Accurate Simulation of a Collaborative Robot Arm with Cartesian Impedance Control

Joel Holmesson

LUND UNIVERSITY

Department of Automatic Control

MSc Thesis
TFRT-6142
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Tryckeriet i E-huset
Lund 2021

Abstract

Simulation of systems is used in several fields of science as a tool for safe and resource-efficient testing, as well as a tool for prediction. In this thesis, the goal is to produce an accurate simulation of a collaborative robot arm, together with a controller solution. The robot is supposed to learn and perform contact-rich tasks. Impedance control is often the suggested control strategy for such tasks, since this type of controller relates kinematics with dynamics to ensure appropriate interaction forces by enforcing the robot to behave like a mass-spring-damper system. The implemented controller for this thesis is based on a Cartesian controller called 'Forward Dynamics Compliance Controller (FDCC)'. It is using ROS and is developed to be a control solution for a wide range of different robot arms. Together with the physics engine ODE, the robotics simulator Gazebo is used as the simulation environment in this project. In this thesis, the controller framework is applied on a KUKA LBR iiwa 7 degree of freedom (DOF) lightweight robot arm. This project is the first known application of the FDCC as a controller to a 7 DOF arm and on a robot without a dedicated end-effector force-torque sensor. Instead of using an end-effector force-torque sensor, the KUKA robot uses an embedded observer for external joint torques to measure interaction forces. In this project, effort is put into processing sensor signals and tuning of control parameters to make the FDCC operate on the KUKA robot. The signal processing consists of frequency filtering and by limiting the rate of change on the sensor signal. Further on, using the same Cartesian impedance controller, the controller parameters are tuned with the result of a stable behaviour and a simulation response closely matching real system response for commanded Cartesian trajectories. Physical Human-Robot Interaction tests also show stable and responsive behaviour. As a final application example, a peg-in-hole insertion task is solved by both a simulated and a real system.

Acknowledgements

I would like to express my thanks to my supervisor Matthias Mayr at the Department of Computer Science at LTH for giving me this opportunity for my thesis. His broad knowledge and tireless support has been a key part in guiding me in this project. I also feel fortunate to have been working in the robot lab, and would like to thank everyone at the Department of Computer Science for making this possible and giving me this opportunity for an unique experience.

I am also very grateful for the support from the Department of Automatic Control at LTH. A big thanks to supervisors Julian Salt, Björn Olofsson and Anders Robertsson for their knowledge and questions that pushed me forward. Their gathered experience has helped me in the decision-making and I feel truly blessed to have had input from all the angles that they provided. Additionally, thanks to examiner Rolf Johansson for the comments and elaborate feedback.

Lastly, I would like to give my thanks to Oussama Chouman for the supporting work with the software. It has also been helpful to have discussions around the Master's thesis process and each other's work.

Contents

1

Introduction

In robotics there are certain tasks that are of high complexity and are difficult for a human to instruct the optimal algorithm or schedule to a robot. Contact-rich tasks are examples of such [Andrychowicz et al., 2020]. Reinforcement learning [Sutton and Barto, 2018] can be used to learn through exploration of the state and action spaces. For performing experiments in robotics in general and reinforcement learning in particular, the system could behave in unexpected ways, e.g., exerting high forces on work-pieces as a result. To eliminate the risks of damage or injury, simulation can be a useful tool that also simplifies the experiment setup process, making it more resource-efficient and repeatable. Simulation also enables parallelization and faster than real-time operation, which for a learning process further improves the learning time.

In this project, a compliant controller for a 7 degree of freedom (DOF) robot arm is to be implemented. The robot is intended to learn contact-rich tasks and the controller is proposed to be based on impedance control. Impedance control is used to control the dynamic relation between force and motion [Hogan, 1985], and is achieved by enforcing mass-spring-damper system characteristics. Impedance control is often implemented for collaborative robots and has in particular shown to improve the learning process for reinforcement learning [Varin et al., 2019].

1.1 Scientific and Technological Basis

This thesis work is based on the development and findings in [Scherzinger et al., 2017] and [Scherzinger et al., 2019], regarding Cartesian control and their controller development[1]. The controller implementation is using impedance control as a tool to handle external interaction. The controller is expected to be used for robot manipulators with a dedicated wrist-mounted force-torque sensor. The implementation in this thesis is not using such a sensor but instead made possible with the

[1] Cartesian Controller library by Scherzinger et al.: `https://github.com/fzi-forschungszentrum-informatik/cartesian_controllers` *(Accessed: 2021-07-31)*

external joint-torque observer which is embedded in the KUKA software. The observer is described and evaluated using the same robot manipulator in [Chawda and Niemeyer, 2017].

A range of different simulators are used for different robotics applications, where Gazebo is the one used in this thesis. It is a commonly used robotics simulator in general and for manipulation tasks in particular [Collins et al., 2021]. For this thesis, Gazebo is suitable for example since it has the force-torque sensor plugin[2] to measure external interaction in simulation. Together with force-torque sensor requirements, the accuracy of simulation is considered one of the most important issues for learning contact-rich tasks in robotics [Collins et al., 2021]. The advantages of using task-space impedance control for reinforcement-learning applications are discussed in [Varin et al., 2019].

1.2 Objective

This project aims to find a solution for Cartesian impedance control for a KUKA LBR iiwa robot arm[3], which should also be available for a simulation that is an accurate dynamic and kinematic copy of the real system. As a core part of Cartesian impedance control, the goal is to handle collisions and unknown environments while still being able to follow commanded Cartesian trajectories. Since there is a large number of different robot arms that are conventionally commanded using joint positions, it motivates this project to find a controller solution that takes input in Cartesian space and computes controller output in joint space. With this structure, solving contact-rich tasks becomes highly intuitive.

While the project will focus on a particular robot setup, the project also aims at finding a solution that will be easily adapted to similar robot systems and bring additional insight into what challenges could occur. Apart from finding a suitable control solution for the objectives, the expected thesis contributions include findings on project-specific challenges. Cartesian force sensing without a dedicated wrist-mounted force-torque sensor, as well as the aspect of accurate force sensing in simulation are particularly interesting examples of challenges to investigate.

[2] Gazebo force-torque sensor plugin: `http://gazebosim.org/tutorials?tut=force_torque_sensor&cat=sensors` *(Accessed: 2021-07-31)*

[3] KUKA LBR iiwa: `https://www.kuka.com/products/robotics-systems/industrial-robots/lbr-iiwa` *(Accessed: 2021-07-31)*

1.3 Problem Formulation

The main objectives of this thesis can be stated as follows:

- Find a suitable Cartesian controller solution working for both the real and the simulated system.
- The implemented controller should handle unknown or dynamic environments and physical human-robot interaction.
- The controller should be evaluated and tuned such that the simulated system is accurately following the behaviour and response of the real system.

1.4 Thesis Outline

The thesis is structured as follows:

2 — Background Presents the theory and notation regarding robot kinematics and dynamics, as well as formulation of relevant concepts from automatic control and motivation for control-design choices.

3 — Implementation Introduces the Forward Dynamics Compliance Controller [Scherzinger et al., 2017], together with motivation and limitations of the design. Project-specific setup properties and solutions are explained, including system comparison of the real and simulated systems.

4 — Results Introduces the setup of the experiments and presents the results.

5 — Discussion Explains the results and discusses general performance of the controller implementation. The objective of the thesis is compared to the implementation results, and suggested subjects for future work are presented.

6 — Conclusions Concludes with a short summary of the thesis, with emphasis on thesis objective, main contributions and results.

2

Background

This chapter includes theoretical background in robotics control, such as expressions and definitions necessary for understanding the content of the following chapters.

2.1 Force Control

One important aspect for solving a task in robotics is finding a suitable control solution. Position or velocity control is often used to get accurate tracking of a desired motion when the contact forces are low or non-existent. This could be the case, for example, for tasks such as welding or painting. On the other hand, force control could be more suitable when it is important with the spatial tracking in combination with tracking desired forces, and where the task assumes or revolves around contact with the environment.

Force control can be divided into *indirect* and *direct* force control [Siciliano and Villani, 2000]. Indirect force control is achieving desired interaction through motion control, whereas direct force control is using a closed force feedback loop to control interaction forces as desired [Siciliano and Villani, 2000]. *Hybrid Position/Force Control* (further referred to as Hybrid Control) was first proposed in [Raibert and Craig, 1981] and is a way to follow positional trajectories while desired forces are controlled along other axes. One application for hybrid control could be for a grinding robot [Xu et al., 2019], and a simple implementation is possible using PID controllers for both position and force errors. Hybrid control is mainly used when the environment is well-known. One of the core problems for assembly tasks and physical Human-Robot Interaction (pHRI) are collisions, both intentional and unintentional [Haddadin et al., 2017]. For this reason *Impedance Control* (introduced in [Hogan, 1985]) is often the suggested control strategy for such tasks. Impedance control is a type of indirect force control and can be used to relate kinematics with dynamics of a system as a mass-spring-damper system and ensures appropriate interaction forces [Zhou et al., 2021], instead of decoupling position and force control as with hybrid control.

2.2 Impedance Control

Impedance control is a type of force control that dynamically relates position/velocity to force, this to achieve a virtual mass-spring-damper behaviour [Scherzinger et al., 2017]. This can be done in Cartesian space by using commanded force that is related to robot position, velocity and acceleration in the following way [Lawrence, 1988]:

$$F_{imp} = K(x - x_d) + B(\dot{x} - \dot{x}_d) + I(\ddot{x} - \ddot{x}_d) \tag{2.1}$$

where K is the stiffness parameter ("spring"), B is the damping parameter and I is the inertia parameter ("mass"). The desired trajectory $(x_d, \dot{x}_d, \ddot{x}_d)$ is compared to the actual values to get a value of the impedance force to correct the error. According to [Scherzinger et al., 2017], the direct dependency on $\dot{x}_d$ and $\ddot{x}_d$ can be omitted to prevent overshoot and get a smoother motion, and they proposed the impedance force to be:

$$F_{imp} = K(x - x_d) - B\dot{x} - I\ddot{x} \tag{2.2}$$

The damping and inertia terms are then acting along the opposite direction of motion, instead of acting to correct the motion errors $(\dot{x} - \dot{x}_d)$ and $(\ddot{x} - \ddot{x}_d)$. The desired motion characteristics $\dot{x}_d$ and $\ddot{x}_d$ are still indirectly implemented in the planned time-stamped trajectory x_d.

2.3 Stability and Implementation of Impedance Control

In [Lawrence, 1988], Cartesian space impedance control is considered, both for position and torque control implementations. It is from a stability-analysis perspective that they conclude that the choice of position or torque control affects the performance of the impedance control for different applications. Position control is better suited for stiff performance (high stiffness and damping parameters), whereas torque control is better for soft interaction (low stiffness and damping parameters) [Lawrence, 1988]. For both cases, it is noted in both theoretical and experimental results that the relation between stiffness K, damping B and system time-delay t_d has a direct impact on system stability. Therefore, there are limitations on selection of these parameters for stability measures. Still, it is possible to choose values of these parameters such that the system is stable. Furthermore, [Lin, 1997] provides additional insight into the fundamentally different tasks that free motion and contact motion are. They suggest that for an impedance controller implementation to handle both free and contact motion, it should adapt and change characteristics based on whether there are contact forces or not.

