MPC-Based Force Control for a Dynamically Stable Robot

Master Thesis

Author(s):
Fäh, Kevin

Publication date:
2020-09-09

Permanent link:
https://doi.org/10.3929/ethz-b-000521579

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

RSL Robotic Systems Lab
Prof. Dr. Marco Hutter

Master Thesis

MPC-Based Force Control for a Dynamically Stable Robot

Spring Term 2020

Supervised by:
Maria Vittoria Minniti
Ruben Grandia

Author:
Kevin Fäh

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

MPC-Based Force Control for a Dynamically Stable Robot

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Fäh	Kevin

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
Volketswil, 15.09.2020	

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.

The Student is acting under the supervision of Prof. Hutter and contributing to research and development in the Research Area at ETH Zürich. Research results of students outside the scope of an employment contract with ETH Zurich belong to the student him- or herself. Results of his collaborations ("Research Results") may be exploited by ETH Zurich, possibly together with other results that are created at ETH Zurich. To facilitate and to enable a common exploitation of all combined research results, the Collaborator hereby assigns his rights to his Research Results to ETH Zurich. In exchange, the Collaborator shall be treated like an employee of ETH Zurich with respect to any income generated due to his Research Results.

This Agreement regulates the participation of the Collaborator in research and development within the above mentioned Research Area and the rights to the Research Results being created.

1. **Collaboration within the Research Area**
   (1) The Student agrees to keep all research results confidential as well as other information from the research group of Prof. Hutter that is declared as confidential. This obligation to confidentiality shall persist until he or she is informed by ETH Zurich that the intellectual property rights to the research results have been protected through patent applications or other adequate measures or that no protection is sought. However, this obligation will not extend over a period exceeding 12 months after completion of his collaboration with Prof. Hutter.

   (2) The Collaborator will make available all information, data and research results out of the collaboration to ETH Zurich.

2. **Intellectual Property Rights**
   (1) The Student assigns his rights to the Research Results, including inventions and works protected by copyright, but not including his moral rights ("Urheberpersönlichkeitsrechte"), to ETH Zurich. Herewith, he/she cedes, in particular, all rights for commercial exploitations of Research Results to ETH Zurich. He/She is doing this voluntarily and with full awareness and he/she acknowledges that there are other options to participate in projects at ETH Zurich that do not require such assignment of rights.

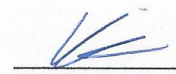
   (2) In exchange, he will be compensated by ETH Zurich in the case of income through the commercial exploitation of Research Results. Compensation will be made as if he was an employee of ETH Zurich and according to the guidelines "Richtlinien für die wirtschaftliche Verwertung von Forschungsergebnissen der ETH Zürich".

   (3) If a patent application is filed for an invention based on the Research Results, the Collaborator will duly provide all necessary signatures. He also agrees to be available whenever his aid is necessary in the course of the patent application process, e.g. to respond to questions of patent examiners or the like.

3. **Amendments and Additions**
   Amendments and/or additions of this Agreement can be made in writing, only.

4. **Settlement of Disagreements**
   Should disagreements arise out between the parties, the parties will make an effort to settle them between them in good faith.
   In case of failure of these agreements, Swiss Law shall be applied and the Courts of Zurich shall have exclusive jurisdiction.

The Collaborator

Zurich, 16.09.2020 Kevin Fäh 

Name Signature

Contents

Acknowledgment

I want to thank Prof. Dr. Marco Hutter, head of the Robotics Systems Lab at ETH Zurich, for the possibility to write this master thesis about the ballbot with the innovative 4-DOF DynaArm and a Robotiq gripper. The possibility to developed and test control and estimation algorithms on a real system of this complexity is a great experience. I highly enjoyed the collaboration with my supervisors Maria Vittoria Minniti and Ruben Grandia. Our weekly meetings and their good input helped me to quickly find the missing aspects in our chosen methods. At the same time, they guided me with the integration of our methods into the complex code basis. I also want to thank Maria Vittoria for taking time to conduct the hardware experiments with me.

Abstract

In this thesis, control methods are investigated that enable a stable and precise robot-environment interaction. Robot-environment interaction control is an important aspect of mobile robotics, since these systems should assist humans during tasks or handle their navigation through unknown environments autonomously. Modern robotics systems use optimal torque control to achieve energy efficient control solutions but also to define and command interaction forces between the robot and the environment. Model Predictive Control (MPC) is the state of the art method to solve such a control problem in an optimal way including constraints on the state and the input. However, a precise model of the robot is needed to have a good performance and as soon as unmodelled dynamics are acting on the robot, the MPC may fail or perform undesirably. We present different interaction control approaches to deal with the unknown environment dynamics while having an MPC-based force controller as the core component. These methods are simulated and tested on a dynamically stable ballbot with a 4-DOF robot arm and a Robotiq gripper attached. We test the interaction controllers on the application cases of the door opening and the lifting of objects of unknown mass which are both navigated precisely by following a target trajectory.

Symbols

Symbols

q	joint position
τ	torque
x	position
$\dot{x}$	velocity
$\ddot{x}$	acceleration
λ_{ee}	end effector force
J_{ee}	end effector jacobian in world frame
m	mass
d	damping
k	spring stiffness
f_s	static force
$\boldsymbol{v}_n$	unit vector along the task trajectory

Acronyms and Abbreviations

COM	Center of Mass
DOB	Disturbance Observer
EKF	Extended Kalman Filter
ETH	Eidgenössische Technische Hochschule
LTI	Linear Time-Invariant
MIAC	Model Identification Adaptive Control
MPC	Model Predictive Control
MRAC	Model Reference Adaptive Control
OCS2	Optimal Control for Switched Systems
PVT	Position Velocity Time
RLS	Recursive Least Squares
RMSE	Root Mean Square Error
TCP	Tool Center Point

Chapter 1

Introduction

1.1 Thesis Introduction and Overview

This thesis deals with Model Predictive Control (MPC)-based force control methods to achieve stable and precise robot-environment interaction tasks. Research in this field is of current interest as modern mobile robots should take over tasks while being in interaction with unknown environments such as the turning of valves and pushing of buttons in industrial applications or the assisted lifting of unknown objects and the autonomous opening of doors in service and inspection tasks. The necessity of direct or indirect force control schemes as described by Siciliano et al. [1], Natale [2] and Vukobratovic et al. [3] for such tasks is well known to limit and define the interaction forces to avoid damage on the environment or the robot. In direct force control, a reference force trajectory is tracked using force feedback from an estimator or sensor, while in indirect force control an impedance model is used that defines the relation between the commanded force and the position, velocity and acceleration tracking errors. These methods can be combined and extended with motion controllers, leading to hybrid controller as presented by Bodie et al. [4].
A state of the art control method for the locomotion and navigation of mobile robots such as the legged robot ANYmal and the ballbot is MPC since it allows to deal with constraints such as the robot dynamics or joint limits and actuator torque constraints by minimizing the cost function of an optimal control problem in a receding horizon fashion. Also the inverse kinematics problem is solved implicitly by adding task space position and orientation cost terms to the control formulation. An important aspect of an MPC solver used for mobile robotics is its capability to allow the online switching between subsystems, such as the change between cost functions, equality and inequality constraints without losing feasibility. This allows the robot to adapt the optimal control problem formulation dependent on the current task (e.g. force vs position control). Farshidian et al. [5], [6] and [7] developed the optimal control for switched systems (OCS2) toolbox which allows to solve an optimal control problem in a receding horizon fashion while switching the subsystem based on a sequential linear quadratic algorithm. We will use this toolbox in our work and discuss it in more detail in section 2.2.
A key point that guarantees stability and high performance of the MPC is to have a very good model of the real system. In the constraint optimal control formulation this affects the robot dynamics equation which is an equality constraint. A good approximation of the robot model is generally easy to acquire through CAD data. However, during contact with an unknown environment there will be a large model mismatch since additional forces will act on the robot.

In this work we are looking for general control strategies to enhance the MPC framework to deal with these unmodelled interaction dynamics. To compare and test the general applicability of the interaction controllers that are presented in this thesis, we apply them on the door opening and object lifting tasks where our goal is to open a door and lift objects of unknown mass. The interaction controller should make it possible to follow a desired trajectories stably and precisely while being in contact with an unknown environment.
We work with the ballbot visible in figure 1.1 which is a robot balancing on a ball with three actuators. The robot was enhanced with an additional 4-DOF robot arm, the DynaArm, on top. We use a Robotiq 2F-85 gripper at the end of the DynaArm for the interaction tasks. In [8] Minniti et al. have already implemented the OCS2 framework for the ballbot with a different arm and demonstrated that this dynamically stable robot can only remain stable with a whole-body controller that optimizes the ball and arm actuator torques at the same time, considering the full robot model.

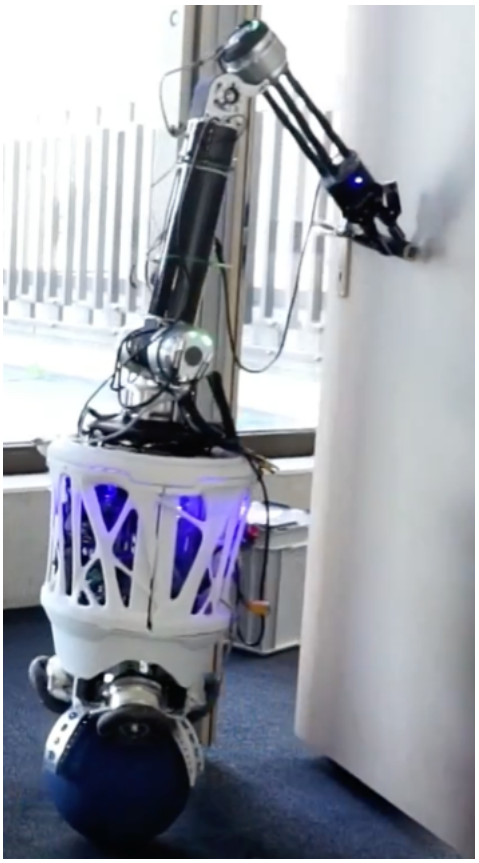


Figure 1.1: The ballbot with the 4-DOF DynaArm and a Robotiq-2F-85 gripper

Our work consists of the following parts: In section 1.2 we review related literature in robot-environment interaction control. In chapter 2 we explain the motivation of using a ballbot and give more insights to its hardware composition in section 2.1. In section 2.2 we explain how we use the OCS2 toolbox for our control problem. In chapter 3 we first present the assumptions in section 3.2 based on which we derived the impedance control approach in 3.4, model identification adaptive control method in 3.5, the model reference adaptive control method in 3.6 and combinations of the

adaptive controllers in 3.7. These controllers are tested in simulation and on the real system on the application cases of the object lifting task in chapter 4 and the opening of a door in chapter 5. In chapter 6 we summarize our results and give an outlook for further research and improvements.

1.2 Research in Robot-Environment Interaction Control

This section refers to current research results in robot-environment interaction control methods applied on mobile robots such as humanoids, quadrupeds and drones. Some of the following papers demonstrate their controller also on the door opening application task. We will summarize their methods and point out their advantages and disadvantages at the end of the section and also evaluate their usability for a whole-body MPC controller and our system.

The opening of a hinged door using MPC was addressed by Lee et al. [9] using a unmanned aerial vehicle with an attached 4-DOF robotics arm. In their approach they enhanced the state space of the system to include the model of the door into the system dynamics of the MPC controller. Therefore, they estimated the angle of the door based on the current end effector position which is valid if the end effector is in contact with the door at the expected point. The optimized position, velocity and force policy of the solver are then used as a reference trajectory for a disturbance observer based controller. The advantage of including the door model into the MPC solver is that collision avoidance constraints can be included and that the solution is optimal for the coupled dynamics between the door and the robot. The main disadvantage of this approach is that including the door dynamics demands a good estimate of the door mass and inertia and possibly static friction, damping and spring stiffness behavior which is further described in section 5.3 of the door opening task. In the work of Lee et al [9], the door was premodeled using a weight measurement and the known door dimensions. Therefore, effects like static friction and damping were not considered in the door model and a strong premodelling was demanded.

The door opening application case using indirect force control on a mobile platform was also described by Lee et al. [10] who opened a door with the compliant humanoid robot (CoMan) in a semi autonomous way. They used a decentralized impedance control approach on joint level with a stiffness matrix that can be computed from a task space stiffness matrix designed in the door frame. This allows to formulate a higher stiffness in the direction of the door path and more compliance in radial direction of the door to avoid high contact forces. The mapping of the task space stiffness to the joint space stiffness is described by the Jacobian from the door hinge frame to the door handle frame. Since the mapping leads to a non-diagonal stiffness matrix on joint level, they use an optimization approach to hold the control of the individual joints independently. For their approach the control of the upper body is separated from the control of the legs and thus not using the full potential of a whole-body controller.

In the paper of Bellicoso et al. [11], the door opening task is addressed with the four-legged mobile robot ANYmal. The locomotion planning is based on a whole-body center of mass optimization to guarantee stability of the platform. The individual control tasks are optimized using a hierarchical optimization based on quadratic programming. This approach allows for prioritising certain tasks over others and to

include constraints dependent on e.g. contact which is important for the stance of the quadruped. The door opening was done using direct force control by tracking desired forces at the end effector. The forces are calculated depending on the current door angle and the difference between desired and actual velocity of the end effector. To compute the current door angle the Taubin circle fit method was applied. Bellicoso et al. regard direct force control as a safe method to interact with environments since it omits controller dynamics (e.g. from an impedance controller) that have to be taken care of for the stability of the coupled system. However, the generation of suitable force trajectories is not trivial without a model of the environment.

Bodie et al. [12] describe two control approaches using active interaction force control for a drone for contact-based inspection tasks. One method referred to as axis selective impedance control solves the inverse dynamics equation for a desired system mechanical impedance based on an estimate of the interaction wrench. This controller allows to move into collision in a stable manner. Since they observed limitations regarding the exact location and precision of the interaction force estimated by the external wrench estimator, they derived a second approach the intentional interaction controller, which uses a force sensor. This approach uses direct force control if the confidence close to the surface is high enough and is combined with the impedance controller. The force applied to the surface is chosen by the user depending on the task.

At the Hannover Fair 2013, Leidner et al. [13] presented the window cleaning task with the mobile robot Justin. The task was addressed from a higher-level point of view, including reasoning about how the robot should locate itself to lift up the cleaning tool as well as planning how to move to the window and locate itself for the cleaning task. Action libraries were used that stored all information about a task depending on the tool that is in application. In this case the cleaning tool contains information about its dimensions, the weight for the lifting and dynamic motion as well as the usage. To clean the window a cartesian force controller with contact stiffness was used that was derived from an attractive potential. Target force and contact stiffness also belong to the action library of the cleaning tool.

All these papers inspired us in finding a control approach that can deal with following trajectories while being in contact with an unknown environment. Regarding that the core controller in our approach will be an MPC, the setups of Lee et al. [9] and Bellicoso et al. [11] which are mostly related to our system. We want to avoid to include overly specific dynamic models, e.g. the door, into the constraint optimization formulation as it was done by Lee et al. [9]. We see the impedance control approach by Lee et al. [10] as an intuitive way to follow a planned trajectory, but kinematic uncertainties demand an estimation of the environment state as used by Bellicoso et al. [11] with the circle fit to update the trajectories, and dynamic uncertainties demand some sort of force feedback as used by Bodie et al. [12]. This would also allow us to estimate the mass of an object online to manipulate it precisely in space without the need of having the mass parameter in an action library as proposed by Leidner et al. [13].
Before we present our controllers to solve the robot-environment interaction problem in a general way in chapter 3, we first define the system we are working with and present the MPC OCS2 formulation in the next chapter.

Chapter 2

System

In this chapter we describe the ballbot and its components that we are working with in section 2.1 and point out the motivation of using a ballbot as a service robot as well as its challenges. We also give a brief overview of how the optimal control for switched systems (OCS2) framework is used in our case in section 2.2. In section 2.3 we state the constraint optimal control formulation which we are using for the robot-environment interaction tasks.

2.1 System Description

We work with the ballbot, a robot that balances on a ball with a 4-DOF arm attached to the base. Such a system can be simply seen as an inverted pendulum in three dimensions. The early research by Fankhauser and Gwerder [14] that only worked with the base of the system resulted in an accurate model for the ballbot and the derivation of a LQR controller to stabilize the robot. Most recently, the system was enhanced with an arm in the work of Minniti et al. [8] where a whole-body MPC controller is used to navigate the system to the desired base and end effector positions in a stable manner.
Many service robots have a base on wheels such as the service robot LIO by F&P Robotics [15] or the humanoid Justin by DLR. Such a base simplifies the controller because the base can be controlled separately from the upper body or the robot's arm without any stability issues. On the other hand, the base area is quite large in order to avoid the overturning of the system, and in case the overturn could happen, the robot does not have the dynamic capabilities to stabilize itself again. Also, navigation through crowded ares is difficult because of the large base area.
These drawbacks are a big motivation to use a ballbot since its base area is very small and the robot has the dynamic capabilities to stabilize itself which is also pointed out by Shomin et al. [16] who used a ballbot for a sit-to-stand assistance task with humans. The motivation behind a whole-body controller is not only the stabilization of the system but also that an optimal control solution, considering the full body, results in human-like motions which include the shifting of the center of mass out of the statically stable area to apply high forces to the environment, e.g. the pushing of a heavy locker. Therefore, the system is sometimes also called the bionic ballbot.
On the other hand, a minimal failure of the controller can lead to a critical damage for the robot or its environment. This risk makes the ballbot a rarely seen mobile platform for commercially oriented applications. Another disadvantage is that the ballbot needs to keep staying in motion to balance itself which makes applications that include visual servoing a bit more challenging. These points motivate us to

strive new limits in the research on the ballbot by investigating new techniques for robot-environment interaction control.

Compared to the paper by Minniti et al. [8], which also worked with the ballbot, the system has now the 4-DOF DynaArm instead of the 3-DOF arm from the paper. The speciality of the DynaArm is that the elbow joint is driven by a belt and the corresponding actuator is located on top of the base of the ballbot. This belt transmission system allows that torques which act on the third joint of the arm do not need to be compensated by the second joint compared to a serial system, but act directly on the base. This allows for a higher payload. Since the URDF conventions, the Gazebo simulation and the OCS2 MPC controller use the serial convention, a mapping for the joint torque, position and velocity was derived in the work of Preisig et al. [17]. This mapping is used whenever actuator data of the real system is used by the MPC controller or MPC controller commands are sent to the actuators. The actuators that are now used on the robot are DynaDrives, developed at the Robotics Systems Lab at ETH, instead of series elastic actuators that were used before. These new drives allow a high nominal torque thanks to an integrated fan.

2.2 Optimal Control for Switched Systems (OCS2)

The OCS2 toolbox developed by Farshidian et al. [5], [6] and [7] enables to solve a constrained optimal control problem of a general robot in a receding horizon fashion. The toolbox allows to switch the constraints, the dynamics and the cost function of a constrained optimal control problem online and is also capable of optimizing over the switching times. The algorithm works by forward integrating the system dynamics around the last optimal control solution, linearizing the system dynamics and computing a second-order approximation of the cost function around the forward integrated state and input. Finally, the constrained optimization is solved with a Lagrange multiplier method leading to an input control law. This controller can be used for any robot that has an URDF representation file by using the automatic differentiation version of the robotics code generator RobCoGen.

2.3 System Model and MPC OCS2 Formulation

The model of the ballbot that is used for the OCS2 MPC controller has 8 degrees of freedom, 2 from the ball x- and y-position, 3 from the base yaw, pitch and roll orientation and additional 3 from the arm's joints positions, where only 3 of the 4 joints are controlled by the MPC. The robot has 6 actuators controlled by the MPC, 3 that act on the ball and 3 on the arm. The model of the ballbot with arm used by the MPC is shown in figure 2.1 which is adapted from Minniti et al. [8].

State and Input

The state vector $x_{ss} = (q, v)$ of the system used in the MPC formulation consists of the robot's generalized coordinates q and generalized velocities v as described by Minniti et al. [8]

$$q = \begin{pmatrix} p_{IS} \\ \theta \\ q_a \end{pmatrix}, \; v = \begin{pmatrix} \dot{p}_{IS} \\ \omega_{IB} \\ \dot{q}_a \end{pmatrix} \tag{2.1}$$

where q consists of the ballbot's ball / sphere $\{S\}$ position p_{IS} in the world frame $\{I\}$, the orientation θ of the base $\{B\}$ in Euler angles in the world frame and the

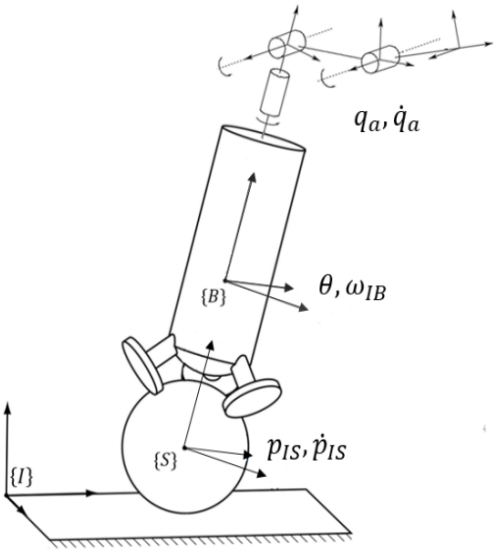


Figure 2.1: The MPC ballbot model adapted from Minniti et al. [8]

angle of the actuators q_a of the arm. The velocity v consists of the ball velocity $\dot{p}_{IS}$, the angular velocity ω_{IB} of the base and the derivative of arm joint coordinates $\dot{q}_a$. The input vector $u = (\tau)$ consists in our formulation of the actuator torques only, compared to the formulation of Minniti et al. [8] which used the formulation $u = (\tau, \lambda_{ee})$ where the end effector wrench applied by the robot to the environment λ_{ee} is also part of the input. We do not consider λ_{ee} as an input in this work but use it as a parameter in the constraints which is described in the following.

Cost Function

The cost function J that is minimized in the constrained optimal control problem over the horizon T is

$$J = \min_{\tau} \int_0^T ||x_{ss}^d \boxminus x_{ss}||_{Q_{ss}}^2 + ||\tau^d - \tau||_R^2 + ||x_{ee}^d \boxminus x_{ee}||_{Q_{ee}}^2 \, dt, \tag{2.2}$$

where x_{ee} is the end effector state consisting of orientation and position. The box minus $\boxminus$ indicates the special formulation for the orientation error as described in the paper of Minniti et al. [8, p. 3] in equation 3. State errors and end effector state errors are penalized with the positive semi-definite matrices Q_{ss} and Q_{ee}. The actuator torque error is penalized with the positive definite matrix R.
For this thesis we will work with the following cost values if not stated differently. The cost matrices that we use are diagonal and therefore only the trace of the matrix is written. The end effector state x_{ee} consists of end effector roll, pitch and yaw as well as x, y, and z position. The end effector cost is

$$Trace(Q_{ee}) = \begin{bmatrix} 0 & 0 & 100 & 200 & 200 & 200 \end{bmatrix}. \tag{2.3}$$

No cost is in the end effector roll and pitch since the system has not enough degrees of freedom to also control these two coordinates in all configurations.
The state cost is

$$\begin{aligned} Trace(Q_{ss\ q}) &= \begin{bmatrix} 0 & 0 & 600 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \\ Trace(Q_{ss\ v}) &= \begin{bmatrix} 20 & 20 & 20 & 10 & 10 & 20 & 3 & 10 \end{bmatrix}, \end{aligned} \tag{2.4}$$

where the cost on the state position coordinates is zero so that the desired end effector state can be optimally tracked. Only the yaw coordinate of the base is given a high cost to avoid oscillations which only occur on the hardware, but not in simulation.
The input cost is

$$Trace(R) = \begin{bmatrix} 2 & 2 & 2 & 0.5 & 0.1 & 0.1 \end{bmatrix}. \tag{2.5}$$

By testing different cost values in simulation, we found that the cost entry for joint 2 of the arm is of major importance for the door opening and lifting tasks since it shapes how much the optimizer will use the base or the arm respectively for the task. The value of 0.1 is a good trade-off.

