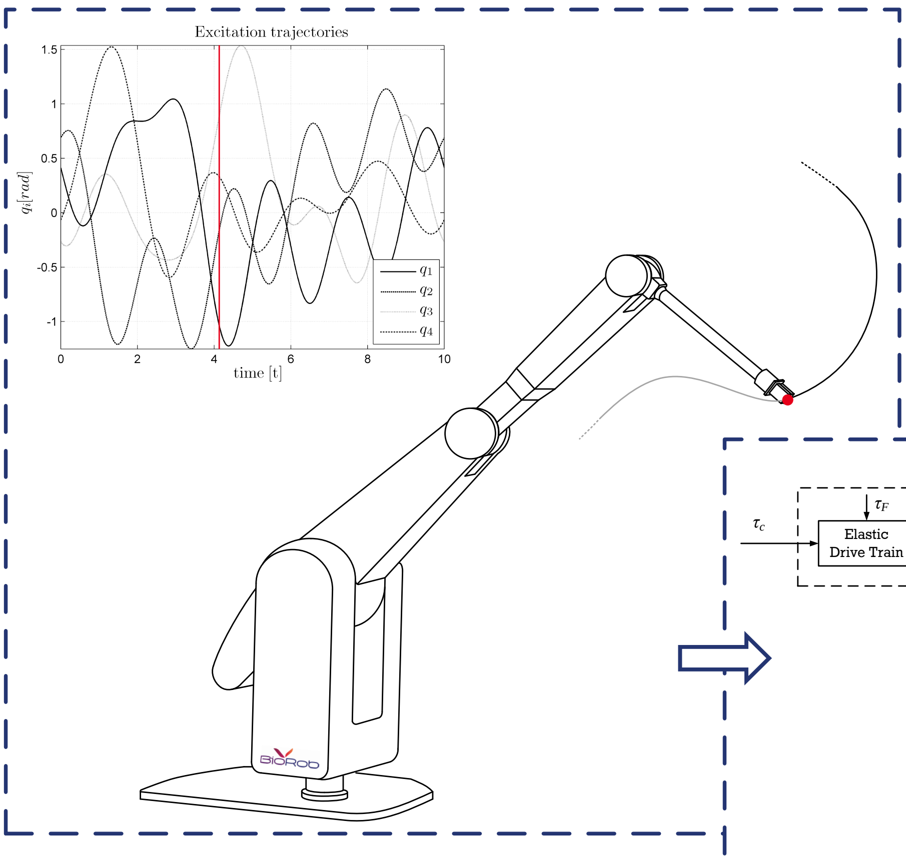
Parameter Identification for a Non-modular Elastic Joint Robot Arm for Observer-based Collision Detection

Parameteridentifikation eines nicht modularen, gelenkelastischen Roboterarms für eine beobachterbasierte Kollisionserkennung

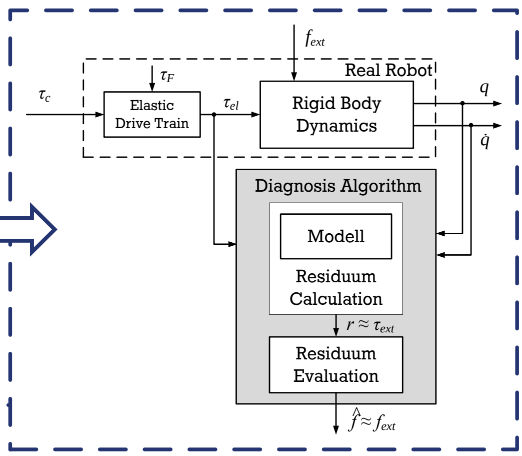
Master Thesis by Jérôme Kirchhoff

April 2011



TECHNISCHE UNIVERSITÄT DARMSTADT

Department of Computer Science
Simulation, Systems Optimization and Robotics Group



Parameter Identification for a Non-modular Elastic Joint Robot Arm for Observer-based Collision Detection
Parameteridentifikation eines nicht modularen, gelenkelastischen Roboterarms für eine beobachterbasierte Kollisionserkennung

Vorgelegte Master Thesis von Jérôme Kirchhoff

Prüfer: Prof. Dr. Oskar von Stryk
Betreuer: Dipl.-Ing. Thomas Lens

Tag der Einreichung: 20. April 2011

Erklärung zur Master Thesis

Hiermit versichere ich, die vorliegende Master Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 20. April 2011

(Jérôme Kirchhoff)

Abstract

Safe physical human-robot interaction gains importance since bringing humans and robots spatially working together provides a high benefit for industry. Here robots can aid humans e.g. as "third hand" or by performing monotonous tasks. To realize this, a certain level of safety has to be ensured. A lightweight design and inherent passive compliance, as present at the BioRob-Arm, helps to reduce the injury risk, which is caused by a robot. Beside this, various performed collision tests from present research showed that a reliable collision detection can reduce the collision force in many cases, or at least dissipate dangerous situations for an involved human. This work implements a model-based disturbance observer for collision detection. Since an accurate model had to be available to ensure a reliable collision detection, a general approach to produce the dynamics model for robots with elastic joints is introduced. To use the model as observer, its parameters have to be identified. For this purpose a method is proposed, that combines two approaches, which separately treat the actuator and load side model to create a linear equation system. These equation systems are then solved using excitation trajectories, which are optimized according to an appropriate observability measure. The model identification process is verified by simulations and experiments. Finally the implemented model-based collision detection is successfully tested during a collision test with an appropriate reaction. The tests have shown that the proposed methods can be used for model identification and collision detection, but the produced model has to be refined to better represent the real behavior. Also the benefit of the collision detection has to be evaluated in further tests with the real robot.

Kurzzusammenfassung

Die sichere Mensch-Roboter-Interaktion gewinnt immer mehr an Bedeutung, denn das gemeinsame Arbeiten von Mensch und Maschine innerhalb eines Arbeitsraumes stellt einen großen Nutzen für die Industrie dar. Hierbei kann ein Roboter zum Beispiel als "dritte Hand" dienen, oder monotonen Aufgaben für den Menschen übernehmen. Um dies zu realisieren, muss ein gewisses Maß an Sicherheit gewährleistet werden. Leichtgewichtige Roboter mit passiver Nachgiebigkeit, wie es der BioRob-Arm darstellt, sind eine Möglichkeit das Verletzungsrisiko zu reduzieren. Zusätzlich durchgeführte Kollisionstests aus bisherigen Arbeiten zeigen, dass eine verlässliche Kollisionserkennung die Kollisionskraft in vielen Fällen reduzieren oder zumindest für den Menschen gefährliche Situationen auflösen kann. Diese Arbeit setzt einen modellbasierten Beobachter für die Kollisionserkennung um. Zur zuverlässigen Kollisionserkennung muss ein akkurates Model zur Verfügung stehen. Zur Ermittlung des Dynamikmodells von Robotern mit elastischen Gelenkten wird ein allgemeiner Ansatz vorgestellt. Bevor das Modell dann im Beobachter heran gezogen werden kann, müssen seine Parameter identifiziert werden. Für diesen Zweck wird eine Methode vorgeschlagen, welche zwei Ansätze miteinander kombiniert, die das antriebs- und abtriebsseitige Modell getrennt betrachten, um für diese jeweils ein lineares Gleichungssystem zur Parameterabschätzung zu erstellen. Um möglichst gute Ergebnisse mit Hilfe der Gleichungssysteme zu produzieren, werden spezielle hierfür optimierte Trajektorien genutzt. Die Ergebnisse der Parameteridentifikation werden sowohl durch Simulationen als auch über Experimente überprüft. Abschließend gilt es die implementierte modellbasierte Kollisionserkennung erfolgreich durch einen Kollisionstest und passende Reaktionsstrategie zu beurteilen. Diese Tests haben gezeigt, dass die vorgestellten Methoden für die Parameteridentifikation und Kollisionserkennung geeignet sind, jedoch das vorliegende Modell weiterentwickelt werden muss, um das reale Verhalten noch besser wiederzugeben. Welchen genauen Nutzen die Kollisionserkennung zur Steigerung der Sicherheit hat, sollte jedoch noch in weiteren Tests mit dem realen Roboter untersucht werden.

Contents

1 Introduction

Nowadays, a machine that autonomously fulfills a task is called robot. But this understanding evolved over time. The term robot was introduced first in 1920 by the Czech writer Karel Capek in his play "Rossum's Universal Robots (R.U.R.)". The word robot emanates from the Slavic word "robota", which means subordinate labour or forced work. In R.U.R. the robots were human like machines (today we would say "androids"). This imagination of the concept robot was refined by Isaac Asimov in the 1940s. This Russian science-fiction writer introduced the well known three laws for the human-robot interaction. Here the human safety is the center of attention. So in the middle 20th century robots were a beautiful conception but unrealizable since the technical requirements were not fulfilled.

The following historical overview summarizes the key statements of [1]. In the following decades the first robotic systems were build. First they only duplicated one-to-one the movement of a human master. With development of integrated circuits computer-controlled robots were designed. These robot arms replaced step-by-step humans in factories and finally in general industry. But the robot found its way out of the industrial environment with new applications like e.g. cleaning, space [1] or search and rescue. After research in the intelligent connection between robot perception (e.g. computer vision) and action, robots are now expected to safely work and life with humans (providing support, service, entertainment, education, etc.).

A reason for the rapid ascent of the robots in the fabrication and other industries is their not diminishing accuracy and the employment in environments dangerous for humans. Humans were replaced at the assembly-line by cheaper robots. Does that mean, robots are "better" than humans? It is true that robots can better handle monotonous and unambitious tasks, because they basically do not fatigue. But in contrast, they are clumsy and dangerous for humans. To cope with these disadvantages one research topic is to design biologically inspired robots. With such inspired bodies the robots should be easily and safely integrated into human environments.

Human-Centered and Life-Like robotics is the research field that covers the vision to leap from personal computers to personal robots. This includes designing biologically inspired robots and safe human-robot interaction. Humanoid robots for instance are capable of bipedal locomotion. They interact with humans via perception systems. These systems should recognize the environment, understand orders by interpreting human language or react on the humans mood[2] (and respond to it appropriately). Humanoids are an example of bio-inspired robots. These robots are reproductions of some natural results, but not necessarily of the underlying means. They tend to adapt traditional engineering approaches to observations of living creatures. On the other side biomimetic robotics tend to replace classical engineering solutions to reproduce the observation of a creature [3]. So biomimetic robots are bio-inspired, but not vice versa.

[1] One example is the mars exploration with "Opportunity" (http://marsrovers.jpl.nasa.gov/home/index.html)

[2] Kismet is a humanoid robot than interacts with humans by simulating emotions: http://www.ai.mit.edu/projects/humanoid-robotics-group/kismet/kismet.html.

[3] Some examples for such biomimetic robots are the "RunBot"from McGill Univerity (http://www.manoonpong.com/Runbot.html), the "Stickybot" from Stanford University Center for Design Research (http://bdml.stanford.edu/twiki/bin/view/Rise/StickyBot) or the "RoboTuna" from Massachusetts Institute of Technology (http://www.manoonpong.com/Runbot.html)

The aim of Human-Centered and Life-Like robots is not possible, if the human-robot interaction is not safe. The following robot safety issues are taken from [2]. Actually industrial robots are far too dangerous to share space with humans. But physical human-robot interaction (pHRI) can be very useful. There are two kinds of pHRI: "hands-off" and "hands-on". "Hands-off" interaction includes tasks where a worker has to enter the robots workspace (e.g. maintenance, repair or test tasks). "Hands-on" interaction is necessary e.g. to work with Intelligent Assist Devices, where humans comanipulate payloads with the device as partner. Robots that are designed to coexist and cooperate with humans can work on applications like assisted industrial manipulation, collaborative assembly or medical applications. Within these applications the importance of safety and dependability increases when human lives are involved. The segregation of humans and robots fails, if they have to share the physical environment to successfully complete their task that requires collaboration. The Holy Grail of pHRI design is intrinsic safety. A robotic device is intrinsically safe if no matter what failure, malfunctioning, or even misuse happen, humans are always safe.

There are various ways to improve the design of intrinsically safe robots. One way is to design an active force control that requires force/torque sensors where ever impacts can occur. This compliance, introduced after sensing an impact, is limited by how the controller can alter the robots behavior. Letting a heavy robot behave gentle and safe is a hopeless task. Another way to achieve safer robots is to construct arms with low inertia of their parts and back-drivability. The psychological acceptability of such arms can be further increased by introducing mechanical compliance. Such compliance realized e.g. by cable transmissions with springs decouples the actuators reflected rotor/gearbox inertia from the links whenever an impact occurs. But this naturally compliant transmissions can diminish performance (decreasing positioning accuracy, velocity of task execution, slow response, increased oscillation, etc.). Since these performance criteria are crucial for most applications, the main research topic is the fast and accurate control of such soft manipulators.

But how do we know when a robot is safe? To design a safe robot one has to know a metric to assess the risk of injuries in accidents. Some severity indices of an impact which can be mapped to the probability of causing a certain level of injury are the Gadd's severity index (GSI), the viscous injury response (VC), or the head injury criterion (HIC, most widely used in the automotive industry). Most of them are related to the tolerance curve developed at Wayne State University (WSTC). This curve (based on experimentally acquired from animal and cadaver head collisions) plots the head acceleration against impact duration. It indicates that very intense head acceleration is tolerable if it is very brief, but that much less is tolerable if the pulse duration exceeds 10 - 15 ms.

In this thesis the biomimetic, partially intrinsically safe robot called BioRob-Arm is subject of research. It consists of links with low inertia and a compliant cable/spring transmission from the motor to the joints. This construction is inspired by the human arm and the corresponding force transmission between muscle and joint. As already mentioned this design decouples the motor/gearbox inertia from the links, and tries to be intrinsically safe. In any case, a collision can occur and harm humans near the robot arm. Even if there are no persons nearby the robot it can collide with obstacles in its workspace. To increase both safety of humans and the robot itself, a collision detection (and appropriate reaction) is needed. One way to realize a collision detection is to use a model to compute a residual between calculated and real arm position [3]. But this detection is just as good as the underlying model.

An accurate robot model is not only necessary for such a collision detection. It is also used to realize model-based robot control schemes as computed-torque or resolved-acceleration. A model is also important to enable off-line programming supported by simulation with accurate motion, which reduces the costs and time of developing a high quality robotic system. To determine a model with highest possible accuracy the model parameters have to be detected. One way to realize this is to disassemble the robot followed by weighing and balancing the components. This is in most cases too complex or not even possible. Alternatively a CAD model can compute the required parameters. This computation also requires an accurate CAD model and material informations that perhaps are not available. This is why an experimental approach is chosen in this thesis to receive the model parameters.

One example of use for physical human-robot interaction is mentioned in [4]. Many small and medium enterprises (SMEs) need robotic automation solutions to increase their cost efficiency. These enterprises have to cope with frequently changing conditions of the production process. In this area service robots are required that fulfill the following key requirements, especially for applications with an unstructured and shared environment for humans and robots:

- Safety: Inherent safety at high speeds and human friendly design boost efficiency and acceptance.
- Flexibility: Mobility, short installation and deployment times allow to quickly change the robot's location and to flexibly react on changing production conditions and current needs.
- Usability: Simple and intuitive programming that can be performed by untrained personnel.
- Performance: Task execution with speed and accuracy comparable to a human arm.

Common industrial robots typically do not meet this criteria or are too expensive for these applications.

Chapter 2

Chapter 2 gives an overview of how human safety can be increased, considering tasks that require cooperation between humans and robots. To know when a robot is safe it has to be investigated what kind of injury it can produce. This chapter shortly presents some safety requirements, which forces are acting during a collision between humans and robots, and which design decision lead to a save robot. Since a reliable collision detection can provide some kind of safety, possible detection schemes are introduced including a model-based one. To realize a model-based collision detection, the model parameters have to be identified first. Approaches which are concerned with this issue are shortly summarized.

Chapter 3

Chapter 3 presents a description how to model a series elastic actuator including the dynamics model and the inverse dynamics. For illustration the BioRob-Arm is modeled. After the kinematic model, the dynamic model configuration is presented containing the elastic transmission, the motor dynamics and the whole equations of motion. How the series elastic actuator influences the inverse dynamics and its usage in the control scheme is shown, as well as how Matlab/Simulink is used to model the robot. Since the Newton-Euler recursion is part of the model parameter identification, it is introduced as procedure to calculate the dynamic equations. Additionally one possibility how to extract the dynamic matrices from the dynamic equations is explained.

Chapter 4

Chapter 4 describes a general possibility to identify the model parameters of a robot with elastic joints. The methodology of the identification process for the actuator and load side are theoretically introduced, before the modified Newton-Euler recursion with only linear model parameters is shown. An identification method containing excitation trajectories and how these are produced is presented, followed by the BioRob-Arm parameter identification on simulation and by experiment.

Chapter 5

In chapter 5 different collision scenarios and a collision detection scheme are introduced. The observer-based detection method and its properties are presented, as well as an appropriate reaction strategy on collisions. Since the joint velocity calculation produces very noisy results, a method using linear regression of the joint positions is presented for this purpose. Finally a collision test is carried out in simulation and which contact model is advisable to use is investigated.

Chapter 6

Chapter 6 summarizes the results of all treated issues and what can be done to further improve the parameter identification and evaluate the collision detection.

2 State of Research

This chapter gives an overview of what can be done to increase human safety in tasks that require cooperation between humans and robots. Further it describes what limitations exist in case of the BioRob-Arm.

As already described in chapter 1, reasons for physical human-robot interaction are to aid humans with routine work, realize human guided teaching, or collaborate assembly. Especially small and medium enterprises need robotic automation solutions to increase their cost efficiency. As shown on [5], examples for service applications are to grip and place chaotically stored work peaces into a machine tool, or the robot forms a worker's third hand. But for all possible applications in which a robot assists a human (also in domestic environments) they must never harm people in their environment.

2.1 Safety Requirements for Physical Human-Robot Interaction

One approach to define safety requirements for industry robots is the ISO 10218 of the European Committee for Standardization [6]. In addition to inherent security requirements for robot parts it restricts the execution to increase safety in collaborative operation with humans. Here the maximum tool center point is restricted to 250 mm/s. Further more, either the maximum dynamic power of 80 W or the maximum static force of 150 N has to be guaranteed. These are very strong restrictions and result in high performance limitations of the robot. Despite such massive constraints it is not assured that nobody is injured during a malfunction either from hardware or from software.