2.4 Rigid-Body Dynamics

For deciding motion of a robot arm, torques for each joint are the physical quantities actuated at the lowest point in the abstraction level. To determine the joint torques τ for a multi-joint arm in free space one uses the rigid-body equation [Parra-Vega et al., 2001]:

$$M(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + g(\mathbf{q}) = \tau + \tau_f \tag{2.3}$$

where M is the inertia matrix, C is the Coriolis and centrifugal matrix and g the gravity component. The joint torques from friction forces, τ_f, are sometimes left out in models due to the complexity of modeling this dynamic. These components depend on the joint states angular position $\mathbf{q}$, velocity $\dot{\mathbf{q}}$ and acceleration $\ddot{\mathbf{q}}$ for each joint, and this is referred to when talking about the joint space. The Equation (2.3) can be used for *inverse dynamics*, i.e., when mapping from joint states to joint torques. The inverse problem is then that of determining the joint states from applied torques, also called *forward dynamics*. Joint accelerations $\ddot{\mathbf{q}}$ can be expressed directly from Equation (2.3) (leaving out friction torques and assuming M is invertible) as

$$\ddot{\mathbf{q}} = M(\mathbf{q})^{-1}(\tau - C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} - g(\mathbf{q})) \tag{2.4}$$

and subsequently joint velocity and position by integrating $\ddot{\mathbf{q}}$. When free space motion cannot be guaranteed, i.e., if the environment is dynamic or unknown, external joint torques τ_{ext} can be added to the model in Equation (2.3) to model the contact with the environment that the robot experiences:

$$\tau + \tau_f - (M(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + g(\mathbf{q})) = \tau_{ext} \tag{2.5}$$

2.5 Kinematics and Transformations

For robot arms with several joints, it is necessary to be able to model or predict motion of joints and/or end-effector, given the other quantity. Depending on what states are of interest to control and which variables that are measurable, and what tasks to be performed, one would want to translate joint states to end-effector states, or vice versa. This is what is referred to when talking about forward and inverse kinematics in robotics and biomechanics. This can be expressed as a function $\mathscr{K}$:

$$\mathbf{x} = \mathscr{K}(\mathbf{q}) \tag{2.6}$$

mapping joint configurations $\mathbf{q}$ to end-effector position and orientation $\mathbf{x}$ in Cartesian space. For an N-joint robot, $\mathbf{q}$ is an N-dimensional vector holding each angular position q_i for joint i. $\mathbf{x}$ is a 6-dimensional vector, holding both position and rotation in Cartesian space $\mathbf{x} = (p_x, p_y, p_z, \alpha, \beta, \gamma)$, where (α, β, γ) as the Euler angles (or the roll-pitch-yaw representation) is one alternative for orientation representation. Equation (2.6) is robot specific, and depends on the length of robot links and

the relation of reference frames in between joints. A relation like Equation (2.6), i.e., end-effector Cartesian kinematics as a function of joint kinematics, is called *forward kinematics*. For velocities, the Jacobian J is used for the corresponding transformation:

$$\mathbf{v} = J(\mathbf{q})\dot{\mathbf{q}} \tag{2.7}$$

where $\mathbf{v}$ is the Cartesian end-effector velocity vector, also called *twist* [Scherzinger et al., 2019]. The Jacobian depends on the joint positions $\mathbf{q}$. A corresponding relation for the acceleration $\mathbf{a}$ is

$$\mathbf{a} = \dot{J}(\mathbf{q})\dot{\mathbf{q}} + J(\mathbf{q})\ddot{\mathbf{q}} \tag{2.8}$$

where one can approximate $\dot{J}(\mathbf{q})\dot{\mathbf{q}} = 0$ to get instantaneous acceleration [Scherzinger et al., 2019]. The relation between joint torques τ and the Cartesian end-effector forces and torques vector $\mathbf{F}$ can be found using the Jacobian [Scherzinger et al., 2019]:

$$\tau = J^T(\mathbf{q})\mathbf{F} \tag{2.9}$$

Further on, when talking about forces on the end-effector $\mathbf{F}$, it is referred to the 6D vector $\mathbf{F} = (f_x, f_y, f_z, \tau_x, \tau_y, \tau_z)$, containing equivalent forces and torques acting along or around each Cartesian axis. Equation (2.9) is a useful mapping between joint space and Cartesian space for forces and torques. This transformation only relies on the Jacobian and measurements or estimations of mentioned forces and torques.

2.6 Redundant Manipulators

Manipulators with different number of joints have different possibilities to reach different end-effector positions and orientations. Some robots have several possible joint configurations to resolve a specific end-effector position and orientation. This is what is called kinematic redundancy and a redundant manipulator is a manipulator that has more degrees of freedom than required to solve the task. Often the task is to achieve specific end-effector position and orientation, which requires 6 degrees of freedom (3 for position and 3 for orientation). Therefore, manipulators with 7 degrees of freedom or higher are often referred to as redundant manipulators. For a redundant manipulator in such an example, it is possible to change joint positions while keeping the same Cartesian position of the end-effector. This is called null-space motion, and results in the main advantage of a redundant manipulator, namely an increased dexterity and a less constrained workspace. In addition, this can also put less demand on motion planning and knowledge of the environment. Additionally, the excess degrees of freedom can be used to satisfy other desired properties of solving the task, such as avoiding obstacles, problematic joint configurations or to minimize energy consumption [Chiaverini et al., 2008]. With these advantages presented, complexity is, however, added to the system and the main issue to resolve is a solution to the inverse kinematics problem.

2.7 Inverse Kinematics and Singularities

The inverse problem of Equation (2.6) is called *Inverse Kinematics* (IK):

$$\mathbf{q} = \mathscr{K}^{-1}(\mathbf{x}) \tag{2.10}$$

As the function $\mathscr{K}$ is highly non-linear, it is difficult to invert and a closed-form solution does not always exist. It is possible to find analytic solutions for certain link chains. Still, one might end up with several possible solutions of joint configurations that lead to the same desired end-effector position [Aristidou et al., 2018]. In particular, this is the case for redundant manipulators, where there could be infinitely many solutions for a given end-effector position and orientation. In general, inverse kinematics has been a significant subject for research for many years with several proposed solutions [Aristidou et al., 2018], where the Jacobian is a central part of the more commonly used numerical solutions. For such solutions, the Jacobian is used iteratively as a linear approximation related to Equation (2.7). One could interpret it as a change in joint positions leads to a change in end-effector position and orientation determined by the Jacobian evaluated at that time instant:

$$\Delta\mathbf{x} \approx J\Delta\mathbf{q} \tag{2.11}$$

The inverse of this approximation (i.e., linear approximation of IK) would require the inverse Jacobian. For some configurations of the joint states, a change in desired end-effector position and orientation cannot be achieved through a change in joint positions. This is, e.g., the case at hardware limits of a manipulator or when joint axes align. This is called a singularity and from the Jacobian is rank-deficient, and similar problems can occur in the neighbourhood of a singularity. Related to equation (2.11) this would correspond to that the inverse is not being defined. Since the Jacobian might not be invertible (e.g., because of a singularity or not being a square matrix), using the approximation (2.11) is not prefered for an IK algorithm. Instead, by using the pseudo-inverse (or *Moore-Penrose* inverse) of the Jacobian [Aristidou et al., 2018], or using a *Jacobian transpose method* [Scherzinger et al., 2019] ensures that there is a solution independent of properties of the Jacobian. These approaches might still have problems of convergence close to singularities. Inside the control or path-planning implementation, singularity avoidance can be implemented to avoid these problems.

2.8 State Observer

In control theory, *state observers* (sometimes called *state estimators* in other fields) are used to estimate a state (physical quantity or signal) for which a sensor is not available. Combining a model of the system and available sensor signals, the state observer is produced and updated, e.g., using statistical methods. In robotics, such an observer could be an estimate of external forces or torques, based on joint states and torques.

2.9 Notch Filter

Signals containing undesired frequency content or noise can be a problem for a control implementation, especially when using a feedback loop with the inherent risk to amplify the noise. For example, sensor signals can be a source of this noise, and low-pass or high-pass filters can be used to filter out the undesired frequency content of noisy signals. Depending on the system signal structure and properties, there might be specific frequencies that one wants to remove from a signal. Then a bandstop or notch filter can be used to decrease the magnitude of signal components with frequencies in a selected frequency interval [Tseng and Pei, 2001]. A transfer function for an infinite impulse response (IIR) implementation of such a filter can in discrete time be expressed as:

$$H(z^{-1}) = \frac{1 - 2\cos(\omega_0/f_s)z^{-1} + z^{-2}}{1 - 2r\cos(\omega_0/f_s)z^{-1} + r^2 z^{-2}} \tag{2.12}$$

where ω_0 is the center frequency of that which is filtered out and f_s is the sampling frequency. The variable r, ranging in value from 0 to 1, is used as the parameter of the width of the frequency band (to be filtered out) around the center frequency ω_0.

3

Implementation

This chapter includes explanation of the controller implementation, project-specific solutions, as well as an in-depth description of the Cartesian controller library introduced in [Scherzinger et al., 2017].

3.1 Forward Dynamics Compliance Controller

Impedance control as described in Chapter 2 is a type of force control. For different applications and robots it might not be the case that forces or torques are measurable or the desired control variables. It is sometimes the case for industrial robots that only joint positions or joint velocities are available as control interfaces. With such limitation, achieving a mass-spring-damper behaviour might not be as straightforward. In [Scherzinger et al., 2017], a suggested control solution they call *Forward Dynamics Compliance Controller* (FDCC) is presented (see control structure in Figure 3.1), using a forward dynamics model to find the equivalent commanded joint states (joint accelerations $\ddot{\mathbf{q}}$, joint velocities $\dot{\mathbf{q}}$ and joint positions $\mathbf{q}$) the force from the impedance controller would correspond to. Either joint positions or joint velocities can then be used as command signals from the FDCC to the robot and still follow the desired dynamic behaviour as expected from an impedance controller. Using a virtual model, the states are tracked and used for forward kinematics to update the impedance control signal, while the only feedback from the real robot comes in the form of the end-effector force sensor signal. In [Scherzinger et al., 2017], they report good stability and performance for a number of different robot arms. However, as mentioned in [Lawrence, 1988], a proper selection of impedance parameters is of importance for such stability. The suggested controller solution is also presented with a well-documented controller library (publicly available at GitHub[1]) with a core goal to be a "power-on-and-go solution" for a wide range of different robot arms. Based on this, this controller library was chosen to develop a Cartesian compliance controller for this thesis.

[1] Cartesian Controller library by Scherzinger et al.: `https://github.com/fzi-forschungszentrum-informatik/cartesian_controllers` *(Accessed: 2021-07-31)*

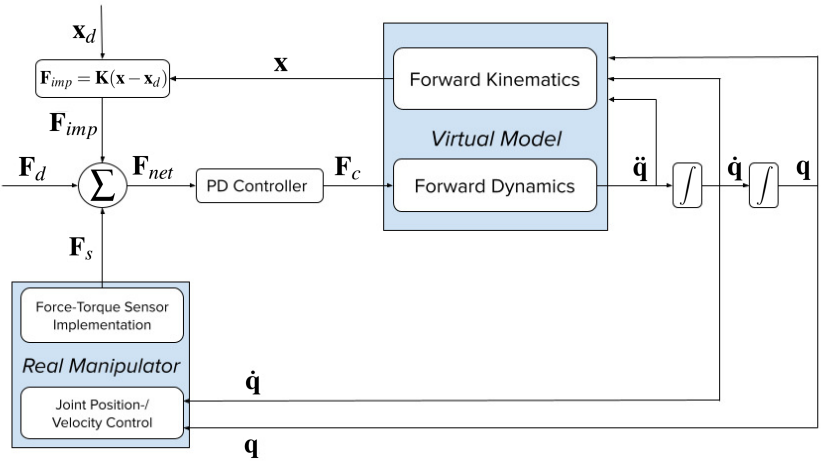

Figure 3.1 Control structure of the FDCC implementation used in this thesis, based on the figure presented in [Scherzinger et al., 2017] of the *Cartesian compliance controller*. The figure includes the virtual model in the top middle and the real robot manipulator which is represented by the block in the bottom left. The "real manipulator" block represents both real and simulated systems in this thesis, whereas the "virtual model" block is a model used inside the FDCC.

3.2 Calculation of the Force Signal

In order to realize a compliant behaviour, the controller incorporates external forces. The main idea for this to be achieved is the summation of external forces $\mathbf{F}_s$ on the end-effector and the mass-spring-damper force $\mathbf{F}_{imp}$ calculated from the impedance formula (2.2). Including a controllable desired force variable as $\mathbf{F}_d$, the net force summation also holds the ability to control applied force, e.g., to a contact surface,

$$\mathbf{F}_{net} = \mathbf{F}_{imp} + \mathbf{F}_s + \mathbf{F}_d \tag{3.1}$$

In the Cartesian controller library[1], the impedance controller is using the simplest model of the environment impedance, which is a spring model

$$\mathbf{F}_{imp} = \mathbf{K}(\mathbf{x} - \mathbf{x}_d) \tag{3.2}$$

As the difference between this formulation and the one in Equation (2.2) relates to Cartesian velocities and accelerations, Equation (3.2) can be a reasonable model as long as the end-effector velocities and accelerations are expected to be low in magnitude. A PD-controller is used to control the net force to zero, such that the system finds equilibrium. Using a gain scaling factor for all dimensions, c, the resulting

force control signal $\mathbf{F}_c$ can be expressed as follows

$$\mathbf{F}_c = c(-\mathbf{k}_p\mathbf{F}_{net} - \mathbf{k}_d\dot{\mathbf{F}}_{net}) \tag{3.3}$$

According to [Scherzinger et al., 2017], $\mathbf{k}_p$ and $\mathbf{k}_d$ should be fairly robot independent and depend more on the desired behaviour than which robot it is controlling. The variable c is a convenient parameter for tuning and finding an appropriate range for the gain.