Constraints

The inequality constraints of the optimization problem are the actuator's position q_{act} limits (2.6) and torque τ limits (2.7)

$$q_{act\ min} \leq \quad q_{act} \quad \leq q_{act\ max}, \tag{2.6}$$

$$\tau_{min} \leq \quad \tau \quad \leq \tau_{max}, \tag{2.7}$$

to avoid self-collision and to respect the physical limits and the saturation of the actuators.
For these values the mapping from the DynaArm to the serial convention that was mentioned in section 1.1 has to be considered since the OCS2 MCP controller uses the serial convention. The equality constraints of the optimization problem are the initial condition (2.8) and the system dynamics (2.9)

$$x_{ss}(0) = x_0 \tag{2.8}$$

$$\dot{x}_{ss} = f(x_{ss}, u). \tag{2.9}$$

The system dynamics $\dot{x}_{ss} = f(x_{ss}, u)$ can be stated as follows

$$\dot{q} = T(q)v \tag{2.10}$$

$$\dot{v} = M(q)^{-1}(-N(q,v) + S^T(q)\tau - J_{ee}^T\lambda_{ee}), \tag{2.11}$$

where $T(q)$ in equation (2.10) is the mapping from the generalized velocities to the derivatives of the generalized coordinates. The second equation (2.11) describes the robot dynamics with joint space inertia matrix $M(q)$ and $N(q,v) = C(q,v)v + G(q)$ with Coriolis and centrifugal terms $C(q,v)v$ and the gravity term $G(q)$. The matrix S^T is the mapping from actuator torques to generalized torques as derived by Fankhauser et al. [14]. Finally, J_{ee}^T is the end effector Jacobian that maps the end effector wrench λ_{ee} to generalized torques.
The OCS2 toolbox uses the automatic differentiation library CppAD for the differentiation of the cost and constraints. The parameters x_{ee} and λ_{ee} do not belong to the state x_{ss} and input u of the MPC control problem formulation, but are fed into the cost and constraints as parameters that can be manipulated online. Since we do not treat the end effector wrench λ_{ee} as an optimization variable in this thesis, we will use this parameter interface to command the desired end effector forces by setting them directly in the system dynamics constraint. In section 3.5 an online system identification method is presented where the robot's system dynamics in equation 2.11 will be enhanced by the dynamics of the estimated environment online. For this purpose the parameter interface and the synchronization module provided by OCS2 are used to set the new dynamics parameters at the beginning of an optimization cycle.

Control

This constraint optimization problem will be solved up to 200 times a second where in each cycle the time horizon T of the cost function is around 1 second. For control the first element of the resulting optimal state and input sequence is used at each cycle. The optimal state and input are mapped to actuator position, velocity and torque in the DynaArm convention. To compute the torque command for the actuators, the optimal torque is used as a feed forward term and the optimal position and velocity are used as references for a PID feedback controller in the drives. This is explained in more detail in the next chapter.

2.4 System Summary

In this chapter we described the ballbot with which we are working in this thesis in section 2.1. We stated the OCS2 MPC formulation in 2.3 which is the core controller based on which we will add additional control structures for interaction tasks with unknown environments in the next chapter.

Chapter 3

Control

In this chapter we describe the control methods, estimation and trajectory generation algorithms that we use for the MPC-based robot-environment interaction control and apply in the door opening and object lifting tasks. The goal is to formulate our method as general and simple as possible so that the same method can also be used for other manipulation and interaction tasks. Therefore, we present the assumptions in section 3.2 that our methods are based on regarding the task formulation, the environment model and the stability of the coupled system. Based on these, we first describe the online trajectory generation method we use in section 3.3. Then, we present the motivation, formulation and stability conditions of our control strategies that use impedance control in 3.4, the model identification adaptive control (MIAC) in 3.5, the model reference adaptive control (MRAC) in 3.6 as well as combinations of MIAC and MRAC in section 3.7.

3.1 Control Task

The control task consists of following a trajectory with the end effector while having contact with an unknown environment at the end effector. The trajectory that we consider is a position velocity time (PVT) trajectory. The trajectory is generated based on prior knowledge about the task or information acquired through a vision system. During the task the trajectory can be updated by using estimation algorithms to better deal with kinematic constraints (e.g. for the door opening). The controller that we want to design should track the desired trajectory adequately by handling the model error in the MPC. For some controllers we also want to estimate a dynamic model of the environment along the trajectory, that could allow to achieve a better performance in a second attempt. In the following, we state the assumptions based on which we design and analyze our controllers.

3.2 Assumptions

The controllers that will be presented in the following sections are derived based on assumptions about the environment model, the MPC and the robot which we state in this section. Since we deal with a ballbot, a robot that is only dynamically stable, the MPC controller has to handle the stability of the robot itself beside the coupled stability of the robot-environment interaction. For the controller architecture that we choose, the whole-body MPC controller will therefore be the core component. The actuator torques τ, which are the optimizers of the constrained optimization problem from section 2.2, will be commanded to the actuators as a feedforward term. Since the MPC only runs up to frequencies of 200 Hz, the actuators are

controlled individually with a PID controller and the feedforward term at a much higher rate of 1500 Hz. The commanded actuator torque τ_{act} is

$$\tau_{act}(t) = \tau_{MPC} + K_P(q_{MPC} - q_{act}(t)) + K_D(\dot{q}_{MPC} - \dot{q}_{act}(t)) + K_I \int_0^T (q_{MPC} - q_{act}(t))dt, \tag{3.1}$$

where the optimal actuator position q_{MPC} and velocity $\dot{q}_{MPC}$ follow from the optimized state policy x_{ss} of the MPC formulation in 2.2. In the results of Minniti et al. [8], figure 7, it was shown that the contribution of the feedforward terms τ_{MPC} in equation (3.1) dominates the commanded actuator torque τ_{act} compared to the feedback terms. We use this result as foundation that the URDF robot model used for the system dynamics in section 2.3 can be very accurate so that the dynamics of the robot are almost perfectly compensated by the MPC controller. Therefore, we will neglect the dynamics of the robot when we analyze the stability of the coupled system when the robot is in interaction with the environment.
Another assumption is that the contact and the interaction is happening only at the end effector of the robot, not far from the tool center point (TCP) of the end effector and the corresponding Jacobian J_{ee}. This allows us to only consider a controller that is a function of the end effector's position, velocity and force and a dynamic model of the environment that is acting at the end effector to analyze their stability.
We will also not consider a contact model and thus any contact dynamics between the robot's TCP and the environment's contact surface since we assume a rigid connection between the two, established by the Robotiq gripper. Although contact dynamics may occur, we assume them to be of minor order, especially for the door opening and object lifting tasks that we tackle in this thesis. This assumption is, of course, not valid for applications that highly depend on a contact model such as the cleaning of a window with a soft sponge. In such an application, pure force control could be used in normal direction of the cleaning plane to push the sponge on the window and to handle the contact dynamics. Our controllers could then be used to move the sponge in the normal plane to deal with static friction and damping from the sponge-window interaction.

3.2.1 Environment Model Along the Trajectory

This section states the reflected environment dynamics at the end effector when the TCP follows the desired trajectory. We assume that the dynamics of the environment projected along the trajectory can be characterized by a linear relationship between force (input u) and the TCP position, velocity and acceleration (state x). Therefore, an adequate and commonly used model is the mass m, damper d and spring k impedance model with spring resting position x_0,

$$m\ddot{x} + d\dot{x} + k(x - x_0) + f_{static} = u. \tag{3.2}$$

In this model x and u are the projected end effector position and force along the desired PVT trajectory with projection vector $\boldsymbol{v}_n$

$$\begin{aligned} x &= \boldsymbol{v}_n^T \boldsymbol{x}_{ee} \\ u &= \boldsymbol{v}_n^T \boldsymbol{\lambda}_{ee}. \end{aligned} \tag{3.3}$$

Besides the mass, damper and spring components, we also consider a static force component f_{static} that could come from e.g. static friction effects or gravity. This model may be oversimplified for environments that are characterized by highly non-linear dynamics, but this model assumption allows us to use tools from the

well established linear control theory and lets us design a general control solution. We also assume that the dynamics of the environment are constant over an interaction interval, making the environment a linear time-invariant (LTI) system. This assumption is needed for the convergence of the estimated parameters using recursive least squares in the model identification adaptive controller in section 3.5 and the convergence of the tracking error for the model reference adaptive controller presented in section 3.6.
In the following we describe how we generate the PVT trajectories that should be followed by our controllers.

3.3 Trajectory Generation

Since the higher level control task is to follow a desired trajectory as described in section 3.1, we need to generate those online with the possibility to update them efficiently. To generate position velocity time (PVT) trajectories with bounded acceleration and velocity online, we use the method derived by Ramos et al. [18]. In the following we summarized their result.

3.3.1 Time-Optimal Trajectory Algorithm

The trajectory generated by the algorithm of Ramos et al. [18] is time-optimal and handles the initial and final constraints which are the initial position and velocity and the final position and velocity as well as the position, velocity and acceleration bounds over the trajectory horizon.
The input of the algorithm is the initial position p_i and velocity v_i and final position p_f and velocity v_f as well as the velocity and acceleration bounds v_{max} and a_{max}. The direction of the trajectory is determined with the sign parameter $\tilde{s}$

$$\begin{aligned} \Delta p &= p_f - p_i \\ \Delta v &= v_f - v_i \\ \Delta p_{crit} &= \frac{sign(\Delta v)(v_f^2 - v_i^2)}{2a_{max}} \\ \tilde{s} &= sign(\Delta p - \Delta p_{crit}). \end{aligned} \tag{3.4}$$

For the brevity of space, we state their result only for the trapezoidal velocity profile case which is the most common profile for longer trajectories where the maximal velocity is reached and held over a certain time phase compared to the triangular profile where the maximal velocity is not reached. In this case the trajectory can be divided into three parts. The acceleration phase till time T_1, the constant velocity phase till time T_2 and deceleration phase till time T_f

$$\begin{aligned} T_1 &= \frac{\tilde{s}v_{max} - v_i}{\tilde{s}a_{max}} \\ T_2 &= \frac{1}{v_{max}}\left(\frac{v_f^2 + v_i^2 - 2\tilde{s}v_{max}v_i}{2a_{max}} + \tilde{s}\Delta p\right) \\ T_f &= T_2 + \frac{v_f - \tilde{s}v_{max}}{\tilde{s}a_{max}}. \end{aligned} \tag{3.5}$$

This results in the following position profile $p(t)$ and velocity profile $v(t)$ and acceleration profile $a(t)$

$$
\begin{aligned}
&For\ 0 \leq t \leq T_1: && && (3.6)\\
&\qquad a(t) &&= \tilde{s} a_{max} \\
&\qquad v(t) &&= \tilde{s} a_{max} t + v_i \\
&\qquad p(t) &&= \frac{1}{2} \tilde{s} a_{max} t^2 + v_i t + p_i \\
&For\ T_1 < t \leq T_2: \\
&\qquad a(t) &&= 0 \\
&\qquad v(t) &&= \tilde{s} v_{max} \\
&\qquad p(t) &&= \tilde{s} v_{max} (t - T_1) + p(T_1) \\
&For\ T_2 < t \leq T_f: \\
&\qquad a(t) &&= -\tilde{s} a_{max} \\
&\qquad v(t) &&= -\tilde{s} a_{max} (t - T_2) + \tilde{s} v_{max} \\
&\qquad p(t) &&= -\frac{1}{2} \tilde{s} a_{max} (t - T_2)^2 + \tilde{s} v_{max} (t - T_2) + p(T_2).
\end{aligned}
$$

The algorithm described above is used to generate the linear trajectories to move the ballbot to certain positions, e.g. to lift up an object, or to generate the circular trajectories for the door opening task. Its simple implementation allows a fast recomputation which is important for the door opening task where the door hinge position and current angle are estimated online and the trajectory thus needs to be recomputed. The parameters that have to be selected with care are the maximal velocity v_{max} and maximal acceleration a_{max} since these have an influence on the control performance and stability. A disadvantage of the algorithm above is that the acceleration behaves like a step response and can therefore not be tracked by a physical system with saturation limits.
In the next section we state the impedance control approach to deal with robot-environment interaction control.

3.4 Impedance Control

One of the most used control approaches for robot-environment interaction control is impedance control. The mechanical impedance is the relation between velocity and force and thus characterizes the behavior of a mechanical system. Task space impedance control for robot-environment interaction applications is described extensively by Natale [2] and Vukobratovic et al. [3]. Recently, impedance control was also used by Angelini et al. [19] to characterize the behavior of the base of the quadruped ANYmal by optimizing the impedance parameters online to create a critically damped behavior of the COM of ANYmal to a disturbance.
We use impedance control to help the MPC controller described in section 2.2 tracking the desired end effector position x_d and velocity $\dot{x}_d$ trajectory generated by the algorithm from section 3.3. The impedance controller is

$$\boldsymbol{f_Z} = K_P(\boldsymbol{x_d} - \boldsymbol{x}) + K_D(\dot{\boldsymbol{x}}_{\boldsymbol{d}} - \dot{\boldsymbol{x}}), \tag{3.7}$$

where K_P is the impedance controller's stiffness matrix with units $[\frac{N}{m}]$ and K_d is the damping matrix with units $[\frac{Ns}{m}]$. The impedance controller will compensate unmodeled environment dynamics by generating a force proportional to the position and velocity tracking error. The stiffness and damping matrices therefore allow to intuitively shape their gains dependent on the task and the desired compliance, for example high stiffness and damping along the trajectory to be tracked and small gains in the orthogonal direction.

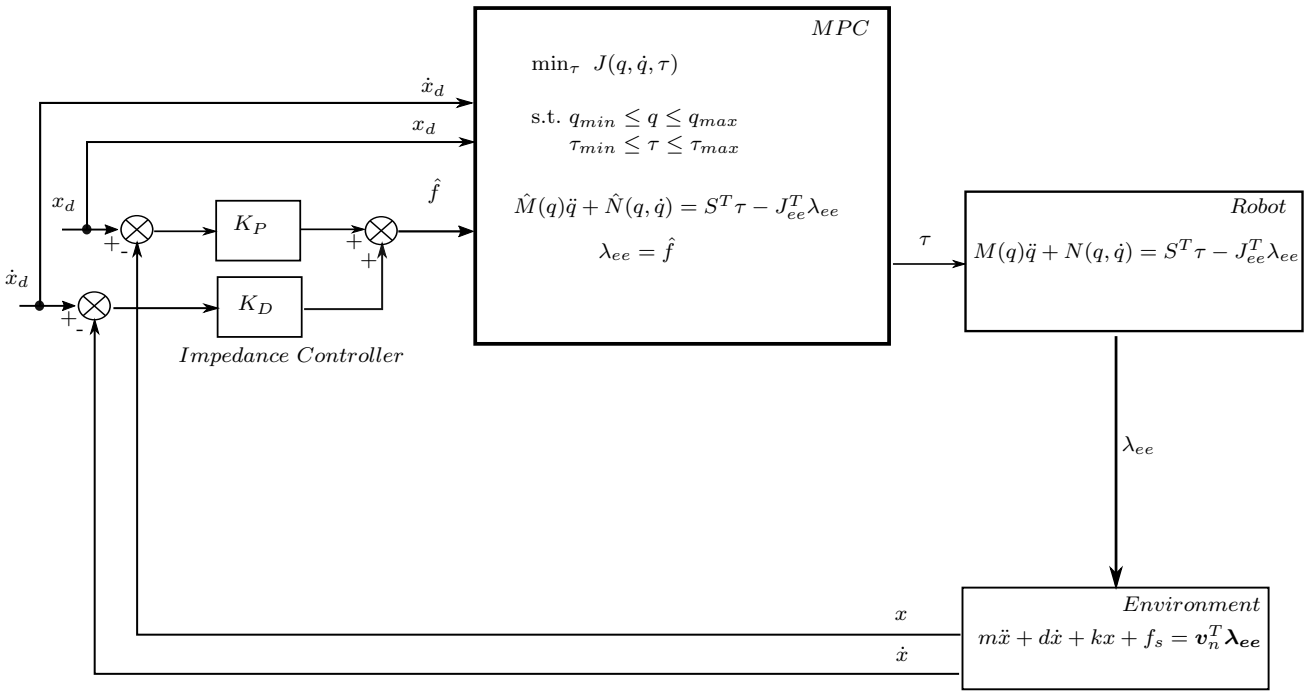


Figure 3.1: Impedance control scheme

The control scheme with the impedance controller together with the MPC is visualized in figure 3.1. The desired end effector position x_d and velocity $\dot{x}_d$ are commanded to the MPC and the desired end effector force $\hat{\boldsymbol{f}} = \boldsymbol{f_Z}$ from the impedance controller is set in the system dynamics to solve the constrained optimization problem from section 2.3 with the additional constraint

$$\boldsymbol{\lambda_{ee}} = \boldsymbol{f_Z}. \tag{3.8}$$

In the next section, we analyze the stability of the coupled system under the assumption stated in section 3.2 that the robot model in the MPC is very accurate

$$\begin{aligned} \hat{M} &\approx M \\ \hat{N} &\approx N \end{aligned} \tag{3.9}$$

and that, therefore, primarily the impedance controller stabilizes the tracking of the desired trajectory.

3.4.1 Stability Analysis

When we project the impedance controller from equation (3.7) along the direction of the trajectory $\boldsymbol{v_n}$, we can analyze the stability of the resulting SISO system

$$m\ddot{x} + d\dot{x} + kx = k_p(x_d - x) + k_d(\dot{x}_d - \dot{x}),. \tag{3.10}$$

where k_p and k_d are the projected stiffness and damping gains. In equation 3.10 we set the static force component to zero $f_{static} = 0$ for simplification and because the impedance controller could not achieve a zero tracking error for a static force. Since the impedance controller is implemented in discrete time running at 400 Hz, a zero order hold (ZOH) element is used to represent the resulting time delay. The delay term is simplified with a Padé approximation to receive a linear transfer function,

$$ZOH = \frac{800}{s + 800}. \tag{3.11}$$

To analyze the stability, the closed loop transfer function $T = \frac{x}{x_d}$ needs to be stable where

$$T = \frac{800(k_d s + k_p)}{ms^3 + (d + 800m)s^2 + (d + k + k_d)800s + (k + k_p)800}. \tag{3.12}$$

A way to formally analyze the stability of the parametric system is to use the Routh Hurwitz determinants criterion. This theorem states that all determinants of the Routh Hurwitz Array need to be larger than zero. Since we know that $m \geq 0$, $d \geq 0$ and $k \geq 0$, the only term that is not greater than zero in any case is the following one which gives an upper bound on the impedance stiffness

$$\frac{k_d A + B}{m} \geq k_p, \tag{3.13}$$

with

$$\begin{aligned} A &= d + 800m \\ B &= d(d + \frac{k}{800} + 800m). \end{aligned} \tag{3.14}$$

This analysis raises the motivation to select the impedance damping k_d very high, but one has to be careful since the velocity estimation of the end effector can be noisy or not continuous due to contact dynamics which then get amplified by k_d and could destabilize the coupled system. Therefore, we try to choose k_p as low as allowable regarding the tracking error and k_d as high as possible while considering the influence of noise.

Although the design of the impedance controller appears intuitive with the stiffness and damping matrices tuned depending on the desired compliance for the task, simulation results reveal that finding stable stiffness and damping matrices with higher gains for good tracking is difficult. Therefore, this method would require a prior tuning for each task, making it not suitable for general applicability. Also the fact that a zero tracking error is not possible in the case of a static force component in the environment's dynamics in equation (3.10) reveals that additional control action is needed. Nevertheless, we consider the impedance controller very useful as a robust control method for the initial contact phase to gain information for the estimators and the adaptive controllers. These methods will be presented in the following sections.

3.5 Model Identification Adaptive Control (MIAC)

In this section we want to improve the impedance controller which fails to deal with environments that have a static force component and which also has a large tracking error if the stiffness and damping matrices are chosen to be more compliant and robust.
One method to improve the tracking and the coupled stability is to gain information about the environment model online. The general idea of model identification adaptive control is described by Öreg et al. [20]. Information about the dynamic parameters of the environment are acquired online. This information can be used to shape a controller, or in our case, to enhance the system dynamics constraint of the MPC controller shown in 2.2. A simpler version of this method would be to use force feedback, but estimating the parameters allows for a more accurate prediction by the MPC. In section 3.5.1 we describe how we used a force sensor in Gazebo which helped us to tune a disturbance observer presented in 3.5.2 that estimates the external wrench at the end effector. Then we describe the recursive least square algorithm in section 3.5.3 that we use to estimate the dynamic parameters based on the end effector's position, velocity and force in discrete time. Finally, the control scheme of MIAC is presented in section 3.5.4.

3.5.1 Gazebo Force Sensor

The simplest way to get an estimate of the forces that the robot raises towards the environment is to use a force torque sensor. In simulation, we use a Gazebo force torque sensor at position p_s, behind the Robotiq gripper with its center of mass at p_g and mass m. The measured wrench $\hat{w}^S$ in the sensor frame is first compensated by the weight of the gripper as described by Erdogan et al. [21]

$$w^S = \hat{w}^S + \begin{pmatrix} I_{3x3} & 0_{3x3} \\ [p_g - p_s]_x & I_{3x3} \end{pmatrix} R_{SI} \begin{pmatrix} 0 & 0 & -mg & 0 & 0 & 0 \end{pmatrix}^T, \tag{3.15}$$

where R_{SI} is the transform from the world frame to the sensor frame. Then the wrench is transformed to the tool center point (TCP) of the end effector at position p_{tcp} and, finally, rotated to the world frame

$$w^I = R_{IS} \begin{pmatrix} I_{3x3} & 0_{3x3} \\ [p_{tcp} - p_s]_{\mathrm{x}} & I_{3x3} \end{pmatrix} w^S. \tag{3.16}$$

The measurements of the force torque sensor in Gazebo are very noisy and the force measurements are varying between $\pm 5\ N$ at a high frequency. We use a second order Butterworth filter with a cut-off frequency at 10 $\frac{rad}{s}$ to remove the high frequency oscillation without having too much delay. This force sensor is used in simulation to tune the disturbance observer presented in the next section.

