqrt{d_{opt} - y_{ee}^2}\right| + \Delta_{opt} \\ \rho_{opt,low} &= \left|\sqrt{d_{opt} - y_{ee}^2}\right| - \Delta_{opt} \end{aligned} \tag{4.6}$$

It has to be verified that the calculated values are within the limits ρ_{max} and ρ_{min}. Furthermore, the calculation only has to be done if y_{ee} is smaller or equal to d_{opt}. Otherwise $\rho_{opt,low} = \rho_{min}$ and $\rho_{opt,up} = \rho_{opt,low} + \Delta_{opt}$. The optimization problem is defined as

$$\begin{aligned} \min_{\rho} \quad & g_(q, p_{ee}, \rho) = |\bar{q}_b - (p_{ee} + \rho)| \\ \text{s.t.} \quad & \rho_{min} \leq \rho_{opt,low} \leq \rho \leq \rho_{opt,up} \leq \rho_{max} \end{aligned} \tag{4.7}$$

where q_b is representing the current base position and $(p_{ee} + \rho)$ the desired base position. Since the desired value of this objective function is zero, ρ can be calculated analytically without the need of using an optimization solver.

4.2 Inverse Dynamics Control

Inverse dynamics control allows solving the motion control problem of a robot in task space, as introduced in 2.2.3. The idea is to calculate based on the desired input position p_d, velocity $\dot{p}_d$ and acceleration $\ddot{p}_d$ the corresponding joint torques τ that create the desired motion, as illustrated in figure 4.4. The equations introduced in

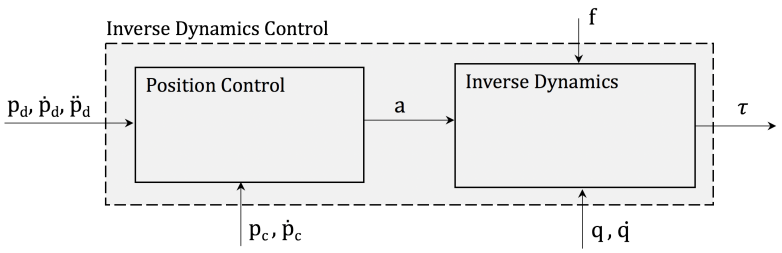

Figure 4.4: The Inverse dynamics control method can be divided into two steps. As a first step, the position control action, based on the desired position p_d and actual position p_c with its respective derivatives, is performed. As a second step, the actual inverse dynamics is implemented considering external forces f on the robot end-effector.

2.2.3 and used to implement the inverse dynamics control, as part of the proposed control architecture, shall be recapitalized in the following. The position control action is represented by a stabilizing control law for a as an outer loop control action, as illustrated in figure 4.4. This is implemented in form of a PD controller as

$$a = \ddot{p}_d + K_P(p_d - p_c) + K_D(\dot{p}_d - \dot{p}_c) \tag{4.8}$$

where K_P and K_D are diagonal gain matrices of the type $diag\{K_1, ...K_n\}$ and p_d the desired as well as p_c the current position, with its respective derivatives. The relation between a, as a value in cartesian space, and the joint accelerations α is defined as [cf. Eq.(2.11)]

$$\alpha = J^{-1}(q)(a - \dot{J}(q, \dot{q})\dot{q})$$

and used as input for the dynamic robot model determined in 3.1.2, as [cf. Eq.(2.18)]

$$\tau = M(q)\alpha + C(q, \dot{q}) + G(q) + J^T(q)f_{ext}$$

to calculate the torque τ for each joint, under consideration of the external forces f_{ext}.

4.3 Force Control

The force control approach presented in this section is based on the direct force control methods, introduced in 2.3. In particular, motion and force control, described in 2.3.3 is customized for the use in the proposed mobile manipulator control architecture. The overall structure of the force control approach, illustrated in 4.5, can be divided into three parts: a collision and contact observer, the actual force controller and parallel composition.

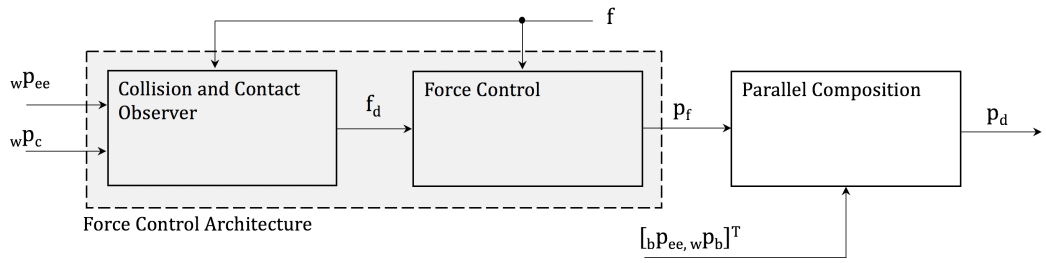

Figure 4.5: Force control architecture, consisting of a collision and contact observer and force controller to determine the compliant position p_f. The parallel composition is adding p_f to p_{ee} in order to obtain the adapted position p_d.

The combination of the collision and contact observer and the actual force controller is referred to as *force control architecture*. First, the force controller, represented by the middle block in figure 4.5, shall be described. The goal of the force controller is to control the contact force f to the desired value f_d. Accordingly, it is aimed to minimize the difference

$$\Delta f = f_d - f \tag{4.9}$$

by applying the force control action. Therefore, a PI-controller is used that can be described as

$$p_f = K_{P_p} \Delta f + K_{I_p} \int_0^t \Delta f dt \tag{4.10}$$

where p_f represents the compliant position, or in other words a position offset in order to minimize the force error Δf. K_{P_p} is the gain value of the proportional component and K_{I_p} is the gain value of the integral component. It has to be pointed out that a desired contact force f_d unequal to zero would cause a force error for end-effector movements in free space, where the measured force f is zero. Thus,

the PI-controller would output a position offset p_f, even though the robot is not interacting with the environment. The result is an adaptation of the end-effector trajectory without relevance and would decline the robot accuracy.

This requires the collision and contact observer to monitor the position offset between the desired ${}_wp_{ee}$ and current ${}_wp_c$ end-effector position, as well as the measured force f. Depending on the monitored values the observer can tell if the robot is moving in free space or in contact with the environment. Accordingly, the observer is switching the value of the desired force f_d between zero and the desired contact force value.

$$f_d = \begin{cases} f_{desired} & \text{for } ({}_wp_{ee} - {}_wp_c) \neq 0 \, and \, f > 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.11}$$

The criteria to select f_d considers an ideal measurement. Thus, in practice, it has to be adapted to be able to handle measurement uncertainties, as well as small forces due to friction in unconstraint directions.

This force control architecture, consisting of an observer and force controller, leads to a problem in case the robot end-effector is changing often between movements with contact and in free space. The integral term of the PI-controller keeps a remainder of its integrated force error due to the transition between contact and free movement. This requires the adding of an integral decay to the PI-control law to allow offset compensation, defined as

$$p_f = K_{P_p} \Delta f + K_{I_p} \int_0^t \Delta f dt - K_d \int_0^{t-1} \Delta f dt \tag{4.12}$$

where $\int_0^{t-1} \Delta f dt$ is the value of the integral of the previous iteration and K_d the decay gain value.

The last component of the parallel control, to combine the desired end-effector position p_{ee} with the compliant position p_f, is the parallel composition. The idea of parallel composition with respect to p_{ee}, is to apply the compliant position p_f to both components of the mobile manipulator. The simplified representation is

$$p_d = p_{ee} + p_f \tag{4.13}$$

where $p_{ee} = [{}_bp_{ee}, {}_wp_b]^T$ is the desired position vector of the arm and the base. It is now crucial, how the position offset p_f is added to p_{ee}. For the following assumption large offsets of p_f are considered. By adding p_f only to the arm position ${}_bp_{ee}$,

collisions could occur where the arm is reaching its workspace limit. Accordingly, the position offset should be applied to the arm and if it reaches its workspace limits the base position should be adapted as well. This can be formulated for the lower workspace limit of the arm with

$$p_d = \begin{pmatrix} {}_b p_{ee_d} \\ {}_w p_{b_d} \end{pmatrix} = \begin{cases} \begin{pmatrix} {}_b p_{ee} + p_f \\ {}_w p_b \end{pmatrix} & \text{for } \rho_{min} \leq {}_b p_{ee} + p_f \\ \begin{pmatrix} \rho_{min} \\ {}_w p_b + (p_f + {}_b p_{ee} - \rho_{min}) \end{pmatrix} & \text{otherwise} \end{cases} \tag{4.14}$$

where ρ_{min} is representing the lower workspace limit of the arm and $({}_b p_{ee}, {}_w p_b)^T$ the desired position vector of the base and the arm. This case represents scenarios where the arm is pushed backward. The parallel composition for the upper workspace limit is defined as

$$p_d = \begin{pmatrix} {}_b p_{ee_d} \\ {}_w p_{b_d} \end{pmatrix} = \begin{cases} \begin{pmatrix} {}_b p_{ee} + p_f \\ {}_w p_b \end{pmatrix} & \text{for } {}_b p_{ee} + p_f \leq \rho_{max} \\ \begin{pmatrix} \rho_{max} \\ {}_w p_b + (p_f + {}_b p_{ee} - \rho_{max}) \end{pmatrix} & \text{otherwise} \end{cases} \tag{4.15}$$

where ρ_{max} represents the upper workspace limit of the mobile manipulator arm. This case represents scenarios where the arm is pulled forwards.

4.4 Mobile Manipulator Control Architecture

Combining the solutions for each subproblem - motion control, force control, and redundancy resolution - introduced in the previous sections, leads to the mobile manipulator control architecture, as illustrated in figure 4.6. The first component in the proposed control architecture is the redundancy resolution, decomposing a trajectory of ${}_w p_{ee}$ into positions ${}_b p_{ee}$ of the arm and positions ${}_w p_b$ of the base. The decomposed trajectory is then input for the position and force controller.

The advantage of having the redundancy resolution independent of the position and force controller is the ability to bypass the redundancy resolution block and control the arm and base independently. This allows not only to track one single

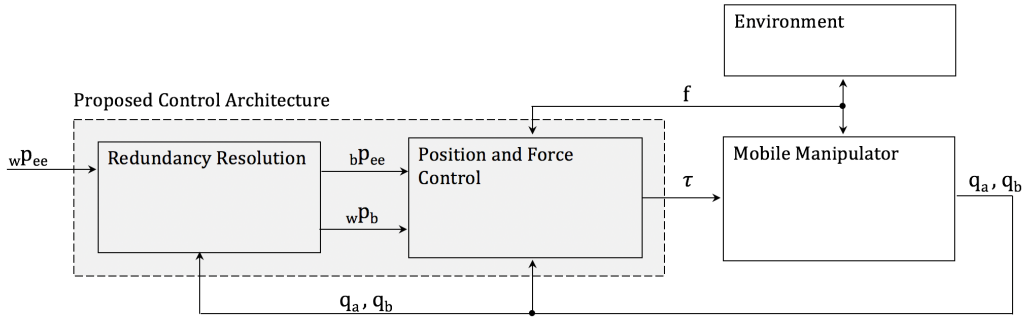

Figure 4.6: Proposed control architecture integrated into the control scheme of the mobile manipulator. Visualization of the redundancy resolution and position and force control as part of the proposed control architecture.

trajectory with the mobile manipulator but being able to easily switch and track a trajectory for the arm and the base independent, regarding the desired task.