After investigation of the effect of robot speed, robot mass, and constraints in the general environment on safety in human-robot interaction during impacts tests [7] conclude, that the requirements introduced by ISO 10218 tend to be unnecessarily restrictive. Crash-Tests of the "DLR-III Lightweight Manipulator" with a dummy at various speed [8] produced the impact characteristics shown in figure 2.1. The black line shows the externally measured force. The red line represents the acting joint torque (where the sensor manifests saturation). The collision decelerates the link and causes a peak (over $\approx$ 4 - 10 ms) in the measured force (black line). The joint torque, effected by the collision, is detected $\approx$ 6 ms delayed after the impact. This shows, that the impact force is transmitted in a very short period. Even if the collision detection would be able to detect the impact faster, the motors could not revert their motion sufficiently fast enough to reduce the transmitted force.

To evaluate the severity of the suffered injuries [8] uses the "Head Injury Criterion", which is common in the automobile industry. This severity criterion indicates the probability of getting injured. A collision experiment was carried out at 2 m/s with the following industrial robot arms: DLR Lightweight Manipuator, KUKA KR3-SI (54kg), KUKA KR6 (235 kg) and the KUKA KR500 (2350kg). The surprising result of this collision tests was that the injury probability of all robot arms was below one percent. Since this result seamed not to be very meaningful for human-robot interaction, [9] and [10] focused on the force that is needed to cause fractures in the facial and cranial bones. The results showed that even moderate velocities of $0.5 - 1.0 m/s$

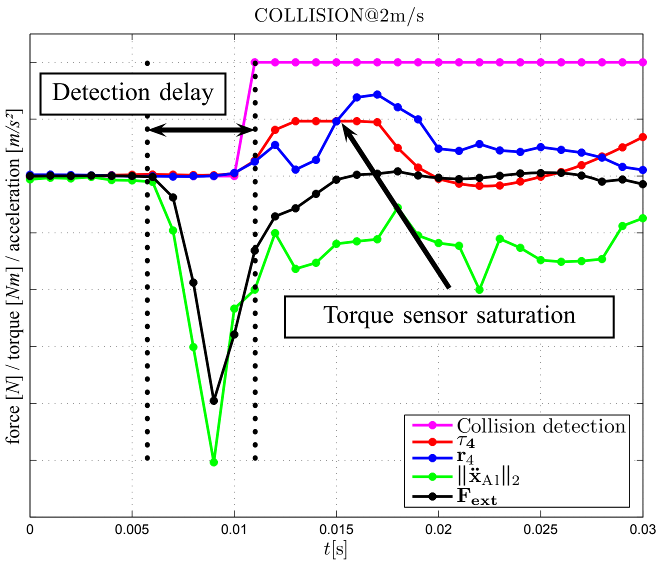


Figure 2.1: Collision characteristics at 2 m/s (from [8])

suffice to cause fractures in all bones accept for the frontal bone. The frontal bone resists impacts up to 2 m/s.

Up to now only safety of the head was investigated. In [11] further experiments took the neck, the chest and the arm into consideration. It was shown that for these body parts the collision detection is able to reduce the collision force. The main reason for this observation is stated in the natural compliance of this body parts, which stretch the force transmission over a longer period (in comparison with the head). Besides the force reduction the collision strategies removed the end-effector from the collision, which resulted in an increased sense of security. Another subject was the force that arrived at the Motor during a collision. It was observed that even with slow velocities the maximum motor torques of rigid robots were exceeded for milliseconds. To reduce this torque peaks the joint stiffness can be reduced or a collision detection can be used.

Besides blunt collisions with humans it is important to consider that the robot arm is acting with sharp tools. [12] tried experiments with various tools from screwdriver to a scalpel and showed that a collision detection and reaction provide a huge benefit. Without collision detection the sharp tool can easily penetrate humans and damage organs resulting in serious injuries.

As mentioned above, no collision detection is fast enough to reduce the transferred force peak. This force is caused by the link and joint (motor, gear) inertia. To have a detailed insight to the involved torques during a collision the explanation of [3] is outlined. Figure 2.2 shows the model of a motor with gear, followed by a link. τ_m describes the motor torque, $\theta, \dot{\theta}, \ddot{\theta}$ the motor position, velocity and acceleration, n_g the gearbox ratio, $q, \dot{q}, \ddot{q}$ the joint position, velocity and acceleration, I_r the rotor inertia, I_g the gearbox inertia an I the link inertia. All inertias are expressed with respect to the same rotation axis.

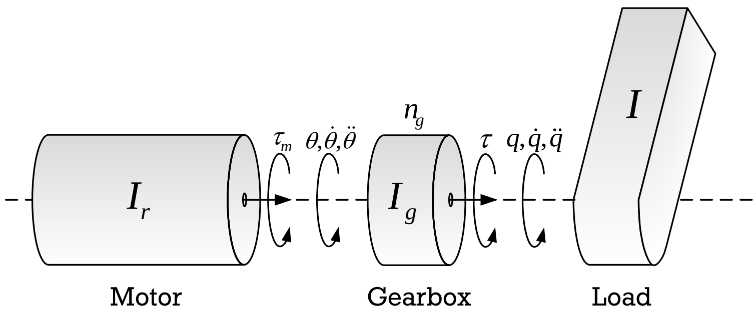


Figure 2.2: Jointscheme (Motor, Gearbox, Load)

The connection between motor and joint velocity and its derivative is given by the gearbox ratio (see equation 2.1).

$$\dot{q} = n_g \cdot \dot{\theta} \qquad \ddot{q} = n_g \cdot \ddot{\theta} \tag{2.1}$$

For the next consideration no load is assumed ($I = 0$) and the gearbox inertia is negligible ($I_g = 0$). If the gearbox is considered idealized with an efficiency of 100 percent[1], the performance has to be maintained over the gearbox. The actuator and output performances are described by $\dot{\theta} \cdot \tau_m$ and $\dot{q} \cdot \tau$ [13], and have to be equal in case of an idealized gearbox:

$$\dot{q} \cdot \tau = \dot{\theta} \cdot \tau_m. \tag{2.2}$$

After inserting equation 2.1 in equation 2.2, the dependency between motor torque and output torque is received (see equation 2.3).

$$\dot{q} \cdot \tau = \frac{\dot{q}}{n_g} \cdot \tau_m \quad \Rightarrow \quad \tau_m = \tau \cdot n_g \tag{2.3}$$

To identify the acting motor inertia in the torque of the output side, one has to insert equation 2.1 and 2.3 into the motor's equation of motion $I_r \ddot{\theta} = \tau_m$ [13] (see equation 2.4).

$$I_r \ddot{\theta} = \tau_m \;\; \Leftrightarrow \;\; I_r \frac{\ddot{q}}{n_g} = n_g \tau \;\; \Leftrightarrow \;\; I_r \ddot{q} = n_g^2 \tau \;\; \Leftrightarrow \;\; \tau = \frac{I_r}{n_g^2} \ddot{q} \tag{2.4}$$

Equation 2.4 shows that the motor torque is modified by the gearbox and results in the real on output side acting motor torque I_r/n_g^2. This torque is denoted as "reflected inertia". This inequality between acting motor torque on actuator and output side is not negligible. Considering a lightweight robot with low link inertia and a high gear reduction, the link inertia I can be smaller than the reflected motor inertia I_r/n_g^2, even if I_r is very small.

[1] The error introduced by the idealization is taken into account in the gearbox friction

Now the acting torques on output side are determined and the remaining inertias ($I_r \neq 0, I \neq 0$) can be considered. Since all inertias are expressed with respect to the same rotation axis they simply can be added up. To illustrate why, consider the volume integral for calculating the separate moments of inertia. Since these volume integrals are build along the same axis, they can be composed to one integral by adding them up. So the resulting torque on output can be expressed as:

$$\tau = (\frac{I_r}{n_g^2} + I_g + I)\ddot{q}. \tag{2.5}$$

All torques involved in a collision are known and we can revisit the collision characteristic. For rigid robot arms, the whole force (caused by all present moments of inertia $I_r/n_g^2 + I_g + I$) is transmitted at once. But if the joint is even little compliant, the rotor continues to turn. How long it continues is determined by the elastic transmission between motor and link. The more compliant the transmission is the more the rotor can continue to rotate, before the arrested joint has effect on the motor. This behavior results in a delayed transmission of the motor and gear inertias into the collision. Such delay achieves more time for both, the collision detection and for reverting the motors, to avoid that the force caused by the motor and gearbox inertia are transmitted into the collision.

[14] researched several joint actuation examples and investigated the limits of performance under safety-enforcing constraints. One of the actuation examples introduced series elastic actuators (SEA) [15]. The result with various transmission stiffnesses showed that the interposition of an elastic transmission between the actuator and the link increased safety with low stiffness but decreased performance at the same time. Hence using the SEA design to decouple the rotor inertia from the link inertia seems to reduce injury risk. Another benefit of the SEA design is the intrinsic low pass filtering of shock loads, which reduce the peak gear forces and low pass filters the shock impulse back driving the actuator [15].

Besides the SEA design two other somewhat more complicated actuation mechanisms were examined in [14]. The distributed macro-mini (DM^2) actuation approach [16] and the variable stiffness transmission (VST) approach. The DM^2 actuation contains two actuators in parallel connection to the same joint, one is devoted to low-frequency components of the required torque, while the other is designed for high-frequency parts. The slow one provides high torques with high rotor inertia and is coupled through a passive elastic transmission (SEA design). The other motor for high frequencies is limited in torque with a very low rotor inertia and rigidly connected to the joint. The VST actuation is an SEA actuation that further allows to vary the transmission stiffness during actuation. The evaluation of both actuation mechanisms resulted in a performance recovering compared to the SEA performance. The DM^2 scheme outperforms SEA in case of large transmission compliance, while there is almost no difference for stiff coupling. The VST with high stiffness at low velocities (harmless) and low stiffness at high velocities (to reduce the transmitted reflected inertia in case of collision), further outperformed the DM^2 actuation, when the stiffness variation range was at certain size. Another advantage of VST is that it allows to put the link in motion swiftly at early acceleration phase, and to minimize oscillations in the final deceleration phase.

Whatever actuation mechanism is chosen for a robot, a collision detection is necessary for further improving safety. It enables to reduce the acting collision forces (in non-rigid case), it increases the sense of security by resolving a dangerous situation and it enables that humans can cooperate with robots equipped with sharp tools.

2.2 Collision Detection and Reaction

There are several approaches to detect collisions by additionally mounting sensors on the robot arm. For example [17] invented a flexible skin. This skin targeted to realize an inexpensive skin to provide the capability of sensing multiple contact locations to increase the level of physical human-robot interaction. As explained in [2] active force control schemes can be used to introduce compliance with respect to the sensed interactions. This approach requires all parts of the arm to be equipped with force/torque sensors. Further there are intrinsic limitations to what the controller can do to alter the behavior of the arm. Another possibility for increasing security and to avoid collisions is to localize obstacle positions in the collaborative workspace [18]. One example of a somewhat appropriate reaction to collisions is proposed by [19]. Here a control scheme for the whole robot surface is proposed that restricts the torque commands to values that comply to preset safety restrictions. So the potential impact force in case of a collision is limited. Another possibility to increase safety during a collision is realized with the KUKA KR 3 SI. This robot arm is equipped with a soft protection cover, capacitive sensors and an autonomous releasing tool fixture. The capacitive sensors enable the robot to detect people in its workspace, and decelerate motion before a collision occurs. The soft cover additionally reduces the collision force.

The simplest way for robots with high path precision, where the joint position, velocity and acceleration can be measured, is described in the following approach. The acting joint torques can be compared with simulated torques from the robot's dynamic equations [20]:

$$\tau = \boldsymbol{M}\left(\boldsymbol{q}\right)\ddot{\boldsymbol{q}} + \boldsymbol{C}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} + \boldsymbol{g}\left(\boldsymbol{q}\right) + \boldsymbol{F}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} \tag{2.6}$$

$$\hat{\tau} = \boldsymbol{M}\left(\boldsymbol{q}_d\right)\ddot{\boldsymbol{q}}_d + \boldsymbol{C}\left(\boldsymbol{q}_d,\dot{\boldsymbol{q}}_d\right)\dot{\boldsymbol{q}}_d + \boldsymbol{g}\left(\boldsymbol{q}_d\right) + \boldsymbol{F}\left(\boldsymbol{q}_d,\dot{\boldsymbol{q}}_d\right)\dot{\boldsymbol{q}}_d \tag{2.7}$$

Here, τ is the joint torque vector corresponding to the measured joint positions, velocities and accelerations, and $\hat{\tau}$ the currently expected joint torque vector. Subtracting one of the other results in the external effected torque

$$\tau_{ext} = \tau - \hat{\tau}. \tag{2.8}$$

But this assumption is only for high path precision true, since then the control error is minimized and the estimation of τ_{ext} as precise as possible. All fast transitions can be indications for a collision. To increase robustness in case of sensor noise, τ_{ext} can be filtered. This basic approach of comparing modeled with measured behavior is not suitable for the BioRob-Arm, since it is only equipped with position sensors. To estimate the missing values one can use numeric differentiation of the joint position. But such joint velocity and acceleration estimation can massively increase the senor noise. To decide whether a collision is occurred or not with such noisy signals is not reliably feasible (see [3] for an example).

All above mentioned collision detection strategies need to assemble additional sensors and thus suitable cables. This is design related impossible for the BioRob arm. To minimize the amount of cables a bus can be used instead. But this requires electronic control systems at the sensors which increases the robot mass. Further, common sensors for torque, velocity and acceleration measurement are not designed for temperatures like in cryogenic applications. Thus a procedure is needed, that gets along with only position sensors.

The actually implemented collision detection also compares a desired with the actual value. Here the joint torque variations are not analyzed, but the expected and actual motor behavior is evaluated. Since the motor velocity is proportional to the applied motor voltage, the current motor velocity can be estimated from a particular applied voltage. Velocity changes at constant voltage, can thus indicate that a collision is occurred. The difficulty is, that not all unexpected variations of the velocity are caused by collisions. Also dynamic effects as gravitation or Coriolis forces can be reasons for such behavior. This results in the task to find an appropriate threshold, that allows variations caused by dynamic effects but further reliable react on collisions. Up to now this task cannot be reliably solved. Such a collision detection principle has been also proposed and implemented by [21]. Another problem with this approach is, that even with exact error detection, the acting joint torques cannot be determined from the estimated motor velocity, thus which joint is involved. So, an collision detection strategy has to be found, that enables reliable collision detection only with motor and joint position sensors.

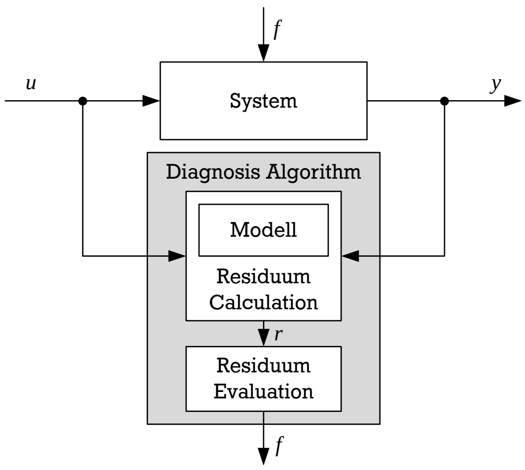


Figure 2.3: Structure for residuum calculation/evaluation from [22] (control variable u, measured system output y, residuum r, error f)

All proposed approaches need either more sensors or cannot reliably determine the error between desired and actual joint torques. For complex dynamic systems, as robots are, this task is called "Fault detection and isolation" (FDI). The error estimation tries to generate a diagnose signal, which expresses the error. This signal is called residuum. While execution the residuum is calculated from input and output. The structure of such a FDI-method is shown in figure 2.3.

As residuum one understands a quantity that shows the variations between the measured process behavior and a model. A residuum hast to be significantly different from zero if an error occurs. If a certain threshold is exceeded one can assume that an error is existent.

One FDI-method which refers to joint torques of robot arms was proposed from [23]. This method, based on the generalized momentum, does not need a joint acceleration estimate. Further it filters the measured values to reduce the influence of sensor noise and increase the robustness. Such created residuum should enable to reliably isolate errors, so that an adequate reaction strategy (like proposed in [24]) can be applied. This approach especially fits to the BioRob-Arm, because high joint elasticities can be simply added in the model. This is why this methodology is used for collision detection. To realize a reliable detection, the model has to be estimated as precise as possible. In the following section procedures for parameter estimation are presented.

2.3 Parameter Identification

As described in the previous section, a reliable observer-based collision detection requires a precise robot model. But also for other applications like model-based control schemes such a model is necessary. The accuracy of the application and hence its robustness or performance depends on the accuracy of the model parameters. Creating a model for a robot arm is a research topic since the mid-eighties. There are two types of model parameters, geometrical and inertial. Geometrical parameters can in most cases simply be measured, e.g. the length of the links. As presented by [25] there are three typical ways to determine the dynamic parameters. The first possibility is to use a CAD model of the manipulator links, but this method relies on model accuracy thus to model small parts as bearings, bolts, etc. The second opportunity consists of physical measuring all parts. [26] disassembled for this purpose the links of a PUMA 560 arm. With this method it is not possible to determine the cross-coupling inertia values of the links. The third and most preferred method contains planed motion of the manipulator and using the dynamic model to calculate the parameters using the executed motion and a least squares approach.

An overview of existing algorithms for parameter estimation is given by [25]. [27] presented an algorithm that use a 6-by-1 one vector of sensed force and torque values to determine the inertial parameters, based on dynamics received by Newton-Euler recursion. [28] proposed an two-step algorithm that first identifies center of mass and Coulomb friction and in a second step the load dynamics and viscous friction parameters. To overcome the non-linearity of the center of mass position in the Newton-Euler recursion, they assume that the manipulator motions are slow and the rotary accelerations are insignificant.