3.3 Forward Dynamics Algorithm

A solution for the forward dynamics block in Figure 3.1 is presented in [Scherzinger et al., 2019] and is implemented in the controller library[2]. Starting from Equation (2.4) with the forward dynamics, this expression is simplified by omitting the Coriolis and centrifugal term as well as the gravity term. This can be motivated by the Coriolis and centrifugal term has a small impact on slower motion and gravity compensation is used as default for any industrial robot arm. This results in the simplified expression of the forward dynamics

$$\ddot{\mathbf{q}} = M^{-1}(\mathbf{q})\tau \tag{3.4}$$

which in the next step uses Equation (2.9) to get an expression for transforming the end-effector forces $\mathbf{F}$ to joint accelerations $\ddot{\mathbf{q}}$ (the dependency on $\mathbf{q}$ in M and J is further on dropped in the notation for readability)

$$\ddot{\mathbf{q}} = M^{-1}J^T\mathbf{F} \tag{3.5}$$

Joint states $\mathbf{q}, \dot{\mathbf{q}}$ and joint angular accelerations $\ddot{q}$ (where joint states are obtained after numerical integration of $\ddot{\mathbf{q}}$) are passed to a virtual model. The controller is using a parameter that sets the number of *internal iterations*, l_{itr}. This means that the virtual model is updated l_{itr} number of times before the joint position/velocity commands are sent to the real robot. Together with the gain scaling parameter c, these parameters make up the main tuning tools for making the system have slower or faster response. As long as there is no external interaction and the position/velocity control on the robot is well-tuned, the virtual model will be very close to the real robot without using position/velocity feedback.

3.4 Virtual Model

Apart from circumventing potential problems with inverse kinematics, the virtual model also makes the implementation less dependent on the real robot dynamics. One also does not need to know the dynamics parameters, e.g., masses of

[2] Cartesian Controller library by Scherzinger et al.: `https://github.com/fzi-forschungszentrum-informatik/cartesian_controllers` *(Accessed: 2021-07-31)*

joints. With the forward dynamics model and the resulting commanded joint positions/velocities, the real robot will approximately follow the virtual model. In this way, rather than making the virtual model be close to the real robot's response from a net force $\mathbf{F}_{net}$, one can choose the system behaviour more freely. In particular, modeling the robot arm with almost zero link masses and instead putting mass centered at the end-effector will result in beneficial, "homogeneous" properties for the inertial matrix $M(\mathbf{q})$, which represents the inertial joint torque τ mapping to joint accelerations $\ddot{\mathbf{q}}$ depending on joint configuration $\mathbf{q}$. By choosing this model, the authors in [Scherzinger et al., 2019] argues that for instantaneous accelerations ($\mathbf{a} = J\ddot{\mathbf{q}}$) an almost constant, diagonal matrix $JM^{-1}J^T$ can be achieved to get uniform Cartesian acceleration $\mathbf{a}$ due to external forces $\mathbf{F}$ throughout the workspace $\mathbf{q}$:

$$\ddot{\mathbf{x}} = JM^{-1}J^T\mathbf{F} \tag{3.6}$$

3.5 Force-Torque Sensor Signal

Since the virtual model is used for state updates, there needs to be some sensor signal from the real robot to notice external interaction, which is not modeled in the virtual model. In the controller setup in Figure 3.1, a force-torque sensor signal is used to get information of interaction with a dynamic or unknown environment, and relating this to desired interaction forces. For this control solution, it is expected that this signal is a 6D vector $\mathbf{F}_s$ with Cartesian force-torque measurements observed at the end-effector frame. Such a sensor is not mounted on the KUKA robot. However, one can use the external torque estimates for each joint as a sensor signal $\hat{\tau}_s$, such that expected sensor signal $\mathbf{F}_s$ can be calculated (as Equation (2.9))

$$\mathbf{F}_s = (J^T)^{\dagger}\hat{\tau}_s \tag{3.7}$$

where $(J^T)^{\dagger}$ is the pseudo-inverse (*Moore-Penrose* inverse to ensure uniqueness) of the transposed Jacobian. Noise in the momentum observer has been observed, both in our experiments and in-depth described and modeled in [Chawda and Niemeyer, 2017]. According to [Chawda and Niemeyer, 2017], the external torques estimates of the real KUKA robot manipulator is using a momentum observer (per joint):

$$\hat{\tau}_s(t) = K_I\left[p(t) - \int_0^t (\tau_J + \hat{\tau}_s)dt - p(0)\right] \tag{3.8}$$

$$p(t) = J_l\dot{q}_l, \qquad \hat{\tau}_s(0) = 0 \tag{3.9}$$

where $p(t)$ is the generalized momentum, K_I is the observer gain and τ_J is the commanded joint torque. These estimates and the momentum observer are described in depth in [De Luca et al., 2006] and [Haddadin et al., 2017]. However, using the external torque estimates $\hat{\tau}_s$ from the KUKA robot to forward as external force signal $\mathbf{F}_s$ (using Equation (3.7)) to the FDCC controller was not possible out of the

box. The main problem observed is that the resulting "sensed" external force $\mathbf{F}_s$ was non-zero and too noisy during free space motion, and especially affected by fast accelerations (e.g., start and stop of a trajectory). The FDCC did not perform as expected with that signal as the external force-torque signals. With no external forces acting on the robot and no commanded trajectory, oscillations in the system were observed as soon as a virtual end-effector force-torque sensor was applied to the FDCC using Equation (3.7).

In [Chawda and Niemeyer, 2017], they are aiming at finding an appropriate torque control solution using the same KUKA robot. A filter and an additive controller for torque control is suggested to handle nonlinearities in the form of vibrations in the joint motor system. Although they also describe non-linear behaviour with non-zero external joint torque signals from the KUKA observer, slightly different challenges are met for our implementation using the FDCC. With our implementation, a decision was made to limit the rate of change of the external force signal $\mathbf{F}_s$ with a rate of change limit parameter $(\Delta\mathbf{F}/\Delta t)^{max}$, such that the transformed signal $\mathbf{F}_s$ is modified:

$$\mathbf{F}_s^{lim}[k] = \mathbf{F}_s^{lim}[k-1] + \text{sign}(\Delta\mathbf{F}[k]) \cdot \max\left(|\Delta\mathbf{F}[k]|, \left(\frac{\Delta\mathbf{F}}{\Delta t}\right)^{\max}\right) \tag{3.10}$$

where $\Delta\mathbf{F}[k] = \mathbf{F}_s[k] - \mathbf{F}_s^{lim}[k-1]$. The idea here is to limit signal peaks/outliers, which are most noticeable when the input is similar to a step-change. It may also limit system oscillations to be fed back into the feedback controller, such that amplified oscillations cannot spiral into instability. Apart from the non-linearities observed in the external torque signal $\tau_{s,obs}$, using the inverse Jacobian could also be a source of undesired signal content, in particular close to singularities.

3.6 Compatibility, Simulation and ROS

The controller implementation is based on ROS and ROS-control[3], which is an interface to send and receive control signals between different software. ROS-control interfaces can either be implemented by drivers of real robots or simulators. Gazebo[4] is one example of a simulator, and the one used in this thesis. This project's system structure and control signals are displayed in Figure 3.2, which also highlights similarities and differences of real and simulated in the implementation. One important feature of this structure is that the signals to and from the controller are of the same type, and that the controller does not need to be aware of whether the system is a real or simulated one. This simplifies comparison of the two systems and reduces the gap between implementations. Update rates for simulation is 1000 Hz

[3] ROS-control: `http://wiki.ros.org/ros_control` *(Accessed: 2021-08-04)*

[4] Gazebo / ROS control: `http://gazebosim.org/tutorials/?tut=ros_control` *(Accessed: 2021-08-04)*

and for the real setup it is 200 Hz. The FDCC controller implementation can pass both joint angular velocities $\dot{\mathbf{q}}$ and positions $\mathbf{q}$ as command signals. On the other hand, the KUKA iiwa ROS system can only receive joint commands in joint angular positions $\mathbf{q}$ or joint torques τ, which is the reason this implementation is using joint position commands. The iiwa ROS[5] software connects ROS with the KUKA software. A URDF (*Unified Robotic Description Format*) file holds a description of a kinematic and dynamic model of the robot and URDF is a standardized description of robots which most systems can read.

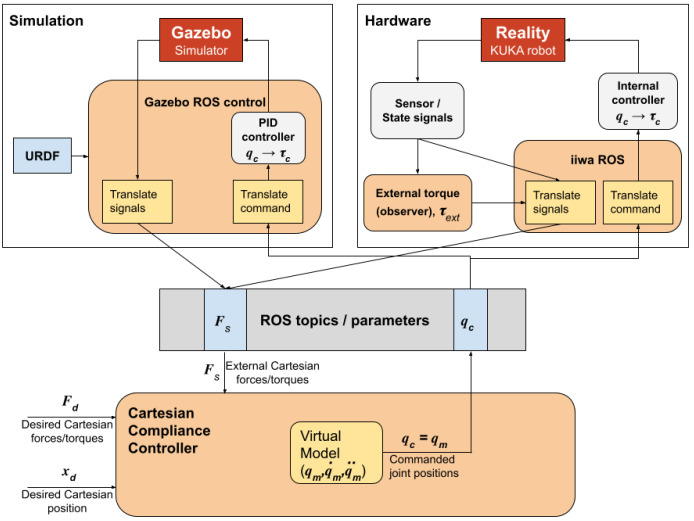

Figure 3.2 System structure of the simulated and real robot processes, including signal scheme for signals relevant to the implemented external Cartesian compliance controller. The virtual model in the Cartesian compliance controller is updated with the correct state at start, then the $\mathbf{F}_s$ signal is the only feedback during execution.

Inner Controller

After interpreting the commanded joint angular positions signal from the FDCC controller, the inner controller will need to control joint torques to achieve the desired joint angular positions. In Gazebo, this is controlled using a PID controller. This means that it is possible to tune this controller to mimic the response of the real system. Although it is theoretically possible to mimic the inner controller response (assuming the inner controller of the real KUKA robot is not overly complex for a

[5] iiwa ROS: `https://github.com/IFL-CAMP/iiwa_stack` *(Accessed: 2021-08-04)*

PID to imitate), it might be difficult to isolate controller characteristics from physics characteristics (real and simulated). In addition to that, as real systems generally have, there are physical or hardware constraints of what experiments are possible to execute. The non-linearities of the system depend on joint states, which further complicates the response analysis. Therefore, the goal of tuning the PID controller in Gazebo is then to get a reasonably close match between the simulated and the real kinematic characteristics, for a chosen set of joint trajectories.With the FDCC controller implementation, the gap between reality and simulation is narrowed and there is more control and knowledge of what is happening in the control loop.

External Forces

The external force-sensor signals $\mathbf{F}_s$ are from different origins in simulation and for the real robot. The real robot uses the external joint-torque observer, and in simulation the sensor signal is available directly through a force-torque sensor plugin. This virtual sensor is applied to the last joint (joint 7), which is the joint connecting two links, one of them being the end-effector link. Initial investigation of this sensor signal in simulation, suggested that the signal should also have limited rate of change (same as was done with the real robot signal in Equation (3.10)) due to the simulated force sensor also showing large spikes and non-zero values for free-space motion. As opposed to the sensor signal for the real robot (see Equation (3.7)), the signal from the sensor plugin in simulation is (at least in theory) independent of the Jacobian. In general, due to the different origin and nature of the external force signal for the real robot and in simulation, it is expected that these feedback signals are not giving the same results. Therefore, the force control might not be expected to produce similar results, but the goal would rather be to make both models have stable Cartesian motion and behaviour in contact. Gazebo is using *Open Dynamics Engine* (ODE)[6] as physics engine.

[6] ODE: `https://www.ode.org/` *(Accessed: 2021-08-04)*

4

Results

In this chapter, procedures and circumstances of tests and measurements are presented, together with the results. The opening part addresses challenges of estimating the external force sensor signal, and presents results of suggested signal-processing methods.

4.1 Experiment Setup

Coordinate Frames

Figure 4.1 shows a joint configuration and Cartesian position and orientation of the end-effector close to the starting state for most of the following experiments. Cartesian motion is defined in relation to *Base frame*, whereas external force-torque $\mathbf{F}_s$ and desired force-torque $\mathbf{F}_d$ are in relation to *Peg frame*. Most importantly to note is that the z-axes in the two frames are oriented pointing in opposite directions, especially in the case for the experiments where the peg is in contact with the surface.

Parameter Settings

As a result of the requirements of the different tests, the controller parameters differ between tests. However, there are parameter settings that are more common than others. These are presented in Table 4.1, and if a parameter setting is not mentioned for a test procedure, the default setting of that parameter is implied.

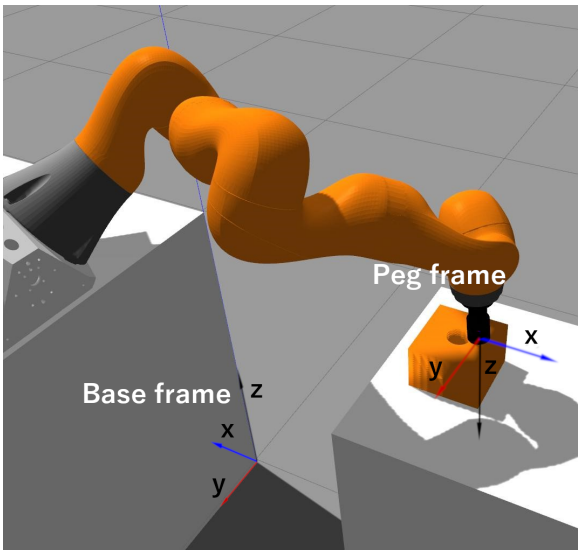

Figure 4.1 Figure shows a state close to the starting state for most of the following experiments. It also shows the two main coordinate frames standard position and orientation. *Base frame* — static world Cartesian reference frame. *Peg frame* — dynamic coordinate frame following position and orientation of the peg attached to the end-effector.