The position and force controller applies the principle of parallel control. The inner loop inverse dynamics control is extended with an outer loop force control architecture, composed by the parallel composition, as illustrated in figure 4.6.

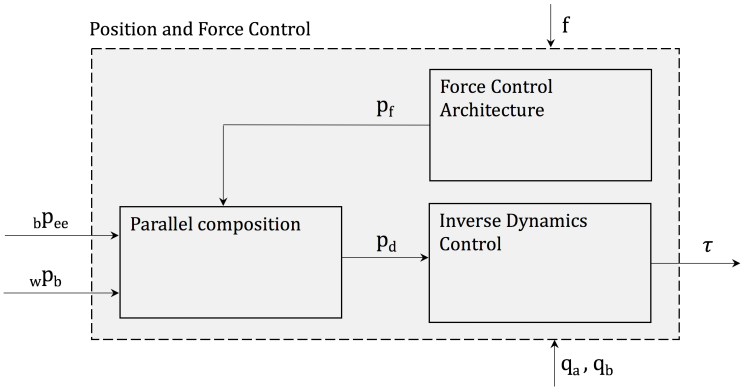

Figure 4.7: Schematic overview of the position and force controller with the parallel composition of the compliant position p_f and the position vector p_{ee} to p_d.

Chapter 5

Simulation

In order to test and evaluate the proposed control architecture, presented in the previous chapter, a Matlab/Simulink model is created. The development of the proposed control architecture and the Simulink model is done in parallel to incrementally test and optimize the architecture design. The foundation of the Simulink model is the kinematic and dynamic modelling of a simplified 3-Dof mobile manipulator, presented in chapter 3.1.

This chapter is structured as follows. First, the developed Matlab/Simulink Model, comprising the proposed control architecture, as well as a model of the mobile manipulator and environment, is described. Afterwards, the simulation results of different scenarios are presented, in order to evaluate the performance of the proposed control architecture.

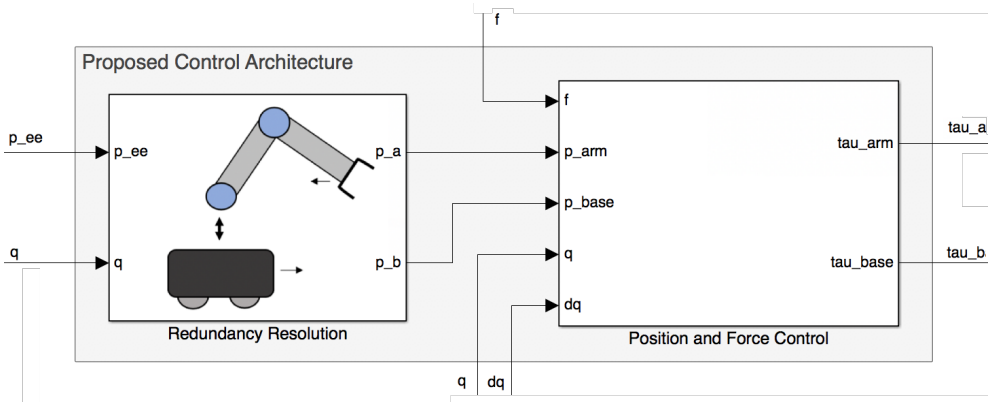

Figure 5.1: Section of the Simulink model, representing the proposed control architecture with redundancy resolution and position and force control. The control architecture calculates the joint torques τ based on the desired trajectory p_{ee}.

5.1 Simulink Model

The structure of the Simulink model is based on the control scheme, as described in chapter 4.4 and visualized in figure 4.6, where it can be divided into the proposed control architecture and the robot model with the environment. The correlated part of the control architecture in the Simulink model is illustrated in figure 5.1, consisting of the redundancy resolution and force and position controller.

The correlated part of the robot model with the environment in the Simulink model is illustrated in figure 5.2. The input of the robot model is the computed torque τ for each joint, as well as the contact force. The output is the current state of the robot q. The model of the environment represents vertical walls to determine the interaction force, based on q. The detailed model of the robot dynamics is shown in figure 5.3. The related equation is based on the modelling in chapter 3.1.2, where $\ddot{q}$ can be calculated as

$$\ddot{q} = M^{-1}(q)(\tau - J^T(q)f - C(q, \dot{q}) - G(q)) \tag{5.1}$$

By double integrating the computed joint accelerations $\ddot{q}$, the current joint velocities $\dot{q}$ and joint angles q of the robot are determined. The robot dynamics is considered to be ideal and no friction is represented.

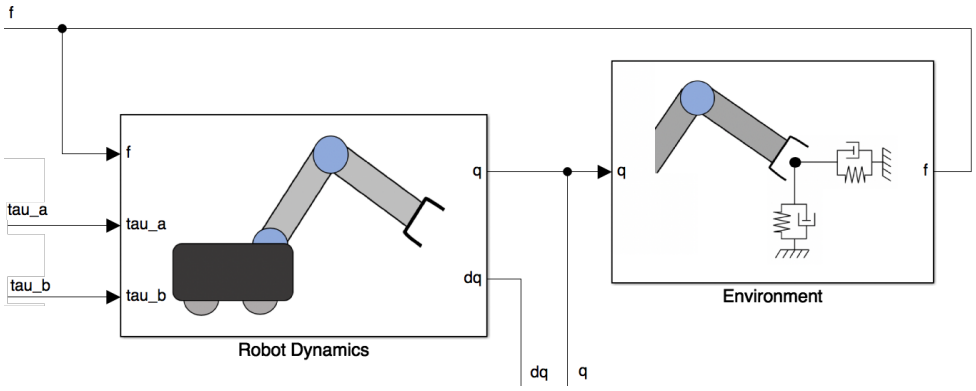

Figure 5.2: Section of the top level Simulink model, representing the plant (robot model) and the environment. The output of this section is the current robot state q, as well as the contact force f. The inputs are the computed joint torques τ for the base and the arm.

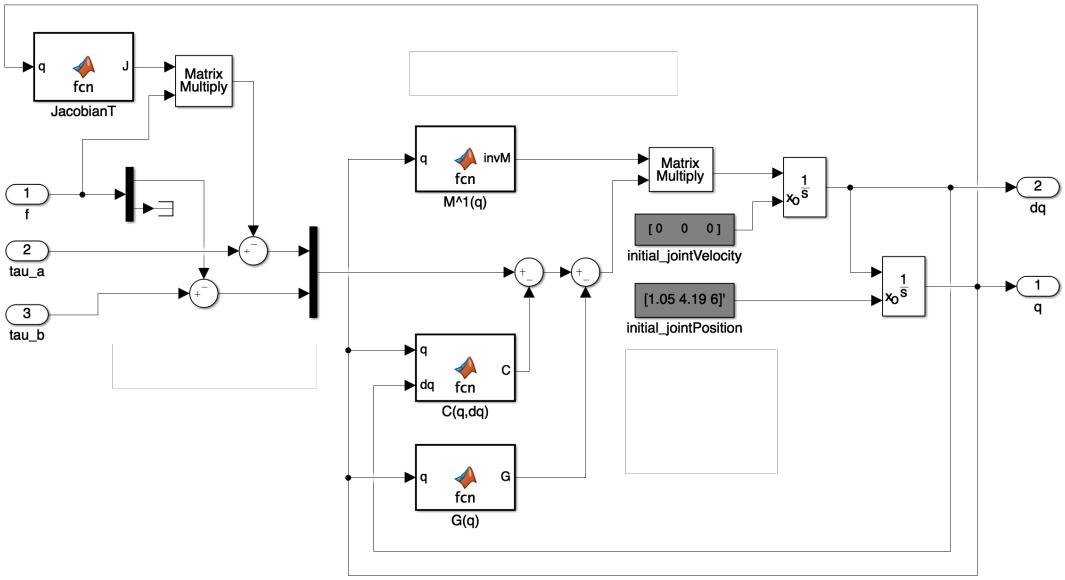

Figure 5.3: Simulink model of the **Robot Dynamics**. The robot dynamics represents the relation between torques, exerted on the robot joints, and the joint accelerations.

The environment is based on the soft contact method, described in 3.1.3. Equation (3.12) represents the behaviour of a compliant planar wall in x- and y-direction

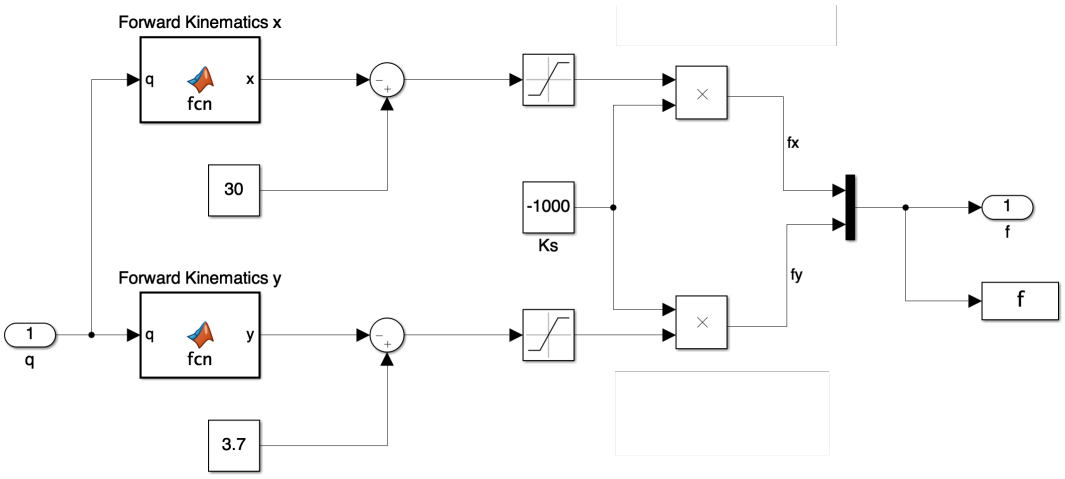

Figure 5.4: Simulink model of the **Environment**, based on the soft contact method. Vertical walls are represented by a spring-system. The starting point of the planar wall in x- and y-direction can be chosen by arbitrary values.