3.5.2 Disturbance Observer (DOB)

Since the real ballbot equipped with a Robotiq gripper does not have a force sensor, an estimator is needed. We use the disturbance observer (DOB) method described by Bodie et al. [12] which has the advantage that no acceleration of the generalized coordinates $\ddot{q}$ needs to be used which are in general very noisy. The update equation of the estimator in discrete time is

$$\hat{\tau}_{ext}[k+1] = K_o(M\dot{q}[k] - \sum_{i=0}^{k}(S^T\tau_{act}[i] - C(q[i], \dot{q}[i])\dot{q}[i] - G(q[i]) + \hat{\tau}_{ext}[i])T_s). \tag{3.17}$$

Bodie et al. show that the equation above (3.17) in continuous time represents a first-order low pass filtered estimate of the true generalized external torque τ_{ext}.

$$\hat{\tau}_{ext} = \frac{K_o}{s+1}\tau_{ext} \tag{3.18}$$

Therefore, K_o can be tuned according to the noise in the state ($q, \dot{q}$ and the actuator torque τ_{act}). We will assume that the estimated generalized external torque is coming from a force acting only at the end effector, as stated in section 3.2. Under this assumption we can compute the external wrench with

$$w_{ext} = J_{ee}^{(-T)}\hat{\tau}_{ext}, \tag{3.19}$$

where the pseudo inverse is used for the inversion of the end effector Jacobian in the world frame. We discovered that the estimation of forces in x- and y-direction works well in this case, but forces in z-direction are estimated badly. The reason is that z-direction forces are underrepresented in the Jacobian since the base cannot move in this direction. Therefore, the least square optimal solution will result in high z-direction errors. To circumvent this issue we use the method presented by Grandia et al. [22] which has also the advantage that forces acting on the end effector and the base can be estimated at the same time. For that purpose a Jacobian J_0 is introduced, relating the floating base and the generalized coordinates. Combining J_0 with the end effector position Jacobian J_{pee} results in an augmented Jacobian J_{aug} that has full rank and relates generalized torques to forces in x- and y-direction acting on the base, moments around x-, y- and z-axis acting on the base and forces in x-, y- and z-direction acting on the end effector at the same time in the inertial frame.

$$\begin{aligned} J_0 &= [I_{5x5}\ 0_{5x3}] \\ J_{aug} &= [J_0;\ J_{pee}] \\ w_{aug} &= J_{aug}^{(-T)}\hat{\tau}_{ext} \end{aligned} \tag{3.20}$$

Commanded Actuator Torque

For the disturbance observer, the measured actuator torque is needed. Hardware tests have revealed that these values are very inaccurate due to a wrong torque constant and that the measured torque also has drift. Therefore, we use the commanded actuator torque instead of the measured one. The actuator commands in the drives are computed with equation (3.1) at 1500 Hz, whereas the estimator is running at 400 Hz. Because of this frequency difference we compute only the PD-feedback terms manually at 400 Hz and neglect the integral term contribution,

$$\tau_{act} = \tau_{MPC} + K_D(\dot{q}_{MPC} - \dot{q}_{act}) + K_P(q_{MPC} - q_{act}). \tag{3.21}$$

Equation (3.21) is thus used as actuator torques τ_{act} in the estimator (3.17).
The estimated interaction wrench described in this section is used as input for the dynamics parameter estimation algorithm using recursive least squares as shown in the next section.

3.5.3 Recursive Least Squares (RLS)

In this section we describe how we use a Bayesian approach to find an estimate of the environment's impedance and static force along the trajectory at each time step by using recursive least squares. This method is equal to the measurement update

step (A Posteriori Update) in the Kalman filter. The algorithm minimizes the mean square error between the measured and estimated state for a linear system. We can bring the system from equation (3.2) into the state-space form with the following substitution,

$$\begin{aligned} x_1 &= x - x_0 \\ x_2 &= \dot{x}_1 = \dot{x} \\ \hat{u} &= u - f_{static}, \end{aligned} \tag{3.22}$$

where x is the position along the trajectory, x_0 is the spring initial position and $\dot{x}$ is the velocity along the trajectory because x_0 is constant. The variable u is the force along the trajectory applied by the robot to the environment. The system from equation (3.2) therefore results in the following continuous time state-space equation,

$$\begin{pmatrix} \dot{x_1} \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{d}{m} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{1}{m} \end{pmatrix} \hat{u}. \tag{3.23}$$

We discretize the system at 400 Hz, which is our estimation frequency, and use the Euler discretization method (first order) since the sampling frequency is high. This yields the difference equation

$$\begin{pmatrix} x_1[k+1] \\ x_2[k+1] \end{pmatrix} = \begin{pmatrix} 1 & T_s \\ -\frac{k}{m}T_s & 1 - \frac{d}{m}T_s \end{pmatrix} \begin{pmatrix} x_1[k] \\ x_2[k] \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{1}{m}T_s \end{pmatrix} \hat{u}[k] + \begin{pmatrix} w_1[k] \\ w_2[k] \end{pmatrix}, \tag{3.24}$$

where w accounts for the measurement noise that is assuemed to be Gaussian. If we look at the second row of equation (3.24) only and re-substitute $\hat{u}$, the following measurement update equation can be extracted,

$$x_2[k+1] = \begin{bmatrix} x_1[k] & x_2[k] & u[k] & 1 \end{bmatrix} \begin{bmatrix} -\frac{k}{m}T_s \\ 1 - \frac{d}{m}T_s \\ \frac{1}{m}T_s \\ -\frac{f_{static}}{m}T_s \end{bmatrix} + w_2[k]. \tag{3.25}$$

We can write the equation above in a simplified way as

$$z_k = H_k\theta_k + w_k. \tag{3.26}$$

The random variable w is the measurement noise with zero mean and variance R. The random variable θ at the time step zero, θ_0, can be initialized with a prior estimate of the environment dynamics' parameters and the known sampling time T_s. Also the initial variance of θ_0, P_0, can be initialized from the variance in the prior estimate of the environment. Finally, the following update equations are used to compute an estimate of the environment dynamics' parameters at each iteration,

$$\begin{aligned} K_k &= P_{k-1}H_k^T[H_kP_{k-1}H_k^T + R_k]^{-1} \\ \theta_k &= \theta_{k-1} + K_k[z_k - H_k\theta_{k-1}] \\ P_k &= [I - K_kH_k]P_{k-1}[I - K_kH_k]^T + K_kR_kK_k^T. \end{aligned} \tag{3.27}$$

The implementation of this filter has a delay of one time step since the velocity x_2 is used as the measurement z but also appears in the observations H. At each

iteration an estimate of the environment's dynamics can be computed from θ_k with

$$\begin{aligned} \hat{m} &= \frac{T_s}{\theta_k(2)} \\ \hat{k} &= \frac{-\hat{m}\theta_k(0)}{T_s} \\ \hat{d} &= \frac{\hat{m}}{T_s}(1-\theta_k(1)) \\ \hat{f}_{static} &= \frac{-\hat{m}\theta_k(3)}{T_s} \end{aligned} \tag{3.28}$$

Estimator Performance Evaluation

For the object lifting task, the performance of the dynamics' parameters estimator is easy to quantify since the true mass can be simply measured and compared to the estimate. But in the door opening case, the true parameters of the door are not available. In simulation we can set the door characteristics, and with the presented door model and relations in section 5.3, we can evaluate how the estimated parameters along the trajectory relate to the door dynamics.
In the next section, we describe how the parameters of the dynamics of the environment that we estimate online are used for control with the MPC controller.

3.5.4 MIAC and MPC Control

In this section we use the estimated parameters of the environment's impedance $\hat{m}$, $\hat{d}$ and $\hat{k}$ and static force $\hat{f}_{static}$ along the task trajectory vector $\boldsymbol{v_n}$ for control. Since our core controller is an MPC we can use the estimated environment parameters in the system dynamics constraint presented in section 2.3. The commanded end effector force in the system dynamics is then

$$\boldsymbol{\lambda_{ee}} = \boldsymbol{f_Z} + \boldsymbol{v_n}(\hat{m}\ddot{x} + \hat{d}\dot{x} + \hat{k}(x - x_0) + \hat{f}_{static}), \tag{3.29}$$

where the estimated parameters are used for the prediction over the horizon and thus for optimization. We still use the impedance controller $\boldsymbol{f_Z}$ from section 3.4 since we need to excite the environment to create data that covers information for the RLS estimator and also for the tracking of the desired trajectory.
The control scheme with the estimator, the impedance controller and the MPC is shown in figure 3.2. The estimated force along the trajectory from the disturbance observer and the position and velocity along the trajectory from forward kinematics are the inputs to the RLS estimator. The estimated dynamic parameters at each time step θ_k are then used in the MPC system dynamics constraint for optimization, making the MPC adaptive to an unknown environment.
Simulation results have shown that the estimated parameters need to be low pass filtered when they are fed back to the MPC to avoid that too much force is applied too fast by the robot. This could lead to a full extension of the DynaArm which makes the re-computation of the end effector force in the DOB (3.21) ill-conditioned and therefore causes a burst of the parameter estimates.
The MIAC control approach offers a great potential for its combination with the MPC since the estimated parameters can be used for prediction. Simulation results revealed that the tracking error is much smaller compared to when only impedance control is used. Thanks to the online identification a broader range of environments can be stably controlled than with the impedance controller only because static force

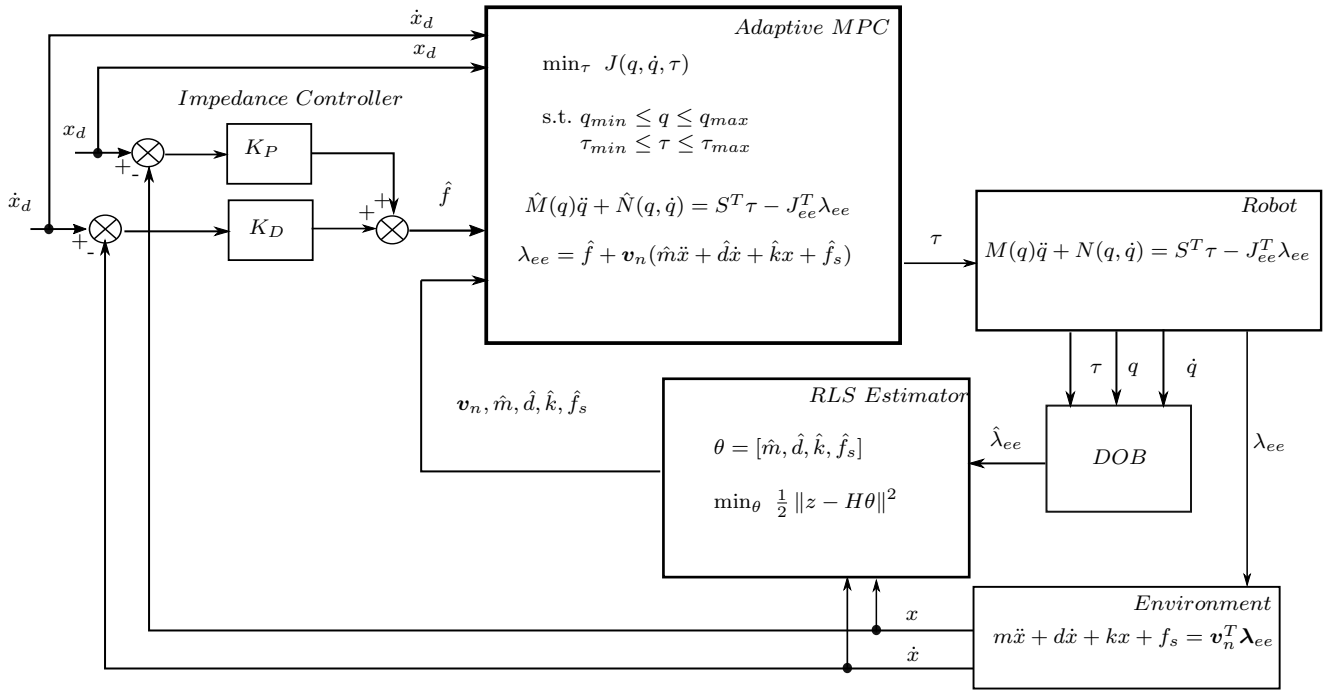


Figure 3.2: MIAC and MPC control scheme

components are compensated. This method also allows the impedance controller design to be very compliant. Nevertheless, in some cases the tracking error could not converge to zero although the estimated parameters converge. This can happen because the linear dynamic model along the trajectory (3.2) may not be correct for certain environments. In the next section, we present model reference adaptive control which will asymptotically minimize the reference tracking error to overcome this issue.

3.6 Model Reference Adaptive Control (MRAC)

The MIAC method from the last section focuses on minimizing the estimation or prediction error of the recursive least squares dynamics estimator. The estimated values are then used to adapt the system dynamics of the MPC controller online. In this section we use a different control method that does not consider the prediction error but the reference tracking errors regarding the desired position and velocity. In the case of model reference adaptive control (MRAC) described by Nguyen [23], a controller can be designed to asymptotically minimize the tracking error to zero while the control law is adapted at each time step with an adaption law. Using MRAC allows to define a reference model of the plant, e.g. a second-order system with desired damping ratio and natural frequency to fulfill a specified rise time and overshoot. The MRAC controller includes adaptive terms which will change dependent on the reference tracking error to make that the closed-loop system follows the reference model. Siciliano et al. [1] demonstrate the use of an MRAC for the control of a full robot, where the adaptive parameters are the joint space generalized mass, non-linear terms and gravity. Inspired by this approach, we formulate an MRAC controller to deal with an unknown environment along the end effector trajectory that can be modeled as a mass-damper-spring model with a static force.

3.6.1 MRAC Controller Synthesis

The general procedure of the design of an MRAC controller as described by Nguyen [23] is to define a control law that is dependent on the reference tracking error and includes adaptive parameters. Then, a Lyapunov function is formulated to derive the adaptive law and that also serves as stability proof. We consider the environment as a structured uncertain second order system

$$m\ddot{x} + d\dot{x} + kx + f_s = u, \tag{3.30}$$

where x is again the position along the trajectory transformed by the initial position of the spring x_0 as done in section 3.5.3. $[m,\ d,\ k,\ f_s]^T = \boldsymbol{\pi}$ is assumed to be constant over a robot-environment interaction interval. The proposed control law is

$$u = \hat{m}\ddot{x}_r + \hat{d}\dot{x}_r + \hat{k}x_r + \hat{f}_s + k_s\dot{\sigma}, \tag{3.31}$$

with adaptive control parameters $[\hat{m}, \hat{d}, \hat{k}, \hat{f}_s]^T = \hat{\boldsymbol{\pi}}$, a positive definite control gain k_s and the following definitions

$$\begin{aligned} e_r &= x_d - x \\ x_r &= x_d + \eta \int e_r dt \\ \dot{x}_r &= \dot{x}_d + \eta e_r \\ \ddot{x}_r &= \ddot{x}_d + \eta \dot{e}_r \\ \sigma &= x_r - x = e_r + \eta \int e_r dt \\ \dot{\sigma} &= \dot{e}_r + \eta e_r \\ \ddot{\sigma} &= \ddot{e}_r + \eta \dot{e}_r, \end{aligned} \tag{3.32}$$

where $x_d,\ \dot{x}_d,\ \ddot{x}_d$ are the reference position, velocity and acceleration and e_r is the reference tracking error and η is a positive definite tuning parameter.
Setting the controller (3.31) into the system (3.30), we receive the following system with $\tilde{m} = \hat{m} - m,\ \tilde{d} = \hat{d} - d,\ \tilde{k} = \hat{k} - k,\ \tilde{f}_s = \hat{f}_s - f_s$ and $[\tilde{m},\ \tilde{d},\ \tilde{k},\ \tilde{f}_s]^T = \tilde{\boldsymbol{\pi}} = \hat{\boldsymbol{\pi}} - \boldsymbol{\pi}$,

$$m\ddot{\sigma} + d\dot{\sigma} + k\sigma = -\boldsymbol{Y}(x_r, \dot{x}_r, \ddot{x}_r)^T\tilde{\boldsymbol{\pi}} - k_s\dot{\sigma}, \tag{3.33}$$

where $\boldsymbol{Y}(x_r, \dot{x}_r, \ddot{x}_r)^T = [\ddot{x}_r, \dot{x}_r, x_r, 1]$. This structure is very similar to the method presented by Siciliano et al. [1, p. 338]. To prove stability of the control system and derive the adaption law of the parameters $\tilde{\boldsymbol{\pi}}$, the following Lyapunov function V is used which depends on the tracking error, the adaptive parameters and a positive definite matrix K_π,

$$V(e_r, \tilde{\pi}) = \frac{1}{2}\dot{\sigma}^2 m + \frac{1}{2}\sigma^2 k + e_r^2 \eta k_s + \frac{1}{2}\tilde{\boldsymbol{\pi}}^T K_\pi \tilde{\boldsymbol{\pi}}. \tag{3.34}$$

The derivative of the Lyapunov function is

$$\dot{V}(e_r, \tilde{\pi}) = \dot{\sigma}\ddot{\sigma} m + \dot{\sigma}\sigma k + 2\dot{e}_r e_r \eta k_s + \tilde{\boldsymbol{\pi}}^T K_\pi \dot{\tilde{\boldsymbol{\pi}}}. \tag{3.35}$$

Equation (3.33) can be solved for $\ddot{\sigma} m + \sigma k$ and substituted into the derivative of the Lyapunov (3.35) function

$$\dot{V}(e_r, \tilde{\pi}) = -\dot{\sigma}^2 d - \dot{e}_r^2 K_s - 2e_r^2 \eta^2 k_s + \tilde{\boldsymbol{\pi}}^T (\dot{\tilde{\boldsymbol{\pi}}} K_\pi - \boldsymbol{Y}(x_r, \dot{x}_r, \ddot{x}_r)\dot{\sigma}). \tag{3.36}$$

To prove that the system is stable under the control law (3.31), the Lyapunov function needs to be positive definite which is given. The Lyapunov function V also needs to have a finite limit as $t \to \infty$,

$$V(e_r(t \to \infty), \tilde{\pi}(\to \infty)) < \infty. \tag{3.37}$$

And the Lyapunov function derivative $\dot{V}$ needs to be uniformly continuous so that from Barbalat's lemma, as described by Nguyen [23], it can be followed that the tracking error e_r is asymptotically stable

$$\dot{V}(e_r, \tilde{\pi}) \to 0 \Rightarrow e_r(t) \to 0 \; as \; t \to \infty. \tag{3.38}$$

These conditions imply that $\dot{V}(e_r, \tilde{\pi}) \leq 0$ and therefore the adaption law

$$\dot{\tilde{\boldsymbol{\pi}}} = \dot{\hat{\boldsymbol{\pi}}} = K_\pi^{-1} \boldsymbol{Y}(x_r, \dot{x}_r, \ddot{x}_r)\dot{\sigma}, \tag{3.39}$$

because $\boldsymbol{\pi}$ is assumed to be constant along the trajectory as stated in section 3.2. Since this is a direct adaptive control method which only aims to minimize the tracking error, the adaptive parameters $\hat{\boldsymbol{\pi}}$ will not converge to the true values $\boldsymbol{\pi}$ as $t \to \infty$.
Nevertheless, the controller in equation (3.31) can be seen as a reference feedforward controller which compensates the dynamics of the environment with an additional PD term $k_s \dot{\sigma}$. If we project the task space impedance controller from equation (3.7) along the normal vector of the environment interaction trajectory $\boldsymbol{v_n}$, the controller gain k_s is equal to the damping of the task space impedance controller $k_s = \boldsymbol{v_n}^T K_D$ and the relation to the task space stiffness is $k_s \eta = \boldsymbol{v_n}^T K_p$. This result shows that the impedance controller is part of the MRAC stability analysis since it is part of the controller together with the adaptive term $f_A = \boldsymbol{Y}(x_r, \dot{x}_r, \ddot{x}_r)^T \hat{\boldsymbol{\pi}}$. Therefore, the MRAC controller can be written as

$$\boldsymbol{\lambda_{ee}} = \boldsymbol{f_Z} + \boldsymbol{v_n} f_A, \tag{3.40}$$

with impedance controller $\boldsymbol{f_Z}$. In the next section, we present the control scheme using MRAC and MPC together.

3.6.2 MRAC and MPC Control

The control scheme that combines the MRAC controller with the MPC is shown in figure 3.3. The commanded end effector force in the system dynamics is now given by equation (3.40). This controller will asymptotically converge to a zero reference tracking error. The tuning parameters of the adaptive controller beside the impedance controller are the initial values of the adaptive terms $\hat{\pi}_0$, the parameter η and the adaption rate K_π.

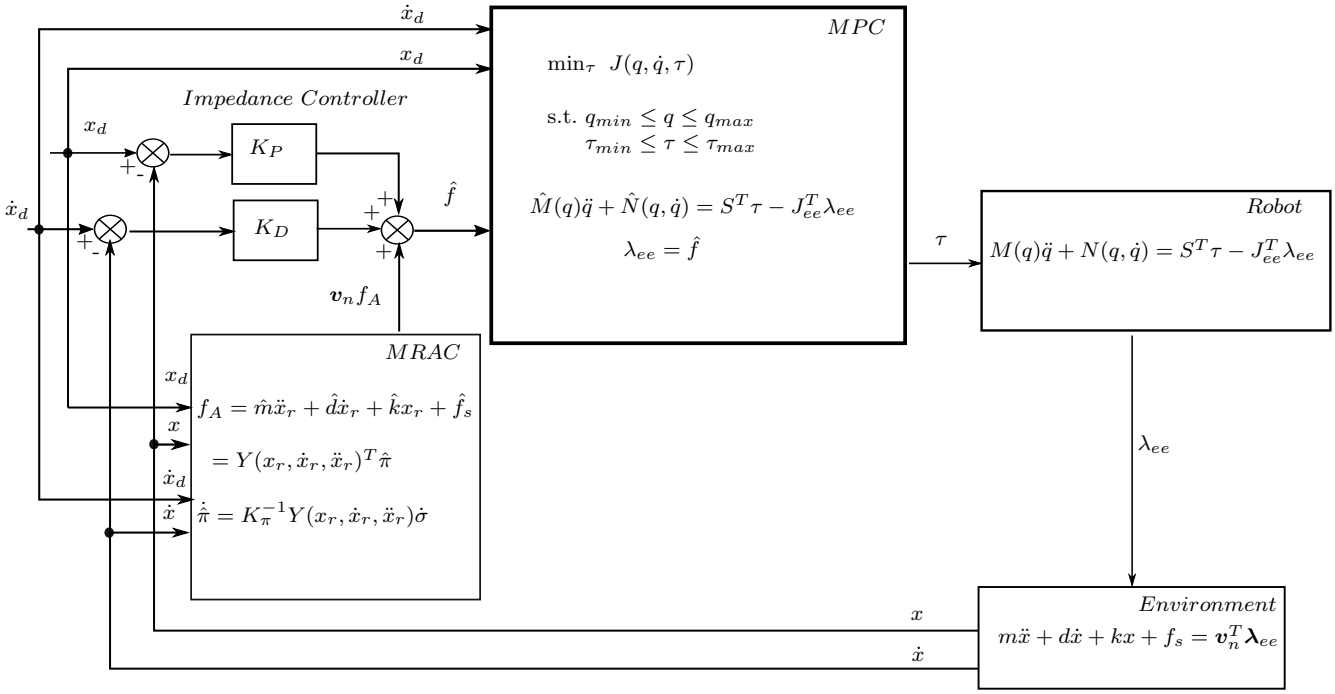


Figure 3.3: MRAC and MPC control scheme

Simulation results have shown that the integral terms in equation (3.32) for x_r and σ can lead to instabilities since it is too aggressive for higher gains of η and leads to a blow-up of $\hat{k}$ if the tracking error does not converge fast enough. We could resolve this problem by limiting the size of the integral error.