A lot of effort to identify the model parameters is done by Gautier et al. based on the robot energy and the Lagrange-Euler algorithm. The research done consists of identifying the inertial parameters [29], and calculation of the minimum inertial parameter set, that can be identified [30]. Further in [31] the direct and inverse dynamic model were combined for identification in an optimization loop that only required torque data. To improve the estimations of the classical least squares approach [32] addresses to the problem of a noisy regression matrix and the resulting biased least squares estimation by building an instrument matrix that removes this bias. Besides the Lagrange-Euler approach the Newton-Euler recursion can be used to calculate the robot dynamics. [33] and [34] developed an efficient estimation method based on a modified Newton-Euler recursion to overcome the non-linearity of the center of mass position.

Based on the parameter categorization [35] into identifiable, identifiable in linear combinations and unidentifiable parameters [36] used a maximum likelihood estimation ([37]) of the inertial parameters for a 6-DoF rigid robot arm. All described methods only deal with rigid robot arms. In context with humanoid robots, [38] proposed a two-step algorithm founded in the serial elastic actuation design. First the approach identifies the actuator parameters including stiffness of the elastic transmission, and then considers the load side. In contrast to [36], here the minimal parameter set is not calculated numerically, but symbolically and hence is not build for a particular trajectory.

The approach used in this master thesis tries to combine the latter two mentioned procedures. It is based on the modified Newton-Euler approach used in [36] to get the rigid dynamic model of the load side and reduce this model with the symbolic approach used by [38]. This symbolic post-processing and its results rely on the quality of symbolic simplification.

3 Modeling of Series Elastic Actuators

3.1 Introduction

The involved forces during a collision are determined by the load and motor inertia, further no collision strategy is fast enough to reduce the transmitted collision force caused by the link (see chapter 2). One way to reduce the links inertia is to move the actuators from the joints to the basis, and e.g. transmit the actuation torques via cables. Further, the safety for physical human-robot interaction can be increased by including passive elasticity in the robots design [2]. One way to realize such elasticity is the series elastic actuator (SEA) mechanism [15]. SEAs introduce a elastic element (e.g. springs) between the output side of the gearbox and the link (as shown in figure 3.1)

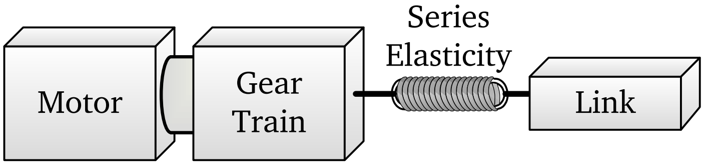


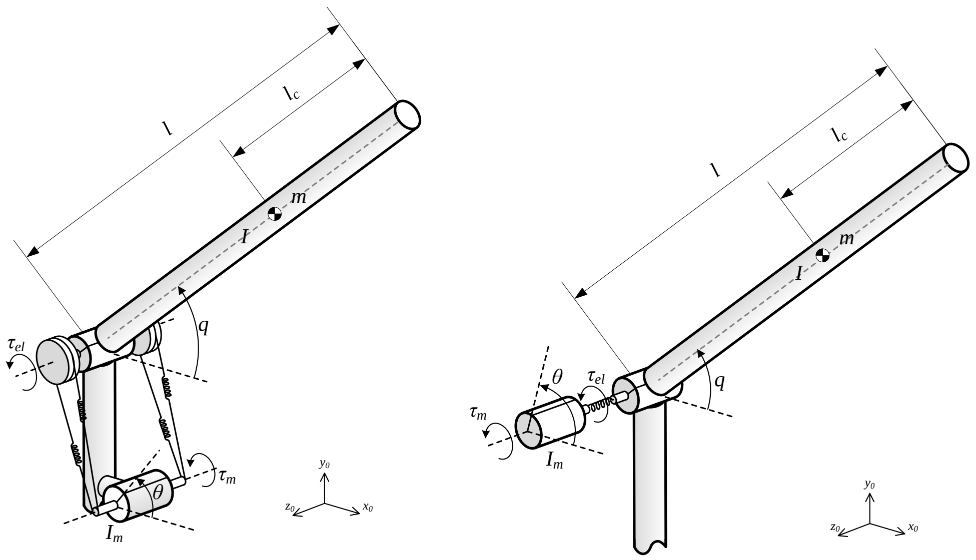
Figure 3.1: Block diagramm of searies elastic actuator (from [15])

As already described, one result of [14] is that an elastic transmission increased safety with low stiffness but decreased performance at the same time. This decoupling of rotor and link inertia seems to reduce injury risk. [15] additionally stated that another benefit of SEA is that it low pass filters the shock loads in both directions, to the load side and back to the actuator side. [39] highlighted that to describe the motion of an elastic robot arm, one has to introduce additional coordinates. Such coordinates do not need a high elasticity to be justified, even small elasticities e.g. from harmonic drives need a special control action to avoid oscillations or instabilities.

To facilitate modeling of a series elastic actuated joint with compliant cables, [40] relocated the motor position for modeling in the joint (see figure 3.2). The precondition for this simplification is that the kinetic energy of the elastic transmission can be neglected in comparison with the kinetic energy of the other mechanical parts. To consider this change in the model e.g. the motor mass can be added to the link and the transmission ratio of the pulley to the gearbox ratio. In the following two paragraphs the dynamic modeling of elastic systems will be described, following the descriptions of [39] and [41].

Dynamic modeling

Here, a robot with flexible joints will be considered as an open kinematic chain with $N+1$ rigid bodies interconnected by N revolute joints. Accordingly to a rigid model, N frames are attached to the N joints, hence the standard Denavit-Hartenberg convention can be used. All joints are actuated by electrical drives. As mentioned above additional coordinates are needed to describe an elastic robot system. One realization are two N-by-1 vectors, $\boldsymbol{q}$ for the link positions and $\boldsymbol{\theta}$ for the motor positions. $\boldsymbol{\theta}$ is reflected through the transmission gears. Reasons for this choice of variables are given by [39]:



(a) Construction of one BioRob-Arm joint

(b) BioRob-Arm joint actuated by a series elastic actuator

Figure 3.2: Relocation of the motor into the joint to facilitate modeling adapted from [40]

1. after reflection, the model will be formally independent of the transmission ratios;
2. the chosen position variables will have a similar dynamic range;
3. the kinematics of the robot will be a function of the link variables $\boldsymbol{q}$ only so that all issues related to direct/inverse kinematics will be identical to the case of fully rigid robots.

To model an elastic system [39] proposed some assumptions that can be made.

(A1) The actuators' masses are rotationally symmetric and their center of masses are located on the rotation axes.

This first assumption implies the independence of the inertia matrix and the gravity term from the motors position. Further the rotor inertia matrix is diagonal (consist only of principal moments of inertia)

To neglect the dynamic coupling of the inertial components (links and rotors) a second assumption can be made.

(A2) The angular velocity of the rotors is due only to their own spinning instead of their own together with the link velocity.

Since usually motors with a high gearbox ratio are used to transform a fast rotation with low torque into a slow rotation with higher torque (ratio of 1:50 to 1:200), this assumption is often correct. So the influence of the robot motion on the rotors can be neglected.

Taking these assumptions into account and considering the robots energy [39] received the dynamic model of a elastic robot, coupled by a compliant transmission (idealized without friction):

$$I_m \ddot{\theta} + \tau_{el} = \tau_m \tag{3.1}$$

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau_{el} \tag{3.2}$$

Equation 3.1 describes the motor dynamics with the diagonal rotor inertia matrix I_m, the torque caused by the elastic transmission τ_{el} and the motor torque τ_m. Equation 3.2 describes the rigid robot dynamics with the mass matrix $M(q)$, the matrix $C(q,\dot{q})$ of the centrifugal and Coriolis terms and the gravity torque vector $g(q)$ coupled with the elastic transmission τ_{el}.

Inverse dynamics

The inverse dynamics problem computes the nominal torque needed to reproduce a desired motion (given by q, $\dot{q}$, $\ddot{q}$). In contrast to rigid robots, the inverse dynamics for elastic robots is not straight forward. Since the motor trajectory is not known, this has to be computed in an additional step with the torques produced by the elastic transmission. The elastic transmission with springs can simply be formulated as:

$$\begin{aligned}\tau_{el} &= K \cdot (\theta - q)\\ \Rightarrow \theta &= K^{-1} \cdot \tau_{el} + q\end{aligned} \tag{3.3}$$

Now the motor position can be determined only from the joint position and the elastic torque transmission. K is the diagonal spring stiffness matrix. After differentiating equation 3.3 two times and insertion into the motor dynamics (equation 3.1), the motor torque τ_m can be estimated from:

$$\begin{aligned}\tau_m &= I_m \ddot{\theta} + \tau_{el}\\ &= I_m \cdot \left(K^{-1} \cdot \ddot{\tau}_{el} + \ddot{q}\right) + \tau_{el}\end{aligned} \tag{3.4}$$

As can be seen the elastic transmission force can be computed from the rigid dynamic model (equation 3.2) and its second derivative ($y^{[i]} = \mathrm{d}^i y/\mathrm{d}t^i$ denotes the i-th derivative, set $N(q,\dot{q}) = C(q,\dot{q})\dot{q} + g(q)$ for compactness, cf. [39]):

$$\tau_{el} = M(q)\ddot{q} + N(q,\dot{q}) \tag{3.5}$$

$$\dot{\tau}_{el} = M(q)q^{[3]} + \dot{M}(q)\ddot{q} + \dot{N}(q,\dot{q}) \tag{3.6}$$

$$\ddot{\tau}_{el} = M(q)q^{[4]} + 2\dot{M}(q)q^{[3]} + \ddot{M}(q)\ddot{q} + \ddot{N}(q,\dot{q}) \tag{3.7}$$

After insertion of equation 3.5 and 3.7 in equation 3.4 the final equation to compute the necessary motor torque to reproduce a desired motion is achieved:

$$\begin{aligned}\tau_m = I_m K^{-1} &\left(M(q)q^{[4]} + 2\dot{M}(q)q^{[3]} + \ddot{M}(q)\ddot{q} + \ddot{N}(q,\dot{q})\right) +\\ &I_m\ddot{q} + M(q)\ddot{q} + N(q,\dot{q})\end{aligned} \tag{3.8}$$

Since equation 3.8 contains the computation of $\boldsymbol{q}^{[4]} = \mathrm{d}^4\boldsymbol{q}/\mathrm{d}t^4$ it requires a higher degree of smoothness of the desired trajectory.

The next sections show how a robot with elastic joints can be modeled and how the dynamic matrices are created. Further the needed control law is briefly introduced. As example the BioRob-Arm is described.

3.2 Modeling the BioRob-Arm

The BioRob-Arm [42] is an 4 DoF series elastic actuator (SEA) arm, which is inspired by the human arm in sense of construction and field of application. Its lightweight design allows to manipulate objects up to a mass of 1 kg without difficulty with a dead weight of only 4 kg. This performance also describes the field of application for this arm. Analog to the human arm it has to perform "Pick-and-Place" tasks, e.g. at an assembly line. Since the robot is lightweight and portable it can be deployed at different locations. New tasks can simply be created by walk-through teaching, so that the facility time remains as small as possible.

To reduce the links inertia, the robot is constructed very lightweight with rigid links and all motors are placed as close as possible at the robots base. The actuators torques are transmitted to the links by pulleys and cables. To realize the passive compliance between the actuators and links, according to the SEA design, springs are built-in the cables. As summarized in [40], actuation via electrical motors is robust, allows high speeds, exhibits excellent controllability and is well suited for highly mobile applications. In contrast the SEA design with very low stiffness needs special efforts regarding oscillation damping. Further the actuation via cables and pulleys increases friction.

Figure 3.3 shows the mechanical structure of the BioRob-Arm including the fist and second link. The motors for the first and second joint are mounted on the first link, the motors to actuate the third and fourth joint are mounted right behind the rotation axis of the second joint, on the third link. This motor placement enables a very low link inertia. One can see, that the actuation cables of the fourth joint are wrapped around a deflection pulley on joint three. The drive train of the third joint with all its components is separately depicted in figure 3.3.

3.2.1 Kinematic Model

As already proposed in chapter 3.1, the kinematic behavior can be described by the standard Denavit-Hartenberg (DH) convention, which only depends on the joint position $\boldsymbol{q}$ as for rigid manipulators. The kinematic structure according to the DH-parameters is shown in figure 3.4.

The actuation of joint four needs an additional guiding pulley, since the actuator is places on link 2 (see figure 3.3). This deflection pulley (with radius r_{d_3}) couples joint three with joint four and influences the equilibrium position of motor four, as introduced in [40]. The equilibrium positions are the joint and motor positions where the elastic coupling produces no torque. Without an additional deflection pulley the equilibrium position of the joints corresponds to the actual motor position, as it is the case for joint one till three. But for joint four it has to be con-

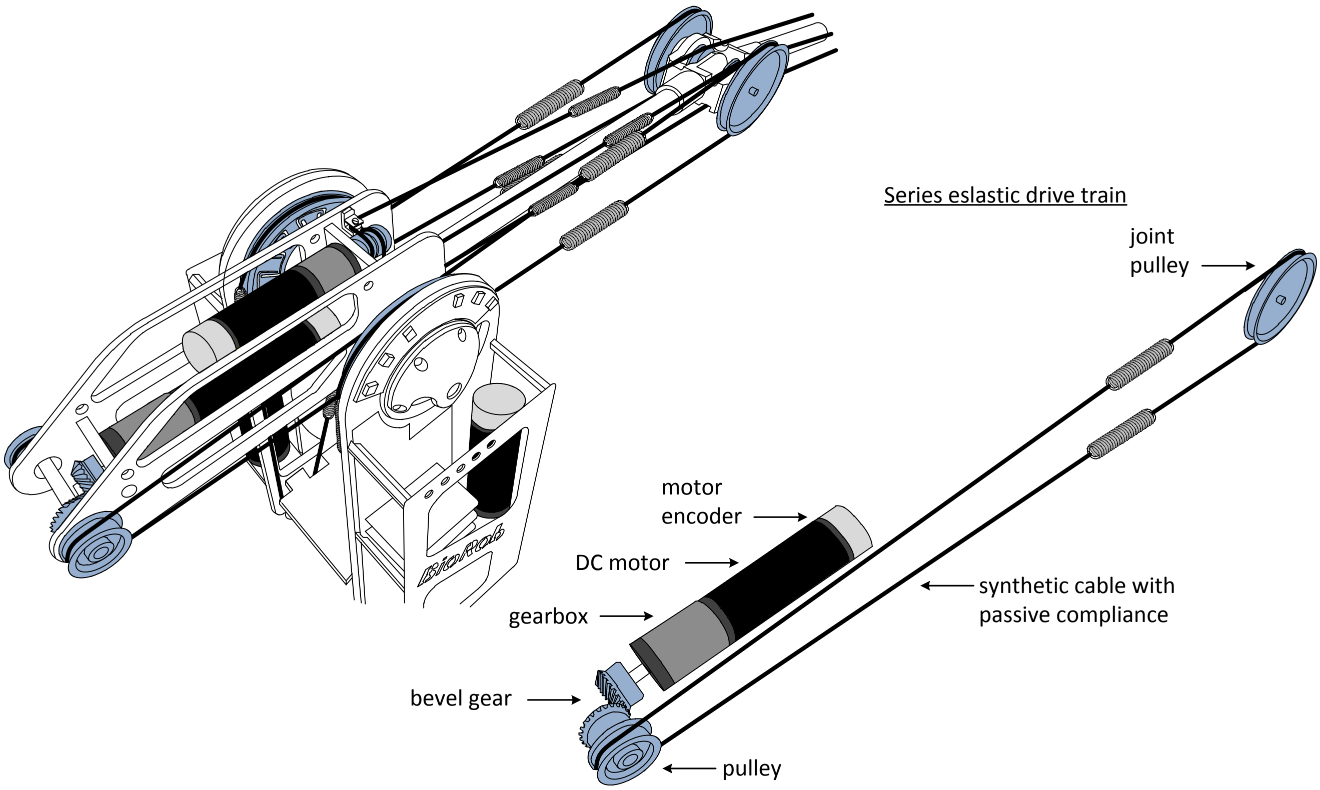


Figure 3.3: Mechanical design with actuation principle of BioRob-Arm

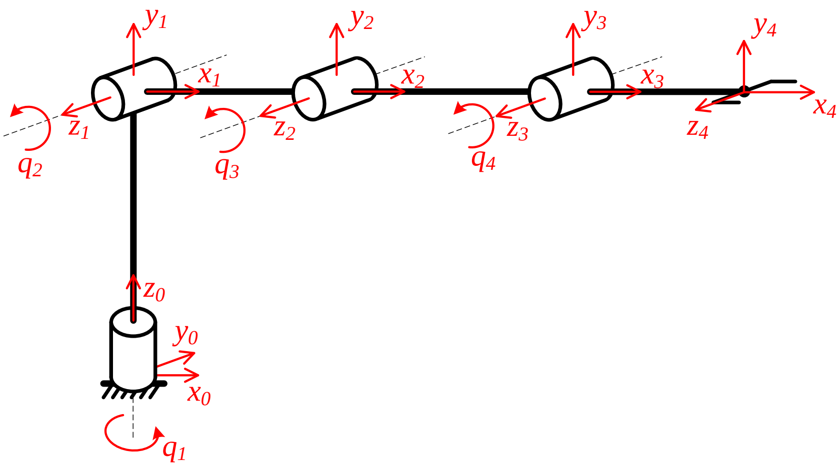
sidered how the guiding pulley in joint three influences the cable wrapped around. The cable length around the guiding pulley is equal to the amount of cable that unwinds from the pulley at joint four, resulting in the following equation:

$$r_{d_3}\, q_3 = -r_4\, \Delta q_4 \quad \Rightarrow \quad \Delta q_4 = -\frac{r_{d_3}}{r_4} q_3. \tag{3.9}$$

This deflection has to be regarded at equilibrium calculation, that is used for control. As vector, the correction term of the motor position to receive the equilibrium position can be written down as:

$$\boldsymbol{\alpha_c}(\boldsymbol{q}) = \left(0, 0, 0, -\frac{r_{d_3}}{r_4} q_3\right)^t \tag{3.10}$$

Now the geometrical structure is modeled. To describe the dynamic model, the motor positions, cables and pulleys have to be abstracted as depicted in figure 3.2. As already described this simplification is only justified, if the kinetic energy of the elastic transmission can be neglected in comparison with those of other mechanical parts, the motor masses are added to the link and the transmission ratio to the gearbox ratio. The resulting schematic model is shown in figure 3.5. As one can see, the links consists of thin bars with mass m_i and inertia I_i around the center of mass. The center of mass position is determined by the vector $\boldsymbol{r}_{c_i}$ relative to the joint



i	θ_i	d_i	a_i	α_i
1	q_1	l_1	0	$\frac{\pi}{2}$
2	q_2	0	l_2	0
3	q_3	0	l_3	0
4	q_4	0	l_4	0

Figure 3.4: Kinematic chain structure of BioRob 4 DOF robot arm with joint frames according to listed DH parameters.