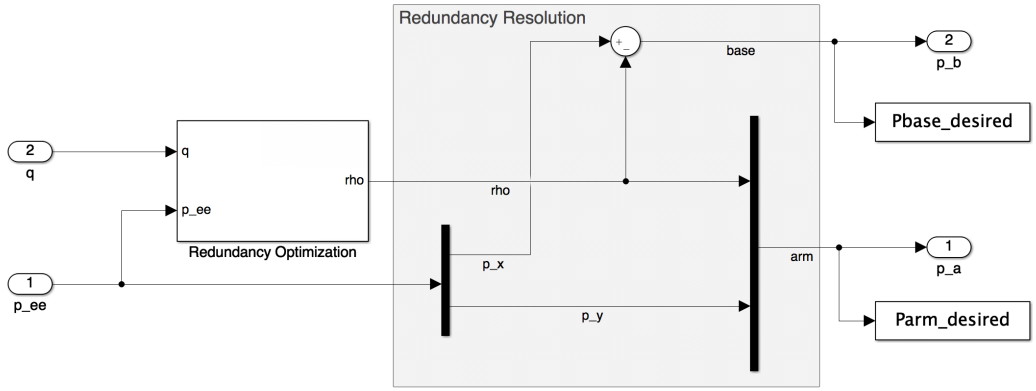

Figure 5.5: Simulink model of the **Redundancy Resolution** as part of the proposed control architecture. As a first step, the optimization problem is solved to determine ρ. As a second step, ρ is used to decompose the desired position p_{ee} into p_a and p_b.

The first part of the proposed control architecture Simulink model in figure 5.1 is the redundancy resolution. The redundancy resolution can be divided into two parts. In the beginning, the optimization problem is solved to determine the value of *rho*. Afterwards, the redundancy is solved by decomposing the desired position p_{ee} into the arm position p_a and the base position p_b by applying equation (3.5).

The basic code, as part of the "Redundancy Optimization" s-function is shown in listing 5.1. The implemented objective function is equation (4.2) to minimize the movement of the base. The boundaries *lb* and *ub* represent the defined workspace limits of the robot. The optimal boundary values to consider the manipulability of the robot, as described in 4.1, are represented by *lrho* and *urho*. In this particular case, the objective function (4.2) is formulated as a quadratic cost function to use the Matlab function *quadprog* in order to solve the optimization problem, since the minimum of a quadratic problem can be found faster than of a non-linear problem. It has to be mentioned that equation (4.2) can be solved analytically as well. However, the idea is to have the Simulink model independent of the used cost function and therefore, an optimization solver instead of the analytical solution is implemented.

```
1 function rho = solveOptimization(qb,xp,yp)
2   l=7;  % Maximum Streched length
3   %Lower boundery of rho and the manipulator workspace
4   lb = 1;
5   %Upper boundery of rho and the manipulator workspace
```

```
6  ub = abs(sqrt(l^2-yp^2));
7  % Optimal distance between ee and pa
8  dopt= sqrt(l1^2+l2^2);
9
10 % Check if yp is smaller than the optimal distance d_opt
11 if yp<dopt
12     rhoOpt = sqrt(dopt^2-yp^2); %Optimal rho in terms of
           manipulability
13 else
14     rhoOpt = lb;
15 end
16
17 offset=1.5; %Optimal Manipulability offset
18 lrho = rhoOpt-offset;
19 urho = rhoOpt+offset;
20
21 % Check if lrho and urho is within the boundaries of the
       manipulator workspace
22 if lrho>lb
23     lb = lrho;
24 end
25 if urho<ub
26     ub = urho;
27 end
28
29 % Quadtratic optimization solver
30 A = [];
31 b = [];
32 Aeq = [];
33 beq = [];
34 H=2;
35 f=2*qb-2*xp;
36 rho = quadprog(H,f,A,b,Aeq,beq,lb,ub);
```

Listing 5.1: Solving the optimization problem to determine rho

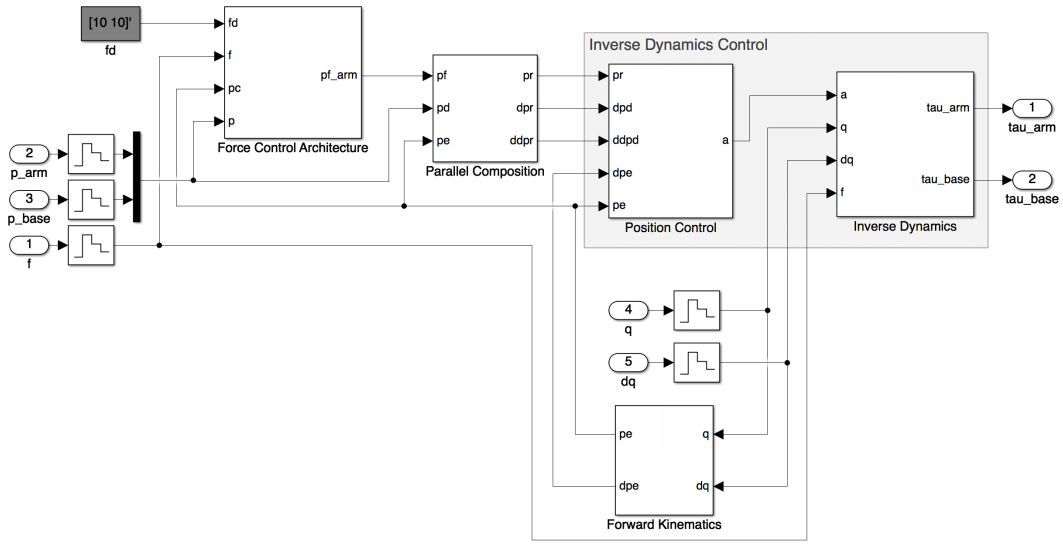

Figure 5.6: Simulink model of the subsystem **Force and Position Control**, consisting of the force control architecture, parallel composition and the inverse dynamics control method.

The second part of the proposed control architecture Simulink model in figure 5.1 is the position and force controller. An overview of this subsystem is shown in figure 5.6, which is based on the structure of figure 4.7. Thus, the subsystem consists of a **force control architecture**, **parallel composition** and **inverse dynamics control**. The **forward kinematics** is used to transform the actual joint angles into an actual end-effector position.

As a first step, the implemented inverse dynamics control block is explained more detailed, based on the description in chapter 4.2. The **position controller** is implemented as a Matlab-function containing the PD-controller represented by equation (4.8). The inverse dynamics is implemented, as shown in figure 5.7 and 5.8. The first part is transforming the computed accelerations in cartesian space a into accelerations in joint space $alpha$, by applying equation (2.11). The second part is the actual inverse dynamics, representing the robot model of equation (2.17), by calculating the vector of joint torques τ based on the desired joint accelerations $alpha$ and under consideration of the measured contact force f.

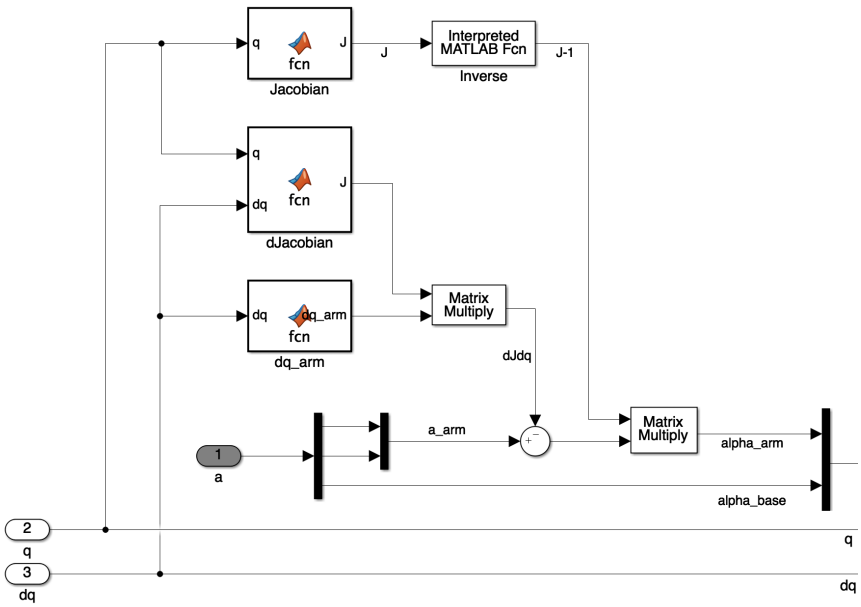

Figure 5.7: Simulink model of **Inverse Dynamics**, showing the implemented transformation between accelerations in task space into accelerations in joint space.

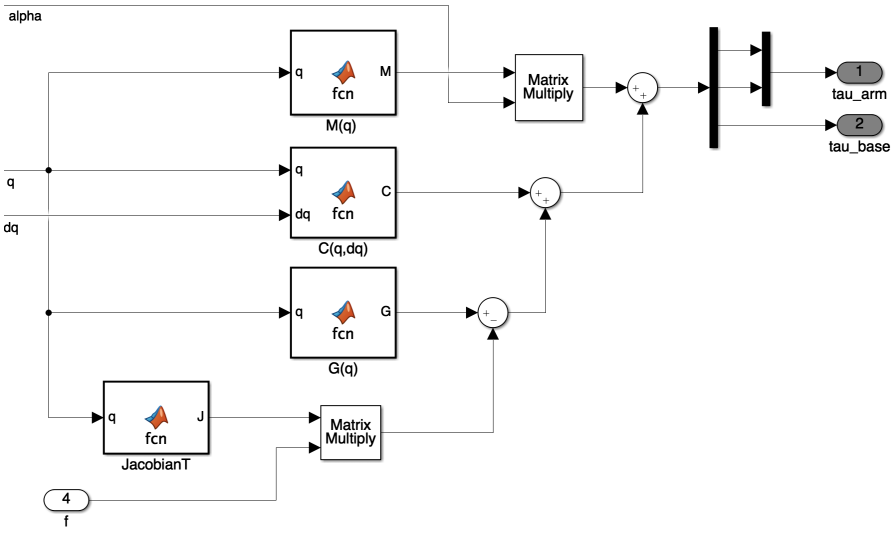

Figure 5.8: Simulink model of **Inverse Dynamics**, showing the implemented robot dynamics to compute the torques τ based on the desired joint accelerations.

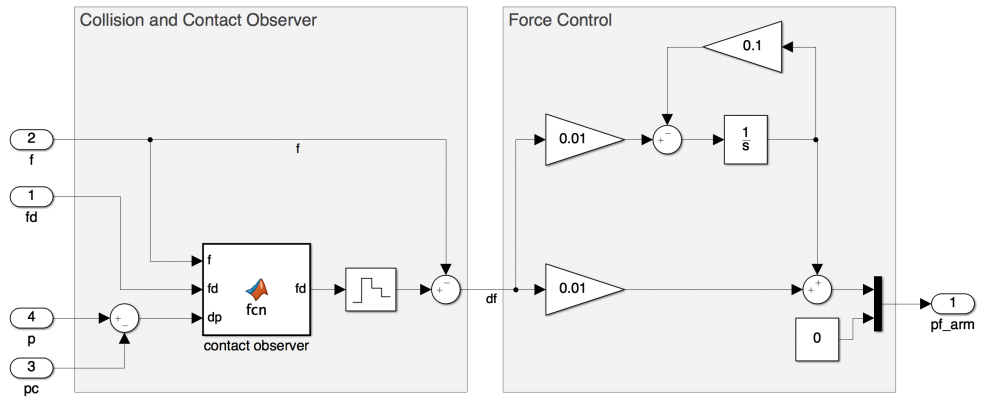

Figure 5.9: Simulink model of the **Force Control Architecture** comprising the collision and contact observer, as well as the force controller.