3.7 Combining MIAC and MRAC

In the sections 3.5 and 3.6, we have presented the MIAC and MRAC controllers that can follow a trajectory with an MPC while having contact with an unknown environment. An impedance controller is part of both approaches and either used to create innovation on the environment interaction in case of the MIAC controller or for the convergence of the adaptive terms in the MRAC controller. In this section we present two approaches to combine both of the previous methods. One way of doing this is to run both controllers together at the same time where feedback control will be created from both side, the reference tracking error minimization and the estimation of the environment dynamics' parameters. Another method is described by Espinoza and Roascio [24] which suggests to linearly combine the estimated parameters of the MIAC controller $\boldsymbol{\theta}$ with the adaptive parameters of MRAC controller $\hat{\boldsymbol{\pi}}$. The scheme of these two combination methods is presented in the following.

3.7.1 Combined Control with MIAC and MRAC

A straightforward combination of the MIAC and MRAC control approaches is to simply run them together which leads to the desired end effector force that is set in the system dynamics as

$$\boldsymbol{\lambda_{ee}} = \boldsymbol{f_Z} + \boldsymbol{v_n} f_A + \boldsymbol{v_n}(\hat{m}\ddot{x} + \hat{d}\dot{x} + \hat{k}(x - x_0) + \hat{f}_{static}). \tag{3.41}$$

The idea is that the combination should lead to a faster convergence of the reference tracking error since additional feedback through the environment's dynamics' parameter estimator is provided. Therefore, the control input from the MRAC controller will be smaller. But simulation results have shown that the adaptive gain K_π has to be chosen with care to avoid oscillations around the reference position during the tracking of a trajectory. The control scheme for this approach is shown in figure 3.4.

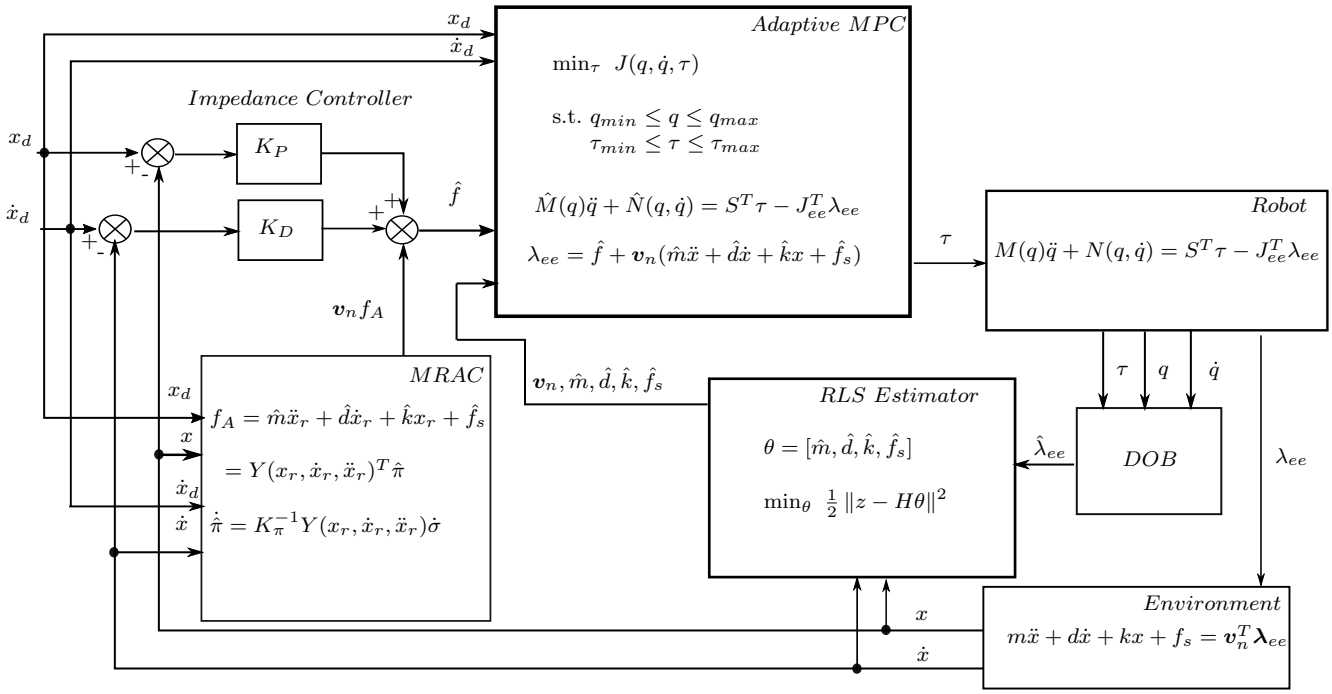


Figure 3.4: Combined MIAC and MRAC control scheme

3.7.2 Combined Adaption with MIAC and MRAC

The method described by Espinoza and Roascio [24] is a combined adaption of the estimated parameters of MIAC $\boldsymbol{\theta}$ with the adaptive parameters of MRAC $\hat{\boldsymbol{\pi}}$

$$\dot{\boldsymbol{\gamma}} = K_a\dot{\hat{\boldsymbol{\pi}}} + K_\theta\dot{\hat{\boldsymbol{\theta}}}, \tag{3.42}$$

with adaptive gains K_a and K_θ which are tuning parameters. According to Espinoza and Roascio, this should avoid bursting events in the adaptive controller and the estimator caused by unfulfilled persistence of excitation. However, in this case parameters that represent a physical system $\hat{\boldsymbol{\theta}}$ are combined with adaptive parameters $\hat{\boldsymbol{\pi}}$ which do not need to represent the real physical environment. The control scheme is shown in figure 3.5. We state this method since it is based on the controllers and estimators presented so far, but we will not apply the linear combined adaption method on the real system for the lifting and door opening tasks. We leave the derivation of proper adaptive gains K_a and K_θ for future research in this field. In the next section, we summarize all the presented interaction control methods.

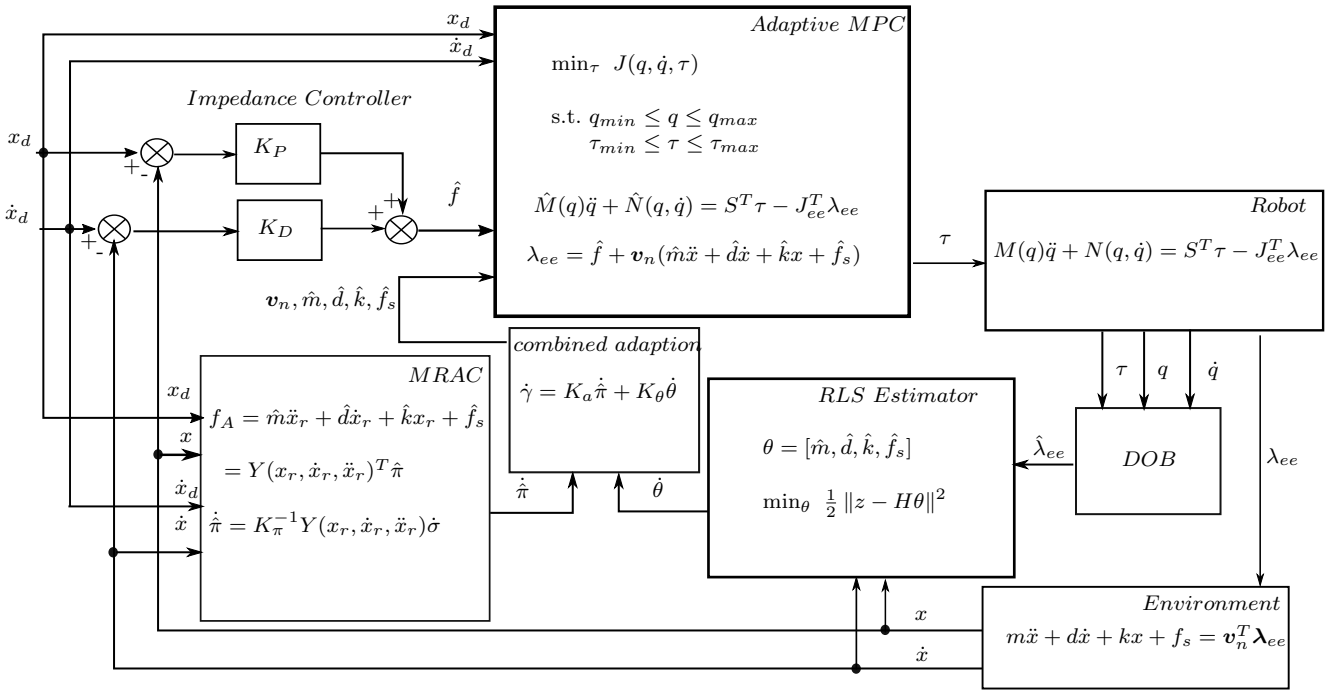


Figure 3.5: Combined MIAC and MRAC adaption scheme

3.8 Control Summary

In this chapter, we have presented the assumptions in section 3.2 based on which we design the controllers to complete the control task named in section 3.1. The trajectory generation algorithm is presented in section 3.3 which trajectories are the target to track under robot-environment interaction. The impedance controller from section 3.4 is part of all the presented controllers since it allows a compliant and robust interaction with the environment and thus creates innovation. This is used in model identification adaptive control (MIAC) presented in section 3.5, where the impedance controller is combined with the recursive least squares algorithm using online system identification to estimate the environment's dynamics' parameters along the task trajectory. These estimates are then used in the MPC controller for prediction and optimization.
In model reference adaptive control (MRAC) presented in section 3.6, the impedance controller is combined with an adaptive controller which adapts its parameters dependent on the reference tracking error. This controller converges to a zero tracking error over time. Finally, two methods are presented to combine the MIAC and MRAC control approaches in section 3.7 which promise either faster convergence or more robustness, while the strength of both controllers is used, the environment' dynamics' parameters estimation of MIAC and the tracking error minimization of MRAC.
In the next chapters, these controllers are applied to the door opening and object lifting tasks in simulation and on the hardware.

Chapter 4

Object Lifting Task

In this chapter we use the derived interaction control approaches from chapter 3 to lift up an object of unknown mass with the ballbot. In section 4.1 we describe how the linear trajectory is generated to move the end effector from a starting to an end point with velocity and acceleration limits. Then the performance of the presented controllers from chapter 3 are compared in simulation in section 4.2. Finally, the hardware test results are presented in section 4.3 and concluded in section 4.4.

4.1 Linear Task Space Trajectory

To move the end effector from an initial task space position $\boldsymbol{x_0}$ to a target position $\boldsymbol{x_1}$, starting with zero initial velocity, a task space trajectory can be parameterized with a control variable s(t) as,

$$\begin{aligned} \boldsymbol{x}(s) &= \boldsymbol{x}_0 + s(t)\frac{(\boldsymbol{x_1} - \boldsymbol{x_0})}{\|\boldsymbol{x_1} - \boldsymbol{x_0}\|} \\ \dot{\boldsymbol{x}}(s) &= \dot{s}(t)\frac{(\boldsymbol{x_1} - \boldsymbol{x_0})}{\|\boldsymbol{x_1} - \boldsymbol{x_0}\|}, \end{aligned} \tag{4.1}$$

for $s(t) \in [0, \|\boldsymbol{x_1} - \boldsymbol{x_0}\|]$. Given a maximal velocity $v_{max} = \dot{s}_{max}$ and maximal acceleration $a_{max} = \ddot{s}_{max}$, the parameter $s(t)$ can be computed with the trajectory generator described in section 3.3. The resulting position velocity time trajectory is then used as end effector reference for the control task.

4.2 Simulation Results

In this section our goal is to track the linear task space trajectory from section 4.1 as accurately as possible in the Gazebo simulation. The following plots and tables show the results of a simulation experiment where the ballbot should follow a trajectory in z-direction while an unknown mass is attached to the end effector which is unmodelled in the MPC controller. The task is to follow this trajectory as accurately as possible while estimating the object's mass. Afterwards, the robot should follow other trajectories in the Cartesian space as accurately as possible while holding the object and using the estimated mass in the MPC controller.
The table 4.1 compares the controller performance w.r.t. the reference tracking error during the lifting task and the accuracy of the estimated mass. The controllers that are compared are the impedance controller from section 3.4, the model identification adaptive controller that estimates parameters with recursive least squares and sends them to the MPC from section 3.5, the model reference adaptive controller which

adapts the parameters dependent on the tracking error described in section 3.6, and the combined MIAC MRAC controller where both methods are running together as stated in section 3.7.1. The true mass of the object that is lifted up is 2 kg. The impedance controller stiffness and damping are

$$\begin{aligned} K_P &= 40 * I_{3x3} \\ K_D &= 10 * I_{3x3}. \end{aligned} \tag{4.2}$$

The RLS estimator is initialized with initial mean and variance

$$\begin{aligned} m_0 &= 0.1 \\ \sigma_m^2 &= 10 \\ d_0 &= 0.0 \\ \sigma_d^2 &= 0.001 \\ k_0 &= 0.0 \\ \sigma_k^2 &= 0.001 \\ f_{s0} &= 1 \\ \sigma_f^2 &= 10 \end{aligned} \tag{4.3}$$

which means that we have the prior knowledge that we lift a mass (no damping and spring characteristics) but no prior knowledge about the object's mass (very small initial mass of 0.1 kg and corresponding static force). For the measurement noise variance we choose

$$R = 10. \tag{4.4}$$

For the PD impedance term of the MRAC controller, the same gains are used as in equation (4.2). The update-gain matrix of the adaptive term in the MRAC controller (3.39) is

$$K_\pi^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 25 \end{pmatrix} \tag{4.5}$$

and the adaptive parameters are initialized with

$$\begin{aligned} \hat{m} &= 0.1 \\ \hat{d} &= 0 \\ \hat{k} &= 0 \\ \hat{f}_s &= 10. \end{aligned} \tag{4.6}$$

To compare the results of the robot-environment interaction controllers, the experiment was also made with the MPC controller alone (MPC only) where no mass was attached. This lead to a root mean square tracking error (RMSE) of 0.014 which indicates a very good tracking. For the experiment with the attached mass, an RMSE of 0.224 was measured with the MPC only which needs to be improved by our interaction controllers.
The results in table 4.1 show that the impedance controller alone improves the tracking but is still much worse than the other controllers since it cannot adapt its gains or use the estimated parameters during the task. The RMSE is thus much larger than for the other controllers, whereas the errors of the other controllers, the

	MPC Only	Impedance Control	MIAC	MRAC	Combined MIAC MRAC
RMSE Tracking Error [m]	0.224	0.149	0.038	0.036	0.036
Estimated Mass [kg]	-	1.66	1.83	1.81	1.87

Table 4.1: The controller performance comparison in simulation for a mass of 2 kg

MIAC, MRAC and combined MIAC MRAC controller, are comparable and much smaller.
A similar distribution is visible for the estimated mass of the recursive least squares estimator. There, the combined MIAC MRAC controller has the closest estimate to the true mass.
The difference between the estimated mass and the true one is dependent on the estimated force from the DOB presented in section 3.5.2 and its rate of convergence and delay but also on the initial values of the environment's dynamics parameters in the RLS estimator. Since the RLS algorithm is minimizing the square error between the measured and predicted values over the whole time interval, it will underestimate the dynamics if the DOB and, therefore, the force along the lifting trajectory is only slowly converging. The convergence of the DOB is shown in figure 4.1 where it is visible that the lifting experiment starts at second 7 and that the convergence needs some time (ca. 7-8 seconds). The observer gain K_o from equation 3.17 is set to identity for our experiments. This gain introduces a substantial delay, but it also reduces noise and high force peaks during the transient that could destabilize the RLS estimator.

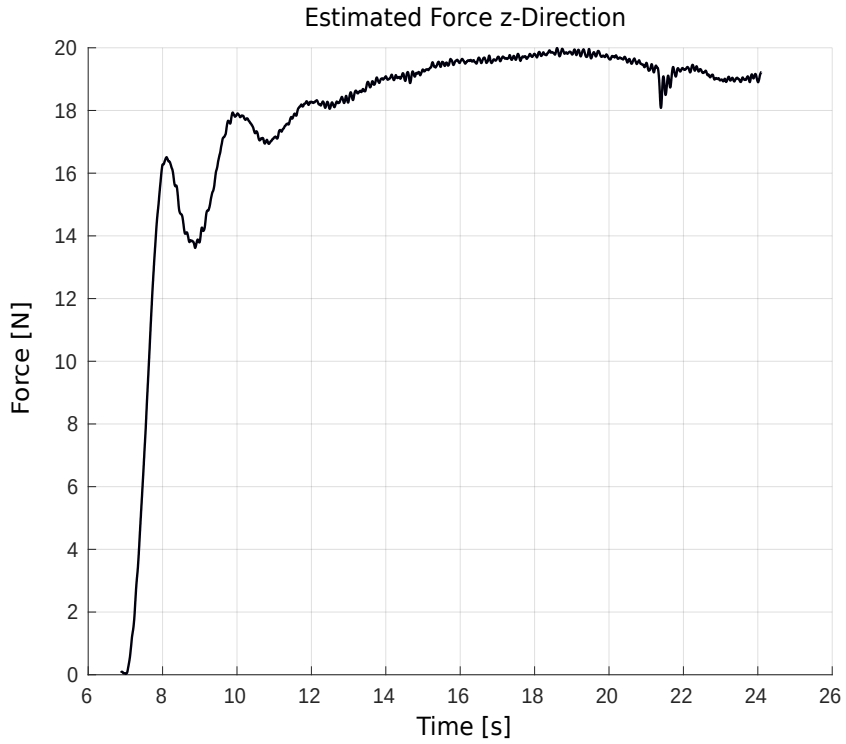


Figure 4.1: The estimated interaction force in z-direction for a mass of 2 kg

The other mentioned influence on the estimated mass comes from the initial values of the RLS estimator. Figure 4.2 shows the convergence of the static force and the mass that is estimated dependent on different initial states of the estimator. For this experiment the MRAC controller is lifting a mass of 2 kg while the RLS estimator is running with initial mass parameters m_0 of 0.1, 0.5, 1 and 2 kg and initial static force parameters f_{s0} of 1, 5, 10, and 20 N. The damping and the spring stiffness are set to zero with a very small variance and will thus remain zero. The lifting starts at second 7. We can see that prior knowledge can improve the rate of convergence and that the estimated parameters are closer to the correct ones. But since the force estimator has a very slow convergence rate due to its low pass characteristics, initializing the estimator with the correct value (2 kg) also leads to a long transient phase.

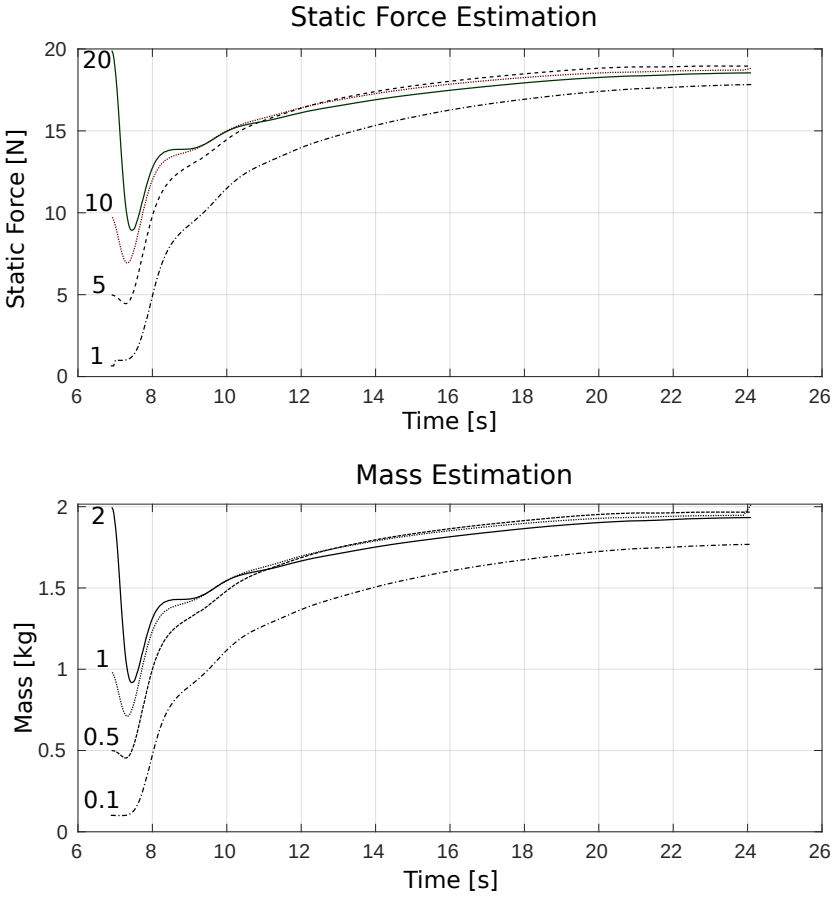


Figure 4.2: Estimated mass and static force convergence for different initial values

In figure 4.3 we show the reference tracking performance of the MRAC controller compared to the pure MPC controller without any feedback about the environment. A mass of 2 kg is pushing the end effector down at second 7 and compared to the pure MPC controller, the MRAC controller makes it possible to follow the desired lifting trajectory (from second 7 to second 24). After second 24, the estimated mass is used by the MPC and an accurate tracking of the reference position in x-, y- and z-direction is possible which is much better than using the MPC alone.
Figure 4.4 shows the commanded force of the MRAC controller that is set in the MPC for this experiment where we see that the impedance controller term $\boldsymbol{v}_n^T f_Z$ converges to zero and the adaptive controller term f_A from equation 3.40 gets static as the tracking error is converging which is also visible in figure 4.3 in the z-position plot after second 16.
As shown in equation (3.1), a PID-controller in the drives regulates the actuator position and velocity to the MPC optimal position and velocity in addition to the feedforward torque of the MPC. Figure 4.5 shows the contribution of the PID-torque in the shoulder flexion extension (SH FLE) and the elbow flexion extension (EL FLE) joint. We can see that when the MRAC controller is in charge, the contribution of the PID-torque is converging close to zero as the tracking error is converging after second 16. On the other hand, when only the MPC controller is in charge, the PID-torque contribution is around 2 Nm for the SH FLE and 4 Nm for

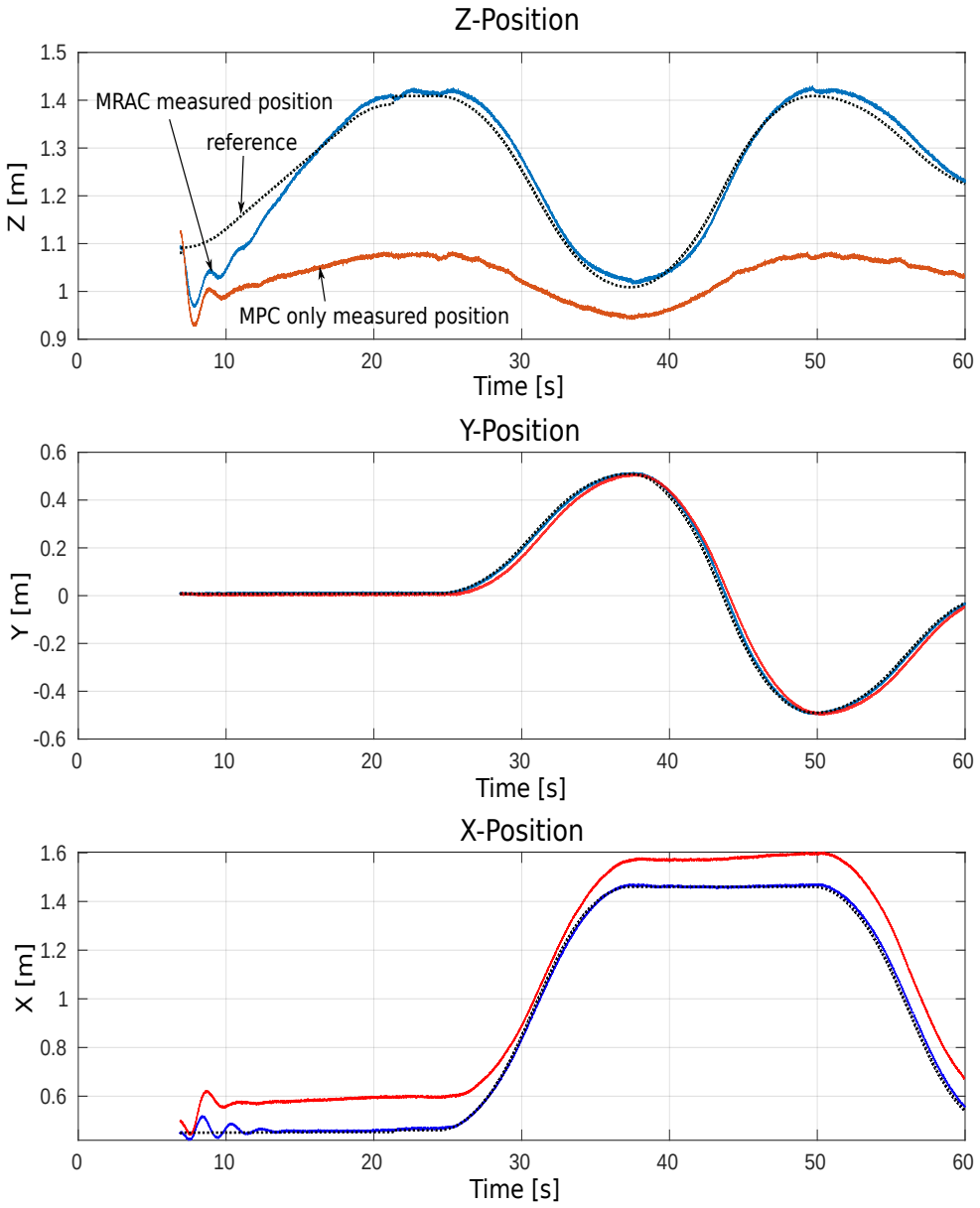


Figure 4.3: The reference tracking during the lifting of the mass (sec 7-24) and with its estimate (sec 24-60) compared between the MRAC controller and the MPC only

the EL FLE actuator on average where the behavior of the controller is very noisy. This additional PID-torque does not make it possible to converge to the reference position, but it prevents that the arm is falling down even more due to the unknown mass.