Table 4.1 Default settings of FDCC setup used in tests and measurements in Chapter 4.

Name	Notation	Default setting
Gain scaling factor	c	1.0
Internal iterations	l_{itr}	10
Translational stiffness	K_t	(x,y,z) = (100,100,100) N/m
Rotational stiffness	K_r	(x,y,z) = (5,5,5) Nm/rad
Translational P gains	$k_{p,t}$	(x,y,z) = (0.015,0.015,0.015)
Rotational P gains	$k_{p,r}$	(x,y,z) = (0.35,0.35,0.35)
Translational D gains	$k_{d,t}$	(x,y,z) = (0.0,0.0,0.0)
Rotational D gains	$k_{d,r}$	(x,y,z) = (0.0,0.0,0.0)
Virtual mass link 7	$m_{v,7}$	1.0 kg
Virtual inertia link 7	$I_{v,7}$	1.0 kg m^2
Virtual mass links 1-6	$m_{v,1} - m_{v,6}$	0.1 kg
Virtual inertia links 1-6	$I_{v,1} - I_{v,6}$	0.01 kg m^2
Force rate limit	$(\frac{\Delta f}{\Delta t})^{max}$	100 N/s
Torque rate limit	$(\frac{\Delta \tau}{\Delta t})^{max}$	10 Nm/s

4.2 External Force-Torque Sensor signal and Filters

As mentioned in Chapter 3, the force-torque sensor signals showed to have some undesired properties. In this section, the challenges and implemented solutions are explained in more detail.

Spike Values and Rate Limiter

The external joint-torque observer signal $\hat{\tau}_s$ was shown to be non-zero, and in particular spiking at start and stop of a commanded trajectory. If this signal is used as the external torque-sensor signal for the FDCC, the controller reacts to this similarly to a collision or interaction when the robot in fact is moving in free space. The rate-limiter filter described in Section 3.5 (Equation (3.10)), is used to decrease the impact of this behaviour of the observer signal, and resulting in a signal with less noise (i.e., still non-zero) that is more suitable and more predictable to use as feedback signal. The expected and observed downside of using the rate limiter is that the controller reacts with some delay to external interaction. This could be noticeable particularly when the real external interaction forces change faster than the upper limit of the rate limiter. Figures 4.2 and 4.3 show one example of the effect of the rate limiter on the Cartesian external force-torque signals $\mathbf{F}_s$ during a free-space motion. In this example, the commanded joint trajectory has a maximum angular velocity of 0.3 rad/s, and uses the high-gain position control for the internal controller on the iiwa side of the system.

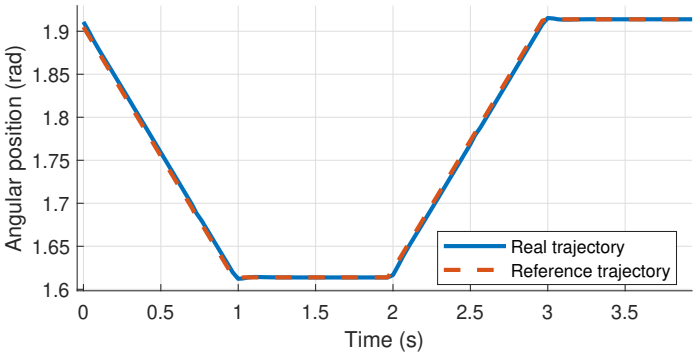

Figure 4.2 Joint-trajectory motion on the real KUKA robot, using the joint-trajectory controller and iiwa position control.

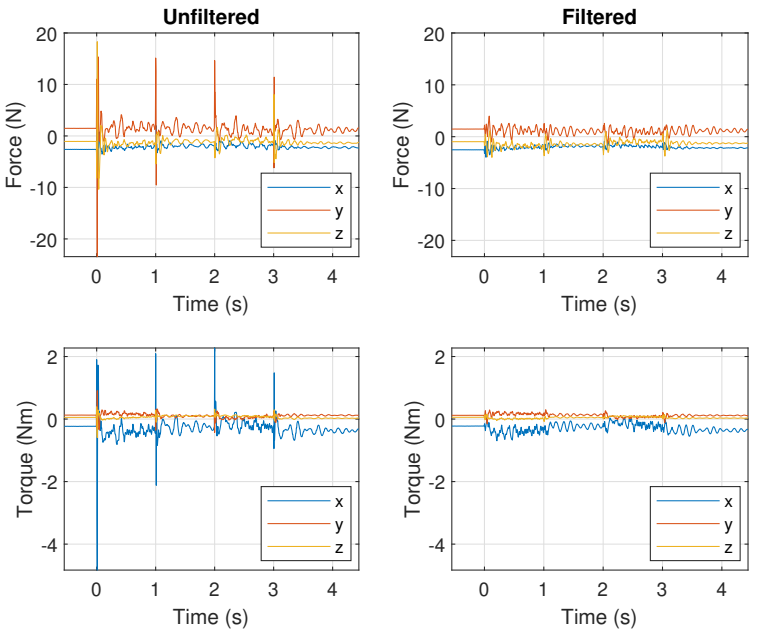

Figure 4.3 External Cartesian force-torque signals, with and without the rate limiter, during joint trajectory motion shown in Figure 4.2. This is to visualize how the external force-torque signal is affected during free-space motion. The slope limit $(\Delta \mathbf{F}/\Delta t)^{max}$ is for forces set to 100 N/s and for torques set to 10 Nm/s.

In Figure 4.3, the effect of the rate limiter is clear, where the problematic peaks in the signal have been suppressed, but with an otherwise similar signal content. A slight offset in both force and torque signals is also visible, which could suggest a calibration error as a part of the gravity-compensation process. When using the unfiltered signal as a force-torque sensor signal for FDCC, the system gets unstable, since large spikes in the sensor signal give rise to fast acceleration. Since fast acceleration gives rise to large spikes in sensor values (as shown in Figure 4.3), there is a positive feedback loop if the rate of change is not limited. Because of this, all of the following tests and measurements were using rate limiter, if the sensor signal was used in net force summation of the controller.

Frequency Analysis and Notch Filter

When the Cartesian external force-torque sensor signal $\mathbf{F}_s$ was used as a feedback to the controller (see Equation (3.1)), noise was observed to be included in the signal and the noise increases in magnitude even without a commanded trajectory and with no external forces. Example of such signals are shown in Figures 4.5 and 4.7, together with the frequency content of these signals shown in Figures 4.6 and 4.8. With a distinct frequency appearing (close to 5.5 Hz), a notch filter (as described in Section 2.9) was therefore used along all dimensions of $\mathbf{F}_s$. The Bode diagrams of the filter's transfer function is shown in Figure 4.4. The test is used as a guiding example of whether the notch filter could have the desired effect on the magnitude of the noise in the signal. Because of the system's non-linear and complex characteristics, the results from this comparison should not be taken too literally, since the response might differ depending on joint configurations and motion. Figure 4.5 shows the force-torque sensor signals $\mathbf{F}_s$ using lower gain ($c = 1.0$) for the PD controller formulation in Equation (3.3), without changing input and without external interaction. The two different signals $\mathbf{F}_s$ (left and right columns) are recorded at separate times with similar joint configuration and using the signals as the force feedback to the FDCC. Figure 4.7 shows the same procedure, but with higher gain ($c = 2.0$). The parameter settings are otherwise the same (default). Then Fast Fourier transform (FFT)([Oshana, 2006]) was used to analyze and compare the frequency content of the signals, after removing the mean.

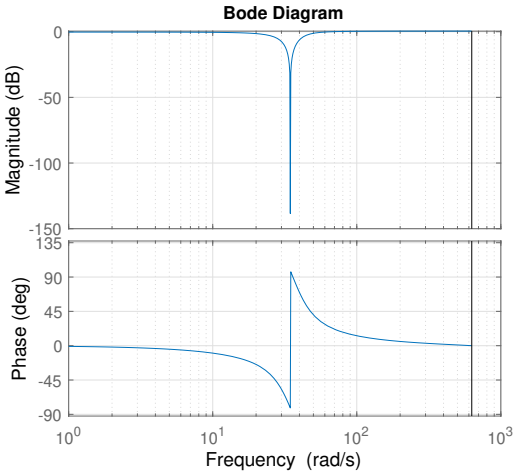

Figure 4.4 Bode plot of the notch filter transfer function $H(z^{-1})$ described in Equation (2.12). Parameters for the filter in the figure are set to $f_0 = 5.5$ Hz, $\omega_0 = 2\pi f_0$, $f_s = 200$ Hz, $r = 0.95$, and this filter is the specific filter used for all following force-torque sensor signals where the notch filter is used.

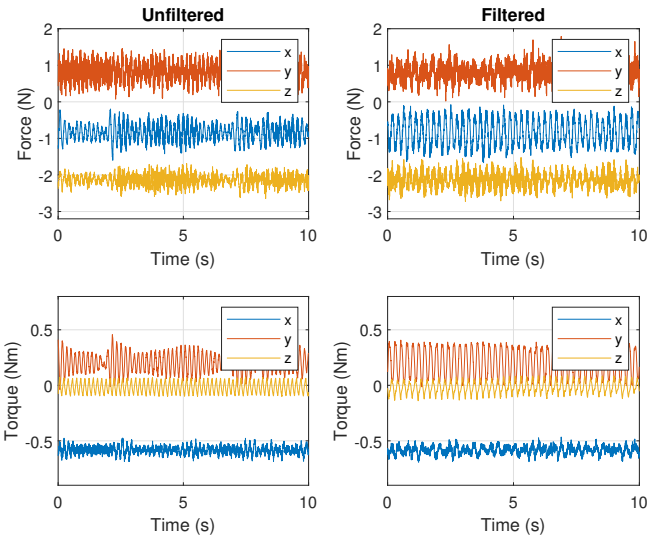

Figure 4.5 External Cartesian force signals $\mathbf{F}_s = (f_x, f_y, f_z, \tau_x, \tau_y, \tau_z)$, both with and without real-time notch filter, recorded at separate times. The gain scaling factor used is the default, $c = 1.0$.

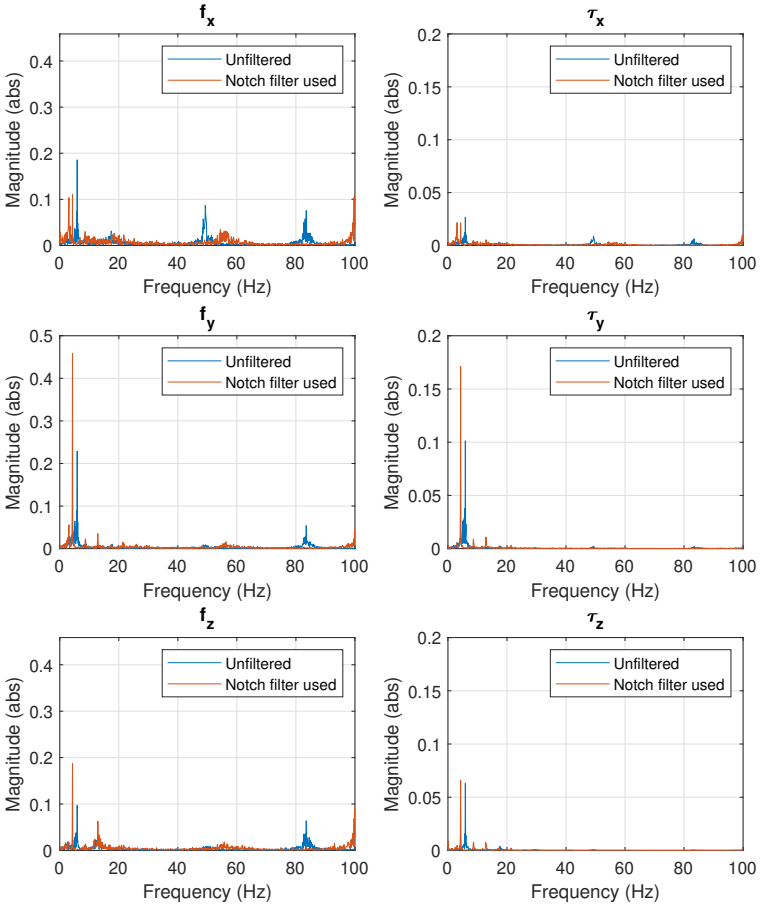

Figure 4.6 Fast Fourier Transform (FFT) of filtered and unfiltered external Cartesian force signals from Figure 4.5, where $c = 1.0$. Before using FFT on the data, the mean was subtracted from the signals. The unfiltered signal has a frequency peak close to 5.5 Hz along each dimension. The filtered signals have frequency peaks at slightly lower frequencies (approximately 4.5 Hz). However, for both y- and z-components, the peaks are unfortunately greater in magnitude than the unfiltered signals' frequency peaks.