The second step comprises the implementation of the **force control architecture**. The Simulink model is the implementation of the approach, presented in chapter 4.3. Figure 5.9 shows the corresponding implementation. The **collision and contact observer** Matlab-function consists of equation (4.11). The **force controller** consists of the control law, described by equation (4.12).

The last step is the **parallel composition** to complete the scheme of parallel control. The Matlab-function of the parallel composition comprises equation (4.14), where only the lower workspace limit is implemented and tested. The tuning of all controller parameters is done experimentally. The simulation is executed with a fixed step size of 0.01s.

5.2 Simulation Results

The testing is divided into two major parts. First, the control architecture is evaluated for trajectory tracking, more specific movements in free space without contact with planar surfaces. Second, the trajectories are used in combination with obstacles, represented by walls, where force is exerted to the robot end-effector in constraint directions.

The **first simulation** consists of a trajectory in free space, where the robot is writing the letters 'DSL' in the air. Figure 5.10 visualizes the corresponding simulation results. The end-effector trajectory can be seen at the top. The trajectory error is caused by the tracking time delay between the desired and actual end-effector position and arises in directions the end-effector is moving. The lower two subplots

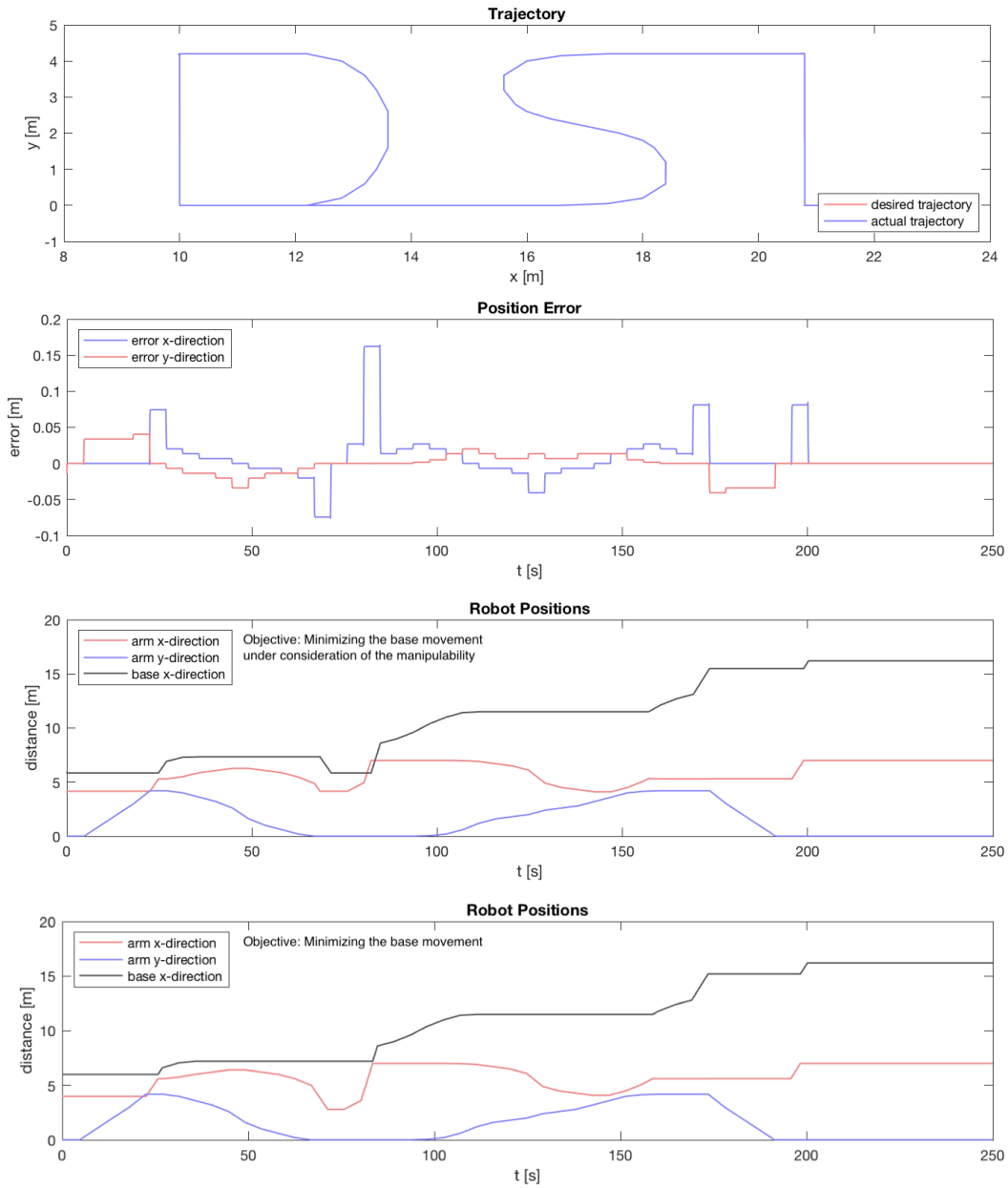

Figure 5.10: Trajectory tracking in free space, where the robot is writing the letters "DSL".

show the relation between arm and base movement, with the objective of minimizing the base movement under consideration of the manipulability and without, as described in chapter 4.1. The gain values of the position controller are $K_P = 1200$ and $K_D = 70$. The **second simulation** is a trajectory with the shape of an ellipse with

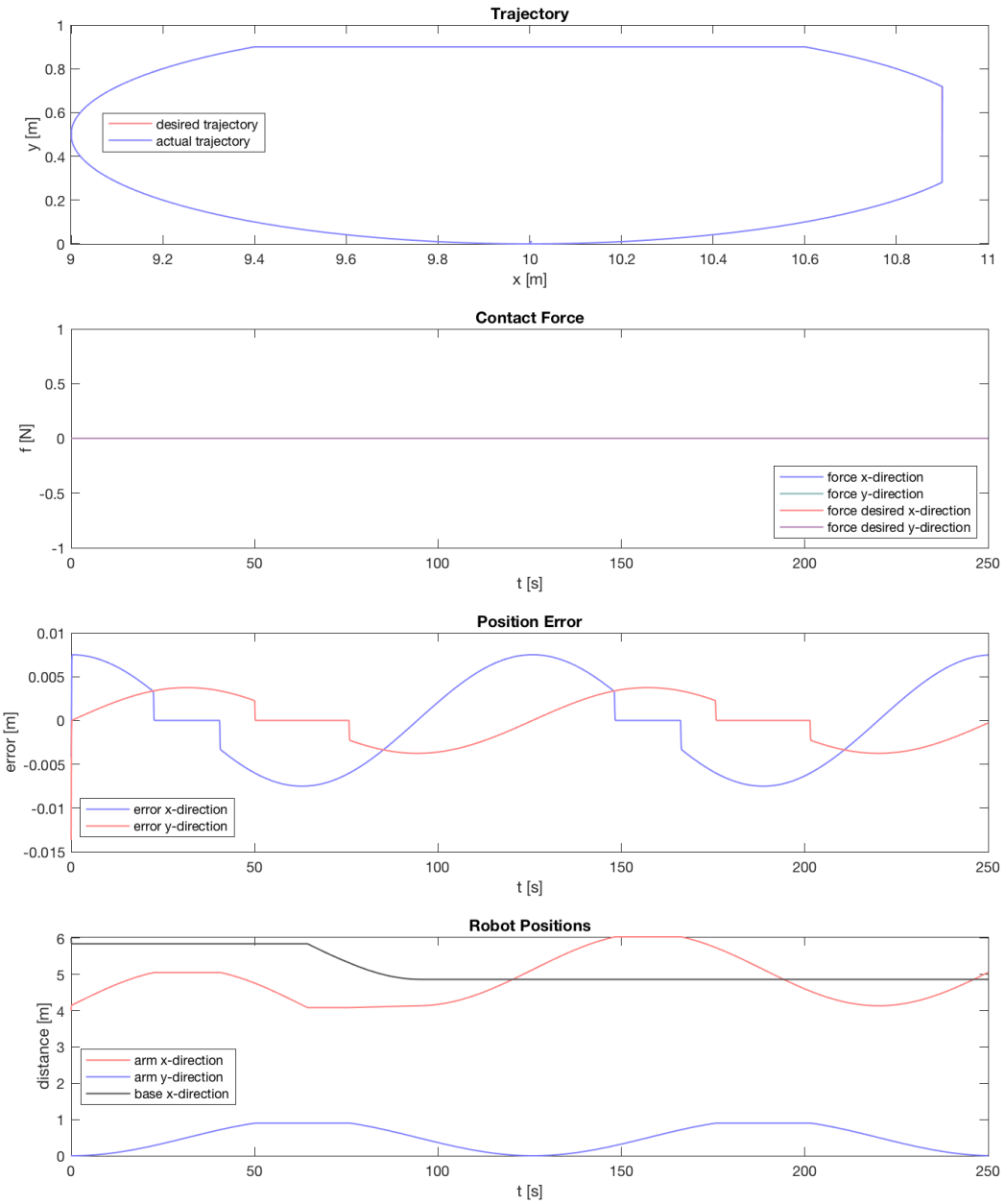

Figure 5.11: Tracking the trajectory of an ellipse with two flattened sides in free space.

two flattened sides. The related plot is shown in figure 5.11. Again, the trajectory error is caused due to the time delay between the desired and actual end-effector position. During the flattened sections of the trajectory this behaviour is emphasised since the end-effector is only moving in one direction inside these segments.