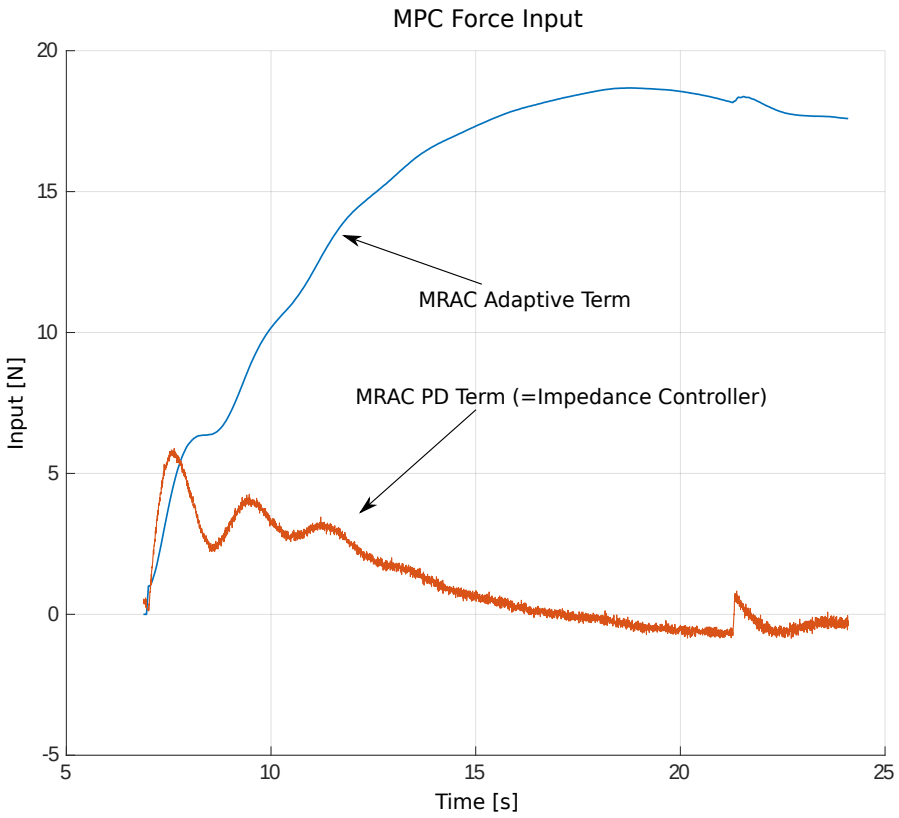


Figure 4.4: The force command of the MRAC sent to the MPC controller during the lifting of the object in simulation

4.3 Hardware Results

On the hardware, we want to track a trajectory in z-direction while the ballbot lifts a unknown mass at the end effector as done in the simulation. During the tests we observed oscillations due to the impedance controller. Therefore, we set the gains of the stiffness and damping matrix of the impedance controller to zero,

$$\begin{aligned} K_P &= 0_{3x3} \\ K_D &= 0_{3x3}. \end{aligned} \tag{4.7}$$

All other gains and parameters of the adaptive controller and the RLS estimator have the same values as described in the simulation section 4.2. As done in the simulation, we lift up a mass of 2 kg along a 30 cm trajectory with all controllers. The tracking error results and the estimated mass are shown in table 4.2. The MRAC controller and combined MIAC MRAC controller show the best trajectory tracking performance while the estimated mass of the MIAC and the combined MIAC MRAC controller is the closest to the real mass of 2 kg. The tracking errors of the lifting in the hardware tests are much smaller than in simulation which is due to the fact that the weight is continuously increased in the hardware tests since the weight is lifted from a table, whereas in simulation the weight of the end effector is changed through a Gezebo plugin with a step function causing the larger error. But

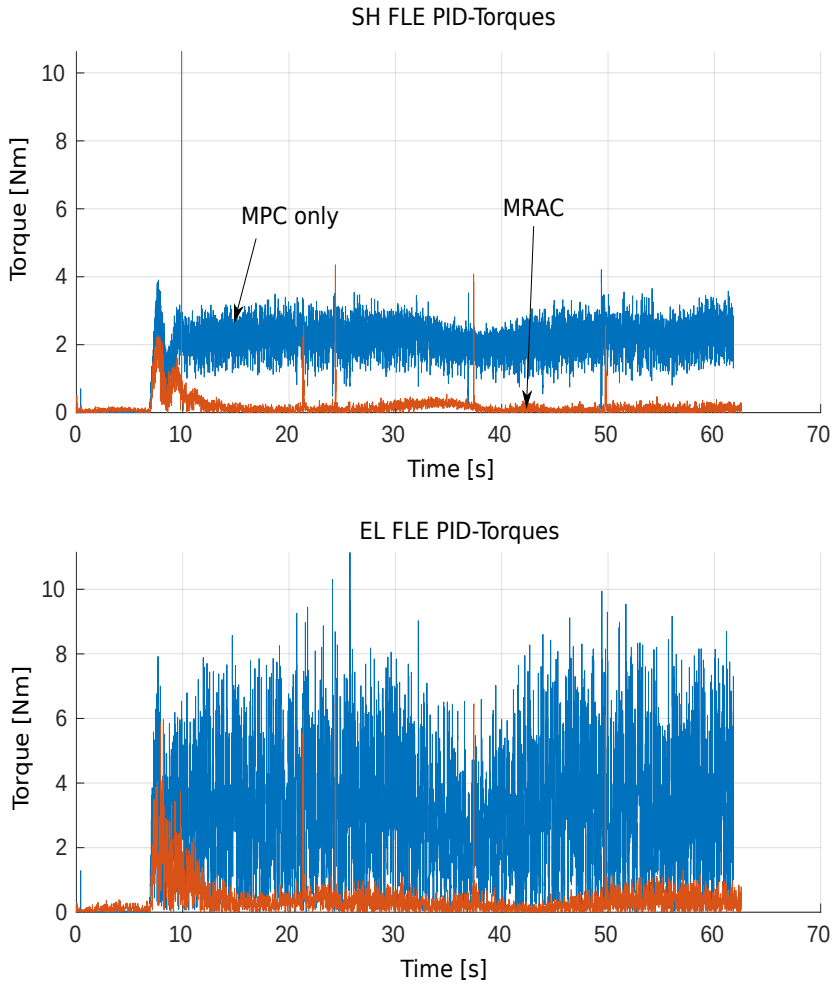


Figure 4.5: The contribution of the PID-torques in the actuators compared between the MRAC and MPC only methods in simulation

also since the PID-controller in the actuator does contribute more on the hardware than in simulation since it runs at higher rates.
We also observe that the mass of the object is underestimated even more on the hardware than in simulation. This is caused by the commanded actuator torque which is used in the DOB equation in section (3.17) instead of the measured actuator torque due to the reasons mentioned in section 3.5.2. The influence of the commanded actuator torque and the PID-controller in the drives will be discussed later in more detail.
Table 4.3 shows the tracking error and the estimated mass for the MRAC controller for weights of 1, 2 and 2.5 kg. We see that the tracking error increases almost in a linear way as the weight increases. Also the estimated mass increases in the same range but it is still far off from the true mass.
Since the MRAC controller has the best performance regarding the tracking error, we want to show the tracking w.r.t. the desired x-, y-, and z-position for this task. In figure 4.6 we see the tracking performance of the MRAC controller compared to the MPC only where no interaction controller is used. The lifting of the object

	MPC Only	Impedance Control	MIAC	MRAC	Combined MIAC MRAC
RMSE Tracking Error [m]	0.075	0.068	0.045	0.024	0.025
Estimated Mass [kg]	0.32	0.57	1.66	1.11	1.51

Table 4.2: The controller performance comparison on the hardware for a 2 kg object

	1 kg	2 kg	2.5 kg
RMSE Tracking Error [m]	0.012	0.024	0.027
Estimated Mass [kg]	0.68	1.11	1.29

Table 4.3: The MRAC controller performance for an object of 1, 2, and 2.5 kg

of 2 kg is finished at second 12. From then on, the estimated mass is set in the MPC system dynamics and the MPC controller should follow additional reference trajectories in the Cartesian space. The z-position plot in figure 4.6 shows that the tracking with the MRAC controller is much better for the lifting of the unknown mass. However, compared to the tracking in the simulation case, the reference tracking after the lifting of the MRAC controller is much worse while the one of the MPC only is much better. In the simulation, the MPC controller running alone without an interaction controller is not able to lift the mass, but on the hardware this works.

The reason why this works on the hardware is shown by figure 4.7 where the absolute values of the PID-torques in the drives from equation 3.1 are shown. These values have to be considered with caution since the measured actuator torque is used to compute them and since the measured torque has drift and a wrong torque constant as mentioned in section 3.5.2. Comparing the PID-torque contribution of the MRAC controller and the MPC only in the shoulder and elbow flexion extension joints (SH FLE, EL FLE), we see that a lot of work is done by the PID-controller if the MPC only is used. Therefore, we have less force control but more position control in the drives. In figure 4.5 we can also see that more work is done by the PID-controller when the MPC only is used. However, in the simulation experiment the mass could not be lifted up compared to the hardware tests. The reason for this could be the high frequency of 1500 Hz with which the PID-controller is running on the hardware which is not the case in the simulation where it runs at around 400 Hz. This high frequency on the hardware increases the influence of the PID-controller, especially of the integral term, which makes it possible to compensate the model inaccuracies in the feedforward torque that come from the MPC controller.

The bigger contribution of the actuator PID-controller on the hardware is also visible for the MRAC controller in figure 4.7 compared to the simulation in figure 4.5. This explains why the mass estimates are worse on the hardware than in simulation. When we compute the commanded actuator torque manually as described in section 3.5.2, we neglect the integral term because of the frequency differences and therefore do not cover its high contribution when we estimate the forces at the end effector.

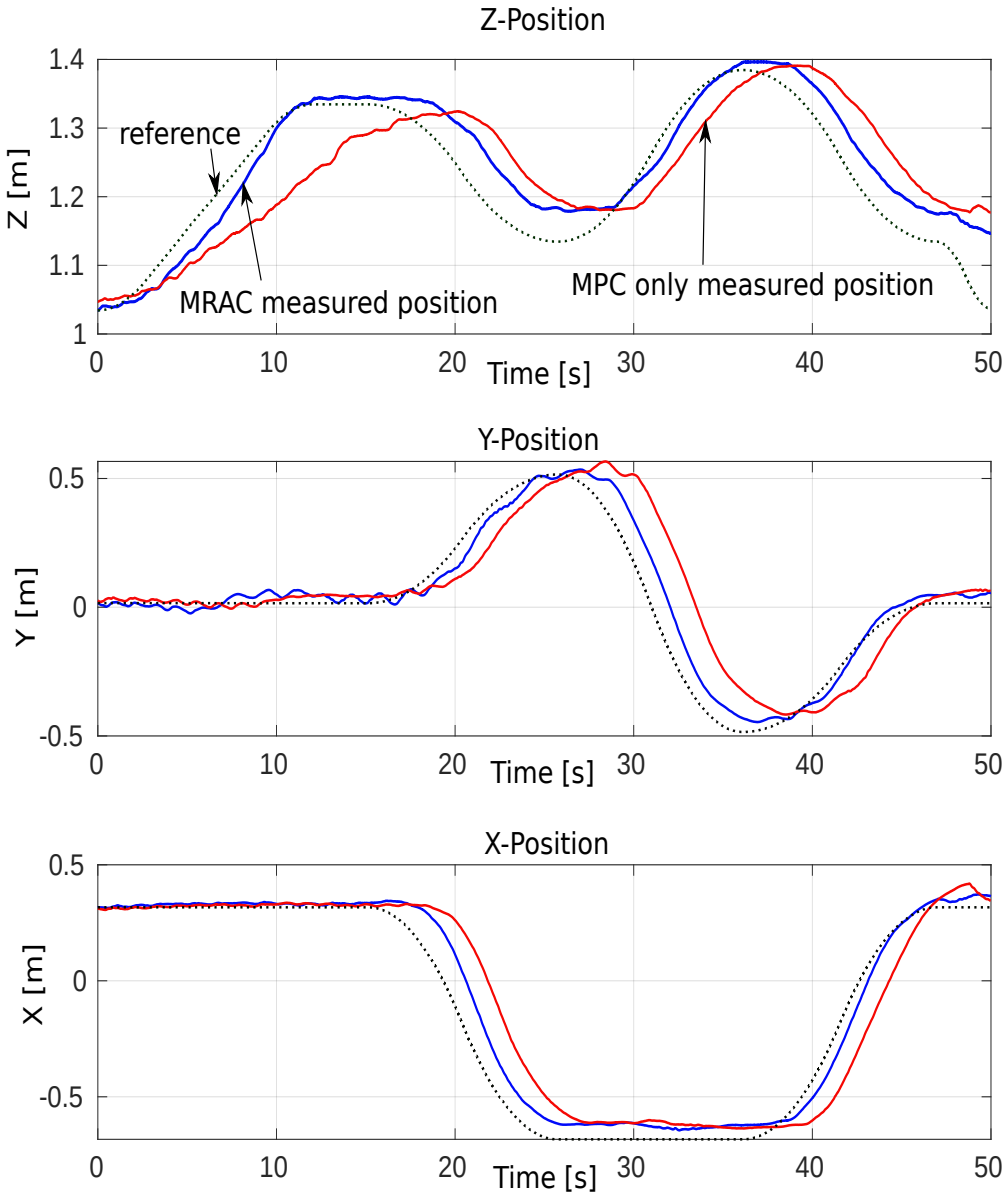


Figure 4.6: The reference tracking during the lifting of the object (sec 0-12) and with its estimated mass (sec 12-50) compared between the MRAC controller and the MPC only

This is visible for the estimation of the force in z-direction in figure 4.8 when the MRAC controller is used. In this figure we see that the estimated force does not reach the correct interaction force of around 20 N in z-direction which corresponds to the mass of 2 kg attached to the end effector, but only estimates around 13 N with a slow rate of convergence. Compared to figure 4.1 where the estimated force in simulation is shown, we see how much more we underestimate the interaction force

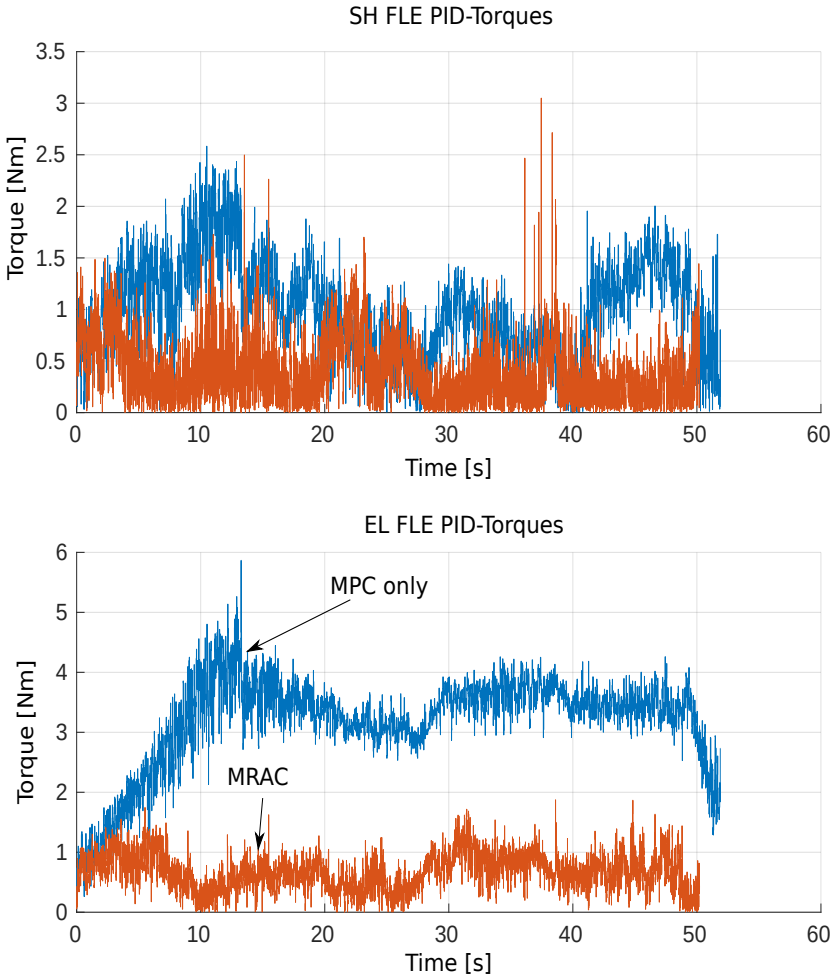


Figure 4.7: The contribution of the PID-torques in the actuators compared between the MRAC and MPC only methods

on the hardware than in simulation. Figure 4.9 shows the force input of the MRAC controller that is sent to the MPC which is also much smaller than in the simulation shown in figure 4.4 because of the higher contribution of the PID-controller in the drives. Since the impedance controller stiffness and damping matrix is zero, only the adaptive term of the MRAC is contributing. In the next section, we want to summarize and conclude our results.

4.4 Object Lifting Task Conclusion

In this chapter our goal was to follow a linear task space trajectory as presented in section 4.1 while an unknown mass is lifted. This mass should be estimated during the lift and used afterwards in the MPC controller to track additional trajectories. We observe a difference between the simulation and the hardware tests w.r.t. the contribution of the PID-controller in the actuators which originates from the much higher sampling frequency on the hardware than in simulation and the influence of

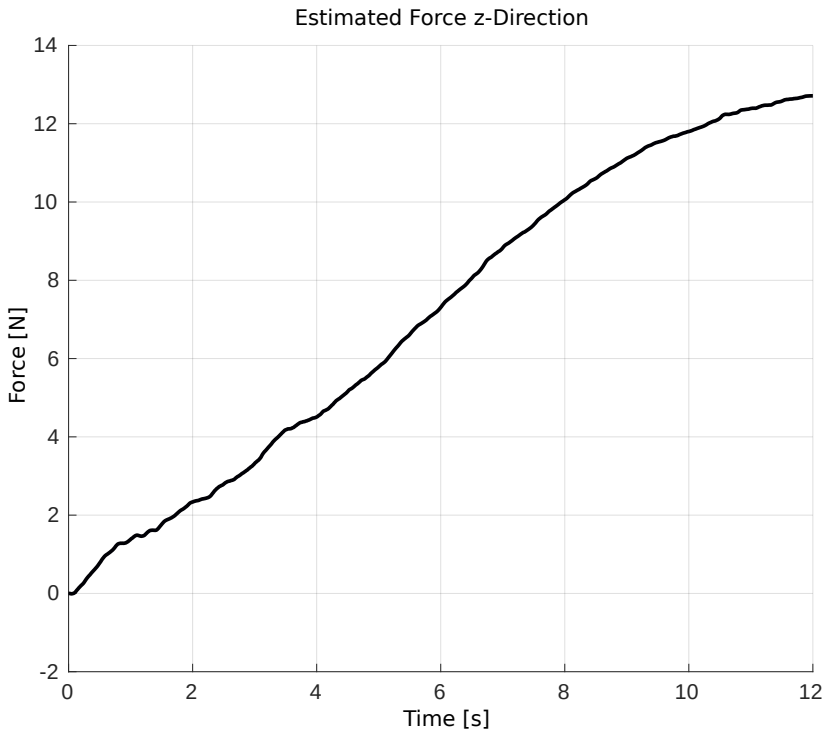


Figure 4.8: The estimated interaction force in z-direction for a mass of 2 kg

the integral term. Regarding the tracking performance of the lifting, the MRAC and the combined MIAC MRAC controller have the best performance of all control approaches. All interaction controllers presented in chapter 3 performed better than the MPC alone. Nevertheless, the bad end effector force estimation on the hardware diminishes the comparability between the control methods since the MIAC controller depends on an accurate estimate. This is also visible in the overall worse mass estimation on the hardware than in simulation. In summary, we can conclude that the MRAC controller is favourable for the lifting if no or only inaccurate end effector force estimation is possible. An improvement of the force estimation or the use of a force sensor would improve the mass estimate and therefore also the tracking after the lifting. This could also further improve the behavior of the MIAC and the combined MIAC MRAC controllers and is therefore a promising attempt for further research.