Frame S_i located at joint $i+1$. The compliance of the elastic transmission can simply be modeled as a mass-spring-damper system with spring constant k_{e_i} and damping coefficient d_{e_i}. Not shown in figure 3.5 is the joint friction, which is defined as a viscous damping with coefficient d_i. Further the rotor and gearbox have the inertia I_{r_i} and I_{g_i} respectively.

3.2.2 Dynamic Model

For the sake of simplicity the dynamic model is developed for one joint and then transferred into matrix form for n joints. The next paragraphs explain the components of the dynamic model one by one and merge them together to create the dynamic equations.

Elastic transmission

As mentioned above the elastic transmission can be modeled as a mass-spring-damper system. A model of such a system with all acting forces is displayed in figure 3.6. Here, the variable x denotes the elongation taking affect on the spring and damper. Hooke's law supplies the spring equation $\tau_s = k_e \cdot x$, further the viscose damping is described by $\tau_d = d_e \cdot \dot{x}$. These forces result in $\tau_{el} = \tau_s + \tau_d = k_e \cdot x + d_e \cdot \dot{x}$, the force of the elastic transmission. Since the elongation is determined by the motor and joint displacement, the final equation is:

$$\tau_{el} = k_e \left(\theta - q\right) + d_e \left(\dot{\theta} - \dot{q}\right) \tag{3.11}$$

Motor dynamics

The motor positions θ were mentioned as reflected, so that all position variables have the same range and are independent of the transmission ratio. For the same reasons the torques generated by the motors and their friction are reflected through the elastic coupling. To facilitate understanding, the reflection is explained with a model of the elastic drive train, shown in figure

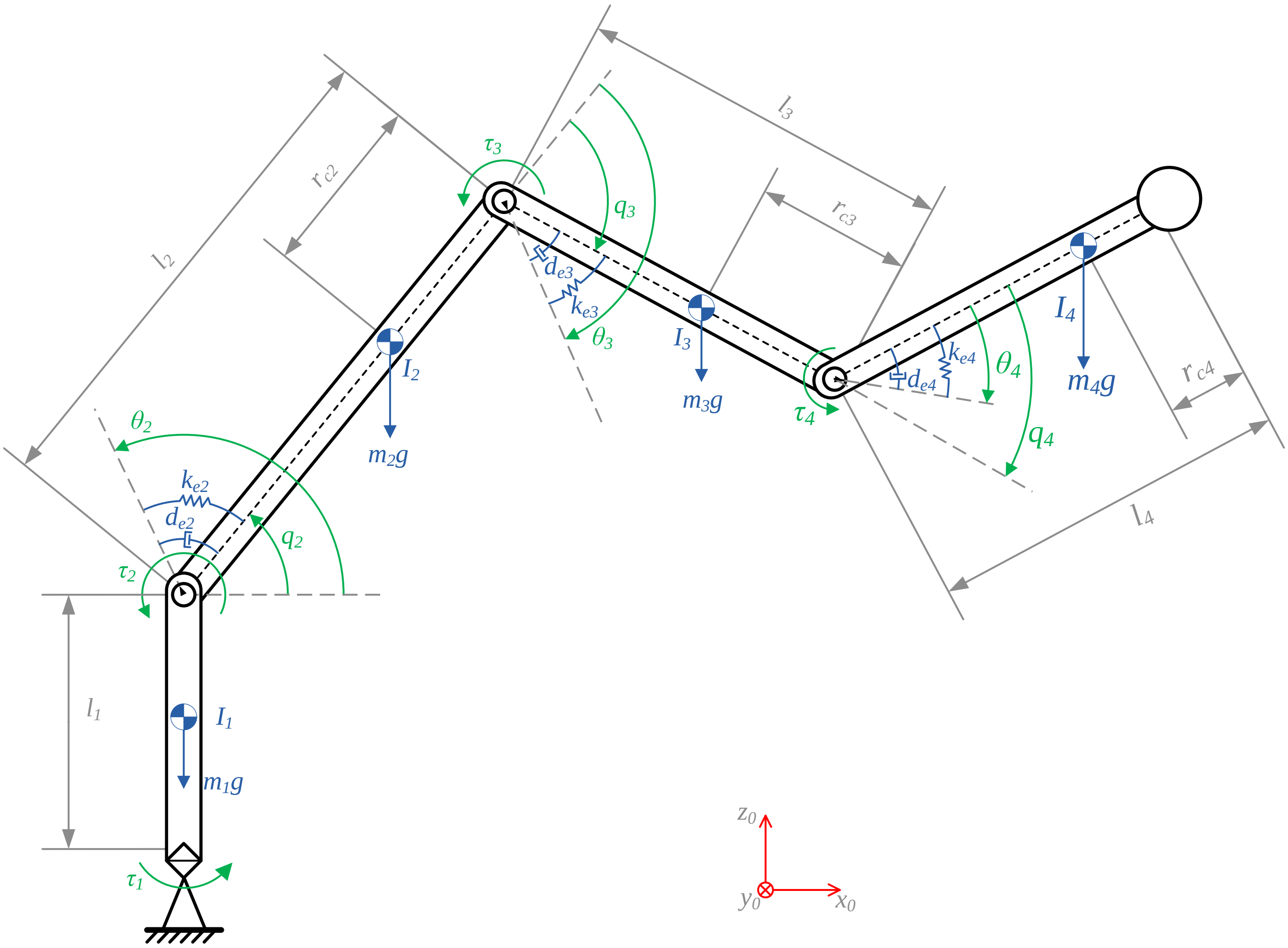


Figure 3.5: Series elastic 4 DoF robot arm

3.7. As explained in [3] the motor dynamics can be derived from the conservation of angular momentum [13]. The time derivative of the angular momentum determines the torque: $\tau_r = \frac{\mathrm{d}L}{\mathrm{d}t}$. The angular momentum $\boldsymbol{L}$ of a rigid body (e.g. the rotor) is the product of its moment of inertia and the angular velocity $L = I_r\dot{\theta}_r$. After insertion, the rotor dynamic equation is derived as $\tau_r = I_r\ddot{\theta}_r$. After introducing a general motor friction term depending on the rotor velocity, the mechanical motor equation becomes (the subscript r indicates that variable belongs to the rotor):

$$\tau_r = I_r\ddot{\theta}_r + f_r(\dot{\theta}_r) \tag{3.12}$$

To achieve the required motor torque, the fast motor velocity has to be reduced via a gearbox. Additionally the elastic transmission further reduces speed. The allover transmission ratio is

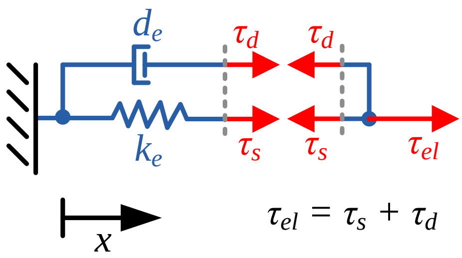


Figure 3.6: Mass-spring-damper model with all acting forces

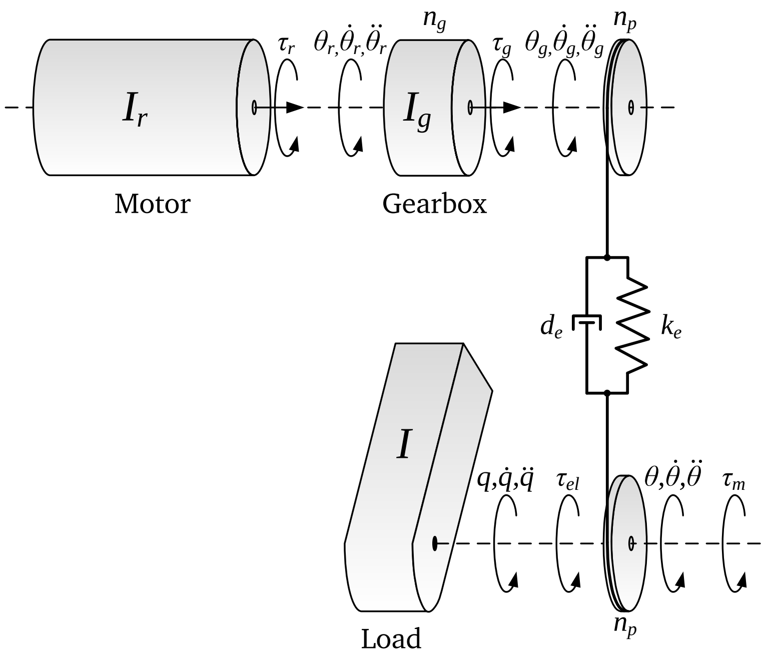


Figure 3.7: Motor model and elastic drive train with all acting forces

called z, with $z = n_g \cdot n_p > 1$, n_g the gearbox ratio, n_p the pulley ratio, and influences the rotor speed as follows:

$$\dot{\theta} = \frac{1}{z} \cdot \dot{\theta}_r \quad \Leftrightarrow \quad \dot{\theta}_r = \dot{\theta} \cdot z \tag{3.13}$$

$$\ddot{\theta} = \frac{1}{z} \cdot \ddot{\theta}_r \quad \Leftrightarrow \quad \ddot{\theta}_r = \ddot{\theta} \cdot z. \tag{3.14}$$

Since the speed is reduced, the rotor torque is amplified according to the same transmission:

$$\tau_m = z \cdot \tau_r \quad \Leftrightarrow \quad \tau_r = \frac{1}{z} \cdot \tau_m. \tag{3.15}$$

Using a gearbox to reduce speed and amplify torque introduces two additional values, the gearbox friction $f_g(\dot{\theta}_r)$ and inertia I_g. Both values are expressed with respect to the rotor axis and also have to be reflected to the load side. Since the inertia terms are expressed with respect to the same axis, they simply can be added up to $\hat{I}_r = I_r + I_g$. One possibility to model friction is to assume a viscous damping $d_v \cdot \dot{\theta}_r$ with coulomb friction $d_C \cdot \mathrm{sign}(\dot{\theta}_r)$. Since both friction terms act based on the same velocity $\dot{\theta}_r$, the friction coefficients can also be added up, resulting in the following friction model:

$$\begin{aligned}\hat{f}_r(\dot{\theta}_r) &= f_r(\dot{\theta}_r) + f_g(\dot{\theta}_r)\\ &= d_{v,r} \cdot \dot{\theta}_r + d_{C,r} \cdot \mathrm{sign}(\dot{\theta}_r) + d_{v,g} \cdot \dot{\theta}_r + d_{C,g} \cdot \mathrm{sign}(\dot{\theta}_r)\\ &= \left(d_{v,r} + d_{v,g}\right) \cdot \dot{\theta}_r + \left(d_{C,r} + d_{C,g}\right) \cdot \mathrm{sign}(\dot{\theta}_r)\end{aligned} \tag{3.16}$$

Insertion of the combined friction and inertia equation for the rotor and gearbox into equation 3.12 leads to:

$$\tau_r = \left(I_r + I_g\right) \cdot \ddot{\theta}_r + \left(d_{v,r} + d_{v,g}\right) \cdot \dot{\theta}_r + \left(d_{C,r} + d_{C,g}\right) \cdot \mathrm{sign}(\dot{\theta}_r) \tag{3.17}$$

Reflecting the rotor torques, velocity and acceleration according to 3.13, 3.14 and 3.15 through the elastic transmission leads to the final motor dynamics equation (subscript m) with reflected variables marked with braces:

$$\begin{aligned}\frac{\tau_m}{z} &= \left(I_r + I_g\right) \cdot \ddot{\theta} \cdot z + \left(d_{v,r} + d_{v,g}\right) \cdot \dot{\theta} \cdot z + \left(d_{C,r} + d_{C,g}\right) \cdot \mathrm{sign}(\dot{\theta} \cdot z)\\ \tau_m &= \underbrace{z^2 \cdot \left(I_r + I_g\right)}_{I_m} \cdot \ddot{\theta} + \underbrace{z^2 \cdot \left(d_{v,r} + d_{v,g}\right)}_{d_{v,m}} \cdot \dot{\theta} + \underbrace{z \cdot \left(d_{C,r} + d_{C,g}\right)}_{d_{C,m}} \cdot \mathrm{sign}(\dot{\theta})\\ \tau_m &= I_m \cdot \ddot{\theta} + d_{v,m} \cdot \dot{\theta} + d_{C,m} \cdot \mathrm{sign}(\dot{\theta})\end{aligned} \tag{3.18}$$

Rigid joint dynamics

The last part to complete the whole one joint elastic dynamics model is the link dynamics equation. This is taken from [20]:

$$\tau = I\ddot{q} + d\dot{q} + mgl\,\cos(q). \tag{3.19}$$

Resulting dynamics

Till now three systems and their dynamics equation have been derived. To achieve the whole model of one elastic joint, this systems have to be coupled together. For this purpose, the schematic representation of all systems and the acting torques are depicted in figure 3.8. The sum of force/toques can only be built on one point or solid body. The motor torque transmission to the link happens only through the elastic drive chain, because of the contact point between motor ⟷ elastic coupling and elastic coupling ⟷ link. At these points the equilibrium is required and leads to the following equations:

$$\tau_1 = \tau_{el} \quad \text{and} \quad \tau_2 = \tau_{el} \tag{3.20}$$

As shown in figure 3.8 the first equation acts at the contact point of the motor and the second equation at the contact point of the link. Inserting both equations into the motor and link

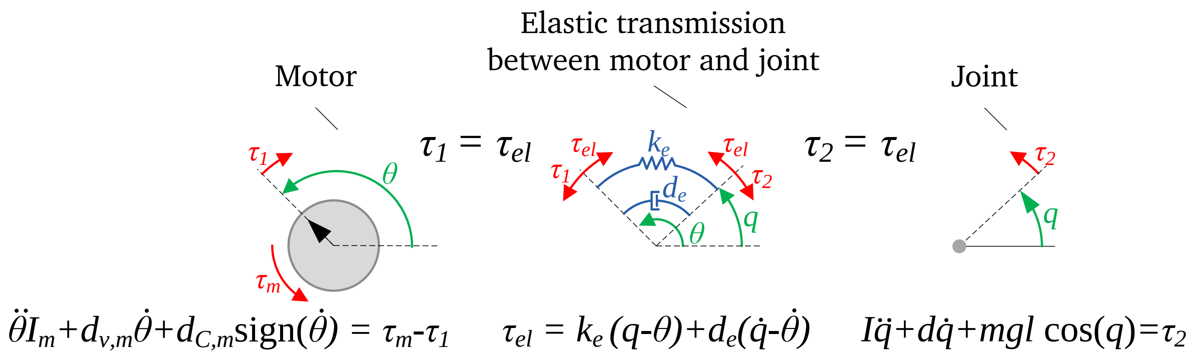


Figure 3.8: Free body diagram of an elastic joint with acting torques

equations, as illustrated in figure 3.8, results in the whole dynamics equation of one elastic joint, coupled via an elastic transmission:

$$I_m\ddot{\theta} + d_{v,m}\dot{\theta} + d_{C,m}\mathrm{sign}(\dot{\theta}) + k_e\left(\theta - q\right) + d_e\left(\dot{\theta} - \dot{q}\right) = \tau_m \tag{3.21}$$

$$I\ddot{q} + d\dot{q} + mgl\cos(q) = k_e\left(\theta - q\right) + d_e\left(\dot{\theta} - \dot{q}\right) \tag{3.22}$$

Now this joint model has to be extended for n joints. First it hast to be ensured, that all derived equations also hold for higher degrees of freedom. Since assumption (A2) from chapter 3.1 holds, the motor model can also be used for motors moving in space. The transmission ratios of the BioRob-Arm (about $i = 100 : 1$) causes the motors to rotate very fast in comparison to the joints, further the inertial coupling of motors and joints can be neglected. The elastic transmission is only assumed, but does not exist in the real joints and thus has no mass. With this absence of mass, motions of the elastic coupling cannot influence the motion of joints in space. So the elastic coupling in all joints can be modeled by a mass-spring-damper model. The robot dynamics equation for an n-DoF robot arm can be formulated as matrix equation [20]. Altogether this leads to the used dynamics model in this work:

$$\boldsymbol{I_m}\ddot{\boldsymbol{\theta}} + \boldsymbol{D_{v,m}}\dot{\boldsymbol{\theta}} + \boldsymbol{D_{C,m}}\mathrm{sign}\left(\dot{\boldsymbol{\theta}}\right) + \boldsymbol{K}\left(\boldsymbol{\theta} - \boldsymbol{q}\right) + \boldsymbol{D}\left(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}\right) = \boldsymbol{\tau_m} \tag{3.23}$$

$$\boldsymbol{M}\left(\boldsymbol{q}\right)\ddot{\boldsymbol{q}} + \boldsymbol{C}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} + \boldsymbol{F}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} + \boldsymbol{g}\left(\boldsymbol{q}\right) = \boldsymbol{K}\left(\boldsymbol{\theta} - \boldsymbol{q}\right) + \boldsymbol{D}\left(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}\right) \tag{3.24}$$

where $\boldsymbol{M}\left(\boldsymbol{q}\right)$ is the mass matrix, $\boldsymbol{C}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}}$ the Coriolis matrix, $\boldsymbol{F}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}}$ the friction and damping of the joints and $\boldsymbol{g}\left(\boldsymbol{q}\right)$ the gravity torque vector. The mass-spring-damper model of the elastic transmission is represented by

$$\boldsymbol{\tau_{el}} = \boldsymbol{K}\left(\boldsymbol{\theta} - \boldsymbol{q}\right) + \boldsymbol{D}\left(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}\right), \tag{3.25}$$

with $\boldsymbol{K}$ the positive definite diagonal matrix of the spring stiffness and $\boldsymbol{D} \geq 0$ the diagonal matrix of viscous friction coefficients. The motor dynamics are described by the diagonal motor inertia matrix $\boldsymbol{I_m}$, the diagonal viscous damping coefficient matrix $\boldsymbol{D_{v,m}}$, the diagonal coulomb friction coefficient matrix $\boldsymbol{D_{C,m}}$ and the motor torque vector $\boldsymbol{\tau_m}$.