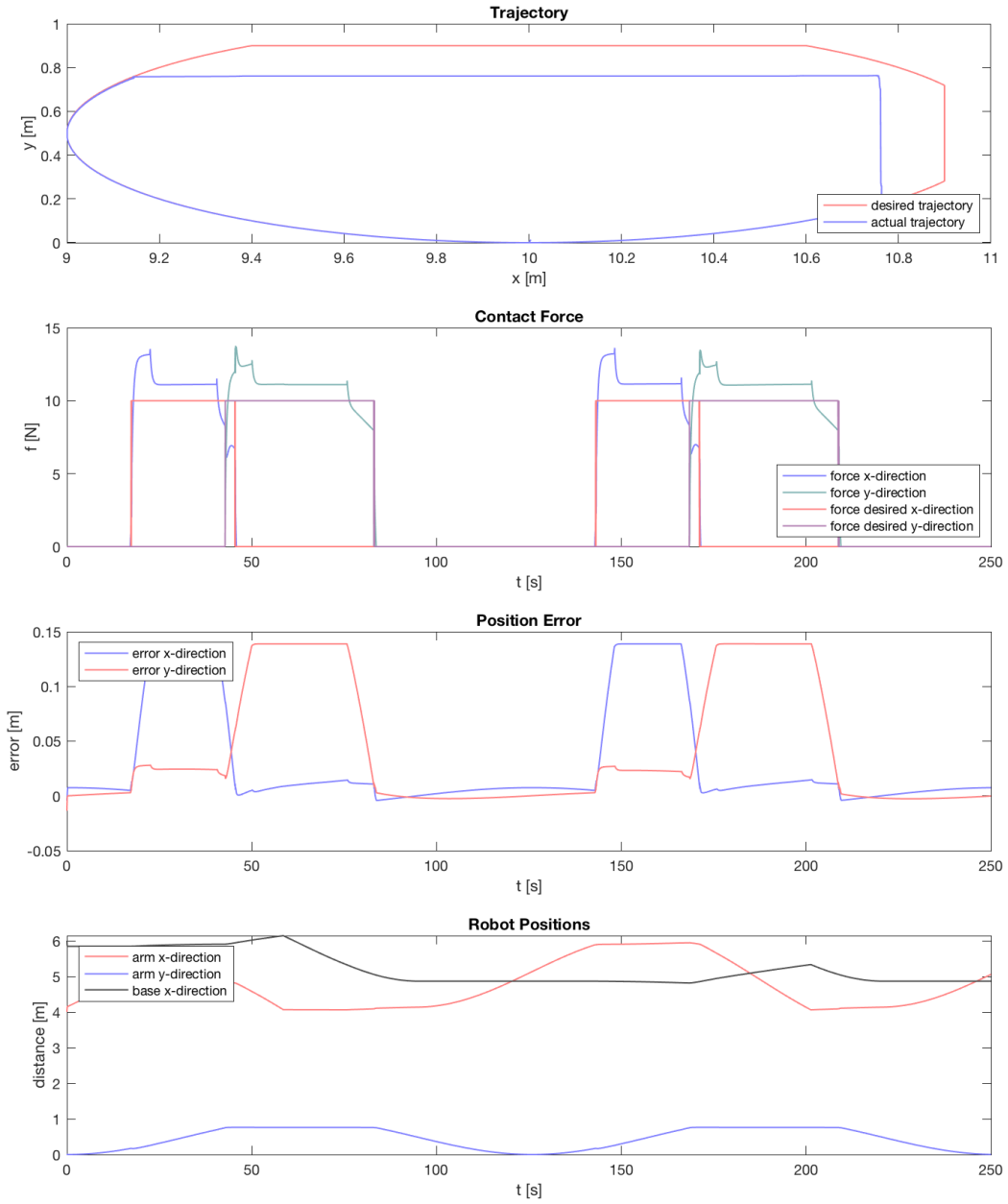

Figure 5.12: Tracking the trajectory of an ellipse with two flattened sides, with constraints in x- and y-directions, due to simulated compliant walls.

Accordingly, the tracking error occurs only in this direction. Adding two planar walls at $x = 10.75$ and $y = 0.75$ constrains the trajectory in both directions, as part of the **third simulation**. Thus, the desired trajectory has to be adapted by the force

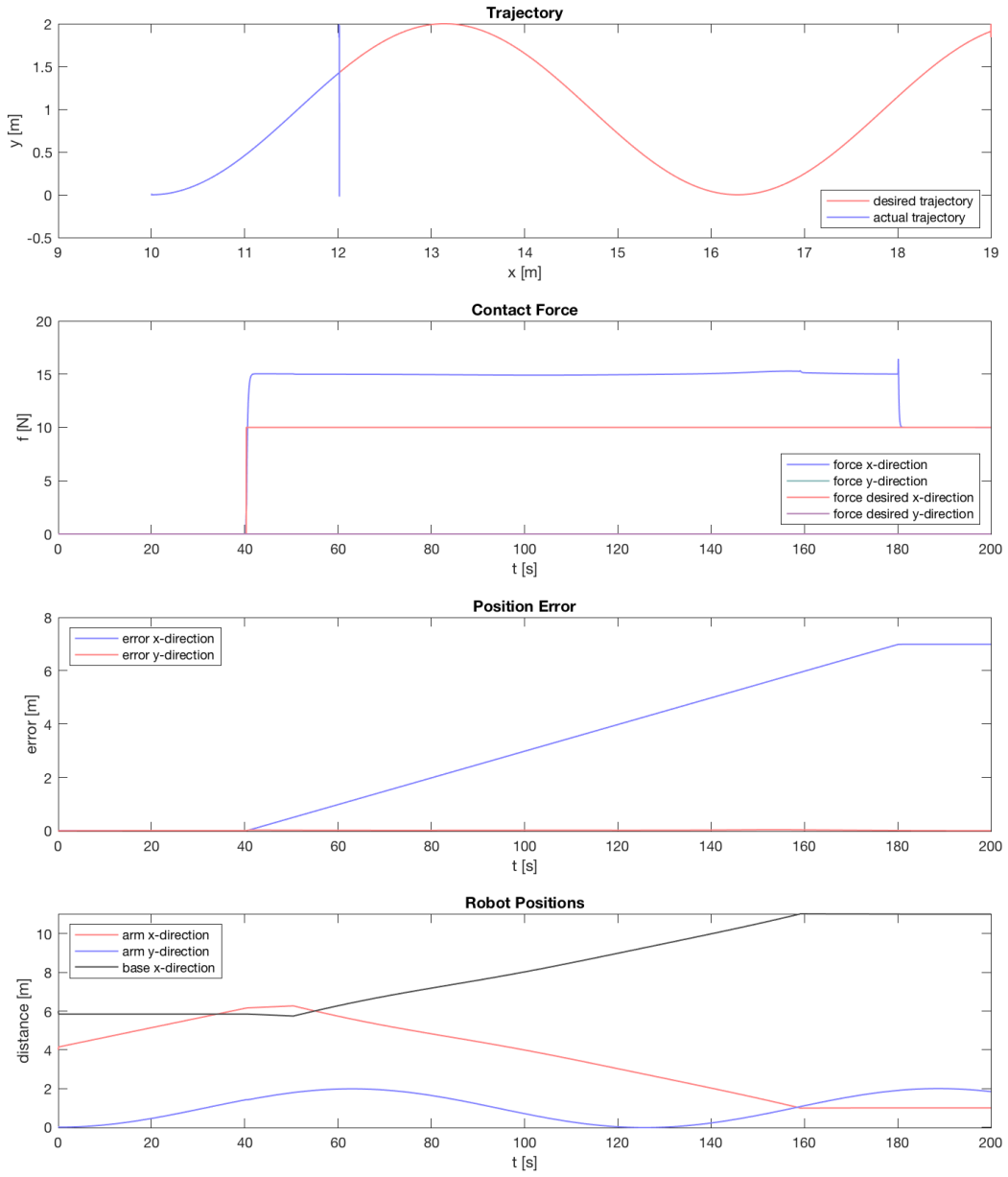

Figure 5.13: Tracking a trajectory blocked by an obstacle, in this particular case a compliant wall at $x = 12$.

controller output to obtain a constant contact force. The gain values of the PI-force-controller are $K_{Pp} = 0.001$, $K_{Ip} = 0.01$ and $K_d = 0.01$. Figure 5.12 illustrates the related Plot. The blue trajectory, as part of the top subplot, visualizes the adapted actual end-effector trajectory due to the walls. The red trajectory represents the sum

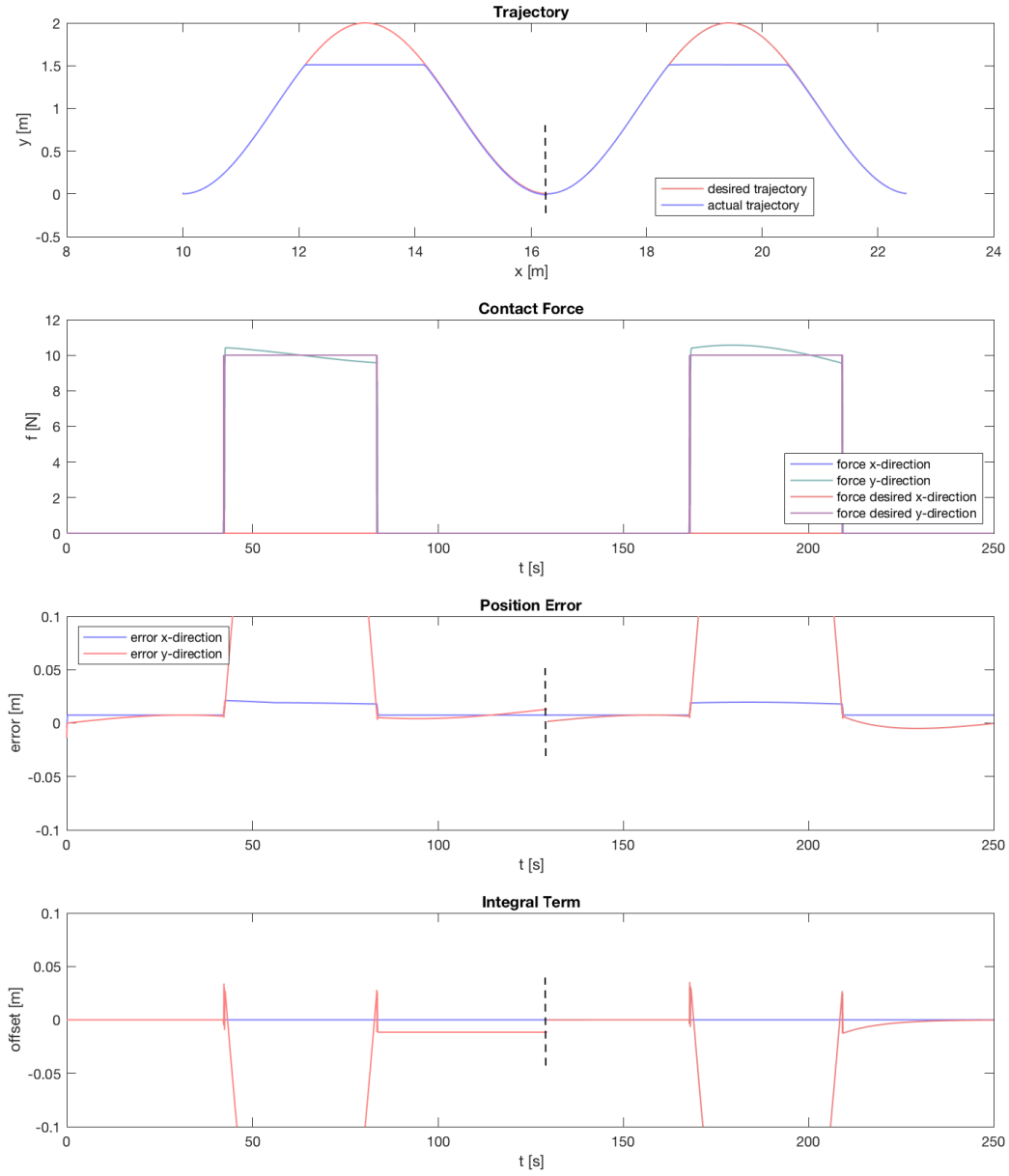

Figure 5.14: Tracking a sinusoidal trajectory with a horizontal wall at $y = 1.5$. The left part of the plot is with $K_d = 0$, the right part with $K_d = 0.01$.

of all desired end-effector positions. The second subplot visualizes the desired and actual contact force. In segments where the position offset is constant between the desired and actual end-effector position, caused by the obstacle (wall), the contact force is constant as well. The offset between the desired and actual contact force

is caused by the decay K_d. Thus, a changing position offset, leads to a changing constant force.