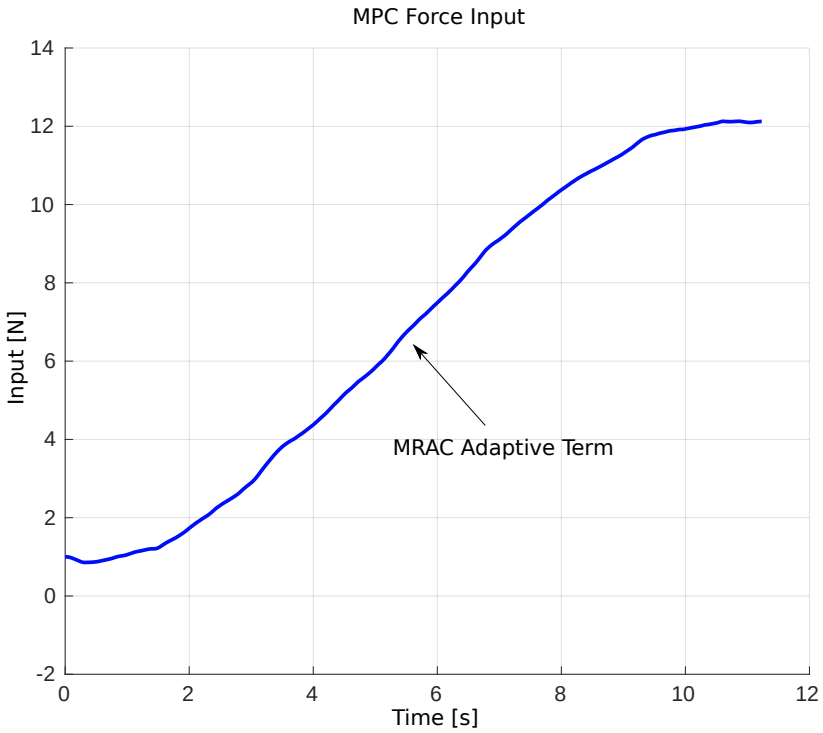


Figure 4.9: The force command of the MRAC sent to the MPC controller during the lifting of the 2 kg mass

Chapter 5

Door Opening Task

In this chapter we use the derived control approaches from chapter 3 to open a door with the ballbot. First, we describe in section 5.1 how we use the trajectory generator from section 3.3 to generate the circular task trajectory which is updated online based on an extended Kalman filter that tracks the door state. This online circle estimator is presented in section 5.2. In section 5.3 we state the assumed door model w.r.t. the door angle coordinates and how the dynamics of the door model relate to the estimated dynamics along the task trajectory. In section 5.5 we compare the different controllers in simulation, and in section 5.6 we present the hardware test results. The results are concluded in section 5.7.

5.1 Door Opening Trajectory

To generate the circular door opening trajectory we assume that a good estimation of the normal vector of the door surface, the door handle position and a rough estimate of the door width can be acquired via a vision system. Based on these the homogeneous transform H_{ID} from the handle frame (D) to the world frame (I) can be computed as well as the homogeneous transform H_{IH} from the door hinge frame (H) to the world frame. From these the position vector from hinge to handle in the hinge frame $\boldsymbol{p}_D^H$ can be computed. The rotation matrix of the handle around the hinge frame is simply a rotation around the z-axis,

$$R_z(\phi) = \begin{pmatrix} cos(\phi) & -sin(\phi) & 0 \\ sin(\phi) & cos(\phi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.1}$$

where ϕ is the door angle. The position of the handle at a certain angle is therefore

$$\boldsymbol{p}_D^H(\phi) = R_z(\phi)\boldsymbol{p}_D^H. \tag{5.2}$$

The velocity of the handle at a certain angle is

$$\dot{\boldsymbol{p}}_D^H(\phi, \dot{\phi}) = J_{HD}^H \dot{\phi}, \tag{5.3}$$

with

$$J_{HD}^H = \begin{pmatrix} -sin(\phi) & -cos(\phi) & 0 \\ cos(\phi) & -sin(\phi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \boldsymbol{p}_D^H. \tag{5.4}$$

To use the time optimal trajectory generation algorithm from section 3.3, a desired door opening angle ϕ_{des} can be chosen as well as a maximal velocity of the door

angle $\dot{\phi}_{max}$ and a maximal acceleration $\ddot{\phi}_{max}$. This will generate a triangular or trapezoidal velocity profile trajectory $(\phi(t), \dot{\phi}(t), t)$ w.r.t the door angle coordinates. The reference task space position trajectory $\boldsymbol{x}_d(t)$ can then be computed with

$$\boldsymbol{x}_d(t) = \boldsymbol{p}_H^I + R_{IH}\boldsymbol{p}_D^H(\phi(t)) \tag{5.5}$$

with hinge position in world frame p_H^I and the task space velocity trajectory $\dot{\boldsymbol{x}}_d(t)$ as

$$\dot{\boldsymbol{x}}_d(t) = R_{IH}\dot{\boldsymbol{p}}_D^H(\phi(t), \dot{\phi}(t)). \tag{5.6}$$

The reference orientation $R_{ID}(t)$ of the end effector can be written as

$$R_{ID}(t) = R_{IH}R_z(\phi(t)) \tag{5.7}$$

where R_{IH} can be extracted from the homogeneous transform H_{IH}.

5.1.1 Stiffness and Damping Matrix Trajectory

Since our main reason for using force control is to prevent high contact forces during the interaction with the environment, we need to design the stiffness matrix K_p and the damping matrix K_d of the impedance controller presented in section 3.4 in that way. The design of the matrices is intuitive in the frame of the handle D (door frame) since we want to achieve very high compliance in radial direction of the door and lower compliance in the direction of the door opening trajectory. The stiffness matrix should thus have a higher gain in tangential and a lower gain in radial direction of the door opening path. Therefore, the stiffness and damping matrices can be designed as diagonal matrices in the door frame K_p^D, K_d^D, but then they need to be rotated to the world frame dependent on the angle $\phi(t)$, similar to as it was done by Lee et al. [10] with

$$\begin{aligned} K_p(t) &= R_{ID}(t)K_p^D R_{ID}(t)^T \\ K_d(t) &= R_{ID}(t)K_d^D R_{ID}(t)^T \end{aligned} \tag{5.8}$$

where $R_{ID}(t)$ is given by equation (5.7).
Since the desired trajectories $(\boldsymbol{x}_d(t), \dot{\boldsymbol{x}}_d(t), t)$ are based on prior estimates of the door width, the normal vector of the door and the handle position under uncertainties, we use an extended Kalman filter presented in the next section to observe the state of the door and to recompute the trajectories to prevent high interaction forces due to kinematic constraints.

5.2 Online Circle Estimation

To generate the door opening trajectory as presented in the last section, the position of the door hinge and the initial position of the handle have to be known or acquired via a vision system. Since such a system can be imprecise, we want to estimate the door hinge position, radius and door angle online. In the work of Bellicoso et al. [11], the Taubin method is used which is a fitting method to fit a circle to a data set that is able to achieve good results for data points coming from a small angle only. In this work we use an extended Kalman filter (EKF) to estimate the circle since this allows us to include prior knowledge about the initial hinge and handle position and the normal vector of the door surface.
We use the following coordinate system for the EKF. The state of the EKF x is the circle radius r, the current angle ϕ, and the circle center in x-direction h_x and

y-direction h_y. The process model of the filter uses the end effector velocities v_x and v_y

$$\begin{aligned} v_x &= rcos(\phi)\dot{\phi} \\ v_y &= rsin(\phi)\dot{\phi}. \end{aligned} \tag{5.9}$$

The process model for the radius r and the circle center h_x and h_y are trivial since they should not change. For the angle the process model is

$$\dot{\phi} = \tilde{s}\sqrt{\frac{v_x^2 + v_y^2}{r^2}} \tag{5.10}$$

where $\tilde{s} \in \{-1, 1\}$ is determined depending on the halfplane we are located on and on the choice if we close or open the door.
If we discretize with sampling time T_s, the discrete process model with process state x^p is received

$$\begin{aligned} x_k^p &= q_{k-1}(x_{k-1}^m, E(v_{k-1})) \\ \phi[k] &= \phi[k-1] + \tilde{s}\sqrt{\frac{v_x^2 + v_y^2}{r[k-1]^2}}T_s + \bar{v}[k-1], \end{aligned} \tag{5.11}$$

where $v[k]$ is the process noise and $\bar{v}$ is its mean value.
Using the nonlinear process model (5.11), the derivative w.r.t the state around the current measurement state x^m, the expectation of the process noise $\bar{v}$ which is assumed to be zero and the process noise variance Q is used to compute the process state variance P_k^p.

$$\begin{aligned} A_{k-1} &= \frac{\partial q_{k-1}}{\partial x_{k-1}}(x_{k-1}^m, \bar{v}_{k-1}) \\ L_{k-1} &= \frac{\partial q_{k-1}}{\partial v_{k-1}}(x_{k-1}^m, \bar{v}_{k-1}) \\ P_k^p &= A_{k-1}P_{k-1}^m A_{k-1}^T + L_{k-1}Q_{k-1}L_{k-1}^T \end{aligned} \tag{5.12}$$

The nonlinear measurement model $z_k = h_k(x_k^p, w_k)$ of the EKF with end effector position p_x and p_y as measurement $z_k = \begin{bmatrix} p_x & p_y \end{bmatrix}^T$ and measurement noise $w = \begin{bmatrix} w_1 & w_2 \end{bmatrix}^T$ is

$$\begin{aligned} p_x &= r[k]sin(\phi[k]) + h_x[k] + w_1[k] \\ p_y &= -r[k]cos(\phi[k]) + h_y[k] + w_2[k]. \end{aligned} \tag{5.13}$$

To compute the update gain matrix K_k, the measurement update state x_k^m and the measurement variance P_k^m, the measurement model is linearized around the current process state and measurement noise with expectation $\bar{w}_k = 0$ and variance Q_k with

$$H_k = \frac{\partial h_k}{\partial x_k}(x_k^p, \bar{w}_k) \tag{5.14}$$

$$M_k = \frac{\partial h_k}{\partial w_k}(x_k^p, \bar{w}_k). \tag{5.15}$$

Given a new measurement z_k, the measurement update is computed with

$$\begin{aligned} K_k &= P_k^p H_k^T [H_k P_{k-1}^p H_k^T + M_k R_k M_k^T]^{-1} \\ x_k^m &= x_k^p + K_k[z_k - h_k(x_k^p, \bar{w}_k)] \\ P_k^m &= [I - K_k H_k] P_k^p. \end{aligned} \tag{5.16}$$

The estimated door angle, radius and hinge position allows us to recompute the trajectory from section 5.1 at a desired rate.

5.3 Door Model

In this section we describe the linear door model that we assume to encounter during the door opening task and explain what this model implies on the estimated parameters. For a door of mass m_d, with width w and thickness t, the linear dynamic equations of the door with door angle ϕ can be written as

$$M\ddot{\phi} + D\dot{\phi} + K(\phi - \phi_0) + \tau_{static} = \tau_{ext}. \tag{5.17}$$

The door inertia M $[\frac{Nms^2}{rad}]$ is

$$\begin{aligned} M &= m_d(\frac{w}{2})^2 + I_{zz} \\ I_{zz} &= \frac{1}{12} m_d(w^2 + t^2) \end{aligned} \tag{5.18}$$

and the door has potentially some damping D $[\frac{Nms}{rad}]$, a spring K $[\frac{Nm}{rad}]$ and static friction τ_{static} $[Nm]$. If we assume that the force is applied by the robot only in door surface normal direction at a distance r from the hinge, the environment model along the trajectory from section 3.2.1 leads to the following equation in the hinge frame H

$$\begin{aligned} (J_{HD}^H)^T \lambda_{ee}^H &= ru = \tau_{ext} \\ r(m\ddot{x} + d\dot{x} + k(x - x_0) + f_{static}) &= M\ddot{\phi} + D\dot{\phi} + K(\phi - \phi_0) + \tau_{static}. \end{aligned} \tag{5.19}$$

With $\ddot{x} = r\ddot{\phi}$ and $\dot{x} = r\dot{\phi}$ the following relations are obtained,

$$\begin{aligned} m &= \frac{M}{r^2} \\ d &= \frac{D}{r^2} \\ f_{static} &= \frac{\tau_{static}}{r}. \end{aligned} \tag{5.20}$$

These relations imply that the estimation of the mass, the damping and the static force along the trajectory shown in section 3.2.1 should lead to by r or r^2 scaled values of the door model parameters. More importantly, it also shows that the parameters' relation are constant along the trajectory as assumed in section 3.2 which makes the estimated model also correct in this circular motion.
Only for the spring stiffness, the linear stiffness value k will not have a constant

relation to circular stiffness value K since $x - x_0 = \boldsymbol{v}^T(\boldsymbol{x} - \boldsymbol{x}_0) = rsin(\phi)$ and for $\phi_0 = 0$ therefore

$$k = \frac{1}{r^2}\frac{K\phi}{sin(\phi)}. \tag{5.21}$$

Since the estimated parameter is now also dependent on the current angle $\phi(t)$, our assumption is violated and the estimated stiffness is not a constant scale of the true one.
In equation (5.21) the projection of the Cartesian position error along the trajectory vector $\boldsymbol{v}$ is used. If, on the other hand, the L2 norm is used, the relation between k and K for $\phi_0 = 0$ would be

$$\begin{aligned} x - x_0 &= ||\boldsymbol{x} - \boldsymbol{x}_0|| \\ x - x_0 &= r\sqrt{2(1 - cos(\phi))} \\ k &= \frac{1}{r^2}\frac{K\phi}{\sqrt{2(1 - cos(\phi))}}. \end{aligned} \tag{5.22}$$

This formulation would be a better choice for a circular trajectory as long as $\phi < \frac{\pi}{2}$. The relations (5.21) and (5.22) are visualized in figure 5.1. This figure shows the arc lengths for $r = 1$ where 1) is the correct arc length $l = r\phi$, 2) is the L2 norm approximation from equation (5.22) where $l = r\sqrt{2(1 - cos(\phi))}$ and 3) is the projected approximation from equation (5.21) where $l = rsin(\phi)$.

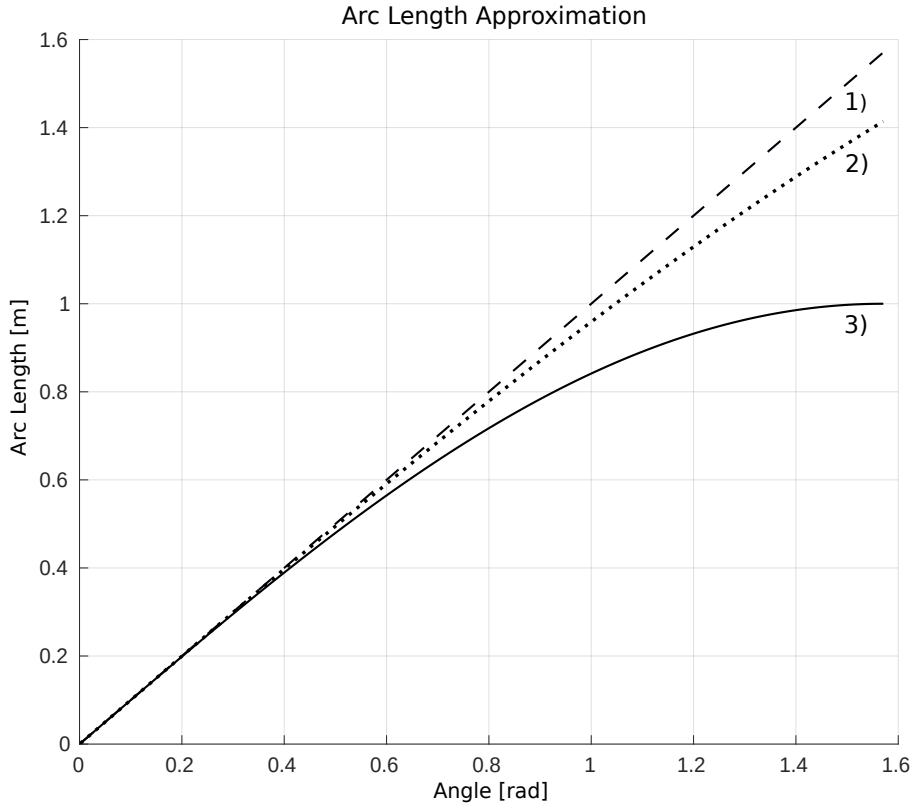


Figure 5.1: Arc length approximation for radius $r = 1$

Because we want to be consistent regarding the choice of our model, which is a more general formulation, we use the projection approach that results in equation (5.21), and we are therefore aware that this should introduce an error in the dynamic parameter estimates for a circular trajectory regarding the stiffness parameter.

5.4 MPC Cost for the Door Opening

In this section we describe a main challenge of using force control on the ballbot for the door opening task and how we try to solve it.
When the ballbot should apply a high force in the transverse plane, the optimizer will re-orientate the ball and base position so that the robot can shift its whole body center of mass towards the direction of the applied force. This is a natural and optimal behavior to apply high forces. In case of the door opening, the main problem with this is that the base or the ball could crash with the door frame. Since the distance between the end effector and the ball will increase if higher forces need to be applied and because the force is applied on a circular trajectory, it is difficult to avoid such a collision.
One solution would be to introduce a constraint on the base position in the optimal control formulation. This approach did not yield good simulation results because it influences all the cost parameters which are not trivial to tune.
Another approach that worked better in simulation was to keep the desired end effector yaw straight, up to a certain angle, where we are safe to turn the ballbot so that from then on forces can be applied in an optimal way. This method introduces high torques around the yaw and roll direction when the yaw is held straight. To avoid that the ballbot tilts too much and looses stability, the cost on the base roll derivative term is increased from 10 to 100 which changes the MPC state cost from equation (2.4) to

$$Trace(Q_{ss\ v}) = \begin{bmatrix} 20 & 20 & 20 & 10 & 100 & 20 & 3 & 10 \end{bmatrix}. \tag{5.23}$$

This approach works well in simulation, but it also reaches its limit when the door demands very high forces to be applied. In the next section, we present the simulation results of the derived controllers on the door opening task.

5.5 Simulation Results

To simulate a door in Gazebo and compare the controllers derived in chapter 3, a door model is created in Gazebo with inertia M as shown in equation (5.18). The damping D and static friction τ_s of the door are set with the joint dynamics tag in the URDF file of the door model. To simulate a spring behavior, the door hinge joint is proportionally controlled with a stiffness gain K. To compare the performance of the environment interaction controllers, two different door models are selected. The goal is to follow the desired door angle trajectory as accurately as possible but also to open to door precisely to the desired final angle. We also want to see if the controllers that use the estimated environment dynamics at the end effector perform better in a second attempt when they are initialized with already estimated parameters.
The diagonal stiffness and damping matrix gains of the impedance controller from section 3.4 are

$$\begin{aligned} Trace(K_p^D) &= \begin{bmatrix} 40 & 10 & 40 \end{bmatrix} \\ Trace(K_d^D) &= \begin{bmatrix} 10 & 3 & 10 \end{bmatrix}, \end{aligned} \tag{5.24}$$

since for the door opening the matrix K_p^D and K_d^D are defined in the door frame as described in section 5.1.1 and we want high compliance in the radial direction of the door. The RLS estimator to estimate the environment's dynamics' parameters is initialized with initial mean and variance

$$\begin{aligned} m_0 &= 0.1 \\ \sigma_m^2 &= 0.1 \\ d_0 &= 0.1 \\ \sigma_d^2 &= 0.1 \\ k_0 &= 0.1 \\ \sigma_k^2 &= 0.1 \\ f_{s0} &= 1 \\ \sigma_f^2 &= 0.1 \end{aligned} \tag{5.25}$$

and the measurement noise variance is set to $R = 0.1$. This initialization means that we have no prior knowledge about any parameter, but that we assume that all of them can occur in the dynamics of the environment. The adaptive controller has the same parameters as used for the lifting task in equation (4.5) and (4.6).

In the following we present the simulation results conducted on two different doors which we refer to as the light and heavy door due to their different dynamics' characteristics.

Light Door

The door model used for the first experiment has the parameters $M = 9.6\frac{Nms^2}{rad}$, $D = 3\frac{Nms}{rad}$, $K = 0\frac{Nm}{rad}$ and $\tau_s = 5Nm$ and the goal is to open the door by 70 degrees.
The results of the simulation experiment for the light door are visualized in table 5.1. The root mean square error (RMSE) of the tracking of the angle trajectory

	MPC Only	Impedance Control	MIAC	MRAC	Combined MIAC MRAC
RMSE Tracking Error [deg]	12.79	4.36	1.31	1.81	2.10
Final Angle Error [deg]	-12.15	-4.07	0.58	1.55	1.36
Estimated Mass [kg]	-	-	0.39	0.39	0.40
Estimated Damping [Ns/m]	-	-	3.65	3.87	3.63
Estimated Spring Stiffness [N/m]	-	-	0.66	0.63	0.70
Estimated Static Force [N]	-	-	4.40	4.41	4.50

Table 5.1: Controller performance comparison for the light door

shows an improvement by using the impedance controller from section 3.4 over the MPC only method where no interaction controller is used. Also the final angle error becomes much smaller. The MIAC controller from section 3.5, the MRAC controller from section 3.6 and the combined MIAC MRAC controller presented in section 3.7.1 provide a major improvement compared to the impedance controller. The final angle error is below 2 degrees for all three controllers. In this experiment the MIAC controller performed best.
The estimated dynamic parameters of the door (mass, damping, spring stiffness and static force) are consistent between the controllers. Since the door radius r is in this case 1 m, the relations from section 5.3 would imply that the mass, damping and static force can be compared without any scaling to the simulation parameters of the door. We can observe that the damping, spring stiffness and static force are close to the correct one. Only the mass is highly underestimated.
To reinforce our motivation that MIAC and the combined MIAC MRAC controller can use the estimated parameters to perform better in a second attempt, the simulation is run again while the RLS estimator is initialized with the values from the first run. The results are visualized in table 5.2. We see that the RMSE tracking error is smaller than in the first attempt, but the final angle error is slightly larger in both cases. While the mass, spring stiffness and static force converged again close to the estimates from the first run, the damping increased. This will be discussed after the results of the heavier door experiment in the next section.