3.2.3 Inverse Dynamics

As described in chapter 3.1 the inverse dynamics problem is calculated using the elastic transmission model. Equation 3.3 provides the motor position as a function of the joint position. This can be insert into the motor dynamics equation 3.1. The remaining unknown variable to calculate the motor torque is the torque applied by the elastic transmission. To calculate this torque the rigid body dynamics of equation 3.5 can be used.

Since the model created for the BioRob-Arm additionally contains damping in the elastic transmission, obtaining the motor position as a function of the joint position from equation 3.25 is not readily possible. Solving equation 3.25 results in $\boldsymbol{\theta} = \boldsymbol{K}^{-1} \cdot \boldsymbol{\tau}_{el} + \boldsymbol{q} - \boldsymbol{D}\left(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}\right)$, with the motor position as a function of the joint position and velocity, as well as of the motor velocity. As described in [39] for a given $\dot{\boldsymbol{\theta}}(0)$, its solution $\dot{\boldsymbol{\theta}}(t)$ is needed to evaluate the nominal torque. In the context of control, this labor can be saved. The spring damping in equation 3.23 and 3.24 is considered as external force resulting in control errors.

So for control purpose the described procedure in chapter 3.1 can be used to obtain the nominal motor torque corresponding to a given trajectory. As final refinement the correction term to receive the equilibrium position has to be considered in torque calculation (as explained in [40]). This results in the following equation, determining the motor position as a function of only the joint position:

$$\boldsymbol{\theta} = \boldsymbol{K}^{-1}\left(\boldsymbol{M}\left(\boldsymbol{q}\right)\ddot{\boldsymbol{q}} + \boldsymbol{C}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} + \boldsymbol{F}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} + \boldsymbol{g}\left(\boldsymbol{q}\right)\right) + \boldsymbol{q} + \boldsymbol{\alpha}_c(\boldsymbol{q}) \tag{3.26}$$

To receive the desired motor torques for the given trajectory equation 3.23 is used, by insertion of the motor velocity and acceleration (achieved from equation 3.26) and elastic torque (achieved from equation 3.25):

$$\boldsymbol{I}_m\ddot{\boldsymbol{\theta}} + \boldsymbol{D}_{v,m}\dot{\boldsymbol{\theta}} + \boldsymbol{D}_{C,m}\mathrm{sign}\left(\dot{\boldsymbol{\theta}}\right) + \boldsymbol{\tau}_{el} = \boldsymbol{\tau}_m. \tag{3.27}$$

How this inverse dynamics computation is used in the control scheme will be shortly explained in the next section.

3.2.4 Position Control

The actual used control scheme is a state space controller. As described in [40] each elastic joint can be described by a system of four first order ordinary differential equations. This determines the length of the state vector. Since the motor and joint positions are provided, these and their derivatives ($\boldsymbol{q}\,\dot{\boldsymbol{q}}\,\boldsymbol{\theta}\,\dot{\boldsymbol{\theta}}$) are used to describe the complete state space.

The trajectory planner only creates the joint position and velocity. As consequence the motor position and velocity have to be calculated. Here the inverse dynamics presented above is used and further simplified. As described in the previous section the damping term, in motor position calculation, is neglected and treated as a control error. Besides the damping term, also the dynamic terms can be ignored, when only using the steady state torque ($\dot{\boldsymbol{q}} = \ddot{\boldsymbol{q}} = 0$). Only the force, caused by gravitational acceleration, influences the steady state torque. This leads to the following simplification of the desired motor position calculation 3.26:

$$\boldsymbol{\theta}_d = \boldsymbol{K}^{-1}\left(g\left(\boldsymbol{q}_d\right)\right) + \boldsymbol{q}_d + \boldsymbol{\alpha}_c(\boldsymbol{q}_d) \tag{3.28}$$

The control loop is split into two parts. The part already described, builds the steady state controller. The second part is an approximative gravitational compensation. With this compensation the gravitational force in the actual position is calculated separately and then added to the controlled force, to receive the whole acting forces in the joint (static and gravitational). This is done to facilitate the controller parameter adjustment independently from the actual gravitational force. Considering the gravitational forces within the control loop can result in inaccurate controlling for arm configurations with low gravitational forces (needs low control parameters) or high gravitational forces (need high control parameters).

All control system parts and how they interact is shown in figure 3.9. The whole state vector is led back and controlled using a P-controller except for the joint positions, these are controlled with a PI-controller. Since the inverse dynamics calculation only provides the desired value of the motor positions, these have to be differentiated to get the motor velocities.

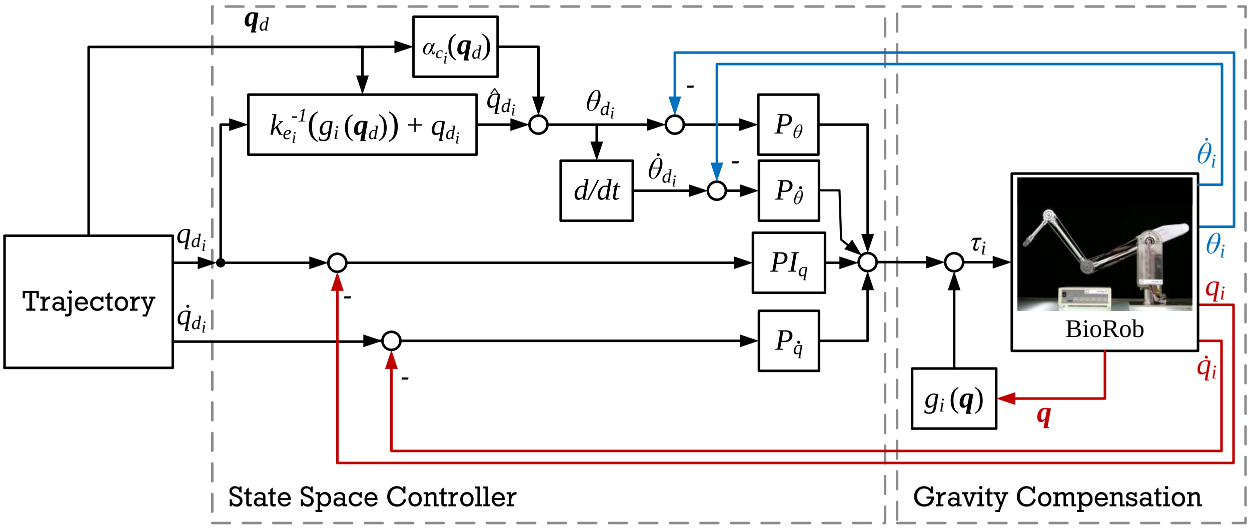


Figure 3.9: Multivariable control structure on joint level (adapted from [40])

3.2.5 Modeling with Matlab/Simulink

To test the parameter estimation and collision detection on the real robot arm, it is useful to create a model for simulation. On this model unpredictable behavior during an execution does not effect the hardware or harm people near the robot arm. Another benefit is that all changes can directly be implemented and their effects simply be observed or analyzed. To realize this kind of model, one can use the Matlab/Simulink application. It allows to build the physical structure using the SimMechanics Toolbox. This physical construction delivers the usually provided sensor data similar to a real system.

To facilitate testing, the Simulink model is build out of blocks, that represent the single parts of the physical structure. On actuator side first, a actuator model is encapsulated in a block (an electric or mechanic model is available). This actuator is further encapsulated within the elastic coupling, to build a whole series elastic actuator model. Another block is build to represent the physical structure of a rigid joint including the succeeding link. As the real prototype evolved, this block also evolved. The modular model design enables one to build an series elastic actuator with an arbitrary number ob joints just by couple one after another. If an algorithm or a specific behavior has to be tested, one can first consider only one or two joints to reduce complexity, before inspecting the complete robot. Apart from the physical structure other blocks have been created, e.g. a controller, a collision detection block and some reaction strategies.

Before using the created model, one has to be sure, that this model represents the realistic behavior of the robot. To do so, the analytical robot dynamics equations can be consulted. These equations can be solved to get the joint acceleration $\ddot{\boldsymbol{q}}$, which can be numerically integrated two times to receive the joint velocity $\dot{\boldsymbol{q}}$ and position $\boldsymbol{q}$. After receiving the joint position and derivatives, these can be compared with the produced corresponding values of the Simulink model. The inverse robot dynamics is also modeled as a block, which requires mass and Coriolis matrix, as well as friction and gravitation vector. These are build automatically from a Newton-Euler recursion for the robot arm with given DH parameter. How these matrices can be extracted from the dynamics equations is described in the next section.

The constructed model is the basis for the investigated algorithms and procedures in this work. How the model is used for the specific task will be considered in the corresponding chapter.

3.3 Calculating Dynamics Equations

There are two fundamental approaches to derive the robots dynamic equations. As described in [43] the Lagrange formulation starts from the total Lagrangian of the system and is energy based. The Newton-Euler in contrast is based on a balance of all the forces acting on the manipulator links. This "force-balance" approach uses elementary dynamics formulas: Newton's second law and Euler's equation. The energy based Lagrange method will not be described here (see [44] or [43]). The Newton-Euler recursion is introduced, because a modification of this method is used for the parameter estimation. An equivalent formulation of this modified method also exists for the Euler-Lagrange approach [43], but will not be discussed here.

Since a force balance is used to derive the equations of motion, the Newton-Euler approach is recursively structured. During the forward recursion the link velocities and accelerations were propagated from the base to the end-effector. This information is used to perform a backward recursion to calculate all acting forces and torques. Figure 3.10 gives an overview of all parameters that are used to analyze each joint during the recursion. All kinematic and dynamic parameters used for the recursion are shown in image 3.10, and listed in the table 3.1 below:

The inverse dynamics computation tries to find the joint moments/forces according to a given trajectory. To calculate these forces first the links linear and angular velocities/accelerations are calculated along the kinematic chain from the base to the end-effector, called "forward recursion". During this recursion, both Newton's second law and Euler's equation are used

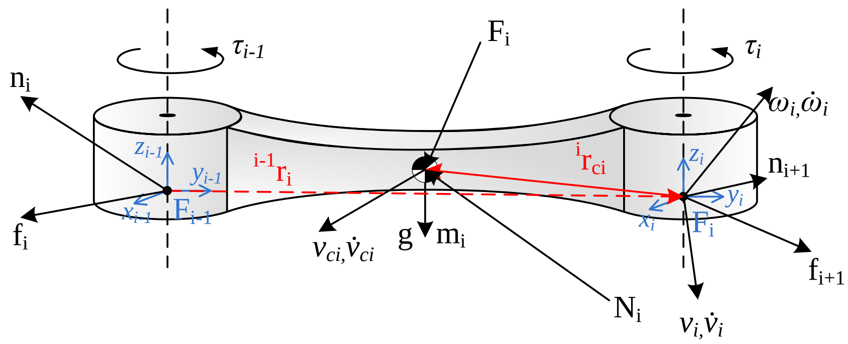


Figure 3.10: Separated link with linear and angular velocities/accelerations, joint torques, center of mass, forces and torques

m_i	Total mass of link i.
τ_i	Joint torque/force at joint i.
$\omega_i, \dot{\omega}_i$	Angular velocity and acceleration of the i-th coordinate frame S_i.
$v_i, \dot{v}_i$	Linear velocity and acceleration of the i-th coordinate frame S_i.
$v_{ci}, \dot{v}_{ci}$	Linear velocity and acceleration of the center of mass of link i.
F_i, N_i	Net force and torque exerted on link i.
f_i, n_i	Force and torque exerted on link i by link $i-1$.
r^i_{ci}	Position of the center of mass of link i.
z_0	z vector of 3-by-3 identity matrix, $z_0 = (0, 0, 1)^T$.
R^{i-1}_i	Orthogonal rotation matrix, which transforms a vector in the i-th coordinate frame to a coordinate frame, which is parallel to the $(i-1)$-th coordinate frame, for $i = 1, 2, \ldots, n$, where $R^n_{n+1} = I$.
I^{ci}_i	Inertia tensor of link i expressed about the center of mass of link i.

Table 3.1: Kineamtic and dynamic parameters for Newton-Euler recursion

to calculate the forces and torques at the links' center of masses. The following well known equations are used to determine the needed values:

$$\omega_i = R^i_{i-1}\left(\omega_{i-1} + \sigma_i z_0 \dot{q}_i\right) \tag{3.29}$$

$$\dot{\omega}_i = R^i_{i-1}\left(\dot{\omega}_{i-1} + \sigma_i\left(z_0\ddot{q}_i + \omega_{i-1} \times \left(z_0\dot{q}_i\right)\right)\right) \tag{3.30}$$

$$\begin{aligned}\dot{\boldsymbol{v}}_{\boldsymbol{i}} = R^i_{i-1}\dot{\boldsymbol{v}}_{\boldsymbol{i}} + \dot{\omega}_i \times \left(R^i_{i-1}\boldsymbol{r}^{\boldsymbol{i-1}}_{\boldsymbol{i}}\right)\omega_i \times \left(\omega_i \times \left(R^i_{i-1}\boldsymbol{r}^{\boldsymbol{i-1}}_{\boldsymbol{i}}\right)\right) + \\ \left(1-\sigma_i\right)\left(1\omega_i \times z_0\dot{q}_i + z_0\ddot{q}_i\right)\end{aligned} \tag{3.31}$$

$$\dot{\boldsymbol{v}}_{\boldsymbol{ci}} = \dot{\boldsymbol{v}}_{\boldsymbol{i}} + \dot{\omega}_i \times \boldsymbol{r}^{\,i}_{ci} + \omega_i \times \left(\omega_i \times \boldsymbol{r}^{\,i}_{ci}\right) \tag{3.32}$$

$$\boldsymbol{F}_{\boldsymbol{i}} = m_i \dot{\boldsymbol{v}}_{\boldsymbol{ci}} \tag{3.33}$$

$$\boldsymbol{N}_{\boldsymbol{i}} = I^{ci}_i \dot{\omega}_i + \omega \times \left(I^{ci}_i \omega_i\right) \tag{3.34}$$

The calculated forces and torques acting at the center of mass are now used to determine the forces and torques at the joints. This is done in opposite direction from the end-effector to the base ("backward recursion"), where external forces can be considered. To calculate the joint torques the following equations are used:

$$\boldsymbol{f_i} = R^i_{i+1}\boldsymbol{f_{i+1}} + \boldsymbol{F_i} \tag{3.35}$$

$$\boldsymbol{n_i} = R^i_{i+1}\boldsymbol{n_{i+1}} + \left(R^i_{i-1}\boldsymbol{r}_i^{i-1} + \boldsymbol{r}^{\,i}_{ci}\right) \times \boldsymbol{F_i} + \left(R^i_{i-1}\boldsymbol{r}_i^{i-1}\right) \times \left(R^i_{i+1}\boldsymbol{f}_{\,i+1}\right) + \boldsymbol{N_i} \tag{3.36}$$

$$\tau_i = \begin{cases} f_i^t \left(R_i^{i-1}\right)^t z_0, & \sigma_i = 0 \\ n_i^t \left(R_i^{i-1}\right)^t z_0, & \sigma_i = 1 \end{cases} \tag{3.37}$$

where the external force and torque at the end-effector are expressed with f_{n+1} and n_{n+1} respectively. The parameter σ_i represents the joint type: $\sigma_i = 0$ for translational and $\sigma_i = 1$ for a revolute joint. Further the angular velocity and acceleration of the robot's base can be assumed as $\omega_0 = 0$ and $\dot{\omega}_0 = 0$ if the robot arm is fixed. In contrast to these values, the linear acceleration of the robot's base is not set to zero, but equals the gravitational acceleration $\boldsymbol{\dot{v}_0} = (g_x, g_y, g_z)^t$.

To carry out the Newton-Euler recursion the following parameters are required: inertia tensor I_i^{ci}, mass m_i, center of mass position r^i_{ci}, as well as the local rotation matrix R_i^{i-1} and translation vector r_i^{i-1}, which describes the frame transformation from S_i to S_{i-1}. Additional information and the derivation of the listed equations, as well as examples are described e.g. in [44] or [43].

Extracting dynamic matrices from dynamic equations

The resulting equations from the Newton-Euler recursion are not in closed form (grouped corresponding to the dynamic matrices). The simplest way to receive the dynamic matrices is to eliminate all expressions that are not contained in a matrix. This approach holds for the mass matrix, Coriolis and gravitation vector. But if the Coriolis matrix is needed, one has to use the mass matrix and the Christoffel symbols.

The general matrix form of the dynamic equation is:

$$\boldsymbol{\tau} = \boldsymbol{M}\left(\boldsymbol{q}\right)\ddot{\boldsymbol{q}} + \boldsymbol{C}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} + \boldsymbol{g}\left(\boldsymbol{q}\right). \tag{3.38}$$

One simply sees, that only the mass matrix is multiplied by the joint acceleration. If the joint velocity and the gravitation vector are set to zero $\dot{\boldsymbol{q}} = \boldsymbol{g} = 0$ in each dynamics equation the reminder left hand side only contains this multiplication. For joint i this can be written as follows:

$$\sum_{j=1}^{n} d_{ij}(\boldsymbol{q})\ddot{q}_j. \tag{3.39}$$

To get the matrix entry in row i and column j, the joint acceleration $\ddot{q}_j$ with $i = j$ has to be set to one and those for $i \neq j$ to zero. If this is done for each joint i, one receives the mass matrix with all entries i, j.