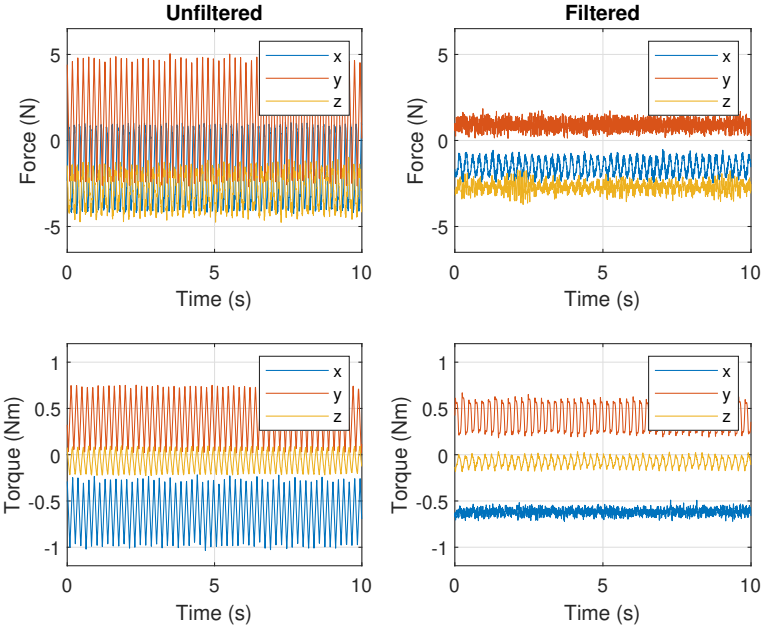

Figure 4.7 External Cartesian force signals $\mathbf{F}_s = (f_x, f_y, f_z, \tau_x, \tau_y, \tau_z)$, both with and without real-time notch filter, recorded at separate times. The gain scaling factor was used twice the default, $c = 2.0$. The noise magnitude is clearly greater along each dimension of the unfiltered signals than those of the filtered signals.

It seems that the system with the default settings has properties that amplify frequencies close to 5.5 Hz, and that increased gain further amplifies those frequencies. The implemented notch filter performs well at canceling those frequencies, which can be seen in Figures 4.7-4.8. However, the filtering also results in new low-frequency content. Therefore, the notch filter could in general be assumed to be better suited for higher-gain settings.

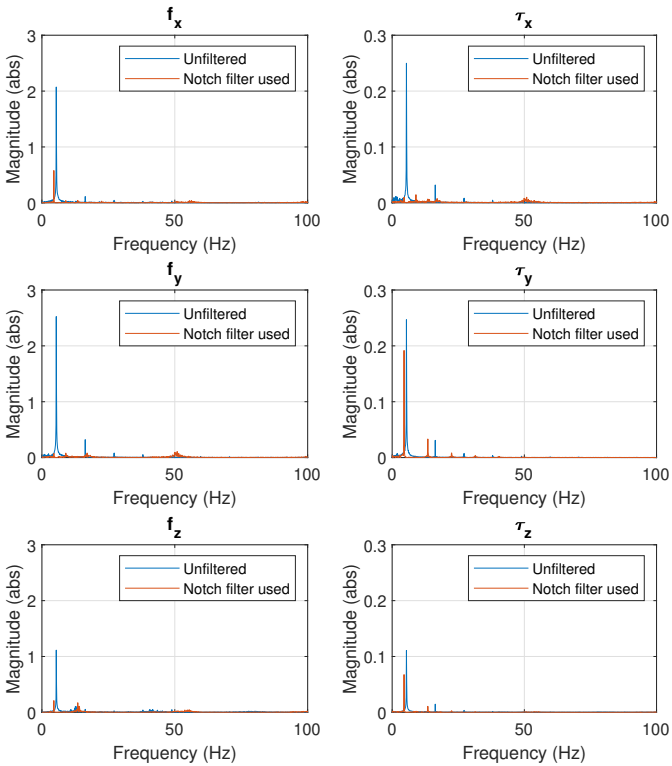

Figure 4.8 Fast Fourier Transform (FFT) of filtered and unfiltered external Cartesian force signals from Figure 4.7, where $c = 2.0$. The filter used is the notch filter described in Equation (2.12). Before using FFT on the data, the mean is subtracted from the signals. This figure shows that the unfiltered signal has frequency content focused close to 5.5 Hz. The frequency content for the filtered signals has in general lower magnitude.

4.3 Control Performance

Force Control

In this part, the performance of controlling forces at the end-effector is presented. As described in Chapter 3 and seen in Figure 3.2, the desired force $\mathbf{F}_d$ is the input used for these tests. The procedure of the test is the following:

- Start position with end-effector just above surface, using low translational stiffness settings along the Cartesian z-direction.
- Command desired force $\mathbf{F}_d$: +5 N along positive z-direction in the peg frame.
- Examine external force signal $\mathbf{F}_s$, indicating interaction forces with the surface.

The external force-torque signal $\mathbf{F}_s$ was used to evaluate how well the resulting Cartesian forces from the end-effector match the desired forces $\mathbf{F}_d$. The desired forces $\mathbf{F}_d$ are measured in what forces the end-effector will exert on the environment in the frame of the end-effector. Therefore, if there are interaction forces, this will be seen in the force-torque sensor signal with opposite sign of the desired forces. Since there are no dedicated external force sensors or other way to more accurately evaluate interaction forces, the estimated force-torque sensor signal is the prefered way to compare the real and simulated systems' force-control behaviour with a similar procedure and measurements. Stiffness settings for both real and simulated tests:

- Translational stiffness, K_t: (x,y,z) = (300,300,10) N/m
- Rotational stiffness, K_r: (x,y,z) = (30,30,30) Nm/rad

The gain scaling factor was set individually for the real and simulated tests:

- Simulation: $c = 0.8$
- Real: $c = 1.0$

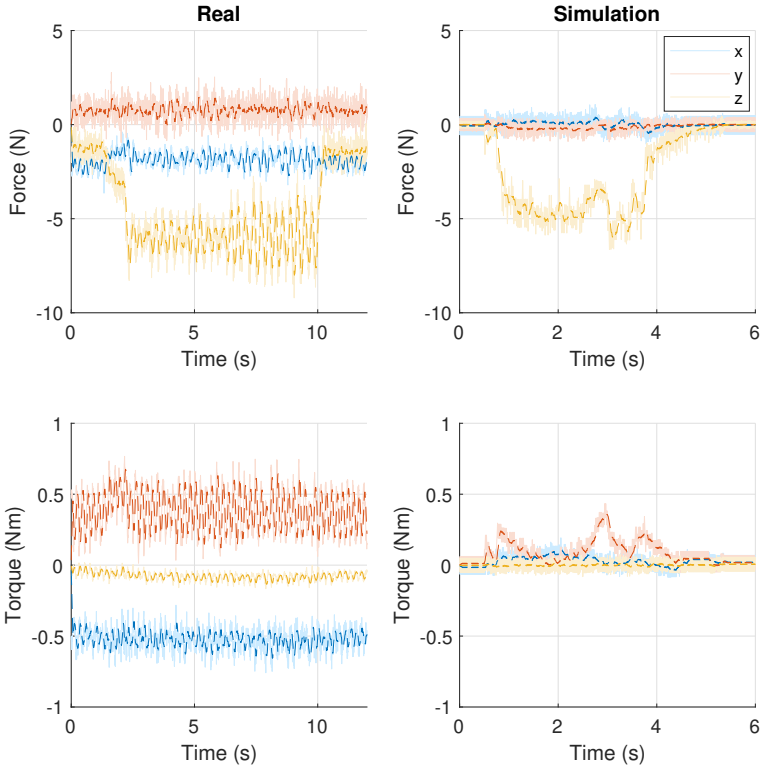

Figure 4.9 External Cartesian force-torque signals $\mathbf{F}_s = (f_x, f_y, f_z, \tau_x, \tau_y, \tau_z)$, during force-control test in simulation. Shaded lines are the signals used for feedback. The dashed line shows the moving average. The offset of sensor signals for the real robot is because of calibration errors of the iiwa joint-torque measurements, or by uncertainty in the estimation of end-effector force-torque signal.

Figure 4.9 shows the resulting force-torque sensor signals during the task of controlling interaction forces. +5 N was commanded to be exerted by the end-effector along z-direction (see Figure 4.1 for orientation of frame) and the sensed forces from the force-torque sensor signals show approximately -5 N, i.e., corresponding force the end-effector experience when exerting +5 N to the contact surface. For the real robot, the noise has a higher magnitude and with offset values. The offset is expected to be mainly because of calibration errors of the iiwa joint torque measurements, but could also be a result of the uncertainty of the external joint-torque estimation or the transformation of it (see Equation (3.7)).

Cartesian Position Tracking

To evaluate Cartesian position tracking with the FDCC approach, a linear trajectory was commanded along each Cartesian direction, separately. The trajectory starts from the current Cartesian pose, and ends 0.1 m from the start position. The external force-torque signal $\mathbf{F}_s$ of the real system has previously been identified as non-zero for free-space motion. Therefore, the trajectories were performed both with and without force feedback. The tests were done both in simulation and with the real robot, and results are presented in Figure 4.10. The stiffness settings for both real and simulated tests were:

- Translational stiffness, K_t: (x,y,z) = (500,500,500) N/m
- Rotational stiffness, K_r: (x,y,z) = (50,50,50) Nm/rad

The gain scaling factor was set individually for real and simulated tests:

- Simulation: c = 0.3
- Real: c = 1.6

In Figure 4.10, it can be seen that all performed trajectories show matching response to the commanded trajectory, with the exception of the real robot trajectory using force feedback (green). It shows slight deviation or even stationary error, which are both explained by the non-zero force-torque sensor signal previously observed for the real system. The sensor signal is previously seen to be changed in value during a free-space motion, in addition to the stationary offset previously explained by calibration errors. The errors are still fairly small, and as impedance controllers are generally not very accurate on position trajectories the performance is considered good enough. The large difference of the gain scaling factor c, was through tuning produced to get this accurate match of real and simulated response to a commanded Cartesian trajectory. It is expected to be affected by the controller update-rate difference for the real and simulated system. Since the controller is using the virtual model inside the controller to update joint states, the difference of a free-space motion of the real robot and in simulation should not be significant for the same controller settings. However, with the simulation using an update rate of 1000 Hz and the real robot 200 Hz, an approximately five times slower response for the real robot could be expected with this update-rate difference. Therefore, the approximately five times lower gain in simulation ($c^{sim} = 0.3, c^{real} = 1.6$) could compensate for the update-rate difference.

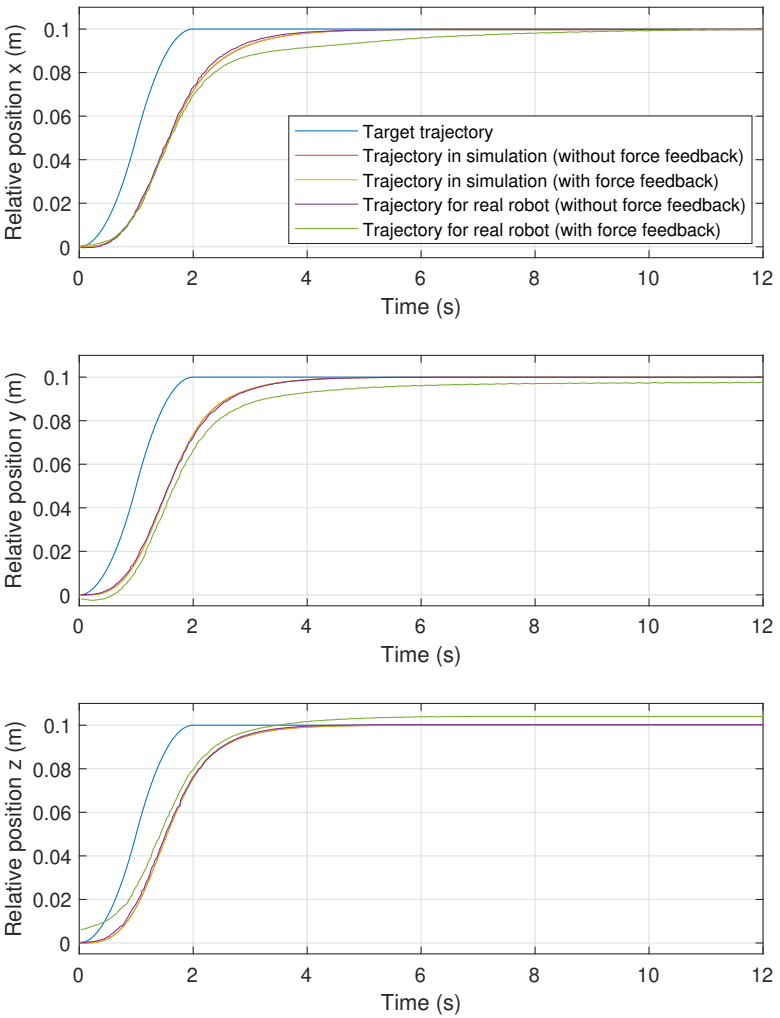

Figure 4.10 Commanded trajectories along x, y and z directions, separately executed. The response of simulation and for the real robot are compared, as well as with and without force-sensor feedback sent to the controller.