The **fourth simulation** represents a trajectory, blocked by a compliant vertical wall at $x = 12$. Figure 5.13 illustrates the simulation results. The gain values are chosen as $K_{Pp} = 0$, $K_{Ip} = 0.01$ and $K_d = 0$. This simulation proofs the insight of simulation three, where the contact force is not constant while the position offset between the desired and actual position is changing. Furthermore, the simulation shows the behaviour of the mobile manipulator moving towards a wall. The lower subplot in figure 5.13 visualizes the arm position with respect to the base and the base position with respect to the world. After reaching the wall, the arm (red line) moves backwards until its defined workspace limit, while the base is moving forward (black line). After the arm reaches its workspace limit, both components stop to move. This proofs the wanted behaviour, implemented in the parallel composition, described in 4.3.

The **fifth simulation** demonstrates the effect of the decay K_d as part of the PI-force-controller, shown in figure 5.13. The gain values of the left side are $K_{Pp} = 0$, $K_{Ip} = 0.1$ and $K_d = 0$. The gain values of the right side are $K_{Pp} = 0$, $K_{Ip} = 0.01$ and $K_d = 0.1$. The lower subplot visualizes that the integral term of the PI-controller remains with a value, causing a constant error. As part of the right side of the plot, considering the decay $K_d = 0.1$, this remaining value is slowly decreasing back to zero.

Chapter 6

Experiments on the Robot

The experiments, in order to test the proposed control architecture, have been done in collaboration with Adam Heins, from the dynamics systems lab, who implemented and integrated the code into the software architecture of the robot.

The mobile manipulator, used for the experiments, consists of a UR10 from Universal Robots [24] and a Ridgeback base from Clearpath robotics [6], as shown in figure 6.1. The UR10 is a 6-Dof industrial manipulator and the Ridgeback base an omnidirectional mecanum wheeled base with three degrees of freedom. Thus, the mobile manipulator has nine degrees of freedom in total. For the experiments, the wrist joints of the manipulator are set to a defined value and only the first three joints $q = (q_1, q_2, q_3)$ are considered. Accordingly, the MM can be considered as equal to the MM modeled in 3.2.

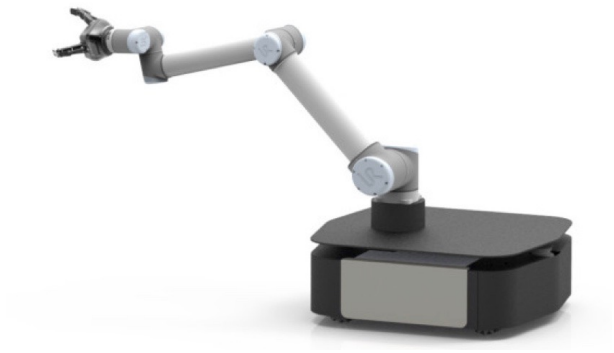

Figure 6.1: Mobile Manipulator consisting of a UR10 robot arm from Universal Robots and Ridgeback base from Clearpath Robotics [6]

Each of those two components has its own hardware interface in order to control the arm and the base, specified by the manufacturer. The possible input parameters of the robot arm are the joint angles $q_a = (q_1, q_2, q_3)$ and velocities $\dot{q}_a$. The input parameters of the mobile base are the position and orientation of the base $q_b = (x, y, \theta)$. The hardware platform does not allow to access and control the joint torques τ. Thus, the proposed control architecture has to be adapted, in order to be able to integrate it into the software architecture of the mobile manipulator. The adapted control architecture is illustrated in figure 6.2, where the inverse dynamics is replaced by an inverse kinematics to transform the position commands into joint commands and use the computed q as input for the arm and the base respectively.

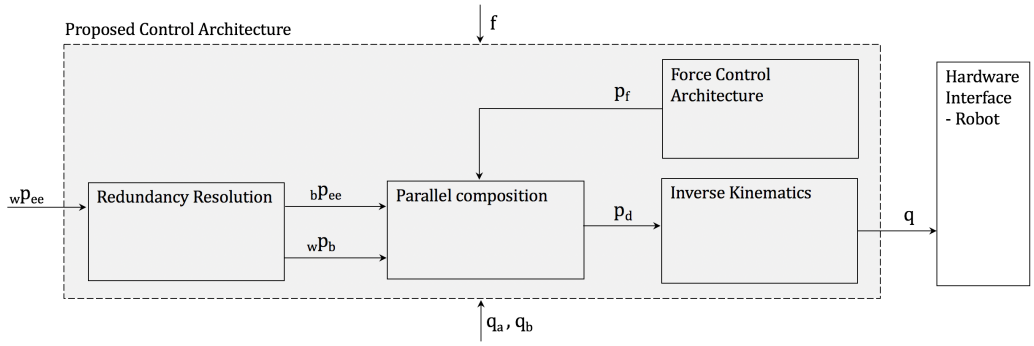

Figure 6.2: Adapted control architecture to test the performance on the real robot. The inverse dynamics, visualized in 4.7, is replaced by inverse kinematics to meet the requirements of the robot hardware interface and deliver the required control commands q.

The goal of the experiments is to confirm the simulation results of the previous chapter. Therefore, two major parts of the control architecture have to be tested. First, the redundancy resolution between the arm and the mobile base, with the objective to minimize the base movement. Second, the position and force control, to show how the robot is able to track trajectories in unconstraint directions and apply force control in constraint directions. Thus, two experiments are defined:

- **Redundancy resolution (E1)** Tracking a sinusoidal trajectory in free space greater than the workspace of the manipulator, what requires to move the arm and the mobile base.
- **Position and force control (E2)** Tracking a trajectory in unconstraint directions, while applying force control in constraint directions, by writing on a whiteboard.

6.1 Experimental Results

6.1.1 Redundancy Resolution

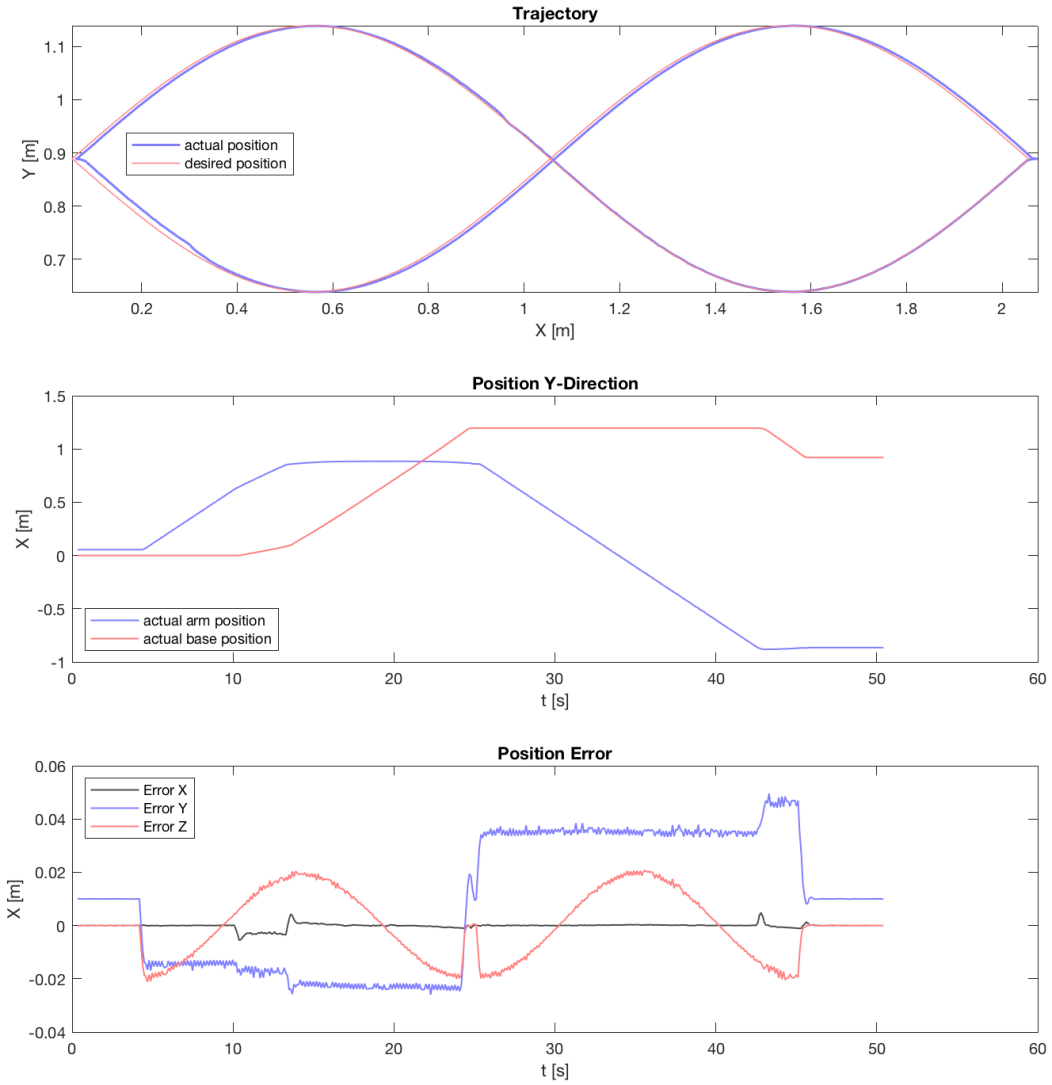

Figure 6.3: Experimental results of the redundancy resolution by tracking a sinusoidal trajectory (top plot), greater than the workspace of the robot arm.

Figure 6.3 visualizes the results of the experiment (E1), testing the redundancy resolution. It can be noticed (middle plot), that first the arm is moving and after reaching its limit the base continues. However, the position error (lower plot) indicates, that the transition between base and arm movement is causing a jerk with

impact to the end-effector position. During this transition, the end-effector has a jump and oscillation, especially in x- and y-direction.