The next matrix considered is the Coriolis matrix. Using the Christoffel symbols, as defined in [43] the terms of the dynamics equation each joint i, corresponding to the Coriolis matrix, can be written as:

$$\sum_{j=1}^{n}\sum_{k=1}^{n} c_{ijk}(\boldsymbol{q})\dot{q}_k\dot{q}_j, \quad \text{where} \quad c_{ijk} = \frac{1}{2}\left(\frac{\partial m_{ij}}{\partial q_k} + \frac{\partial m_{ik}}{\partial q_j} - \frac{\partial m_{jk}}{\partial q_i}\right), \tag{3.40}$$

with c_{ijk} known as Christoffel symbols. The term $c_{ijj}(\boldsymbol{q})\dot{q}_j^2$ represents the centrifugal effect induced on joint i by velocity of joint j. Further the term $c_{ijk}(\boldsymbol{q})\dot{q}_k\dot{q}_j$ represents the Coriolis effect induced on joint i by velocities of joint j and k. As in the common matrix-vector form the Coriolis term is written as $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$, the elements of matrix $\boldsymbol{C}$ satisfy the equation

$$\sum_{j=1}^{n} c_{ij}\dot{q}_j = \sum_{j=1}^{n}\sum_{k=1}^{n} c_{ijk}(\boldsymbol{q})\dot{q}_k\dot{q}_j. \tag{3.41}$$

As consequence the (i,j)-th element of the matrix $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ is defined as

$$c_{ij} = \sum_{k=1}^{n} c_{ijk}(\boldsymbol{q})\dot{q}_k = \sum_{k=1}^{n} \frac{1}{2}\left(\frac{\partial m_{ij}}{\partial q_k} + \frac{\partial m_{ik}}{\partial q_j} - \frac{\partial m_{jk}}{\partial q_i}\right)\dot{q}_k \tag{3.42}$$

In contrast of using the Christoffel symbols to create the Coriolis matrix the Coriolis vector $\boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}})$, where the joint velocities are already multiplied with the Coriolis matrix, can be extracted directly from the dynamic equations. The terms that belong to the mass matrix or the gravitation vector can be deleted by setting $\ddot{\boldsymbol{q}} = \boldsymbol{g} = 0$. The remaining terms directly form the Coriolis vector entries. Since the combination of how the joint velocities are multiplied within the Coriolis vector is not unique, it is not possible to reconstruct the Coriolis matrix out of this vector. That is the reason why the Christoffel symbols are used.

Analog to the Coriolis vector, the gravitational vector can be achieved. The gravitational parts of the dynamic equations are independent on joint velocity and acceleration. Setting these to zero $\ddot{\boldsymbol{q}} = \dot{\boldsymbol{q}} = 0$ yields to the gravitational vector entries.

After these steps, the mass and Coriolis matrix, as well as the Coriolis and gravitational vector are received and can be used e.g. in the inverse dynamics model.

3.4 Conclusion

This chapter introduced a compact description of how to model a series elastic actuator. The introduction first gave an overview what a SEA consists of and what assumptions can be made to facilitate modeling. After a general explanation how to model the dynamics and how to receive the inverse dynamics, the BioRob-Arm was introduced as example for a SEA.

Besides the mechanical design of the drive train and other design choices made for the BioRob-Arm, all general modeling steps mentioned in the introduction were explained in detail for the BioRob-Arm. First the kinematic model was derived using the standard Denavit-Hartenberg notation. To calculate the equilibrium positions the coupling of the fourth and third joint has to be considered.

To receive the dynamics model, all parts were investigated separately and then combined. This step included to design the elastic transmission and how to reflect all motor parameters through this transmission. The reflected values were derived step by step by considering the whole elastic drive train and the inherent transmission ratio. To couple the actuator and load side dynamics into one equation system the force/torque transmission has been observed, resulting in the complete dynamics model

$$\begin{aligned} \boldsymbol{I}_m\ddot{\boldsymbol{\theta}} + \boldsymbol{D}_{v,m}\dot{\boldsymbol{\theta}} + \boldsymbol{D}_{C,m}\operatorname{sign}\left(\dot{\boldsymbol{\theta}}\right) + \boldsymbol{K}\left(\boldsymbol{\theta} - \boldsymbol{q}\right) + \boldsymbol{D}\left(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}\right) &= \boldsymbol{\tau}_m \\ \boldsymbol{M}\left(\boldsymbol{q}\right)\ddot{\boldsymbol{q}} + \boldsymbol{C}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} + \boldsymbol{F}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} + \boldsymbol{g}\left(\boldsymbol{q}\right) &= \boldsymbol{K}\left(\boldsymbol{\theta} - \boldsymbol{q}\right) + \boldsymbol{D}\left(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}\right), \end{aligned}$$

containing the elastic transmission

$$\boldsymbol{\tau}_{el} = \boldsymbol{K}\left(\boldsymbol{\theta} - \boldsymbol{q}\right) + \boldsymbol{D}\left(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}\right).$$

After deriving the dynamics equations, the inverse dynamics has been focused. The motor position estimation is a nontrivial task, if damping is included in the elastic transmission. Since the inverse dynamics is used in case of the control system, damping can be neglected and assumed as control error. With this assumption, the motor position has been formulated as a function of only joint variables and their derivatives:

$$\boldsymbol{\theta} = \boldsymbol{K}^{-1}\left(\boldsymbol{M}\left(\boldsymbol{q}\right)\ddot{\boldsymbol{q}} + \boldsymbol{C}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} + \boldsymbol{F}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} + \boldsymbol{g}\left(\boldsymbol{q}\right)\right) + \boldsymbol{q} + \boldsymbol{\alpha}_c(\boldsymbol{q})$$

Subsequent to the inverse dynamics of the elastic system, the controller was introduced. The control system is a state space controller that consists of two parts. The first part linearizes the dynamic system so that it can be described by $\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\theta}$ and $\dot{\boldsymbol{\theta}}$ in state space. Each joint can then be controlled by a linear controller for all states. Besides this part, additionally a gravitation compensation is build into the control loop.

After all dynamic parts have been created, a short overview of the built Matlab model for simulation has been given. Finally, the Newton-Euler recursion has been introduced to create a rigid dynamics model. Since the produced dynamics equation is not in matrix vector form, a simple procedure is shown to receive the dynamic matrices and vectors, as post-processing of the recursion. The next chapter deals with the parameter identification of a series elastic actuator.

4 Parameter Identification

4.1 Introduction

Parameter identification of a robot model can be done in various ways. As already described in chapter 2.3 the geometrical or inertial parameters can be extracted from a CAD model. But the accuracy of the extracted parameter depend on the information available to build a CAD model. Perhaps material informations are not known or the modeling of all parts are not possible. Also disassembling is not an alternative of choice because it is time consuming and the interdependent parameters (inertia values) can not be determined. Swevers ([37]) claimed that an accurate dynamic robot model is required to increase quality, reduce costs and time of manufacturing, since such model enables the support of off-line programming by simulation and accurate motion control. Further he argued that experimental identification is the only efficient way to obtain an accurate model.

There are two ways in identifying model parameters. Static methods estimate parameters when the robot is not moving. For example the spring stiffness can be determined, by fixing the robot motor positions in a particular configuration and weight a link down with a certain load. This moves the link and elongates the spring between link and motor. As the elongation can be measured with the motor and joint position sensors and the weight and lever arm is known, the resulting torque can be computed. If this is done for various loads, the spring characteristics is detected. This procedure is simple but time consuming. Such experiments further allows to identify the geometrical parameters. For example the length and weight of links can simply be measured before assembling. This approach is not applicable to determine dynamic parameters e.g. the spring damping or inertial parameters.

In contrast to static experiments, parameters can be identified with the robot in motion. This motion is created with a trajectory, so that the dynamic parameters are estimated as accurate as possible. Aim of the parameter identification introduced in this work is to estimate the rigid body inertias, as well as the motor frictions and elastic transmission parameters. The basis for this estimation is a linear dynamics model. As described in [45] all parameters can be combined in a $N_{par} \times 1$ parameter vector $\boldsymbol{\vartheta} = \left(\vartheta_1, \ldots, \vartheta_{N_{par}}\right)^T$. The output of the system is represented by the $n \times 1$ vector $\boldsymbol{\tau}$ and the coefficients of the linear dynamic model by an $n \times N_{par}$ matrix $\boldsymbol{\phi}$:

$$\boldsymbol{\tau} = \boldsymbol{\phi}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\boldsymbol{\vartheta}. \tag{4.1}$$

One possibility to determine the parameters is to take P measurements during the motion of the system (e.g. the torques, joint position, velocity and acceleration) and stack the resulting equation 4.1 for each measurement to a new linear equation system:

$$\boldsymbol{\tau_{tot}} = \boldsymbol{\phi_{tot}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\boldsymbol{\vartheta}, \tag{4.2}$$

with $\boldsymbol{\tau_{tot}}$ is an $n \cdot P \times 1$ vector of all output measurements and $\boldsymbol{\phi_{tot}}$ a $n \cdot P \times N_{par}$ matrix. One solution for the parameter estimation is to use ordinary least squares:

$$\boldsymbol{\vartheta} = (\boldsymbol{\phi_{tot}}^T \boldsymbol{\phi_{tot}})^{-1} \boldsymbol{\phi_{tot}}^T \boldsymbol{\tau_{tot}}. \tag{4.3}$$

Since the dynamics equation is rewritten as linear equation system 4.1 that is used to perform a linear regression (solving 4.1 with least squares), 4.1 is called regressor form of the dynamics equation and the matrix $\boldsymbol{\phi}$ is called regressor. Rank deficiencies can cause problems in inverting $\boldsymbol{\phi_{tot}}^T\boldsymbol{\phi_{tot}}$. Reasons for such rank deficiencies (extracted from [45]) can be inadequate data or unidentifiable parameters. To evaluate the data quality an observability index, such as the condition number of the regressor matrix can be used. Because of the robot's structure some parameters could not be identifiable. These parameters have to be detected and eliminated from estimation procedure.

The identification procedure presented in this work tries to combine the two approaches presented in [38] and [36]. As discussed in [38] the parameter estimation for series elastic actuators can be divided in two parts. First, the parameters (including motor friction and elastic transmission stiffness/damping) on actuator side are identified after creation of the regressor form for the actuator side dynamics. This regressor form is build with the symbolic algorithm proposed in [38]. After this identification, the load side parameters can be examined. Here, [38] received the symbolic load dynamic equations from a Lagrange-Euler algorithm. Using these created equations does not take into account, that the center of mass position vector is nonlinear. To get all load side parameters linear in the estimation process, the proposed procedure of [36] is used. Here the link inertia tensors are expressed about the link coordinate frames instead of the center of mass frames. To reduce the created regressor model [36] categorizes the parameter in identifiable, identifiable in linear combinations and unidentifiable considering a random trajectory. This trajectory is sampled and the measured values stacked as described above. On base of the regressors rank, the parameters are categorized. Since this approach is based on a particular trajectory, the categorization can be different for different trajectories. This work used the proposed procedure to get a regressor form with linear parameters but categorized the parameters inspired from the algorithm in [38]. This combination of both approaches led to an identification framework, that identifies the parameters with a general symbolic regressor form where all parameters appear linear.

4.2 Methodology of Parameter Identification

<table>
<tr><th>Actuator side</th><th>Load side</th></tr>
<tr><td>Create dynamic equation</td><td>Build regressor form of dynamic equation with modified Newton-Euler recursion.</td></tr>
<tr><td>Build regressor form</td><td>Reduce model to the minimal set of dynamic parameters (base parameter set)</td></tr>
<tr><td colspan="2">Determine the optimal excitation trajectory for parameter estimation</td></tr>
<tr><td colspan="2">Estimation of model parameters</td></tr>
</table>

Table 4.1: Identification steps on actuator and load side

As outlined in chapter 4.1, this work tries to combine two parameter estimation approaches, to build a general framework for creation of a robot dynamics model with the highest possible accuracy. In the following section, the theoretical background of the combined parts are explained step by step, building the whole framework. Since the parameter estimation on actuator and load side are treated differently, they will be explained separately, starting on actuator side. If all parameters on actuator side are estimated, they are used to identify the parameters of the rigid load side structure. The identification process contains the steps described in table 4.1.

4.2.1 Build Regressor Form of Actuator Dynamics

The actuator dynamics for one joint (see equation 3.21) consists of three parts on the right hand side:

the motor dynamics	$I_m\ddot{\theta}$,
the friction model	$d_{v,m}\dot{\theta} + d_{C,m}\text{sign}(\dot{\theta})$,
the elastic transmission model	$k_e\left(\theta - q\right) + d_e\left(\dot{\theta} - \dot{q}\right)$,

and the resulting motor torque τ_m on the left hand side. Besides the motor dynamics, the friction and elastic transmission model can be freely formulated or replaced by other models (that depend linear on the model parameters). For that reason these parts have to be provided by the user. Out of these terms the framework then automatically builds the whole actuator dynamic equation.

As described in [38], it is easy to find the regressor form by simply rearranging the terms that consists of parameter independent expressions, unknown parameters or products with unknown parameters. Using the algorithm proposed in [38] the dynamic equation is treated as a text string and than analyzed as follows. First the string is divided in subexpressions separated by $+$ and $-$ operators, followed by further dividing the subexpressions separated by $*$ and / operators. Each expression is then put into one of the three groups (parameter independent expression, unknown parameter or product with unknown parameter). These groups exactly correspond to the different parts of the regressor form: Parameter independent expressions belong to the torque vector $\boldsymbol{\tau}$ of the regressor form, expressions that form products with unknown parameters constitute the matrix $\boldsymbol{\phi}$, and unknown parameters produces the vector $\boldsymbol{\vartheta}$.

For the above one joint example this rearranging can simply be done and results in the following regressor form:

$$\tau_m - I_m\ddot{\theta} = \left[\dot{\theta} - \dot{q}, \dot{\theta}, \theta - q, \text{sign}(\dot{\theta})\right] \cdot \begin{bmatrix} d_e \\ d_{v,m} \\ k_e \\ d_{C,m} \end{bmatrix}. \tag{4.4}$$

The regressor form of the actuator side (equation 4.4) directly represents the base regressor form, because it consists of the minimal number of unknown parameters that describes the dynamic equations. Since there are no mutual dependencies between the actuator parameters as well as no linear dependencies between the regressor rows (in the n-DoF case), the regressor form (with the used friction and transmission models) cannot be further reduced. Nevertheless the framework tries to find a base regressor model according to the algorithm of [38] because this could be necessary for other actuator models. The reduction process will be discussed later in detail in context of the load side dynamics.

4.2.2 Build Regressor Form of Load Dynamics

The actuator dynamic parameters all appear linearly in the dynamic equations. This is not the case for the rigid load side dynamics. The produced equations of motion from Newton-Euler recursion or Lagrange-Euler algorithm are nonlinear in the center-of-mass vectors. To overcome with this [36] used a modified Newton-Euler recursion proposed by [34] to receive a model that is linear in all dynamic parameters. This modified Newton-Euler recursion will be described in the following paragraphs. Each link can be described by ten inertial parameters ([44],[36]): the total mass, the three center-of-mass coordinates scaled by the link mass, and the six entries of the inertia tensor (three inertia moments and three inertia products).

As shown in chapter 3.3 the joint forces and torques within the Newton-Euler recursion are calculated as follows:

$$\boldsymbol{f_i} = R^i_{i+1}\boldsymbol{f_{i+1}} + \boldsymbol{F_i} \tag{4.5}$$

$$\boldsymbol{n_i} = R^i_{i+1}\boldsymbol{n_{i+1}} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i + \boldsymbol{r}^i_{ci}\right) \times \boldsymbol{F_i} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \left(R^i_{i+1}\boldsymbol{f}_{i+1}\right) + \boldsymbol{N_i} \tag{4.6}$$

If the inertia tensor, as described in [34], is expressed about the link coordinate frame instead of the center-of-mass frame, the resulting model is linear in all dynamic parameters. The inertia tensor expressed in the link coordinate frame can be computed according to the parallel-axis theorem (Steiner's law) as:

$$\boldsymbol{I'_i} = \boldsymbol{I^{ci}_i} + m_i\left((\boldsymbol{r}^i_{ci})^T\boldsymbol{r}^i_{ci}\boldsymbol{E} - \boldsymbol{r}^i_{ci}(\boldsymbol{r}^i_{ci})^T\right), \tag{4.7}$$

with the 3×3 identity matrix $\boldsymbol{E}$.