4.4 Physical Human-Robot Interaction (pHRI)

Interaction Test A — Kinesthetic Teaching and Compliance

Kinesthetic teaching is a widely used method for programming manipulators to solve assembly tasks [Muxfeldt et al., 2014]. This method calls for smooth, responsive and intuitive interaction behaviour. Figure 4.11 shows the test process of the robot with low stiffness being guided to the two goal positions indicated by the green cylinders on the table. This mimics, for example, a pick-and-place task where the Cartesian positions are taught. The focus is on the qualitative analysis of the response and motion of the robot during interaction. Figure 4.12 shows joint angular positions and velocities, respectively, during the experiment illustrated in Figure 4.11. Figure 4.12 also shows Cartesian external forces and torques on the end-effector $\mathbf{F}_s$. The stiffness settings for Interaction Test A were:

- Translational stiffness, K_t: (x,y,z) = (10,10,10) N/m
- Rotational stiffness, K_r: (x,y,z) = (2,2,2) Nm/rad

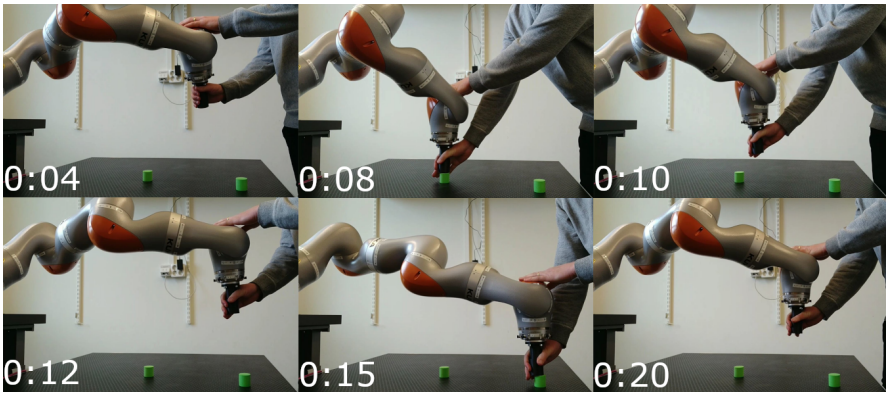

Figure 4.11 Snapshots of Interaction Test A, with the robot using low stiffness settings. Joint states $\mathbf{q}, \dot{\mathbf{q}}$ and external forces on the end-effector $\mathbf{F}_s$ during this process are shown in Figure 4.12. Time-stamps in the snapshot images indicate approximate corresponding points along the timeline of Figure 4.12.

After the first goal point was reached (0:08), the starting joint states are approached again, which can be seen in the upper-left graph over joint angular positions in Figure 4.12. The joint angular velocities show main changes during externally induced Cartesian motion, but some oscillations were observed at the impact at the second goal point (0:15), where joint 6 (light blue) seems to show the most noticeable variations and rapid change of velocity. At the same point in time, the force-torque sensor graphs also show faster oscillations. The z direction (yellow)

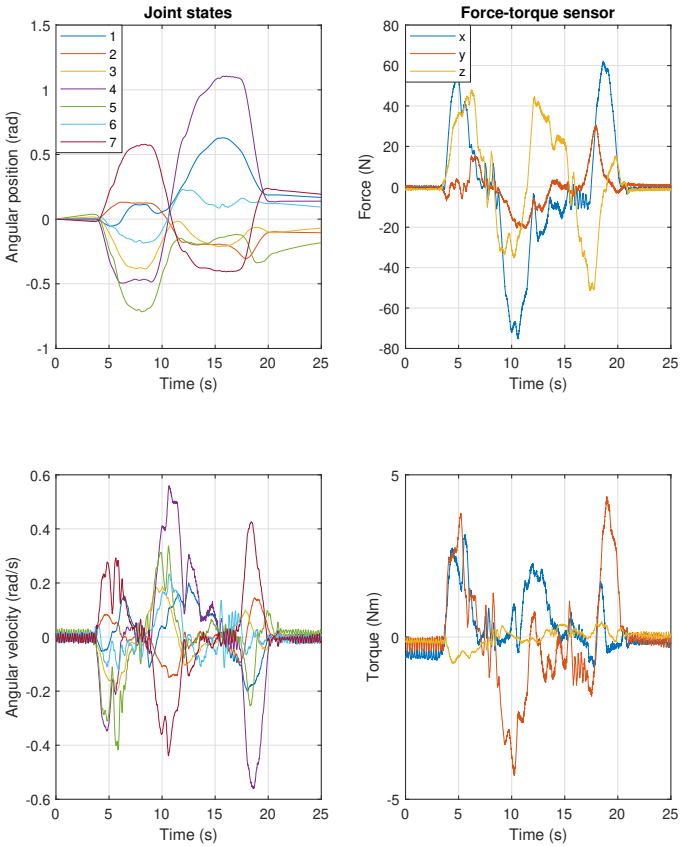

Figure 4.12 The figure shows joint states (joint angular position $\mathbf{q}$ and joint angular velocity $\dot{\mathbf{q}}$) during Interaction Test A. Zero radians is defined as the starting angular position for each joint. To the right is the force-torque sensor signal $\mathbf{F}_s = (f_x, f_y, f_z, \tau_x, \tau_y, \tau_z)$, which shows the human interaction in the frame of the end-effector.

of the force-sensor graph in the upper-right plot shows where the pulling up and pushing down is exerted using external forces. The sensor is displayed in the frame of the end-effector, which means that positive force corresponds to force pushing towards the table, and negative force to pulling away.

The interaction with the robot was smooth and the system felt stable. Applying force to the robot resulted in an intuitive response of motion, and the interaction

experience was characterized by the fact that it was reactive in each joint, as a result of the external joint-torque observer. Even though the external joint-torque observer signal is transformed to work in the Cartesian end-effector frame, the robot joints feel compliant with this low stiffness setting. This led to being able to decide the direction of which the elbow joint should be folded, through the use of externally applying forces at intuitive parts of the robot arm. This would not be possible for a robot arm which would only use a dedicated end-effector force-torque sensor. For such a manipulator, the rest of the joints are completely stiff to external interaction. This finding would also indicate that this FDCC implementation using external joint torque (such as the iiwa) to implement compliance, would handle collisions on the whole arm in a way that manipulators with only dedicated end-effector force-torque sensors never could, when using the same controller.

Interaction Test B — Singularity Test

As a part of interaction and compliance testing, the real system was moved by hand into a joint configuration close to or at a singularity. Figure 4.13 shows the test process in images. The stiffness settings for interaction test B were:

- Translational stiffness, K_t: (x,y,z) = (10,10,10) N/m
- Rotational stiffness, K_r: (x,y,z) = (2,2,2) Nm/rad

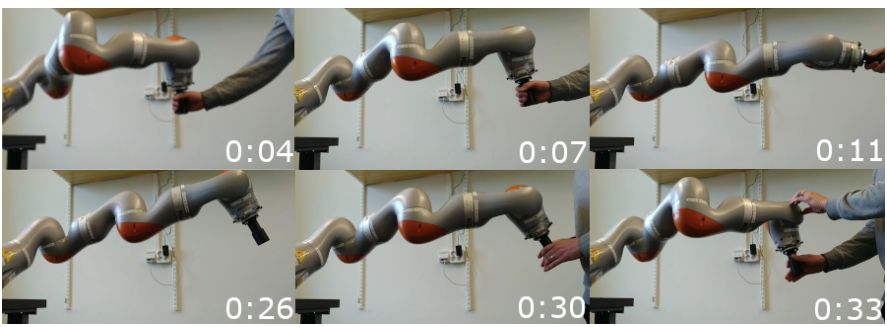

Figure 4.13 Snapshots of Interaction Test B, with the robot using low stiffness settings. Joint states $\mathbf{q}, \dot{\mathbf{q}}$ and external forces on the end-effector $\mathbf{F}_s$ during this process are shown in Figure 4.14. Time-stamps in the snapshot images indicate approximate corresponding points in the timeline of the graphs in Figure 4.14.

Compared to interaction test A, interaction and feel to the robot was similar, except for when the robot approached a singularity when stretching it out. At that point, the robot was released and left to observe it while it oscillated on its own. It also slowly re-adjusts the orientation (motion of joint 6 in the upper-left graph in Figure 4.14), due to the non-zero (still low) stiffness settings. It does not seem to be

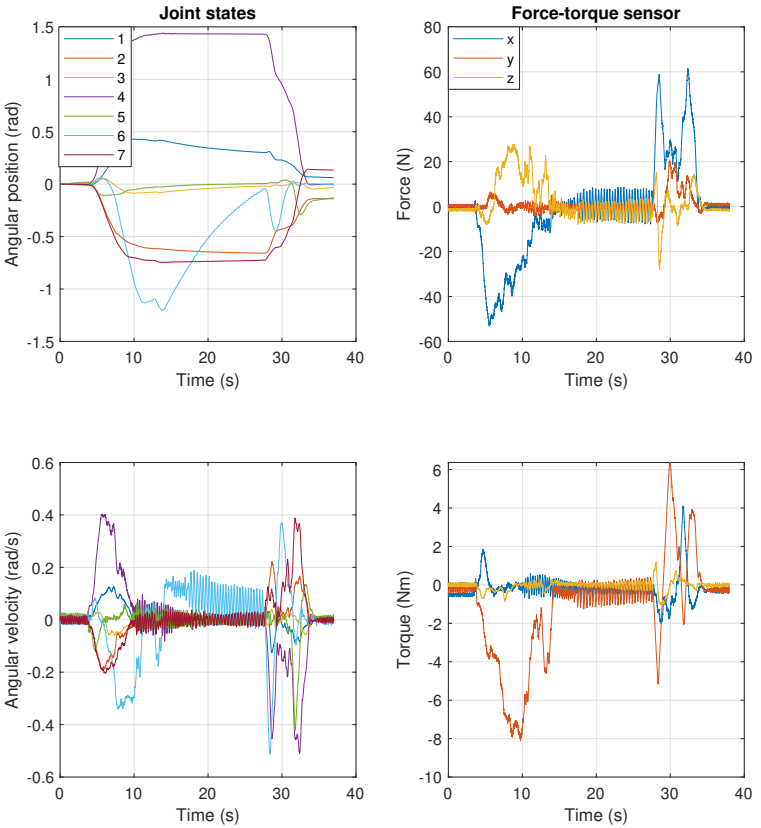

Figure 4.14 The figure shows joint states (joint angular position $\mathbf{q}$ and joint angular velocity $\dot{\mathbf{q}}$) during Interaction Test B. Zero radians is defined as the starting angular position for each joint. To the right is the force-torque sensor signal $\mathbf{F}_s = (f_x, f_y, f_z, \tau_x, \tau_y, \tau_z)$, which shows the human interaction in the frame of the end-effector.

able to readjust for the translational Cartesian error, but this is expected behaviour close to a singularity. The oscillations do not seem to change in magnitude, but rather continued in an undamped manner. The contact was then reinitiated and the oscillations are visible and noticeable when touching the robot, but the forces acting to produce the oscillations feel small. At the end, the robot was moved out of the singularity with ease by hand, back to the initial Cartesian pose.

4.5 Peg-in-Hole Task

Peg-in-hole assembly is a common task in industrial environments, which often relies on physical contact and force control. The goal of the task is to insert a peg through contact with the surroundings of the hole. This testing was done to combine previously isolated experiments (path tracking and contact motion), and evaluate it in a qualitative manner. The circumstances and procedures of this test were simplified. The Cartesian position inside the hole was known, and only the end-effector reference position $\mathbf{x}_d$ was changed for a successful insertion. A spiral motion was commanded once the peg is close to the hole, which in turn resulted in a search motion while also overcoming friction forces. The test gave a good idea of how the setup and controller could perform in a more generalized peg-in-hole insertion task, or in a reinforcement learning process. It is worth noting that this is only one example out of many tasks that can be solved with a Cartesian impedance-based controller, and similarly Cartesian impedance controller is only one example of many controller solutions that can solve similar tasks.