Heavy Door

The door model used for the second experiment has the parameters $M = 24\frac{Nms^2}{rad}$, $D = 6\frac{Nms}{rad}$, $K = 3\frac{Nm}{rad}$ and $\tau_s = 15Nm$ and the goal is to open the door by 70 degrees. The handle is again at a 1 m distance from the hinge. All controllers use the same parameters as for the light door in section 5.5.
The results of the second experiment are shown in table 5.3. Since the door is heavier, it demands a larger amount of force. Therefore, the MPC alone performs poorly, but also the impedance controller does not yield the desired results without reshaping the stiffness and damping matrices. On the other hand the MIAC, the

	MIAC	Combined MIAC MRAC
RMSE Tracking Error [deg]	0.68	2.90
Final Angle Error [deg]	0.71	1.39
Estimated Mass [kg]	0.43	0.43
Estimated Damping [Ns/m]	4.41	4.26
Estimated Spring Stiffness [N/m]	0.60	0.68
Estimated Static Force [N]	4.73	4.76

Table 5.2: Controller performance comparison for the light door where controllers are initialized with already estimated parameters

	MPC Only	Impedance Control	MIAC	MRAC	Combined MIAC MRAC
RMSE Tracking Error [deg]	33.63	12.71	3.24	2.22	2.87
Final Angle Error [deg]	-47.37	-15.47	0.83	2.29	1.82
Estimated Mass [kg]	-	-	1.45	1.41	1.50
Estimated Damping [Ns/m]	-	-	9.90	8.21	9.83
Estimated Spring Stiffness [N/m]	-	-	2.47	2.40	2.55
Estimated Static Force [N]	-	-	13.93	13.91	14.54

Table 5.3: Controller performance comparison for the heavy door

MRAC and the combined MIAC MRAC controllers again performed very well with a small angle tracking error. They also ended with a reasonable small angle error at the end of the task.
The estimated parameters of the spring stiffness and static force are close to the real values which was also the case for the lighter door. However, the mass is highly underestimated and the damping is overestimated. Using the estimated parameters for a second run improved the angle tracking as shown in table 5.4. As for the light door, all parameters but the damping remain close to the initial values, while the estimated damping increased.
We think that the damping overestimation occurs because the mass is underestimated. This could be related to the choice of our estimation model described in section 3.5.3. Since we estimate the parameters in discrete time, we do not use

an acceleration measurement but only the velocity of the end effector. This could impair the accuracy of the mass estimation if also the damping is estimated, and it leads to an overestimation of the damping term. In case of the lifting task in section 4.2, the mass estimation is accurate, but no damping is estimated there. To overcome this issue, a lower variance could be chosen for the damping parameter in the second run, when the estimator is initialized with already estimated parameters.

	MIAC	Combined MIAC MRAC
RMSE Tracking Error [deg]	1.67	2.22
Final Angle Error [deg]	1.32	1.27
Estimated Mass [kg]	1.53	1.50
Estimated Damping [Ns/m]	13.06	12.83
Estimated Spring Stiffness [N/m]	2.89	2.80
Estimated Static Force [N]	14.10	13.98

Table 5.4: Controller performance comparison for the heavy door where controllers are initialized with already estimated parameters

Since the MIAC controller performed best for both door models, we want to present the door angle tracking for the heavier door model in figure 5.2. The EKF for the angle estimation is initialized with a 5 cm offset in x- and y-direction. In simulation the state of the measured door angle is available which is marked with “Measured” in figure 5.2. However, the door angle state of the EKF is used as measured value for the controller since when a real door is opened no measurement of the door angle is available. The door opening starts at second 7 and the estimated door angle converges to the desired one at second 22. The desired trajectory is recomputed with a rate of 4 Hz given the current estimates of the door radius, hinge center position and the current desired position.

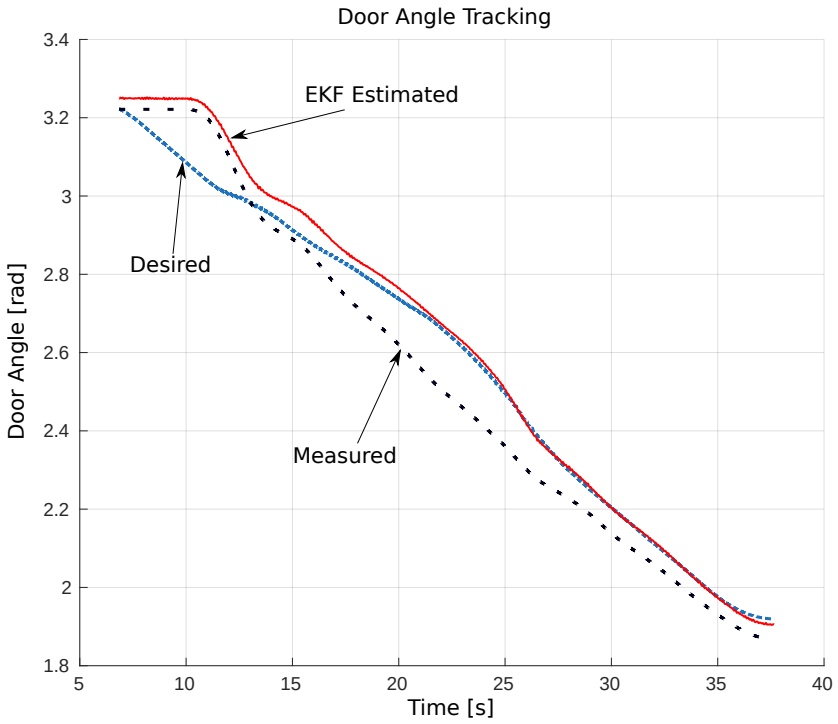


Figure 5.2: Door angle tracking with the MIAC controller

In figure 5.3 the estimated dynamics parameters of the MIAC controller which are fed back to the MPC system dynamics, are plotted over time. We see that all parameters converged except the spring stiffness which was still increasing until the end of the task. Figure 5.4 shows the corresponding input force of the impedance controller which is initially large when the tracking error is large but converges to zero as the estimated environment parameters converge. In the next section the performance of the interaction controllers is presented on the hardware tests.

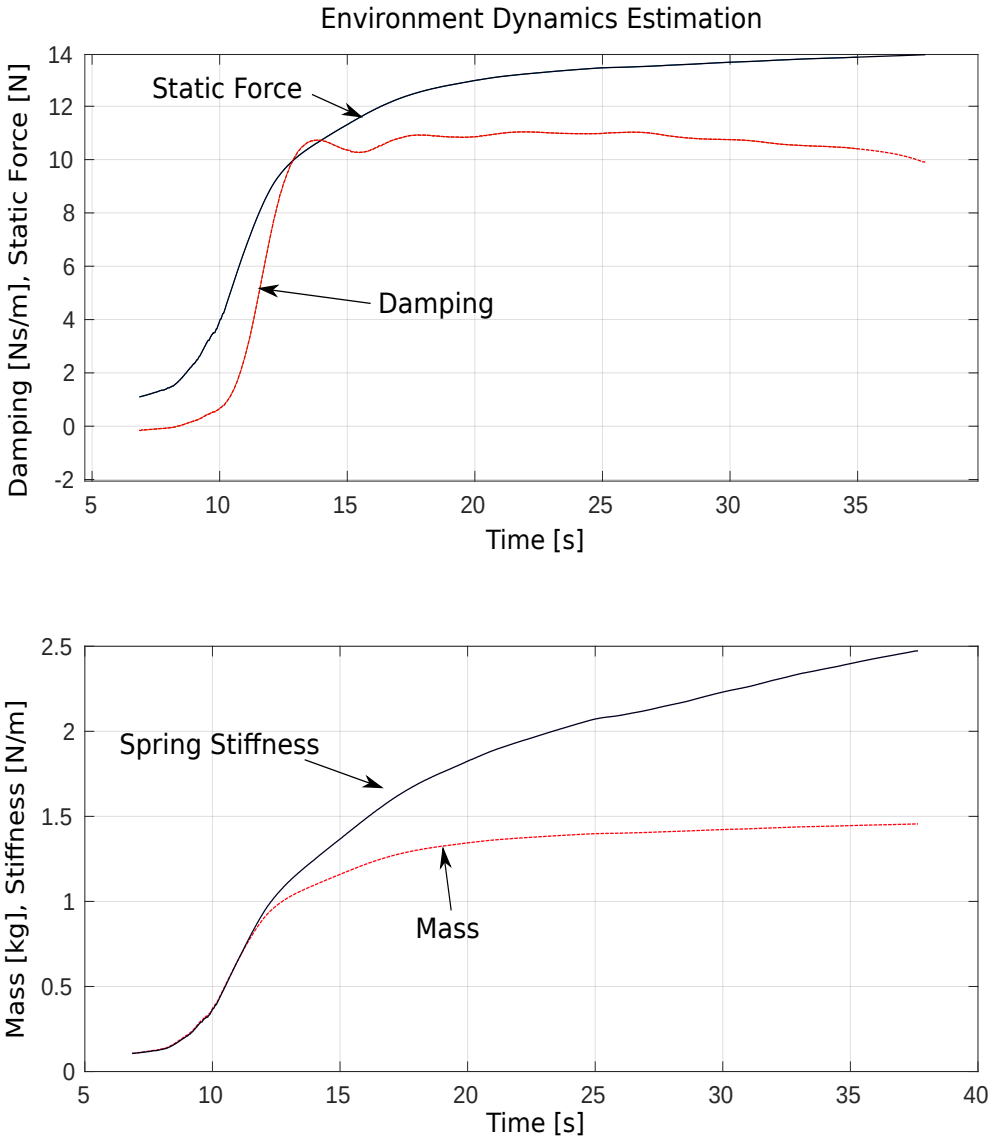


Figure 5.3: Estimated parameters with the MIAC controller that are set in the MPC system dynamics

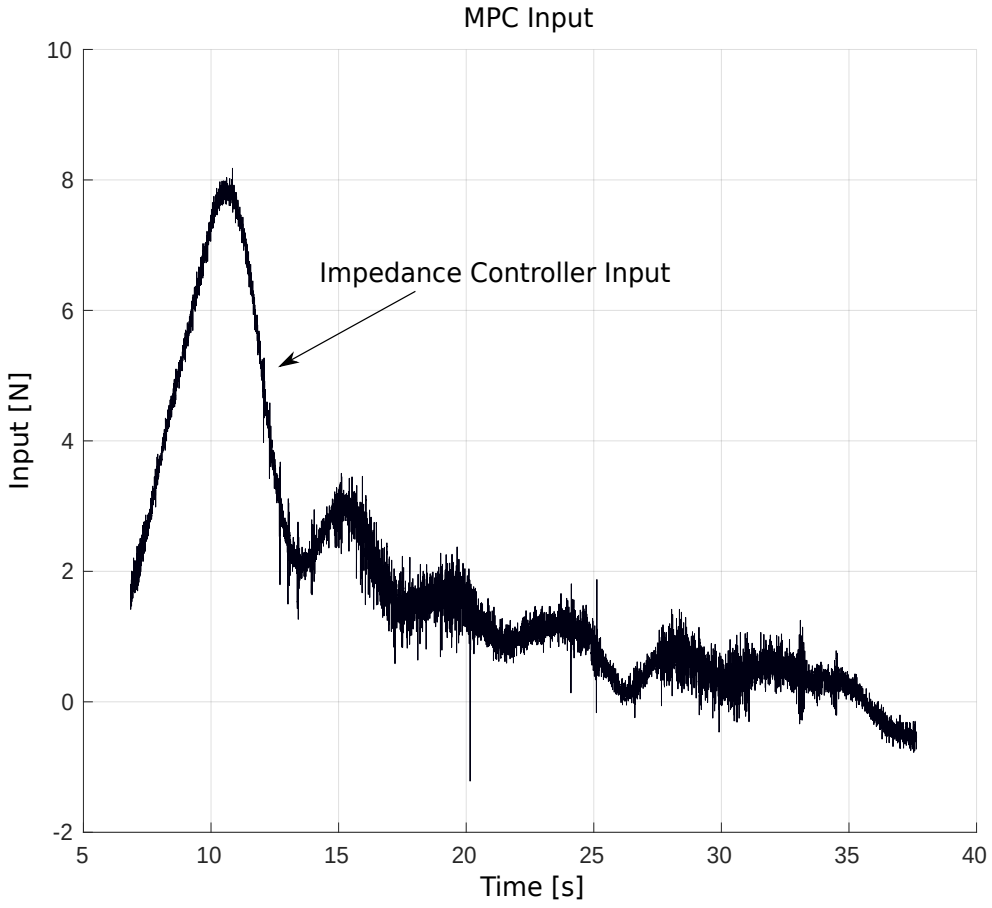


Figure 5.4: The commanded force of the MIAC sent to the MPC controller during the door opening

5.6 Hardware Results

The door opening with the ballbot hardware is performed on a wooden door shown in figure 5.5 which has a width of 90 cm and a handle that is located at around 84 cm away from the door hinge position. This door does not have a door closer and thus no spring behavior. As in simulation, the goal is to open the door by 70°. Since the door alone is very light to open, it does not represent a challenging environment and the robot-environment interaction controllers from chapter 3 behave all equally good. Therefore, we place two boxes of different sizes and weights behind the door which makes the task more challenging.

Figure 5.5: The door with the heavy box

The two boxes used with their weights and dimensions are described in table 5.5. Since it is difficult to avoid a collision by the ball with the door frame due to the robot's limited DOF and the small distance between the handle and the door frame, we extend the handle of the door with the setup shown in figure 5.6. In the following, we present the results for the light and the heavy box. The controllers and estimators are all initialized the same way as described in the simulation results in section 5.5. These parameters were found to yield good results in simulation and, therefore, we want to test their general applicability to the two different doors.

	Light Box	Heavy Box
Weight	1.5 kg	4.2 kg
Base Size	16.5 x 27 cm	27 x 37 cm

Table 5.5: The box weight and dimension

Figure 5.6: The handle extension

Light Box

For the light box, forces around 10 - 15 N need to be applied to move the door. This is a force range that is applied well by the ballbot without the needing to incline a lot. The table 5.6 summarizes the results.
The tracking error results show a similar distribution as in the simulation. For the impedance controller, the tracking error is very large, whereas for the MIAC, the MRAC and the combined MIAC MRAC controllers, it is comparably small. Different from the simulation, the MRAC controller has the smallest tracking error instead of the MIAC controller. Also, the final door angle error for these three controllers is in an acceptable range below 5 degrees. The estimated dynamic parameters of the door are consistent between the controllers regarding the mass and static force. The damping parameter is estimated much higher in the combined MIAC MRAC controller than in the individual ones. Also, the spring stiffness estimate becomes negative in the combined approach.

	Impedance Control	MIAC	MRAC	Combined MIAC MRAC
RMSE Tracking Error [deg]	13.81	2.98	1.81	2.56
Final Angle Error [deg]	-16.21	1.07	1.60	3.90
Estimated Mass [kg]	-	1.44	1.33	1.45
Estimated Damping [Ns/m]	-	12.97	13.62	27.88
Estimated Spring Stiffness [N/m]	-	1.02	1.51	-0.32
Estimated Static Force [N]	-	10.71	11.00	10.04

Table 5.6: Controller performance comparison for the light box

To compare the behavior and characteristics of the different approaches, figure 5.7 shows the cumulative door angle tracking error of all controllers in %. We can observe at second 12.5, which is approximately in the middle of the task, that the cumulative tracking error of the impedance controller marked as "PD" is below 50% which means that most of the error is yet to come and, thus, this controller does not improve over time.
The cumulative tracking error of the MRAC and combined MIAC MRAC controllers is around 50% and for the MIAC controller it is close to 90%. This result implies that the estimated parameters that are used in the MIAC controller seem to be physically correct because the error curve is flattening as time evolves, and the tracking becomes nearly perfect. Although the MRAC controller derivation promises the tracking error to converge to zero, we observe small oscillations around the reference point. This can be related to the choice of the update gain matrix K_π which is either too high or too low, but also to the fact that the reference point is not constant but advancing over time. The combined MIAC MRAC controller has a very constant tracking error over the task that does not converge which implies that the high damping that is estimated is incorrect and could be caused by the interaction of the controllers due to the closed-loop system identification.

Heavy Box

For the heavier box, forces in the range of $20 - 30N$ need to be applied by the robot. In simulation such a demanding door could not be simulated successfully because of the contact between the gripper and the handle which was not rigid enough for these forces. In some runs of the hardware tests, the door could not be opened because the ball collided with the door frame and blocked which lead to a controller failure. This happened sometimes for the MRAC and the combined MIAC MRAC controller since the adaptive term can lead to an oscillation of the commanded forces which peaks can reach $30N$. At these peak forces the ballbot is inclined so much that a door opening becomes impossible. Table 5.7 summarizes the results in the cases where the door could be opened successfully.

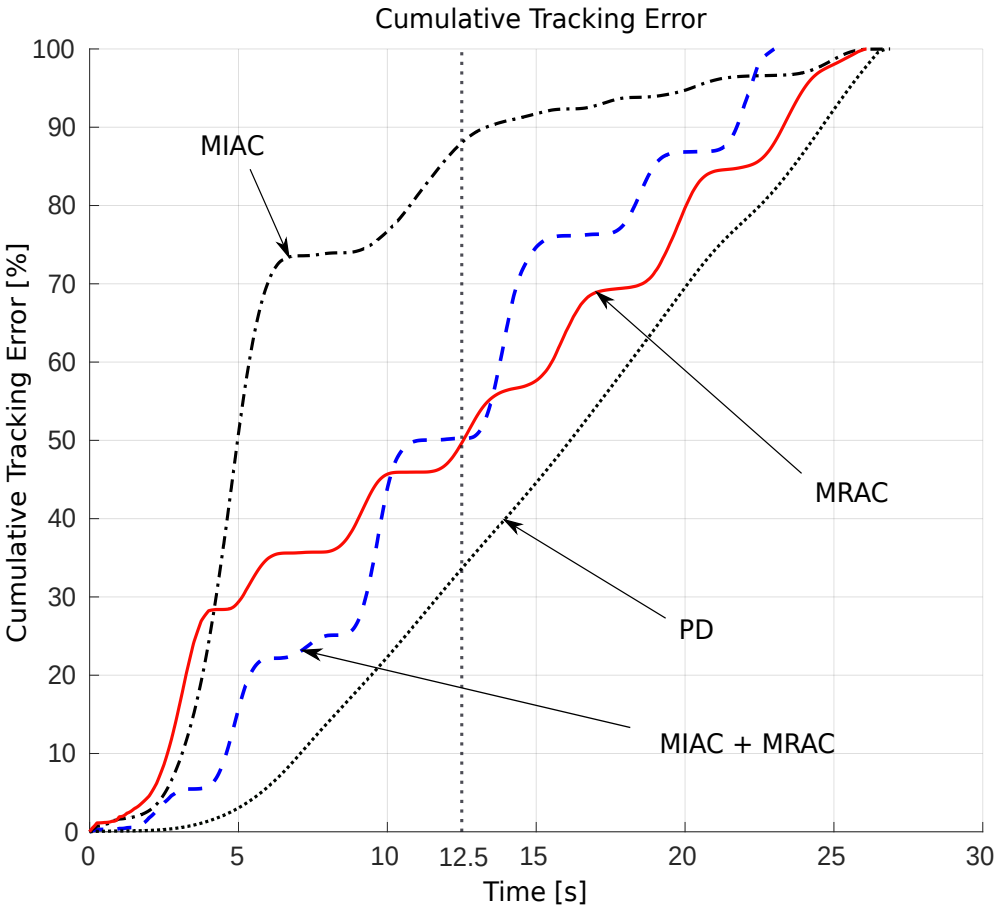


Figure 5.7: Cumulative tracking error comparison between all controllers for the light box

	Impedance Control	MIAC	MRAC	Combined MIAC MRAC
RMSE Tracking Error [deg]	16.86	5.13	2.219	5.73
Final Angle Error [deg]	-14.9	2.58	2.69	-8.59
Estimated Mass [kg]	-	3.3	2.5	4.82
Estimated Damping [Ns/m]	-	22.13	22.07	58.57
Estimated Spring Stiffness [N/m]	-	6.2	0.6	-7.03
Estimated Static Force [N]	-	21.28	16.06	19.4

Table 5.7: Controller performance comparison for the heavy box

For the heavier box, the tracking error and the final door angle error of the MIAC and MRAC controllers increased only a bit which still resulted in a very good performance. The combined MIAC MRAC controller has larger oscillations around the reference point and similar to the light box, the damping is overestimated and the spring stiffness becomes negative. The tracking of the MIAC controller and the evolution of the estimated parameters for the two doors with different boxes shows a very similar behavior to the simulation results in section 5.5. The sequence of the door opening with the MIAC controller is shown in figure 5.8.

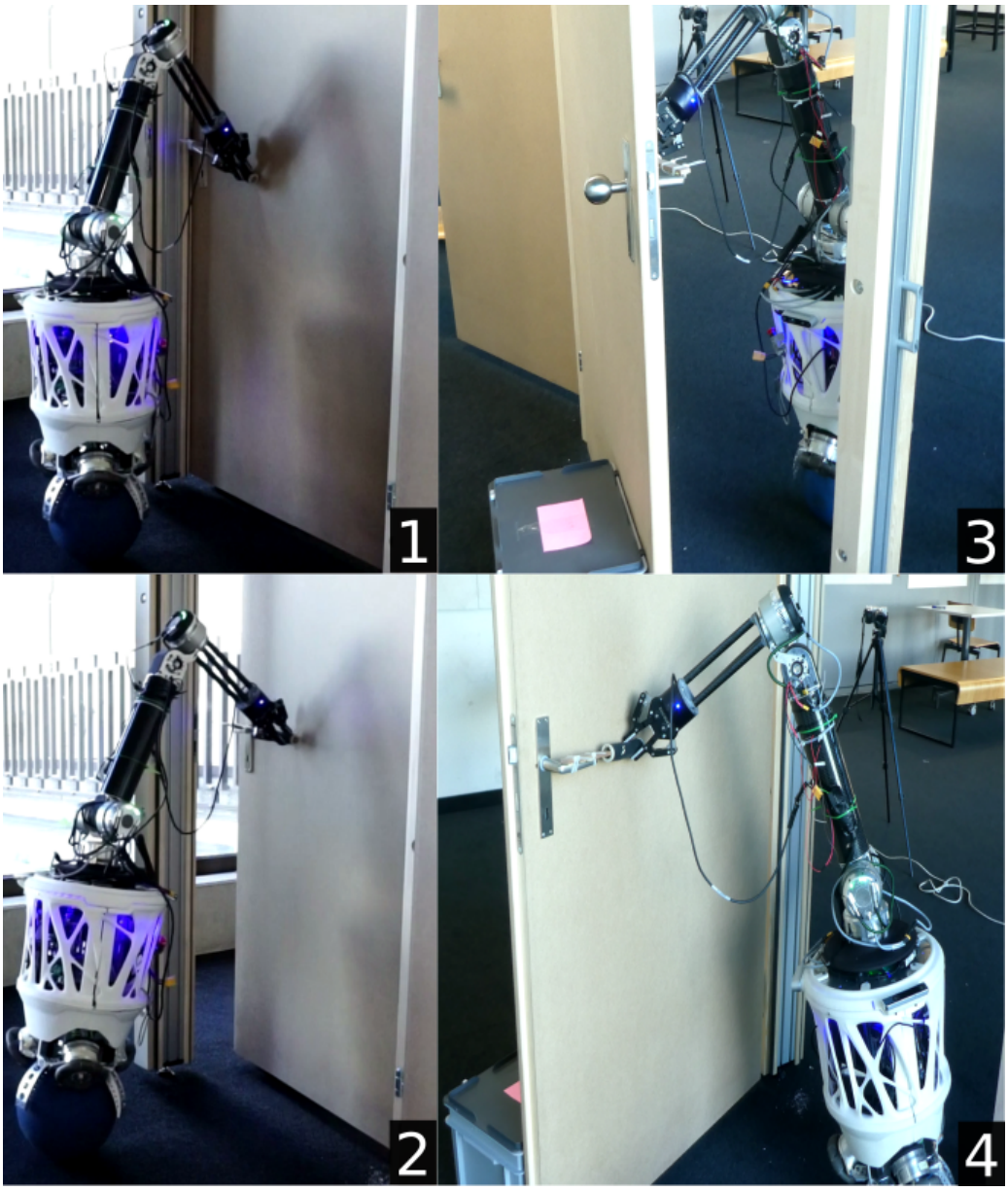


Figure 5.8: The door opening sequence using the MIAC controller with the heavy box behind the door

Since the MRAC controller performed best for this task, we present the data of the angle tracking in figure 5.9 and the commanded force of the MRAC controller that is set in the MPC system dynamics figure 5.10. We can see that the tracking error is not converging to zero and that the force from the adaptive controller is oscillating. This could be caused by the update gains of the adaptive terms. Nevertheless, the tracking error is very small and the door can be accurately opened.