Since the joint force $\boldsymbol{f_i}$ is not influenced by the inertia tensor, equation 4.5 has not to be changed. Equation 4.6 is modified as follows.

$$\boldsymbol{n_i} = R^i_{i+1}\boldsymbol{n_{i+1}} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i + \boldsymbol{r}^i_{ci}\right) \times \boldsymbol{F_i} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \underbrace{\left(R^i_{i+1}\boldsymbol{f}_{i+1}\right)}_{=\ f_i - F_i \quad (4.5)} + \boldsymbol{N_i} \tag{4.8}$$

$$\Leftrightarrow \boldsymbol{n_i} = R^i_{i+1}\boldsymbol{n_{i+1}} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i + \boldsymbol{r}^i_{ci}\right) \times \boldsymbol{F_i} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \left(\boldsymbol{f_i} - \boldsymbol{F_i}\right) + \boldsymbol{N_i} \tag{4.9}$$

$$\Leftrightarrow \boldsymbol{n_i} = R^i_{i+1}\boldsymbol{n_{i+1}} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \boldsymbol{F_i} + \boldsymbol{r}^i_{ci} \times \boldsymbol{F_i} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \boldsymbol{f_i} - \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \boldsymbol{F_i} + \boldsymbol{N_i} \tag{4.10}$$

$$\Leftrightarrow \boldsymbol{n_i} = R^i_{i+1}\boldsymbol{n_{i+1}} + \boldsymbol{r}^i_{ci} \times \boldsymbol{F_i} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \boldsymbol{f_i} + \boldsymbol{N_i} \tag{4.11}$$

After substitution of $\boldsymbol{N_i}$ and $\boldsymbol{F_i}$ from equation 3.34 and 3.33 in equation 4.11, one obtains:

$$\boldsymbol{n_i} = R^i_{i+1}\boldsymbol{n_{i+1}} + \underbrace{\boldsymbol{r}^i_{ci} \times m_i \dot{\boldsymbol{v}}_{\boldsymbol{ci}}}_{=\ m_i \boldsymbol{r}^i_{ci} \times \dot{\boldsymbol{v}}_{\boldsymbol{ci}}} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \boldsymbol{f_i} + I^{ci}_i \dot{\omega}_i + \omega \times \left(I^{ci}_i \omega_i\right). \tag{4.12}$$

Further substitution of $\dot{\boldsymbol{v}}_{\boldsymbol{ci}}$ from equation 3.32 leads to:

$$\begin{aligned}\boldsymbol{n_i} = R^i_{i+1}\boldsymbol{n_{i+1}} + &\underbrace{m_i \cdot \boldsymbol{r}^i_{ci} \times \left(\dot{\boldsymbol{v}}_{\boldsymbol{i}} + \dot{\omega}_i \times \boldsymbol{r}^i_{ci} + \omega_i \times \left(\omega_i \times \boldsymbol{r}^i_{ci}\right)\right)}_{(1)} \\ &+ \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \boldsymbol{f_i} + \underbrace{I^{ci}_i \dot{\omega}_i + \omega \times \left(I^{ci}_i \omega_i\right)}_{(2)}.\end{aligned} \tag{4.13}$$

To facilitate the formulation, the following identities are introduced:

$$\begin{aligned} a \times (b \times (b \times a)) &= b \times \left[a^T a \boldsymbol{E} - a a^T\right] b \\ a \times (b \times a) &= \left[a^T a \boldsymbol{E} - a a^T\right] b \end{aligned}$$

Substituting this identities in equation 4.13 sub term (1) and 4.7 into sub term (2) one gets:

$$\begin{aligned} \boldsymbol{n_i} = R^i_{i+1}\boldsymbol{n_{i+1}} &+ \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \boldsymbol{f_i} + m_i \boldsymbol{r}^i_{ci} \times \dot{\boldsymbol{v}}_{\boldsymbol{i}} \\ &+ m_i \left[(\boldsymbol{r}^i_{ci})^T \boldsymbol{r}^i_{ci} \boldsymbol{E} - \boldsymbol{r}^i_{ci}(\boldsymbol{r}^i_{ci})^T\right] \dot{\omega}_i + m_i \omega_i \times \left[(\boldsymbol{r}^i_{ci})^T \boldsymbol{r}^i_{ci} \boldsymbol{E} - \boldsymbol{r}^i_{ci}(\boldsymbol{r}^i_{ci})^T\right] \omega_i \\ &+ I^{'}_i \dot{\omega}_i - m_i \left[(\boldsymbol{r}^i_{ci})^T \boldsymbol{r}^i_{ci} \boldsymbol{E} - \boldsymbol{r}^i_{ci}(\boldsymbol{r}^i_{ci})^T\right] \dot{\omega}_i - m_i \omega_i \times \left[(\boldsymbol{r}^i_{ci})^T \boldsymbol{r}^i_{ci} \boldsymbol{E} - \boldsymbol{r}^i_{ci}(\boldsymbol{r}^i_{ci})^T\right] \omega_i \\ &+ \omega \times \left(I^{'}_i \omega_i\right) \\ \Leftrightarrow \boldsymbol{n_i} = R^i_{i+1}\boldsymbol{n_{i+1}} &+ \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \boldsymbol{f_i} + m_i \boldsymbol{r}^i_{ci} \times \dot{\boldsymbol{v}}_{\boldsymbol{i}} + I^{'}_i \dot{\omega}_i + \omega \times \left(I^{'}_i \omega_i\right) \end{aligned} \tag{4.14}$$

which is the final joint torque equation with the inertia tensor expressed about the link coordinate frame. For the next formulation, one has to insert equation 4.5, 3.33 and 3.32 into 4.14. This leads to:

$$\begin{aligned} \boldsymbol{n_i} &= R^i_{i+1}\boldsymbol{n_{i+1}} + m_i \boldsymbol{r}^i_{ci} \times \dot{\boldsymbol{v}}_{\boldsymbol{i}} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \left(R^i_{i+1}\boldsymbol{f_{i+1}} + \boldsymbol{F_i}\right) + I^{'}_i \dot{\omega}_i + \omega \times \left(I^{'}_i \omega_i\right) \\ \Leftrightarrow \boldsymbol{n_i} &= R^i_{i+1}\boldsymbol{n_{i+1}} + m_i \boldsymbol{r}^i_{ci} \times \dot{\boldsymbol{v}}_{\boldsymbol{i}} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \left(R^i_{i+1}\boldsymbol{f_{i+1}}\right) + \\ &\qquad \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \boldsymbol{F_i} + I^{'}_i \dot{\omega}_i + \omega \times \left(I^{'}_i \omega_i\right) \\ \Leftrightarrow \boldsymbol{n_i} &= R^i_{i+1}\boldsymbol{n_{i+1}} + m_i \boldsymbol{r}^i_{ci} \times \dot{\boldsymbol{v}}_{\boldsymbol{i}} + \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \left(R^i_{i+1}\boldsymbol{f_{i+1}}\right) + \\ &\qquad m_i \left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \left(\dot{\boldsymbol{v}}_{\boldsymbol{i}} + \dot{\omega}_i \times \boldsymbol{r}^i_{ci} + \omega_i \times \left(\omega_i \times \boldsymbol{r}^i_{ci}\right)\right) + I^{'}_i \dot{\omega}_i + \omega \times \left(I^{'}_i \omega_i\right). \end{aligned} \tag{4.15}$$

After this, one can define a six by one vector $\boldsymbol{\gamma_i}$ as

$$\boldsymbol{\gamma_i} = \left[\boldsymbol{f_i} \quad \boldsymbol{n_i}\right]^T, \tag{4.16}$$

and combine 4.5 and 4.15 into a single matrix-vector equation

$$\boldsymbol{\gamma}_i = \boldsymbol{D}^i_{i+1}\boldsymbol{\gamma}_{i+1} + \boldsymbol{\Gamma}_i, \tag{4.17}$$

where $\boldsymbol{D}^i_{i+1}$ is the six by six pseudo-rotation matrix

$$\boldsymbol{D}^i_{i+1} = \begin{pmatrix} \boldsymbol{R}^i_{i-1} & \boldsymbol{0} \\ \left[\left(\boldsymbol{R}^i_{i-1}\boldsymbol{r}^{i-1}_i\right)x\right]\boldsymbol{R}^i_{i-1} & \boldsymbol{R}^i_{i-1} \end{pmatrix} \tag{4.18}$$

and $\boldsymbol{\gamma}_i$ is defined as

$$\boldsymbol{\Gamma}_i = \begin{pmatrix} \boldsymbol{F}_i \\ \boldsymbol{N}'_i \end{pmatrix}, \tag{4.19}$$

with $\boldsymbol{N}'_i$ being

$$\boldsymbol{N}'_i = m_i\boldsymbol{r}^i_{ci} \times \dot{\boldsymbol{v}}_i + m_i\left(R^i_{i-1}\boldsymbol{r}^{i-1}_i\right) \times \left(\dot{\boldsymbol{v}}_i + \dot{\omega}_i \times \boldsymbol{r}^i_{ci} + \omega_i \times \left(\omega_i \times \boldsymbol{r}^i_{ci}\right)\right) + I'_i\dot{\omega}_i + \omega \times \left(I'_i\omega_i\right) \tag{4.20}$$

and $\boldsymbol{F}_i$ is given by 3.33. In this modified procedure, equation 4.17 represents the backward recursion. In the next step the recursion is expanded to receive a explicit expression for vector $\boldsymbol{\gamma}_i$. Further the vector $\boldsymbol{\Gamma}_i$ is decomposed in a matrix-vector product, of matrix $\boldsymbol{K}_i$ and the dynamic parameters vector $\boldsymbol{\vartheta}_i$ for joint i. With this matrix-vector product and no externally applied forces and torques, the recursion e.g. for a 4-DoF manipulator is:

$$\begin{aligned}
\boldsymbol{\gamma}_4 &= \boldsymbol{D}^4_5\boldsymbol{\gamma}_5 + \boldsymbol{\Gamma}_4 = \boldsymbol{K}_4\boldsymbol{\vartheta}_4 \\
\boldsymbol{\gamma}_3 &= \boldsymbol{D}^3_4\boldsymbol{\gamma}_4 + \boldsymbol{\Gamma}_3 = \boldsymbol{D}^3_4\boldsymbol{K}_4\boldsymbol{\vartheta}_4 + \boldsymbol{K}_3\boldsymbol{\vartheta}_3 \\
\boldsymbol{\gamma}_2 &= \boldsymbol{D}^2_3\boldsymbol{\gamma}_3 + \boldsymbol{\Gamma}_2 = \boldsymbol{D}^2_3\boldsymbol{D}^3_4\boldsymbol{K}_4\boldsymbol{\vartheta}_4 + \boldsymbol{D}^2_3\boldsymbol{K}_3\boldsymbol{\vartheta}_3 + \boldsymbol{K}_2\boldsymbol{\vartheta}_2 \\
\boldsymbol{\gamma}_1 &= \boldsymbol{D}^1_2\boldsymbol{\gamma}_2 + \boldsymbol{\Gamma}_1 = \boldsymbol{D}^1_2\boldsymbol{D}^2_3\boldsymbol{D}^3_4\boldsymbol{K}_4\boldsymbol{\vartheta}_4 + \boldsymbol{D}^1_2\boldsymbol{D}^2_3\boldsymbol{K}_3\boldsymbol{\vartheta}_3 + \boldsymbol{D}^1_2\boldsymbol{K}_2\boldsymbol{\vartheta}_2 + \boldsymbol{K}_1\boldsymbol{\vartheta}_1
\end{aligned} \tag{4.21}$$

With $\boldsymbol{D}^i_j = \boldsymbol{D}^i_{i+1}\boldsymbol{D}^{i+1}_{i+2}\ldots\boldsymbol{D}^{j-1}_j$ this can be written as:

$$\begin{pmatrix} \boldsymbol{\gamma}_1 \\ \boldsymbol{\gamma}_2 \\ \boldsymbol{\gamma}_3 \\ \boldsymbol{\gamma}_4 \end{pmatrix} = \begin{pmatrix} \boldsymbol{K}_1 & \boldsymbol{D}^1_2\boldsymbol{K}_2 & \boldsymbol{D}^1_3\boldsymbol{K}_3 & \boldsymbol{D}^1_4\boldsymbol{K}_4 \\ \boldsymbol{0} & \boldsymbol{K}_2 & \boldsymbol{D}^2_3\boldsymbol{K}_3 & \boldsymbol{D}^2_4\boldsymbol{K}_4 \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{K}_3 & \boldsymbol{D}^3_4\boldsymbol{K}_4 \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{K}_4 \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{\vartheta}_1 \\ \boldsymbol{\vartheta}_2 \\ \boldsymbol{\vartheta}_3 \\ \boldsymbol{\vartheta}_4 \end{pmatrix} \tag{4.22}$$

or generally

$$\boldsymbol{\gamma}_i = \begin{pmatrix} \boldsymbol{D}^i_1\boldsymbol{K}_1 & \boldsymbol{D}^i_2\boldsymbol{K}_2 & \ldots & \boldsymbol{D}^i_n\boldsymbol{K}_n \end{pmatrix} \cdot \boldsymbol{\vartheta}_i, \tag{4.23}$$

with $\boldsymbol{D}_j^i = \mathbf{0}$ if $j < i$ and $\boldsymbol{D}_j^i = \boldsymbol{E}$ if $j = i$ ($\boldsymbol{E}$: 6×6 identitiy matrix).

As mentioned above the vector $\boldsymbol{\Gamma_i}$ and thus $\begin{pmatrix} \boldsymbol{F_i} & \boldsymbol{N_i'} \end{pmatrix}^T$ can be decomposed as the product of matrix $\boldsymbol{K_i}$ and the parameter vector

$$\boldsymbol{\vartheta_i} = \begin{pmatrix} m_i & m_i r_{ci,x}^i & m_i r_{ci,y}^i & m_i r_{ci,z}^i & I_{i,xx}' & I_{i,yy}' & I_{i,zz}' & I_{i,xy}' & I_{i,xz}' & I_{i,yz}' \end{pmatrix}^T .$$

Before introducing matrix $\boldsymbol{K_i}$ of this decomposition, the following notation is used to denote the cross product of two vectors and the multiplication of a vector by a matrix. If ω_i and $\boldsymbol{a}$ are 3×1 vectors, then $\omega_i \times \boldsymbol{a} = \left[\omega_i x\right] \boldsymbol{a}$ and the multiplication of 3×3 inertia matrix $\boldsymbol{I_i}$ with the vector ω_i is denoted as $\boldsymbol{I_i}\omega_i = \left[\bullet\omega_i\right] \boldsymbol{I_i}$, where

$$\left[\omega_i x\right] = \begin{pmatrix} 0 & -\omega_{i,z} & \omega_{i,y} \\ \omega_{i,z} & 0 & -\omega_{i,x} \\ -\omega_{i,y} & \omega_{i,x} & 0 \end{pmatrix} \qquad \left[\bullet\omega_i\right] = \begin{pmatrix} \omega_{i,x} & 0 & 0 & \omega_{i,y} & \omega_{i,z} & 0 \\ 0 & \omega_{i,y} & 0 & \omega_{i,x} & 0 & \omega_{i,z} \\ 0 & 0 & \omega_{i,z} & 0 & \omega_{i,x} & \omega_{i,y} \end{pmatrix} \tag{4.24}$$

and

$$\boldsymbol{I_i} = \begin{pmatrix} I_{i,xx} & I_{i,yy} & I_{i,zz} & I_{i,xy} & I_{i,xz} & I_{i,yz} \end{pmatrix}^T .$$

With this notation the vectors $\boldsymbol{F_i}$ and $\boldsymbol{N_i'}$ can be rewritten as:

$$\begin{aligned} \boldsymbol{F_i} =& \boldsymbol{\dot{v}_i} m_i + \left[\dot{\omega}_i x\right] m_i \boldsymbol{r}_{ci}^i + \left[\omega_i x\right] \left[\omega_i x\right] m_i \boldsymbol{r}_{ci}^i \\ \boldsymbol{N_i'} =& \left[\left(R_{i-1}^i \boldsymbol{r}_i^{i-1}\right) x\right] \boldsymbol{\dot{v}_i} m_i + \left[-\boldsymbol{\dot{v}_i} x\right] m_i \boldsymbol{r}_{ci}^i + \left[\left(R_{i-1}^i \boldsymbol{r}_i^{i-1}\right) x\right] \left[\dot{\omega}_i x\right] m_i \boldsymbol{r}_{ci}^i + \\ & \left[\left(R_{i-1}^i \boldsymbol{r}_i^{i-1}\right) x\right] \left[\omega_i x\right] \left[\omega_i x\right] m_i \boldsymbol{r}_{ci}^i + \left[\bullet\omega_i\right] \boldsymbol{I_i'} + \left[\omega_i x\right] \left[\bullet\omega_i\right] \boldsymbol{I_i'} \end{aligned} \tag{4.25}$$

and simply be decomposed into

$$\begin{aligned} &\boldsymbol{\Gamma_i} = \begin{pmatrix} \boldsymbol{F_i} \\ \boldsymbol{N_i'} \end{pmatrix} = \boldsymbol{K_i} \boldsymbol{\vartheta_i} = \\ &\begin{pmatrix} \boldsymbol{\dot{v}_i} & \left[\dot{\omega}_i x\right] + \left[\omega_i x\right] \left[\omega_i x\right] & \mathbf{0} \\ \left[\boldsymbol{p_i} x\right] \boldsymbol{\dot{v}_i} & \left[-\boldsymbol{\dot{v}_i} x\right] + \left[\boldsymbol{p_i} x\right] \left[\dot{\omega}_i x\right] + \left[\boldsymbol{p_i} x\right] \left[\omega_i x\right] \left[\omega_i x\right] & \left[\bullet\omega_i\right] + \left[\omega_i x\right] \left[\bullet\omega_i\right] \end{pmatrix} \boldsymbol{\vartheta_i}, \end{aligned} \tag{4.26}$$

with $\boldsymbol{p_i} = R_{i-1}^i \boldsymbol{r}_i^{i-1}$. Matrix $\boldsymbol{K_i}$ can be semantically divided into submatrices as follows:

$$\boldsymbol{K_i} = \begin{pmatrix} \boldsymbol{K_{i,11}} & \boldsymbol{K_{i,12}} & \boldsymbol{K_{i,13}} \\ \boldsymbol{K_{i,21}} & \boldsymbol{K_{i,22}} & \boldsymbol{K_{i,23}} \end{pmatrix} \tag{4.27}$$

where $\boldsymbol{K_{i,11}}$ and $\boldsymbol{K_{i,21}}$ are of dimension 3×1 multiplied by m_i, $\boldsymbol{K_{i,21}}$ and $\boldsymbol{K_{i,22}}$ are of dimension 3×3 multiplied by $m_i r_{ci}^i$ and $\boldsymbol{K_{i,13}}$ and $\boldsymbol{K_{i,23}}$ are of dimension 3×6 multiplied by $\boldsymbol{I_i'}$.

After calculation of the backward recursion 4.22, the acting torques for joint i are chosen from the following matrix-vector equation:

$$
\begin{aligned}
\boldsymbol{\tau}_i &= \begin{pmatrix} \left(\boldsymbol{R}_i^{i-1}\right)^T (1-\sigma_i)\,\boldsymbol{z}_0 \\ \left(\boldsymbol{R}_i^{i-1}\right)^T \sigma_i \boldsymbol{z}_0 \end{pmatrix}^T \cdot \boldsymbol{\gamma}_i \\
&= \underbrace{\begin{pmatrix} \left(\boldsymbol{R}_i^{i-1}\right)^T (1-\sigma_i)\,\boldsymbol{z}_0 \\ \left(\boldsymbol{R}_i^{i-1}\right)^T \sigma_i \boldsymbol{z}_0 \end{pmatrix}^T \cdot \begin{pmatrix} \boldsymbol{D}_1^i \boldsymbol{K}_1 & \boldsymbol{D}_2^i \boldsymbol{K}_2 & \dots & \boldsymbol{D}_n^i \boldsymbol{K}_n \end{pmatrix}}_{=\boldsymbol{\phi}_i} \boldsymbol{\vartheta}_i
\end{aligned} \tag{4.28}
$$

with $\boldsymbol{\phi}_i$ represents the $1 \times 10n$ regressor row of joint i. The whole dynamic model 4.1 received with this modified Newton-Euler recursion is now linear in all parameters and directly provides the $n \times 10n$ regressor matrix.