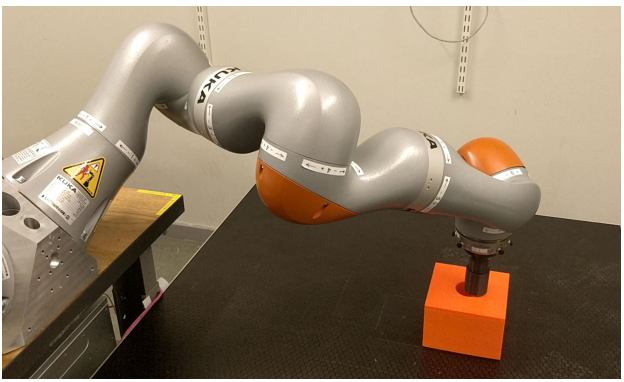

Figure 4.15 Robot during search motion of peg-in-hole task.

In this test, a stiff cylindrical peg with diameter of 45 mm was attached to the end-effector link of the robot setup. The goal was a hole with a diameter of 50 mm, which gave the peg a clearance of 5 mm. The stiffness settings for both real and simulated tests were:

- Translational stiffness, K_t: (x,y,z) = (700,700,100) N/m
- Rotational stiffness, K_r: (x,y,z) = (100,100,100) Nm/rad

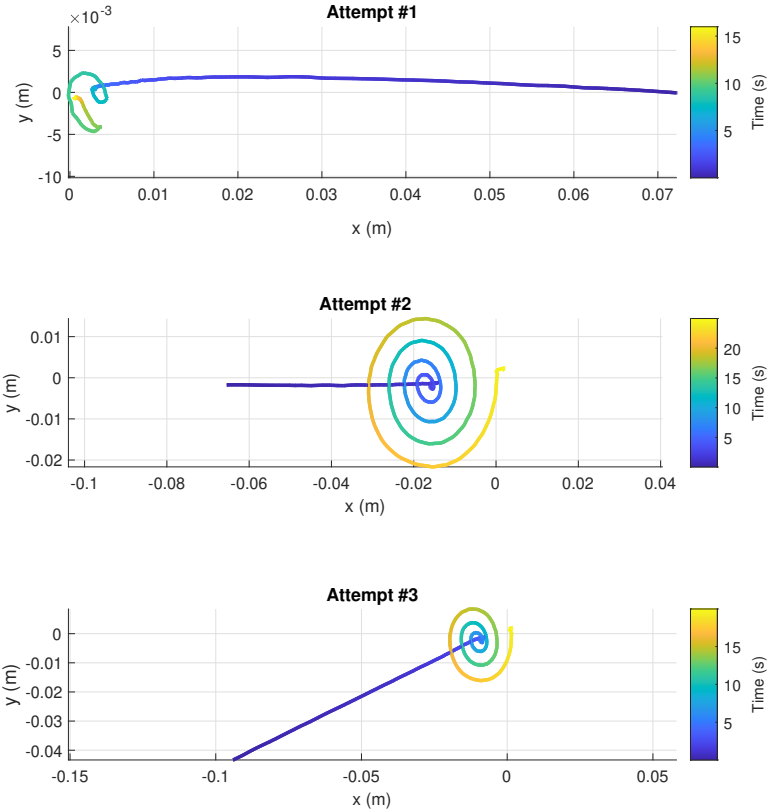

Figure 4.16 Position of end-effector in the xy-plane for the real robot during the peg-in-hole task. The plots show detailed motion during each insert attempt, also displayed in Figure 4.18. The color code of the graphs indicates the passing of time, using start time 0 s to indicate the beginning of each attempt.

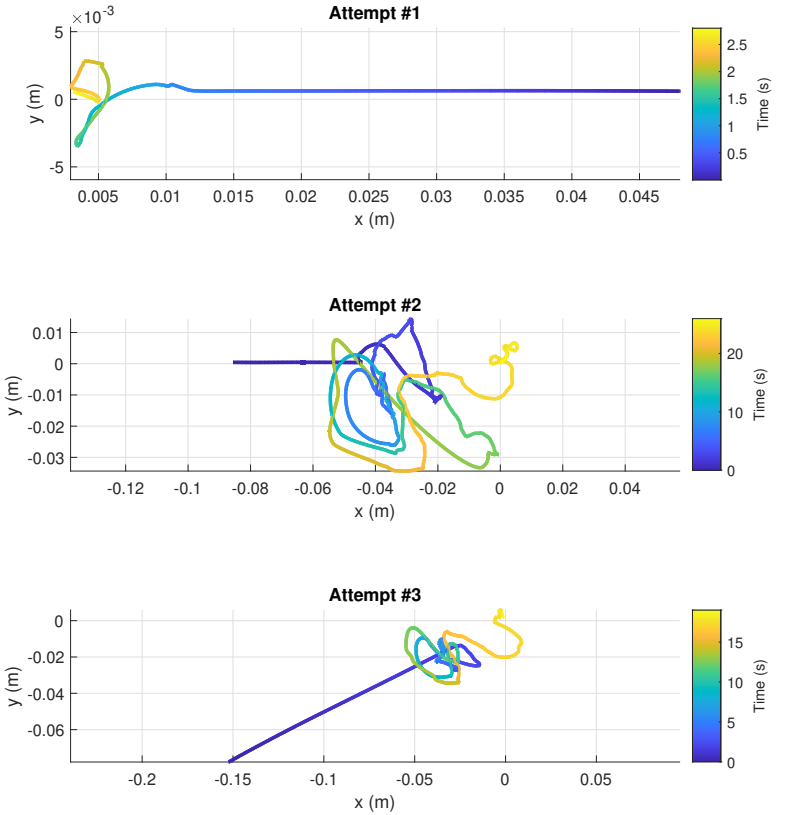

Figure 4.17 Position of end-effector in the xy-plane for the simulated system during the peg-in-hole task. The plot shows detailed motion during each insert attempt, also displayed in Figure 4.19. The color code of the graphs indicates the passing of time, using start time 0 s to indicate the beginning of each attempt.

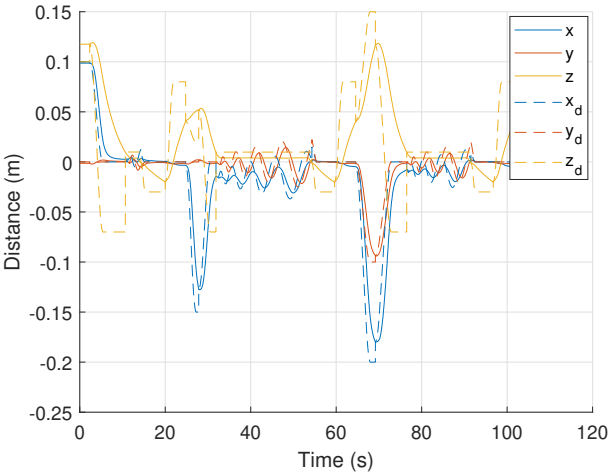

Figure 4.18 Position of end-effector of the simulated robot during a sequence of three peg-in-hole insertion tests. The first attempt starts at 2 s, the second at 28 s and the third attempt starts at 69 s. The figure shows distance from the center of the hole along each direction (full lines), as well as corresponding commanded position (dashed lines) at each point in time.

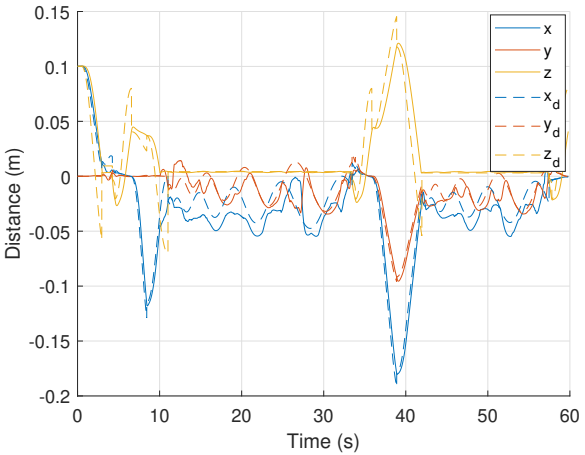

Figure 4.19 Position of end-effector of the simulated robot during a sequence of three peg-in-hole insertion tests. The first attempt starts at 0 s, the second at 8 s and the third attempt starts at 39 s. The figure shows distance from the center of the hole along each direction (full lines), as well as corresponding commanded position (dashed lines) at each point in time.

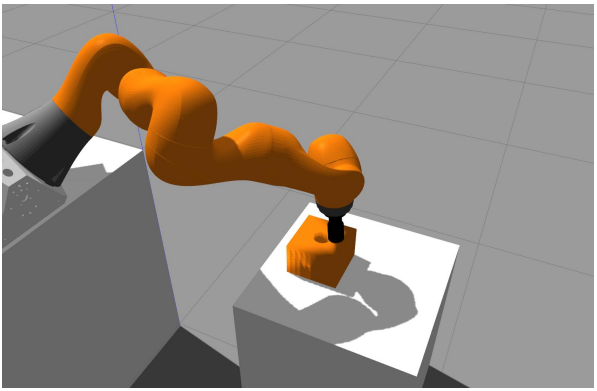

Figure 4.20 KUKA robot arm in Gazebo during peg-in-hole task.

The results of the peg-in-hole insertion present consistent successful results of the task, both for the real and simulated system. For the real robot, the motion was following the reference trajectory with a slight lag, possibly due to the lower update rates of the real robot. It can still provide a nice spiral motion in the xy-plane as desired, which theoretically should radially cover a surface consistently. For the simulated system, the graphs in Figure 4.17 show a seemingly chaotic end-effector path in xy-plane, and not very similar to the desired spiral motion seen in the real robot case (see Figure 4.16). However, each attempt in simulation was also successful. Some improvement is needed for simulating the sliding motion; either by tuning controller parameters or by finding a different friction model for simulation.

5

Discussion

This chapter discusses the findings on stability and performance from the experiments in Chapter 4 and a comparison to the thesis objectives is made. Furthermore, the chapter contains suggestions and ideas for future work.

5.1 Objective and Contribution

The objectives stated in the introductory part of this thesis were:

- Find suitable Cartesian controller solution to work for both the real and simulated system.
- Implemented controller should handle unknown or dynamic environments and physical human-robot interaction.
- Evaluate and tune the controller such that the simulated system is accurately following the behaviour and response of the real system.

In this thesis, it was concluded that the FDCC could be a suitable choice. The choice was based on the detailed library documentation, as well as the generality and usage of ROS in the controller development, which facilitates a parallel implementation for a simulated system. Even though this solution was previously tested on other manipulators by the developers of the controller, this project is the first known to implement it on a 7-DOF manipulator. In the documentation of the FDCC library, it also states that it requires an end-effector force-torque sensor. Since this is not attached to the KUKA LBR iiwa robot arm, a solution was developed through transforming and processing of the external joint-torque observer signals provided by the iiwa framework. In general, this thesis has broadened the range of manipulators of which the FDCC could be a viable controller solution. Additionally, the project has given increased insights of the challenges and solutions for accurate simulation of robot force control. The evaluation process consisted of a number of qualitative and quantitative experiments that provided desirable results. These included tests with external interaction to assess reasonable behaviour for contact-rich tasks and

response to a dynamic environment. As a result of an extensive tuning process, there was a good match between the simulated and the real system response.

5.2 Stability

Force-Torque Signal and Rate Limiter

One of the challenges of this thesis has been to handle the external force-torque signal that is needed for the FDCC framework. Without using the force-sensor feedback, the implementation of the controller worked in a "power-on-and-go" manner—the way the developers of the controller intended. However, the system was observed to be unstable when using force-sensor feedback based on estimated joint-torques. Therefore, the rate limiter was needed to avoid feedback of signal spikes that were not induced by external contact.Even though the rate limiter enabled higher gain of the controller, there would still be limitations on controller gain to avoid visible oscillations in joints and thereby end-effector. As an effect of the rate limiter, the controller would also not identify external interactions correctly, if they have a slope greater than the upper limit of the rate limiter. However, this was not observed as a problem in the interaction experiments, as the interactions and contacts were generally not very sudden or large in magnitude. Instead, the main concern about the effects of the rate limiter was rather that it could have induced perceptible latency or lag in the robot response. During pHRI testing, the response of the robot could sometimes feel a bit slow, but similar to a sense of inertia of the end-effector. It was difficult to evaluate whether this was an inherent part of the controller, or either induced by the rate-limit filter or by the limitations on the controller gains. However, for the interaction done with the real robot in this project, the response was smooth and intuitive. Joint configurations close to singularities could give rise to unwanted oscillations, but with no apparent risk of getting into an unstable state on its own. Dedicated joint control could be implemented to be able to avoid such problems induced by singularities.