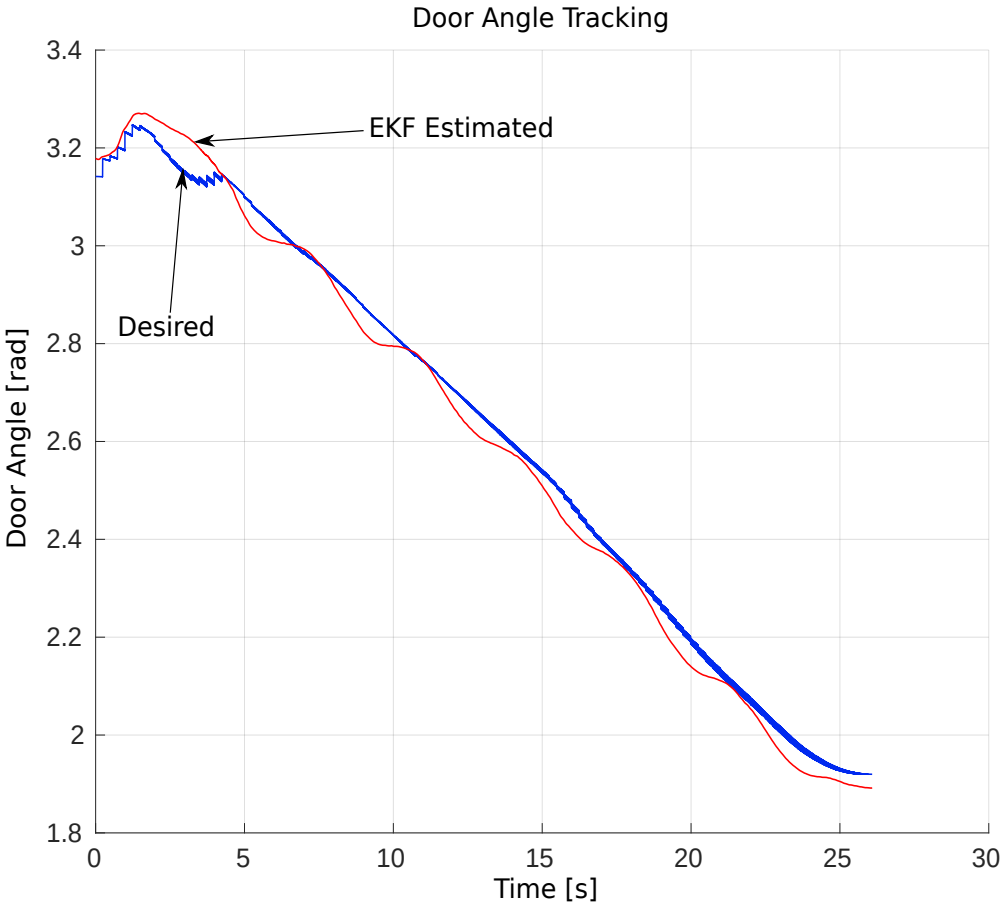


Figure 5.9: Door angle tracking

In section 5.5 we stated the motivation that the MIAC and the combined MIAC MRAC controller can be initialized with the estimated parameters for a second attempt to perform better. This behavior could not be verified on the hardware since the disturbance observer is too slow so that the estimated parameters decrease before they increase again, similar to as it is described for figure 4.1 and figure 4.2 in section 4.2. In the next section, we summarize and conclude the door opening task results.

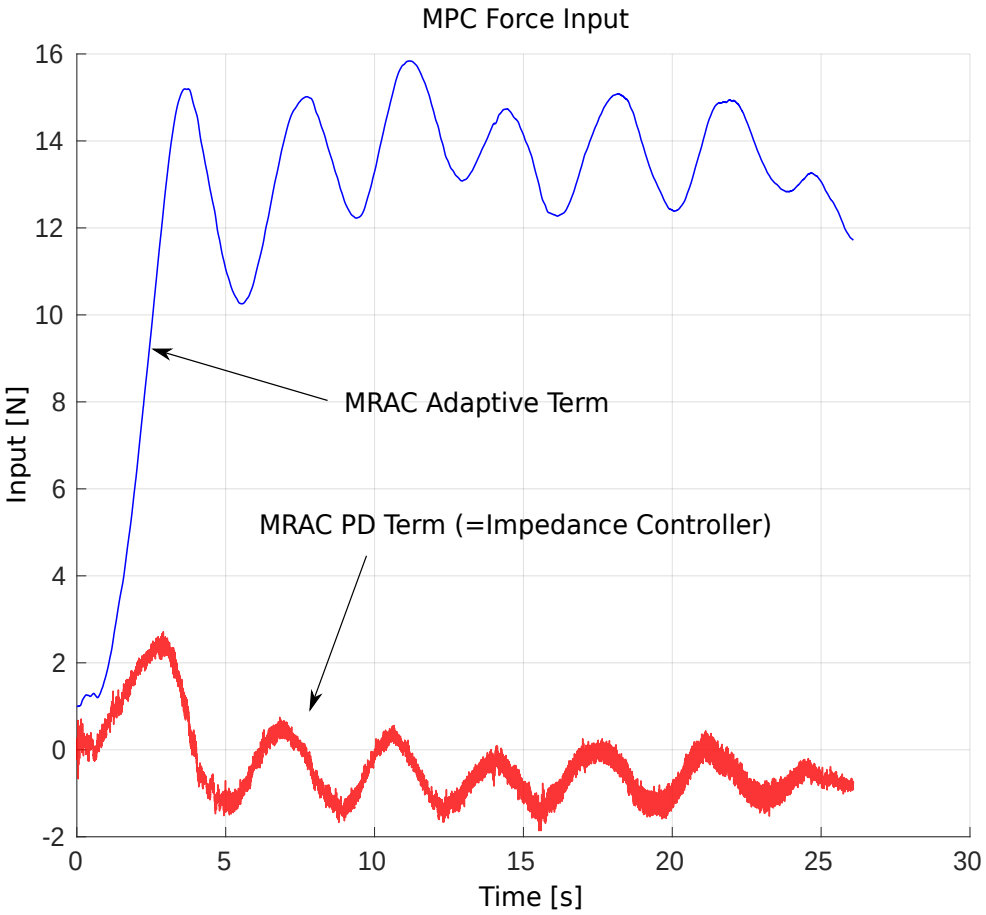


Figure 5.10: The commanded force of the MRAC sent to the MPC controller during the door opening

5.7 Door Opening Task Conclusion

In this chapter we presented how we use the trajectory generation algorithm from section 3.3 to create the circular trajectory with a trapezoidal profile w.r.t. the door angle in section 5.1. To update the trajectory depending on the current door state and the inaccuracies in the prior estimate of the door dimensions, we use an extended Kalman filter presented in section 5.2 to estimate the door radius, angle and hinge position online to recompute the target trajectory.
In section 5.5 we applied the control algorithms from chapter 3 in simulation on two different doors. We could observe that the estimated dynamic parameters along the trajectory cannot be directly related to the dynamic parameters of the door model in the simulation using the relations described in section 5.3. Nevertheless, the estimated parameters along the trajectory that are used for control in the MIAC controller have a positive effect and decrease the tracking error over time which could be shown by the hardware tests and is visible in figure 5.7.
If we compare the hardware test results from section 5.6 with the simulation results, we see that they are very consistent which indicates the accuracy of the Gazebo simulation. On the hardware the MIAC controller did not perform as well as in simulation. This is related to the influence of the PID-controller in the actuators

and its effect on the force estimation, similar to as it is written in the conclusion of the lifting task in section 4.4.

The main problem and challenge of opening a door with the ballbot is to avoid a collision between the ball and the door frame since the ballbot inclines more to apply higher forces. This limits the type of doors that can be opened. To avoid a collision, an additional constraint could be introduced into the MPC problem which limits the allowed ball position during the door opening. However, this could also introduce additional problems regarding the tuning of the whole-body cost parameters described in section 2.3 and the stability of the control approaches derived in chapter 3.
Comparing the tracking error results of the simulation and the hardware test, we observe that the impedance controller derived in section 3.4 is insufficient for an accurate tracking. Reshaping the stiffness and damping matrices to higher values could improve the performance, but it could also diminish the stability for the interaction with other doors. This makes the impedance controller without any adaption or online identification possibilities unusable as a general controller for such robot-environment interaction tasks.
The best controller regarding the tracking error on the hardware for the opening of a door to 70° was the MRAC controller. Although we could not observe a convergence of the tracking error completely to zero over time, the MRAC could quickly adapt and stay close to the reference point. The controller was stably oscillating around the reference point.
Nevertheless, the MIAC controller is an interesting and promising approach in combination with the MPC controller. Figure 5.7 shows that the MIAC controller has a converging tracking error over time, and that parameters that are fed back to the MPC make the MPC aware of the environment and enable a stable interaction. The only drawback of the MIAC controller is that it needs more time for the parameters to converge. If a door is opened only up to e.g. 40° with the same trajectory velocity, the MIAC controller could perform as bad as the impedance controller regarding the tracking because the DOB from section 3.5.2 is too slow, whereas the MRAC controller would have performed well. On the other hand if the door should be opened up to e.g. 120°, the MIAC controller could outperform the MRAC controller because its tracking error is converging as the parameters converge, whereas the MRAC is slightly oscillating around the reference point.
These results suggest to either improve the DOB by making it possible that the measured actuator torque and not the commanded actuator torque can be used or to derive a different formulation of the DOB itself which is not a strong low-pass filtered estimate of the true value and thus does not have a large delay. Another approach would be to use a force sensor behind the gripper which could immediately measure the current robot-environment interaction force.

Regarding this analysis between the MIAC and the MRAC controller, the former expectation of using the combined MIAC MRAC controller was to see a quick initial adaption of the MRAC controller to quickly decrease the tracking error and then a convergence of the tracking error to zero due to the influence of the estimated parameters as for the MIAC controller. However, a better performance of the combined approach could not be observed. A reason for this could be that the update rates of the MRAC controller were too aggressive regarding the slower convergence of the MIAC controller. Another reason could be that since we used the commanded actuator torque instead of the measured one and since we make a closed-loop system identification, the MRAC controller influences the estimation of the parameters. This could have caused the overestimated damping values and thus an incorrect representation of the environment. Nevertheless, we think that

further research regarding a combination of an adaptive controller with online system identification as it is done with the combined MIAC MRAC controller could lead to promising results. We refer to a different combination approach in the next chapter.

Chapter 6

Conclusion and Outlook

In this chapter we summarize and conclude our work and give an outlook for improvements and additional methods that can be used and applied to the ballbot system.

6.1 Summary and Conclusion

In this thesis we successfully derived, implemented and tested control methods for robot-environment interaction control based on an MPC controller using force control. We achieved our goal which was to keep the methods generally applicable which is demonstrated by the fact that the same control methods can be used for the door opening and the lifting task. Other application cases will be suggested in the outlook in section 6.2. Also the premodelling regarding the controllers and the model in the MPC was minimized compared to the work of e.g. Lee et al. [9] so that no mass of the door needs to be known and the door itself is not a part of the MPC controller. The derived controllers are able to adapt to the environment with adaptive control or online system identification and are based on a very general model along the task trajectory.
In section 2.3 we presented the MPC controller that we are working with in this thesis. We set the desired end effector force or the estimated parameters of the environment's dynamics from the interaction controllers, presented in chapter 3, in the system dynamics of the MPC controller. This makes the MPC adaptive to its current environment.
In section 3.2 we state all assumptions based on which we derive and analyze the interaction controllers. A major assumption is that we can describe the environment as a mass, damper, spring, static force LTI-system along the task trajectory. The assumption that the environment can be characterized as a general mechanical system with constant parameters over an interaction interval makes it possible to derive controllers using the classical linear control theory, online system identification and the adaptive control theory.
The simplest and most used control method in the literature about robot-environment interaction is impedance control presented in section 3.4. Although the stability of this controller can be analyzed by assuming that the environment's dynamics' parameters are limited to some upper and lower bounds, we observed that the stiffness and damping gains of the impedance controller have to be chosen compliant to avoid instabilities when it is combined with the MPC controller. But also with a compliant impedance controller, better results regarding the trajectory tracking under robot-environment interaction were achieved than in the cases where the MPC controller was running alone. However, as soon as high static forces are demanded, the

impedance controller does not perform very well which became clear in the simulation and hardware tests of the door opening task in section 5.5 and 5.6 where large tracking errors occurred and the door could not be opened up to the desired angle. A large improvement can be measured when the model identification adaptive controller (MIAC) presented in section 3.5 is used. This controller also uses the impedance controller but estimates at the same time the mass, damping, spring stiffness and static force of the environment's dynamics using recursive least squares. The estimated parameters are fed to the MPC, making the MPC controller adaptive to the environment. The simulation and hardware tests have revealed that as the estimated parameters start to converge, also the tracking error and the contribution of the impedance controller are converging towards zero. Since we use the disturbance observer presented in section 3.5.2 to estimate the interaction forces at the end effector through the generalized torques, the convergence of the MIAC controller is not as fast as the one of the MRAC controller. For both tasks, the door opening and the lifting, a better performance of the MIAC controller could be observed in the simulation than in the hardware tests. A major reason for this is that the commanded actuator torques needed to be used instead of the measured actuator torques because we observed a non-constant offset and drift in the measured actuator torque. Since the actuator commands as presented in equation (3.1) are computed at a very high frequency of 1500 Hz in the drives, while we only approximate the command at 400 Hz, neglecting the integral term of the PID-controller, we lose much of the feedback contribution of the PID-torques. This also explains the worse mass estimation of the lifted object in the hardware tests from section 4.3 than in simulation presented in section 4.2.
The best performance in all hardware tests achieved the model reference adaptive controller (MRAC) derived in section 3.6. The MRAC achieved the best reference tracking because of the fast adaption rate which minimizes the tracking error quickly. We could not see a complete convergence of the tracking error to zero as the derivation of the controller would promise, but measured a small oscillation around the reference point. The reason for this could either be the adaption rate which makes the controller behave differently depending on the environment, but also the reference point which is advanced at each time step and thus does not allow a complete tracking convergence. An advantage of the MRAC controller is that no force estimation is needed.
The last control method that was tested is the combined MIAC MRAC controller described in section 3.7.1. The intuition behind this controller was that the adaptive term should improve the tracking performance in the initial phase similar to the fast adaption of the MRAC controller. This should also stabilize and improve the parameter estimation of the MIAC approach due to the innovation of the force input signal on the environment generated by the MRAC controller. However, a better performance regarding the tracking error could not be observed in the simulation and hardware tests. The combination even had a negative influence on the parameter estimation in the door opening task. While estimating the environment's dynamics' parameters under the MRAC and MIAC controller gave consistent results, the damping was overestimated using the combined approach. It seems that the adaptive controller has a negative influence on the parameter estimation. One reason for this could be related to the adaption rates of the adaptive controller which were chosen too large or too small. Another reason could be that the commanded actuator torque was used for the force estimation and thus a stronger influence of the adaptive controller on the parameter estimator happened since we have a closed-loop system identification. Nevertheless, we believe that a proper combination of the adaptive and the identification adaptive approach could lead to even better results. The simulation and hardware tests of the object lifting task have shown the strength of the combined approach, which had a decent tracking performance with

a good estimate of the mass of the lifted object.
In the next section, we want to give an outlook about which other robot-environment interaction tasks could be attempted with our controllers and what could be improved in future research.

6.2 Outlook

The controller that can be improved the most regarding its performance is the MIAC controller. As we described in the last section, the usage of the commanded instead of the measured actuator torque lead to an underestimation of the object's mass in the lifting task. Improving the quality of the measured actuator torques would make it possible to correctly include the contribution of the PID-terms for the interaction force estimation at the end effector using the disturbance observer (DOB). However, using the DOB introduces a delay in the estimation of the end effector force and, therefore, also affects the performance of the MIAC controller due to the rate of the parameter estimation using recursive least squares (RLS). Therefore, a major improvement could be made by using a force sensor at the end effector. If the filtering of the force sensor data can be done without a large delay while still removing high force peaks that would destabilize the RLS algorithm, the performance of the MIAC controller could be improved a lot.
Although running the combined MIAC MRAC controller as we describe it in section 3.7.1 did not improve the control performance in the expected way, there are other ways on how to combine the adaptive with the identification adaptive method. One promising approach that could not be investigated in more depth in this thesis is the method described by Espinoza and Roascio [24] summarized in section 3.7.2. Maybe it would be possible to formulate the combination of the two approaches as one optimization problem, namely the minimization of the prediction or estimation error of the online system identification method and the minimization of the tracking error as it is done for the adaptive controller to receive optimal values for the adaption rates in the adaptive controller and update rates in the estimator which are then optimally compatible.
To finalize our work, we want to give two additional application cases where our controllers could be directly applied thanks to their general formulation and applicability. The controllers presented in this thesis could be used for a cleaning task similar to the application case described in the paper of Leidner et al. [13]. While in the approach of Leidner et al., the mass of the lifted cleaning tool needed to be known, the mass can be unknown with our approach since it is estimated online and the interaction controllers allow to lift it precisely as it is done in chapter 4. Our controllers could then also be used to track the cleaning trajectory in the window plane, compensating the unknown interaction forces caused by friction and damping between the cleaning tool and the window.
Another application case could be the steering of a service trolley along a trajectory. Our interaction controllers would make it possible to precisely follow the desired trajectory with the robot that grasps the service trolley while adapting to the unknown mass of the trolley, the static force caused by the wheel friction and the damping effects.
These tasks are ideas for further research in the derivation and application of methods for robot-environment interaction control.

Bibliography

[1] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer, 2010.

[2] C. Natale, *Interaction Control of Robot Manipulators: Six-Degrees-of-Freedom Tasks*. Springer, 2003.

[3] M. Vukobratovic, D. Surdilovic, Y. Ekalo, and D. Katic, *Dynamics and Robust Control of Robust Robot-Environment Interaction*. World Scientific Publishing, 2009.

[4] K. Bodie, C. D. Bellicoso, and M. Hutter, ANYpulator: design and control of a safe robotic arm, in *IEEE International Conference on Intelligent Robots and Systems*, 2016, 1119–1125.

[5] F. Farshidian, M. Neunert, A. W. Winkler, G. Rey, and J. Buchli, An efficient optimal planning and control framework for quadrupedal locomotion, in *IEEE International Conference on Robotics and Automation*, 2017, 93–100.

[6] F. Farshidian, E. Jelavic, A. Satapathy, M. Giftthaler, and J. Buchli, Real-time motion planning of legged robots: A model predictive control approach, *IEEE-RAS 17th International Conference on Humanoid Robots*, 2017, 577–584.

[7] F. Farshidian, M. Kamgarpour, D. Pardo, and J. Buchli, Sequential linear quadratic optimal control for nonlinear switched systems, *IFAC-PapersOnLine*, **50**, 2017, 1463–1469.

[8] M. V. Minniti, F. Farshidian, R. Grandia, and M. Hutter, Whole-body MPC for a dynamically stable mobile manipulator, *IEEE Robotics and Automation Letters*, **4**, 2019, 3687–3694.

[9] D. Lee, H. Seo, D. Kim, and H. Jin Kim, Aerial manipulation using model predictive control for opening a hinged door, in *arXiv preprint arXiv:2003.08256*, 2020.

[10] J. Lee, A. Ajoudani, E. M. Hoffman, A. Rocchi, A. Settimi, M. Ferrati, A. Bicchi, N. G. Tsagarakis, and D. G. Caldwell, Upper-body impedance control with variable stiffness for a door opening task, in *14th IEEE-RAS International Conference on Humanoid Robots*, 2014, 713–719.

[11] C. D. Bellicoso, K. Krämer, M. Stäuble, D. Sako, F. Jenelten, M. Bjelonic, and M. Hutter, ALMA - Articulated locomotion and manipulation for a torque-controllable robot, in *IEEE International Conference on Robotics and Automation*, 2019, 8477–8483.

[12] K. Bodie, M. Brunner, M. Pantic, S. Walser, P. Pfändler, U. Angst, R. Siegwart, and J. Nieto, Active interaction force control for omnidirectional aerial contact-based inspection, in *arXiv preprint arXiv:2003.09516*, 2020.

[13] D. Leidner, A. Dietrich, F. Schmidt, C. Borst, and A. Albu-Schäffer, Object-centered hybrid reasoning for whole-body mobile manipulation, in *IEEE International Conference on Robotics and Automation*, 2014, 1828–1835.

[14] P. Fankhauser, C. Gwerder, S. Leutenegger, F. Colas, and R. Siegwart, Modeling and Control of a Ballbot, Bachelor Thesis, ETH Zurich, 2010.

[15] J. Mišeikis, P. Caroni, P. Duchamp, A. Gasser, R. Marko, N. Mišeikiene, F. Zwilling, C. de Castelbajac, L. Eicher, M. Früh, and H. Früh, Lio – A Personal Robot Assistant for Human-Robot Interaction and Care Applications, *IEEE Robotics and Automation Letters*, **5**, 2020, 5339–5346.

[16] M. Shomin, J. Forlizzi, and R. Hollis, Sit-to-stand assistance with a balancing mobile robot, in *IEEE International Conference on Robotics and Automation*, 2015, 3795–3800.

[17] J. Preisig, D. Sako, J.-P. Sleiman, and M. Hutter, Development of the Control Structure for the DynaArm and Visual Servoing guided Apple Grasping, Master Thesis, ETH Zurich, 2020.

[18] F. Ramos, M. Gajamohan, N. Huebel, and R. D'Andrea, Time-optimal online trajectory generator for robotic manipulators, 2013.

[19] F. Angelini, G. Xin, W. J. Wolfslag, C. Tiseo, M. Mistry, M. Garabini, A. Bicchi, and S. Vijayakumar, Online optimal impedance planning for legged robots, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, 6028–6035.

[20] Z. Öreg, H. S. Shin, and A. Tsourdos, Model identification adaptive control - implementation case studies for a high manoeuvrability aircraft, in *27th Mediterranean Conference on Control and Automation*, 2019, 559–564.

[21] C. Erdogan, M. Zafar, and M. Stilman, Gravity and drift in force / torque measurements, Tech. Rep., 2014.

[22] R. Grandia, D. Pardo, and J. Buchli, Contact invariant model learning for legged robot locomotion, *IEEE Robotics and Automation Letters*, **3**, 2018, 2291–2298.

[23] N. T. Nguyen, *Model-Reference Adaptive Control*. Springer, 2018.

[24] A. T. Espinoza and D. Roascio, Concurrent adaptive control and parameter estimation through composite adaptation using model reference adaptive control/Kalman filter methods, in *IEEE Conference on Control Technology and Applications*, 2017, 662–667.