6.1.2 Position and Force Control

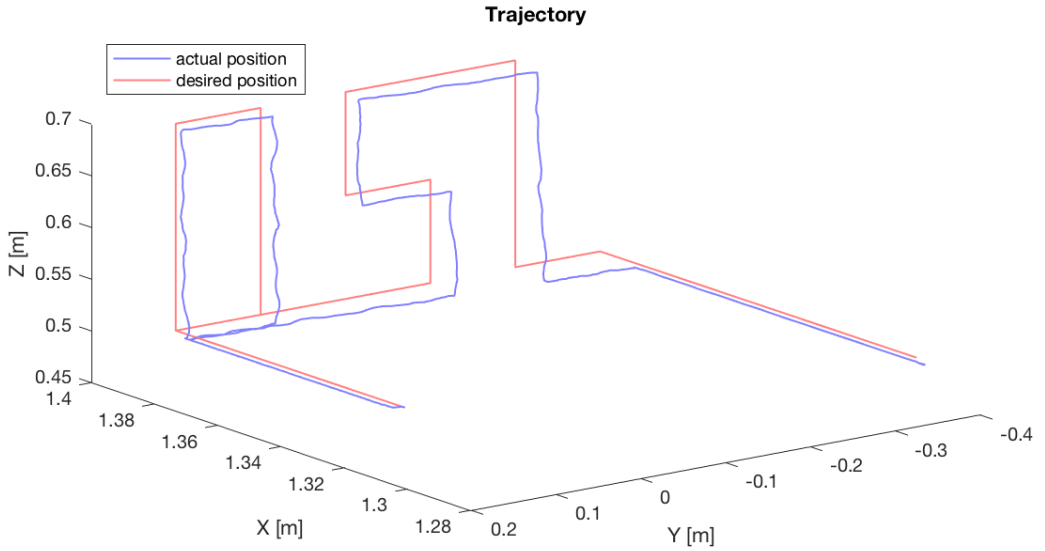

Figure 6.4: Actual and desired end-effector trajectory of the mobile manipulator. The wavy behavior of the actual position is caused by the compliant behavior of the whiteboard and the drifting force measurement and the associated force control output.

Figure 6.4 illustrates the trajectory, to test the position and force controller (E2) of the proposed control architecture. The red line represents the desired trajectory of the robot end-effector. Since the wall has an offset to the front with respect to the x-direction of the desired trajectory, the robot movement is constrained and requires to apply force control after reaching the wall. This is visualized by the blue line, representing the actual end-effector position. The force control parameters are set to $K_P = 0.05$ and $K_I = 0$ and $K_d = 0$, as well as the desired contact force $f_d = 0$.

In figure 6.5 the force and position in x-direction over time are illustrated, as well as the position error in all three directions. The error in y- and z- direction is caused by the time delay between the actual and desired position, in the direction the end-effector is moving. The error in x-direction is caused by the wall and the adapted trajectory due to the force controller. It can be noticed, that the actual measured force in x-direction correlates with the position error in x-direction, which is basically

a wanted behavior. However, the quality of the measured force is not consistent due to the fact that the measured force is drifting and even during movements in free space ranging between $\pm 1N$. This measurement error is causing an unwanted position offset as the output of the force controller. Thus, the accuracy of movements in free space is degraded by the noise of the force measurement.

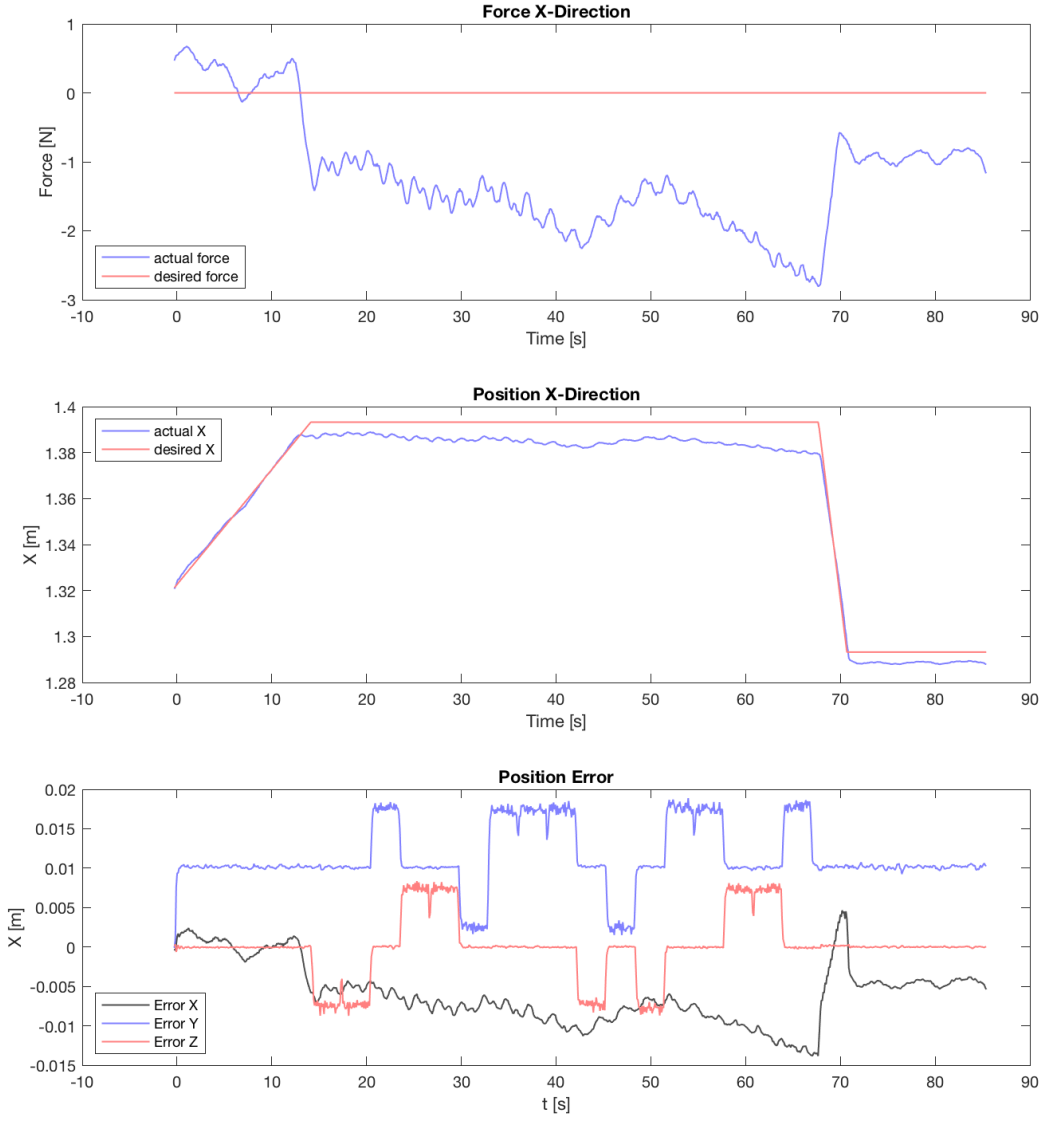

Figure 6.5: Experimental results of the proposed control architecture by writing on a whiteboard. The upper plot visualizes the measured force in x-direction and the middle plot the related end-effector position in x-direction over time. The lower plot shows the position error in each direction.

Even though the result, visualized in figure 6.4 and 6.5, shows potential for improvements, especially to deal with the bad force measurement quality, the task is executed adequately. The robot executing the experiment (E2) and the related result, represented by the quality of writing on the whiteboard, can be seen in figure 6.6.

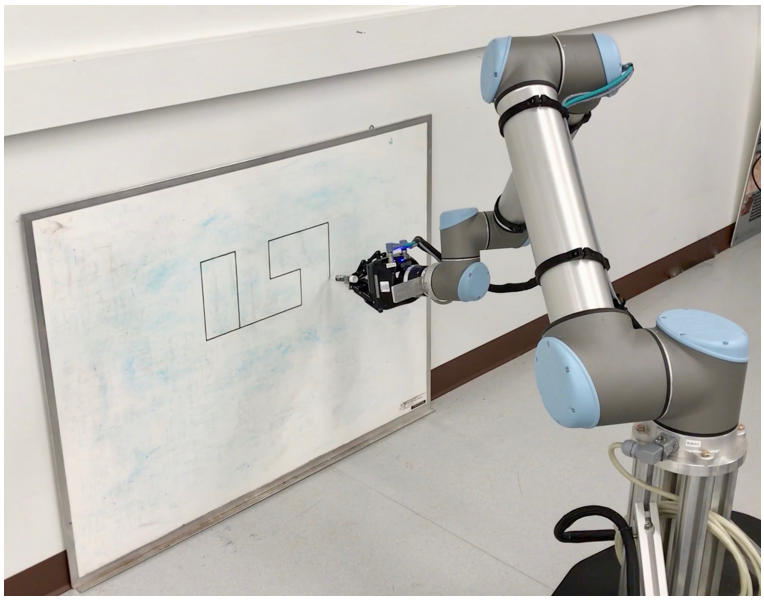

Figure 6.6: Mobile Manipulator writing on the wall during the experiment to test the proposed control architecture

Chapter 7

Conclusion

This thesis considered a control architecture to allow mobile manipulators to track trajectories in unconstrained directions and interact with the environment in constraint directions by facing the problem of position and force control with redundancy resolution. As a first step, a literature review was used to provide an overview of the mentioned problem. Therefore, redundancy resolution methods were classified into position, velocity and acceleration level, where the classification indicates the level of application of the respective method. Furthermore, motion control was grouped into joint and task space control methods, as well as interaction control into direct and indirect force control methods. A kinematic and dynamic model of a 3-Dof mobile manipulator in the 2D-plane, in combination with a model of the environment, has been created to provide the foundation of a later Simulink simulation. A kinematic 6-Dof mobile manipulator model was used for experiments.

Based on the literature review the different solutions for each subproblem have been compared and respectively, a method as part of the proposed control architecture selected. The architecture combines a redundancy resolution method on position level, by using redundancy parameters. These parameters describe the redundancy by the relation between the arm and the base position of the MM. The relation is formulated as a structured optimization problem. The output is a decomposed trajectory for the arm and the base respectively, which is the input of a motion control approach in task space. More specifically, inverse dynamics control was selected as part of the control architecture. A direct force control method is completing the control architecture, by extending the motion controller with an outer loop force controller to allow parallel control.

A Simulink simulation, based on a 3-Dof mobile manipulator model in the 2D-plane, proves the theoretical functionality of the proposed control architecture. Furthermore, experiments on a mobile manipulator research platform demonstrate the performance by applying the control architecture to real applications. As part of the first experiment, the redundancy resolution successfully coordinates the movement between the robot arm and base by tracking a trajectory greater than the manipulator workspace. The second experiment proofs the operability of the position and force controller by writing on a whiteboard.

7.1 Future directions

This subsection provides recommendations for improvements of the proposed control architecture and, in general, potential future research directions about mobile manipulator control.

Especially the experimental results have shown the impact of the force sensor reading quality to the force control results. Since the purchase of very precise force sensors can be expensive the solution might be the integration of force control methods, that do not require a force sensor reading, as presented by [11] for contact tooling applications. In general, the force control method should be able to react fast and stable to unknown collision events. The integration of more advanced collision handling, introduced by [8], would then allow human-robot interaction. This requires not only to monitor the end-effector contact force but the contact force applied to any part of the robot, as presented by [14].