System Oscillations

It is not determined what the origin of the low-frequency oscillations in the real system is. However, there is a hypothesis that the robot mount and table of which it stands on could be one source of the self-oscillatory behaviour. This was not further investigated, but the idea comes from that the table is not completely stiff, and that when the oscillations in the robot became large enough, the table would also start vibrating which could have resulted in the positive feedback behaviour noticed in the initial tests of the project. Another possible source of the oscillations could be time delays in the system. Even though the actual delay was not investigated, it is expected to be at least a couple of ms just by the structure of the system flow shown in Figure 3.2, using ROS and the iiwa software. An example of the impact

the system time-delay has on stability, can be found in the previously mentioned [Lawrence, 1988] regarding impedance-control implementations.

5.3 Task Performance

As a controller implementation for a collaborative robot arm, the tests showed desirable results. Cartesian position tracking showed good results both for the real and simulated robot, with a tracking lag as expected because of the nature of an impedance-based controller. A stationary offset of a few millimeters was occasionally observed with the real robot. This is argued to be due to the calibration error or estimation uncertainty of the external joint-torque observer, which led to an offset in the force-torque sensor signal. Force-control experiments also showed promising results, with similar characteristics for both the real and simulated robot. However, the collision and friction model in the simulation could be investigated further to make this interaction response more predictable and close to reality. It is possible that the physics engine ODE is not preferred for this type of simulation. As an example, the sliding motion element of the peg-in-hole task showed to have different motion characteristics in simulation. Otherwise, the peg-in-hole task showed successful insertion from different start positions.

5.4 Future Work

This thesis resulted in a successful implementation and controller solution of a simulated and real 7-DOF robot arm. However, there are a couple of topics that could be interesting for future work to potentially increase stability and responsive behaviour. To begin with, the filtering process and the use of the external joint-torque signals provided by the iiwa framework could be explored further. If the noise of the signal could be reduced, it should enable increased gain of the controller to increase responsiveness and accuracy of desired forces. The development in [Chawda and Niemeyer, 2017] could be an interesting starting point for another filtering approach. The use of a moving-average filter could be a simple alternative filter implementation.

Moreover, the inclusion of damping in the impedance-controller part could be interesting to examine, because of the theoretical energy dissipating effect. If that would be the case, it could possibly decrease system oscillations or at least have a damping effect.

In this project, a lot of time and effort was put into tuning for both the real and simulated system. One interesting aspect to look into further is the difference of update rates between the real and simulated system. For this thesis, it was not investigated to match the update rates of the systems, but this could result in a closer match of Cartesian tracking. It would at least be expected to be a less tuning-demanding

process in getting an accurate simulation of the real system. To further reduce the effort needed for the tuning process, it could be beneficial for future implementations using this controller to develop a structured way for tuning of controller parameters based on sensor signals, robot dynamics or simulation-specific properties. This could possibly even be automated to some extent by the use of system identification to for example shape interaction forces in a desired way.

As the Cartesian controller library provides an intuitive way of performing tasks in Cartesian space, addition of null-space control in this controller would be very useful, for example, for performing motion along Cartesian reference trajectories that span over a large part of the workspace. Self-collisions or external collisions of non-end-effector parts of the manipulator are often also a pressing issue, and require dedicated joint-space control. One specific example where null-space control can be advantageous is where the desired Cartesian reference can be achieved with multiple elbow configurations, where one of the configurations would result in collision or a singularity. As the FDCC is developed currently, only Cartesian end-effector position is considered, and therefore the resulting joint configurations from Cartesian reference points can be difficult to foresee. Null-space control can therefore be implemented to increase dexterity and enable more independent motion in larger parts of the workspace.

6

Conclusions

This thesis has investigated and implemented a controller solution for a collaborative robot arm based on impedance control, together with an implementation for accurate simulation of the real system. A motivation for the thesis was to investigate possibilities of a controller, which computes joint position or velocity commands to the robot, in the perspective of the large variety of robots that can be commanded with such signals. In addition, the controller should also introduce compliance in Cartesian space to facilitate contact-rich tasks.

The controller solution of choice is based on the Cartesian controller library called 'Forward Dynamics Compliance Controller (FDCC)', and was implemented to be used on an KUKA iiwa robot arm. Together with the physics engine ODE, Gazebo was used as the simulation environment to mimic the behaviour of the real robot system. Matching response and characteristics of the simulated and real system was achieved through tuning of control parameters and processing of sensor signals. The implementation resulted in stable behaviour for contact-rich tasks and during physical Human-Robot Interaction. As the force-torque sensor signals of the simulated and the real system have separate origin and properties, the match between simulation and real performance seemed to be the most difficult to achieve. Moreover, collision and friction models using the adopted simulation tools also differ from the real behaviour. As an example, the sliding-motion task is difficult to perform with consistent results in simulation. However, both the real and simulated system with the implemented controller solution gave consistent successful results for the peg-in-hole task.

Bibliography

Andrychowicz, M., B. Baker, M. Chociej, R. Józefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba (2020). "Learning dexterous in-hand manipulation". *The International Journal of Robotics Research* **39**:1, pp. 3–20. DOI: 10.1177/0278364919887447.

Aristidou, A., J. Lasenby, Y. Chrysanthou, and A. Shamir (2018). "Inverse kinematics techniques in computer graphics: A survey". *Computer Graphics Forum* **37**, pp. 35–58. DOI: 10.1111/cgf.13310.

Chawda, V. and G. Niemeyer (2017). "Toward torque control of a KUKA LBR iiwa for physical human-robot interaction". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (24-28 Sept. - Vancouver, BC, Canada)*, pp. 6387–6392. DOI: 10.1109/IROS.2017.8206543.

Chiaverini, S., G. Oriolo, and I. D. Walker (2008). "Kinematically redundant manipulators". In: Siciliano, B. et al. (Eds.). *Springer Handbook of Robotics*. Springer, Berlin, Heidelberg, pp. 245–268. ISBN: 978-3-540-30301-5. DOI: 10.1007/978-3-540-30301-5_12.

Collins, J., S. Chand, A. Vanderkop, and D. Howard (2021). "A review of physics simulators for robotic applications". *IEEE Access* **vol. 9**, pp. 51416–51431. DOI: 10.1109/ACCESS.2021.3068769.

De Luca, A., A. Albu-Schäffer, S. Haddadin, and G. Hirzinger (2006). "Collision detection and safe reaction with the DLR-III lightweight manipulator arm". In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (9-15 Oct - Beijing, China)*, pp. 1623–1630. DOI: 10.1109/IROS.2006.282053.

Haddadin, S., A. De Luca, and A. Albu-Schäffer (2017). "Robot collisions: A survey on detection, isolation, and identification". *IEEE Transactions on Robotics* **33**:6, pp. 1292–1312. DOI: 10.1109/TRO.2017.2723903.

Hogan, N. (1985). "Impedance Control: An Approach to Manipulation: Part I—Theory". *Journal of Dynamic Systems, Measurement, and Control* **107**:1, pp. 1–7. ISSN: 0022-0434. DOI: 10.1115/1.3140702.

Lawrence, D. A. (1988). "Impedance control stability properties in common implementations". In: *Proceedings 1988 IEEE International Conference on Robotics and Automation (24-29 April - Philadelphia, PA, USA)*. Vol. 2, pp. 1185–1190. DOI: 10.1109/ROBOT.1988.12222.

Lin, S. T. (1997). "Force sensing using Kalman filtering techniques for robot compliant motion control". *Journal of Intelligent Robotic Systems* **18**:1, pp. 1–16. DOI: 10.1023/A:1007946400645.

Muxfeldt, A., J.-H. Kluth, and D. Kubus (2014). "Kinesthetic teaching in assembly operations – a user study". In: Brugali, D. et al. (Eds.). *Simulation, Modeling, and Programming for Autonomous Robots*. Springer International Publishing, Cham, pp. 533–544. ISBN: 978-3-319-11900-7.

Oshana, R. (2006). "4 - overview of digital signal processing algorithms". In: Oshana, R. (Ed.). *DSP Software Development Techniques for Embedded and Real-Time Systems*. Embedded Technology. Newnes, Burlington, pp. 59–121. ISBN: 978-0-7506-7759-2. DOI: https://doi.org/10.1016/B978-075067759-2/50006-5.

Parra-Vega, V., A. Rodiguez-Angeles, S. Arimoto, and G. Hirzinger (2001). "High precision constrained grasping with cooperative adaptive handcontrol". *Journal of Intelligent and Robotic Systems - JIRS* **32**, pp. 235–254. DOI: 10.1023/A:1013987209547.

Raibert, M. H. and J. J. Craig (1981). "Hybrid Position/Force Control of Manipulators". *Journal of Dynamic Systems, Measurement, and Control* **103**:2, pp. 126–133. ISSN: 0022-0434. DOI: 10.1115/1.3139652.

Scherzinger, S., A. Roennau, and R. Dillmann (2017). "Forward Dynamics Compliance Control (FDCC): A new approach to Cartesian compliance for robotic manipulators". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (24-28 Sept - Vancouver, BC, Canada)*, pp. 4568–4575. DOI: 10.1109/IROS.2017.8206325.

Scherzinger, S., A. Roennau, and R. Dillmann (2019). "Inverse kinematics with forward dynamics solvers for sampled motion tracking". In: *2019 19th International Conference on Advanced Robotics (ICAR) (2-6 Dec - Belo Horizonte, Brazil)*, pp. 681–687. DOI: 10.1109/ICAR46387.2019.8981554.

Siciliano, B. and L. Villani (2000). "From indirect to direct force control: A roadmap for enhanced industrial robots". *Invited Paper, Robótica*.

Sutton, R. S. and A. G. Barto (2018). *Reinforcement Learning: An Introduction*. MIT press - Cambridge, MA.

Tseng, C.-C. and S.-C. Pei (2001). "Stable IIR notch filter design with optimal pole placement". *IEEE Transactions on Signal Processing* **49**:11, pp. 2673–2681. DOI: 10.1109/78.960414.

Varin, P., L. Grossman, and S. Kuindersma (2019). "A comparison of action spaces for learning manipulation tasks". *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (3-8 Nov. - Macau, China)*, pp. 6015–6021.

Xu, X., D. Zhu, H. Zhang, S. Yan, and H. Ding (2019). "Application of novel force control strategies to enhance robotic abrasive belt grinding quality of aero-engine blades". *Chinese Journal of Aeronautics* **32**:10, pp. 2368–2382. ISSN: 1000-9361. DOI: https://doi.org/10.1016/j.cja.2019.01.023.

Zhou, H., S. Ma, G. Wang, Y. Deng, and Z. Liu (2021). "A hybrid control strategy for grinding and polishing robot based on adaptive impedance control". *Advances in Mechanical Engineering* **13**:3. DOI: 10.1177/16878140211004034.

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER'S THESIS

Date of issue
June 2021

Document Number
TFRT-6142

Author(s)
Joel Holmesson

Supervisor
Matthias Mayr, Dept. of Computer Science, Lund University, Sweden
Björn Olofsson, Dept. of Automatic Control, Lund University, Sweden
Anders Robertsson, Dept. of Automatic Control, Lund University, Sweden
Rolf Johansson, Dept. of Automatic Control, Lund University, Sweden (examiner)

Title and subtitle

Accurate Simulation of a Collaborative Robot Arm with Cartesian Impedance Control

Abstract

Simulation of systems is used in several fields of science as a tool for safe and resource-efficient testing, as well as a tool for prediction. In this thesis, the goal is to produce an accurate simulation of a collaborative robot arm, together with a controller solution. The robot is supposed to learn and perform contact-rich tasks. Impedance control is often the suggested control strategy for such tasks, since this type of controller relates kinematics with dynamics to ensure appropriate interaction forces by enforcing the robot to behave like a mass-spring-damper system. The implemented controller for this thesis is based on a Cartesian controller called 'Forward Dynamics Compliance Controller (FDCC)'. It is using ROS and is developed to be a control solution for a wide range of different robot arms. Together with the physics engine ODE, the robotics simulator Gazebo is used as the simulation environment in this project. In this thesis, the controller framework is applied on a KUKA LBR iiwa 7 degree of freedom (DOF) lightweight robot arm. This project is the first known application of the FDCC as a controller to a 7 DOF arm and on a robot without a dedicated end-effector force-torque sensor. Instead of using an end-effector force-torque sensor, the KUKA robot uses an embedded observer for external joint torques to measure interaction forces. In this project, effort is put into processing sensor signals and tuning of control parameters to make the FDCC operate on the KUKA robot. The signal processing consists of frequency filtering and by limiting the rate of change on the sensor signal. Further on, using the same Cartesian impedance controller, the controller parameters are tuned with the result of a stable behaviour and a simulation response closely matching real system response for commanded Cartesian trajectories. Physical Human-Robot Interaction tests also show stable and responsive behaviour. As a final application example, a peg-in-hole insertion task is solved by both a simulated and a real system.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-54

Recipient's notes

Security classification