The proposed control architecture is presented as one defined structure. However, the selected modular control components allow a certain flexibility to the architecture design. Currently, the force control output is applied to the decomposed trajectory after solving the redundancy, where the parallel composition handles the adaptation of the base and arm position. It could be investigated how the controller performs by applying the position offset before solving the redundancy and leaving the distribution to the optimization of the redundancy. In this context, the affection of the different optimization objectives can be investigated in order to combine the different objectives to a multi-objective optimization and choose an appropriate weighting.

In order to allow precise tracking, while performing fast dynamic motions, the robot model has to be accurate as well. For instance, the research platform used

doesn't consider the model of the entire mobile manipulator, but separate models for the arm and the base. This can be improved by creating a model of the entire mobile manipulator, as presented in [9] and done for the simulations in this work, or by using learning-based methods to learn the unmodelled effects. Furthermore, it is necessary to provide smooth transitions between the base and arm movements.

Mobile manipulators are extremely flexible systems, due to their moveability. Instead, static manipulators are bound to their location executing repetitive tasks. These tasks are programmed once and executed many times. However, mobile manipulators have to deal with many unknown events and update the planning while executing the desired task. This leads to the integration of a planner as a component that generates the input for the proposed control architecture. Therefore, the redundancy resolution provides a suitable interface for a trajectory of end-effector positions for the entire mobile manipulator. By bypassing the redundancy resolution, the arm and the base can be controlled separately.

Appendix A

Forward Kinematics

The forward kinematics of a manipulator can be obtained by using the DH-Notation. In the following the forward transformation of the 2-Dof and 3-Dof manipulator is determined.

A.0.1 3-DOF Model

A schematic model of the 2-Dof robot arm, as part of the 3-Dof mobile manipulator, is shown in the following figure. The used frames are indicated at each joint of the robot, as well as the world frame ${}_{W}F$ and the end-effector frame ${}_{TCP}F$.

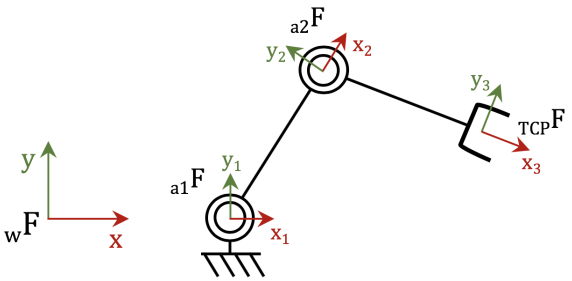

Figure A.1: Schematic model of a 2-Dof robot arm

The corresponding DH-parameter are listed in table A.1, starting at the base frame of the manipulator

i	θ_i	α_i	d_i	a_i
0	0	0	0	d
1	q_1	0	0	l_1
2	q_2	0	0	l_2

Table A.1: DH-parameter of the manipulator with reference to the base frame

Thus, the forward transformation ${}^{b}T_{ee}$ of the end-effector position with respect to the base ${}^{a1}F$ is described by

$$ {}^{b}T_{ee} = \begin{bmatrix} R_{11} & R_{12} & x \\ R_{21} & R_{22} & y \\ 0 & 0 & 1 \end{bmatrix} $$

where the end-effector position is described by the coordinates x and y

$$ \begin{aligned} {}_{b}x_{ee} &= l_1 cos(q_1) + l_2 cos(q_1 + q_2) \\ {}_{b}y_{ee} &= l_1 sin(q_1) + l_2 sin(q_1 + q_2) \end{aligned} $$

A.0.2 6-DOF Model

A schematic model of the 3-Dof robot arm, as part of the 6-Dof mobile manipulator, is shown in the following figure. The used frames are indicated at each joint of the robot, as well as the world frame ${}_{W}F$ and the end-effector frame ${}_{TCP}F$.

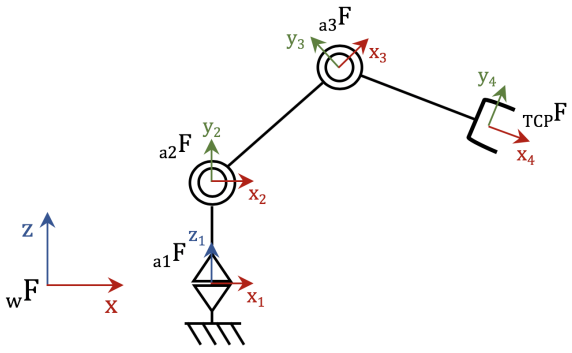

Figure A.2: Schematic model of a 3-Dof robot arm

The corresponding DH-parameter are listed in table A.2, starting at the base frame of the manipulator

i	θ_i	α_i	d_i	a_i
0	0	0	0	d
1	q_1	0	0	l_1
*	90	0	0	0
2	q_2	0	0	l_2
3	q_3	0	0	l_3

Table A.2: DH-parameter of

Accordingly, the forward transformation ${}^{a1}T_{ee}$ of the end-effector position with respect to the base ${}^{a1}F$ is described by

$$ {}^{a1}T_{ee} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & x \\ R_{21} & R_{22} & R_{23} & y \\ R_{31} & R_{32} & R_{33} & z \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

where the end-effector position is described by the coordinates x, y and z

$$ \begin{aligned} {}_a x_{ee} &= l_2 cos(q_2) cos(q_1) + l_3 cos(q_2 + q_3) cos(q_1) \\ {}_a y_{ee} &= l_2 cos(q_2) sin(q_1) + l_3 cos(q_2 + q_3) sin(q_1) \\ {}_a z_{ee} &= l_1 + l_2 sin(q_2) + l_3 sin(q_2 + q_3) \end{aligned} $$

Bibliography

[1] Roberto Ancona. Redundancy modelling and resolution for robotic mobile manipulators: a general approach. *Advanced Robotics*, 31(13):706–715, 2017.

[2] Victor Andaluz, Flavio Roberti, and Ricardo Carelli. Adaptive control with redundancy resolution of mobile manipulators. In *IECON Proceedings (Industrial Electronics Conference)*, 2010.

[3] Victor Andaluz, Flavio Roberti, and Ricardo Carelli. Robust control with redundancy resolution and dynamic compensation for mobile manipulators. In *Proceedings of the IEEE International Conference on Industrial Technology*, 2010.

[4] Rainer Bischoff, Ulrich Huggenberger, and Erwin Prassler. KUKA youBot - a mobile manipulator for research and education. In *2011 IEEE International Conference on Robotics and Automation*. IEEE, may 2011.

[5] Botond Bócsi, Lehel Csató, and Jan Peters. Indirect robot model learning for tracking control. *Advanced Robotics*, 2014.

[6] Clearpath Robotics. Ridgeback - Omnidirectional mobile manipulation robot. `https://www.clearpathrobotics.com/ridgeback-indoor-robot-platform/`.

[7] A. De Luca, G. Oriolo, and P.R. Giordano. Kinematic modeling and redundancy resolution for nonholonomic mobile manipulators. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1867–1873. IEEE, 2006.

[8] S. Haddadin, A. Albu-Schaffer, A. De Luca, and G. Hirzinger. Collision detection and reaction: A contribution to safe physical Human-Robot Interaction. In

2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 3356–3363. IEEE, sep 2008.

[9] Y.M. Hu and B.H. Guo. Modeling and motion planning of a three-link wheeled mobile manipulator. In *ICARCV 2004 8th Control, Automation, Robotics and Vision Conference, 2004.*, volume 2, pages 993–998. IEEE, 2004.

[10] KUKA. Mobile Robotics KMR QUANTEC. `https://www.kuka.com/en-ca/products/mobility/mobile-robots/kmr-quantec/`, 2017.

[11] Yanan Li, Gowrishankar Ganesh, Nathanael Jarrasse, Sami Haddadin, Alin Albu-Schaeffer, and Etienne Burdet. Force, Impedance, and Trajectory Learning for Contact Tooling and Haptic Identification. *IEEE Transactions on Robotics*, pages 1–13, 2018.

[12] Zhijun Li, Shuzhi Sam Ge, and Aiguo Ming. Adaptive Robust Motion/-Force Control of Holonomic-Constrained Nonholonomic Mobile Manipulators. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, 37(3):607–616, jun 2007.

[13] Alessandro De Luca. Robots with kinematic redundancy. `https://www.dis.uniroma1.it/{~}deluca/rob2{_}en/02{_}KinematicRedundancy.pdf`, 2017.

[14] Emanuele Magrini, Fabrizio Flacco, and Alessandro De Luca. Control of generalized contact motion and force in physical human-robot interaction. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2298–2304. IEEE, may 2015.

[15] Alicja Mazur, Mirela Frontkiewicz, and Wojciech Domski. Position-force control of nonholonomic mobile manipulator with simple holonomic constraint. In *2015 10th International Workshop on Robot Motion and Control, RoMoCo 2015*, 2015.

[16] M. H. Raibert and J. J. Craig. Hybrid Position/Force Control of Manipulators. *Journal of Dynamic Systems, Measurement, and Control*, 103(2):126, 1981.

[17] Robotic Systems Lab - ETH Zürich. Robot Dynamics (Lecture Notes). `https://www.ethz.ch/content/dam/ethz/special-interest/mavt/robotics-n-intelligent-systems/rsl-dam/documents/RobotDynamics2017/RD{_}HS2017script.pdf`, 2017.

[18] Christof Rohrig, Daniel Hes, and Frank Kunemund. Motion controller design for a mecanum wheeled mobile manipulator. In *2017 IEEE Conference on Control Technology and Applications (CCTA)*, pages 444–449. IEEE, aug 2017.

[19] Shashank Sharma, Gerhard K Kraetzschmar, Christian Scheurer, and Rainer Bischoff. Unified Closed Form Inverse Kinematics for the KUKA youBot. *Proceedings of Robotik*, pages 139–144, 2012.

[20] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics: Modeling, Planning, and Control*. Springer, 2010.

[21] Bruno. Siciliano and Luigi. Villani. *Robot Force Control*. Springer US, 1999.

[22] Roland Siegwart and Illah Nourbakhsh. *Introduction to Autonomous Mobile Robots*. The MIT Press, 2004.

[23] Mark W Spong, Seth Hutchinson, and M Vidyasagar. *Robot Dynamics and Control*. WILEY, 2004.

[24] Universal Robots. Technical specifications UR10. `https://www.universal-robots.com/media/50880/ur10{_}bz.pdf`.

[25] Dongmei Wei, Chao Ren, Mingyuan Zhang, Xiaohan Li, Shugen Ma, and Chaoxu Mu. Position/force control of a holonomic-constrained mobile manipulator based on active disturbance rejection control. In *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, pages 6751–6756. IEEE, oct 2017.

[26] Wikipedia. Multi-Objective Optimization. `https://en.wikipedia.org/wiki/Multi-objective{_}optimization`.

[27] T. Yoshikawa. Manipulability of Robotic Mechanisms. *The International Journal of Robotics Research*, 4:3–9, 1985.

List of Figures