In case of the BioRob-Arm additional joint friction parameters need to be estimated. For example when a viscous friction model $d_i \cdot \dot{q}_i$ is assumed for each joint, one just has to append the friction coefficient (e.g. $\dot{q}_i$) into the regressor row $\boldsymbol{\phi}_i$ and the friction parameter d_i into the parameter vector $\boldsymbol{\vartheta}$. This results in a rigid dynamic model with friction, where each regressor row consists of $1 \times 11n$ elements. Since perhaps not all parameters are identifiable for one particular robot arm, the so received regressor matrix has to be reduced. How this is realized will be described in the next paragraph.

Receiving the base regressor form

As mentioned in [36] the regressor matrix produced by the modified Newton-Euler recursion is not invertible due to loss of rank from restricted degrees of freedom and linear dependencies between the columns of $\boldsymbol{\phi}$. The model reduction to the minimal set of dynamic parameters that describe the system used in [36] is based on the categorization method that investigates the linear dependencies between the regressor columns numerically for a particular trajectory. This method categorizes the parameters into unknown parameters, parameters that appear in linear combination and parameter that are not identifiable. Since the results of analyzing the linear dependencies depend on the used trajectory, the categorization can change for different trajectories. That is the reason why this work uses a more general approach mentioned by [38] that symbolically reduces the regressor form.

To find the base regressor form, all linear dependencies have to be found. To realize this [38] used a reasoning approach proposed by [46]. In order that the following procedure successfully finds all linear dependencies, all regressor entries have to be simplified regarding the trigonometric function. The reasoning approach is done in two steps, first all fundamental functions included in $\boldsymbol{\phi}$ are identified, then a new matrix is created which is transformed to the Echelon form by a Gauss-Jordan elimination. The resulting upper triangular matrix contains the linear dependencies between all regressor columns. These dependencies are used to get the base regressor form and the minimal set of unknown dynamic parameters. As defined by [46] fundamental function of the regressor are time-varying functions of joint variables and their time derivatives, that satisfies the following two conditions:

1. Any fundamental function can not be represented by a linear combination of the other fundamental functions.
2. All elements of $\boldsymbol{\phi}$ can be represented by a linear combination of the fundamental functions.

Since the regressor entries are analyzed symbolically (analogue to the algorithm described in 4.2.1) the possible base functions out of which the fundamental functions are build (e.g. sin, cos or sign) have to be user provided, to identify the fundamental functions. In addition to the fundamental function, regressor entries can consist of system parameters that are multiplied by certain fundamental functions. After investigation of all entries in $\boldsymbol{\phi}$ all found fundamental functions are stored in the vector $\boldsymbol{f} \in \mathfrak{R}^k$, with k the number of fundamental functions in $\boldsymbol{\phi}$. The corresponding known system parameters are stored in a vector $\boldsymbol{b}_{ij} \in \mathfrak{R}^k$ such that the multiplication of both results in the original regressor entry:

$$\boldsymbol{\phi}_{ij}\left(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}\right) = \boldsymbol{f}^T\left(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}\right) \boldsymbol{b}_{ij} \tag{4.29}$$

The found system parameter vectors $\boldsymbol{b}_{ij}$ of the corresponding regressor entry $\boldsymbol{\phi}_{ij}$ in row i and column j are used to build a new matrix $\boldsymbol{B} \in \mathfrak{R}^{l \times N_{par}}$, $l = nk$:

$$\boldsymbol{B} = \begin{pmatrix} \boldsymbol{b}_{11} & \cdots & \boldsymbol{b}_{1N_{par}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{b}_{n1} & \cdots & \boldsymbol{b}_{nN_{par}} \end{pmatrix} \tag{4.30}$$

In the next step this matrix is transformed into the Echelon form $\boldsymbol{B}_E$ by performing a Gauss-Jordan elimination. The resulting matrix

$$\boldsymbol{B}_E = \begin{pmatrix} \boldsymbol{B}_{Eu} \\ \boldsymbol{0} \end{pmatrix} \tag{4.31}$$

consists of three different types of columns, that describe the linear dependencies between the columns of $\boldsymbol{\phi}$. If all elements equal zero, the corresponding regressor column is not needed an can be removed. Thus the corresponding dynamic parameter is not identifiable. The second type of columns consists of only zero elements except for one entry that is one. For this type the corresponding regressor column is linear independent and necessary for identification. The last column type contains zeros and known parameter elements. Here the corresponding regressor column is linear dependent on other columns and can be removed. The columns on which it is linear dependent is represented by the row index of the elements different from zero. These elements themselves represent the coefficients of the linear combination. After removing all regressor columns that are not necessary, resulting in the base regressor representation $\boldsymbol{\phi}_0$, also the unknown dynamic parameters have to be reduces to achieve the base regressor parameters $\boldsymbol{\vartheta}_0$. This is done by the following multiplication:

$$\boldsymbol{\vartheta}_0 = \boldsymbol{B}_{Eu} \boldsymbol{\vartheta}. \tag{4.32}$$

This multiplication deletes all not identifiable dynamic parameters, and linearly combines them according to the linear dependencies of the regressor columns. In contrast to the used actuator model, where all unknown parameters can be identified separately, the unknown dynamic parameters partially can only be identified in linear combination of other parameters.

In addition to this model reduction, the framework numerically checks for a random trajectory, if the derived regressor has full rank. The model reduction on symbolical expressions requires a certain simplification of the dynamic expressions in the regressor matrix to find all linear dependencies. If this simplification is not effective, the resulting base regressor is not of full rank. In this case the model can be numerically reduced analogue to the symbolical approach by creating the stacked regressor matrix $\boldsymbol{\phi}_{tot}$ (see chapter 4.1). This matrix is then transformed using the Gauss-Jordan elimination to get all linear dependencies. The resulting upper triangular matrix again contains all information to reduce the symbolical regressor as well as to produce the base parameter set. Since this reduction is done based on a particular trajectory the reduction can be different for varying trajectories, as the case in [36].

The next section describes how to find a trajectory, that allows the best possible identification. Such trajectory is called persistently exciting.

4.2.3 Persistently Excitation Trajectories

After creation of the actuator side and load side regressor form, the resulting linear equation system has to be solved. Since this is only possible if the regressor matrix has full rank, the base regressor and base parameter set will be denoted by $\boldsymbol{\phi}$ and $\boldsymbol{\vartheta}$ (instead of $\boldsymbol{\phi}_0$ and $\boldsymbol{\vartheta}_0$). As described in chapter 4.1 the parameters can be determined by taking P measurements during a system motion and stack the regressor form for each measurement. Since the estimation quality varies for different trajectories, one have to find the best suitable one for estimation of the particular robot arm. This is done by finding the trajectory parameters that optimize an observability measure.

As exciting trajectory a parameterized function such as a polynomial [38] or finite Fourier series [37],[36] can be used. The angular position q_i, velocity $\dot{q}_i$ and acceleration $\ddot{q}_i$ for joint i at time t expressed as parameterized polynomial can be written as:

$$q_i(t) = \sum_{l=0}^{n} a_l t^l \qquad \dot{q}_i(t) = \sum_{l=1}^{n} l a_l t^{l-1} \qquad \ddot{q}_i(t) = \sum_{l=2}^{n} (l^2 - l) a_l t^{l-2} \tag{4.33}$$

where n represents the polynomials order. Alternatively a finite sum of harmonic sine and cosine functions like finite Fourier series can describe $q_i, \dot{q}_i$ and $\ddot{q}_i$ as:

$$
\begin{aligned}
q_i(t) &= \sum_{l=1}^{n} \left(\frac{a_{i,l}}{\omega_f l} \sin(\omega_f l t) - \frac{b_{i,l}}{\omega_f l} \cos(\omega_f l t) \right) + q_{i,0} \\
\dot{q}_i(t) &= \sum_{l=1}^{n} \left(a_{i,l} \cos(\omega_f l t) + b_{i,l} \sin(\omega_f l t) \right) \\
\ddot{q}_i(t) &= \sum_{l=1}^{n} \left(-a_{i,l} \omega_f l \sin(\omega_f l t) + b_{i,l} \omega_f l \cos(\omega_f l t) \right)
\end{aligned}
\tag{4.34}
$$

with ω_f the fundamental pulsation of the Fourier series. The periodic functions specified by the Fourier series have a time period $T_f = 2\pi/\omega_f$. Describing the joint motion with this function results in $2n+1$ parameters, $a_{i,l}$ and $b_{i,l}$ for $l = 1, \ldots, n$, which are the amplitudes of the sine and cosine functions, and $q_{i,0}$ which is the offset on the position trajectory. The trajectory parameters for joint i are represented by the parameter vector $\boldsymbol{\delta_i}$.

Compared with this functions, the optimization problem for polynomials contains $n+1$ degrees of freedom, whereas for Fourier series $2n+1$ and therefor is more complex. But on the opposite [37] mentions that, Fourier series are periodic and thus simply allow to repeat a motion for time-domain data averaging. This improves the signal-to-noise ratio of the experimental data.

The trajectory parameters have to be determined in a way, that the parameter estimation results in exact values. To judge how suitable a trajectory is for estimation, there exist several observability indices. To check this indices the actual trajectory is sampled P times and a stacked regression matrix $\boldsymbol{\phi_{tot}}$is produced. As listed in [45], optimal experimental design theory has given rise to several data measures. The most significant alphabet-optimalities are:

- A-optimality: minimize the trace of $(\boldsymbol{\phi}^T \boldsymbol{\phi})^{-1}$
- D-optimality: maximize the determinant of $\boldsymbol{\phi}^T \boldsymbol{\phi}$
- E-optimality: maximize the minimum singular value of $\boldsymbol{\phi}^T \boldsymbol{\phi}$

For robot calibration observability indices were introduces by [47], and shortly described in the next paragraphs. Some of these also have an alphabet-optimality counterpart. Observability index O_1 represents the root of the product of the singular values of $\boldsymbol{\phi}$:

$$
O_1 = \frac{(\sigma_1 \ \sigma_1 \cdots \sigma_m)^{1/m}}{\sqrt{m}} \tag{4.35}
$$

and is similar to d-optimality. Both formulations, d-optimality and O_1, represent the volume of the confidence hyper ellipsoid in the measurements τ with axis lengths corresponding to the singular values. Maximizing O_1 gives the largest hyper ellipsoid volume similar to maximizing the determinant of the information Matrix $\boldsymbol{M} = \boldsymbol{\phi}^T \boldsymbol{\phi}$. Since maximizing the determinant of the information matrix $\boldsymbol{M}$ is equivalent to minimizing the variance $\mathrm{Var}[\boldsymbol{\vartheta}] = \sigma \left(\boldsymbol{\phi}^T \boldsymbol{\phi} \right)^{-1}$ of the estimated parameters, it is claimed that this index is the best.

Another way to measure the observability is to minimize the condition number of $\boldsymbol{\phi}$. This measure does not have an counterpart in optimality alphabet.

$$O_2 = \frac{\mu_1}{\mu_r} \tag{4.36}$$

which measures the eccentricity of the hyper ellipsoid rather that its size, where μ_1 is the biggest and μ_r the smallest singular values. Minimizing the condition number automatically makes all singular values become close to each other and rather forms a hypersphere than a hyper ellipsoid. One point of critique of O_1 has been that it may result in favoring one direction over another to maximize the volume, this is avoided by O_2.

Similar to e-optimality, the minimum singular value μ_r can be maximized:

$$O_3 = \mu_r. \tag{4.37}$$

Here, the aim is to make the shortest axis as long as possible, regardless of the other axis. This tries to improve the worst case. The a-optimality does not have a counterpart in robot calibration literature. Since d-optimality seems to be the best choice, as it maximizes the confidence for all parameters and minimizes the estimation variance, it is used in this work but with a little modification. As proposed in [37] the d-optimality criterion can be expressed by

$$O_1 = -\log \det \boldsymbol{M}. \tag{4.38}$$

This measure keeps the above mentioned properties, especially it is independent of the scaling of the parameters. Additionally the negative log term assists the optimization algorithm, since little changes near the optimum (high determinant value) result in bigger variations of the objective function.

The excitation trajectory should not violate the actuation constraints as maximum and minimum joint position, velocity and acceleration. In Addition the trajectory should not cause collisions with the robot itself or it's the environment. Taking this into account, the complete optimization problem is formulated as:

$$\min_{\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}}} -\log(\det \boldsymbol{M}) \quad \text{s.t.} \quad \begin{array}{l} q_{min} \leq \boldsymbol{q} \leq q_{max} \\ \dot{q}_{min} \leq \dot{\boldsymbol{q}} \leq \dot{q}_{max} \\ \ddot{q}_{min} \leq \ddot{\boldsymbol{q}} \leq \ddot{q}_{max} \end{array} \tag{4.39}$$

where the joint position, velocity and acceleration trajectories are represented by $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ respectively, under lower joint, velocity and acceleration bounds q_{min}, $\dot{q}_{min}$ and $\ddot{q}_{min}$, as well as upper joint, velocity and acceleration bounds q_{max}, $\dot{q}_{max}$ and $\ddot{q}_{max}$ respectively.

As proposed in [36] a first initial guess for the trajectory should not violate the physical robot limits. Analogue to [36] the initial trajecotry parameters were generated using a simple least-squares method, in this work. Since the joint velocities tend to be the limiting constraints on the allowed trajectories these are taken as basis for all joints. To create the least squares solution the finite Fourier-series for the joint velocity is reformulated as a matrix-vector equation:

$$\dot{\boldsymbol{q}}_i = \boldsymbol{A}\boldsymbol{\delta}_i \tag{4.40}$$

with

$$
\begin{aligned}
\dot{\boldsymbol{q}}_{\boldsymbol{i}} &= \begin{pmatrix} \dot{q}_i(t_1) & \cdots & \dot{q}_i(t_P) \end{pmatrix}^T \\
\boldsymbol{\delta}_{\boldsymbol{i}} &= \begin{pmatrix} a_{i,1} & b_{i,1} & \cdots & a_{i,N} & b_{i,N} & q_{i,0} \end{pmatrix}^T \\
\boldsymbol{A} &= \begin{pmatrix} \frac{\sin(\omega_f 1 t_1)}{\omega_f 1} & -\frac{\cos(\omega_f 1 t_1)}{\omega_f 1} & \cdots & \frac{\sin(\omega_f N t_1)}{\omega_f N} & -\frac{\cos(\omega_f N t_1)}{\omega_f N} & 1 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ \frac{\sin(\omega_f 1 t_P)}{\omega_f 1} & -\frac{\cos(\omega_f 1 t_P)}{\omega_f 1} & \cdots & \frac{\sin(\omega_f N t_P)}{\omega_f N} & -\frac{\cos(\omega_f N t_P)}{\omega_f N} & 1 \end{pmatrix}
\end{aligned}
\tag{4.41}
$$

For computation of the initial trajectory parameters a random set of P possible velocities (by equidistant sample times) in between the velocity bounds are generated. The parameter vector $\boldsymbol{\delta}_{\boldsymbol{i}}$ is then determined with the standard least-squares approach:

$$
\boldsymbol{\delta}_{\boldsymbol{i}} = \left(\boldsymbol{A}^T \boldsymbol{A}\right)^{-1} \boldsymbol{A}^T \dot{\boldsymbol{q}}_{\boldsymbol{i}}. \tag{4.42}
$$

The trajectory parameters for each joint are combined into one column vector and passed to the optimization algorithm. This tries to vary the trajectory parameters of all joints to minimize the objective function by taking the joint, velocity, acceleration and Cartesian constraints into account. The resulting excitation trajectories are then used to estimate the parameters, which is described in the next section.

4.2.4 Parameter Estimation

After knowing the most suitable trajectory for parameter estimation, this trajectory is used to move the robot arm and to measure the joint positions, velocities, accelerations and torques at P time instants $t_1, \ldots, t_P$. These measurements are used to build up the whole linear equation system:

$$
\boldsymbol{\tau}_{\boldsymbol{tot}} = \boldsymbol{\phi}_{\boldsymbol{tot}} \boldsymbol{\vartheta},
$$

with

$$
\begin{aligned}
\boldsymbol{\tau}_{\boldsymbol{tot}} &= \begin{pmatrix} \boldsymbol{\tau}_{\boldsymbol{t_1}} \cdots \boldsymbol{\tau}_{\boldsymbol{t_P}} \end{pmatrix}^T \\
\boldsymbol{\phi}_{\boldsymbol{tot}} &= \begin{pmatrix} \boldsymbol{\phi}\left(q_{t_1}, \dot{q}_{t_1}, \ddot{q}_{t_1}\right) \\ \cdots \\ \boldsymbol{\phi}\left(q_{t_P}, \dot{q}_{t_P}, \ddot{q}_{t_P}\right) \end{pmatrix}
\end{aligned}
$$

the stacked joint torque vector $\boldsymbol{\tau}_{\boldsymbol{tot}}$ and regressor matrix $\boldsymbol{\phi}_{\boldsymbol{tot}}$. There are at least the number of base parameters equations (and thus time instants) necessary to solve this equation system. To increase the reliability of the results and decrease influence of noise it is useful to take more measurements into account.

After model reduction the regressor matrix is invertible and the equation system can be solved by a standard linear least squares estimation:

$$\boldsymbol{\vartheta}_{ls} = \left(\boldsymbol{\phi}_{tot}{}^{T}\boldsymbol{\phi}_{tot}\right)^{-1}\boldsymbol{\phi}_{tot}{}^{T}\boldsymbol{\tau}_{tot}. \tag{4.43}$$

This least squares estimation is sensitive to noise in joint torque, position, velocity and acceleration measurement. To overcome this [37] proposed an iterative procedure that estimates the Fourier series parameters and robot parameters simultaneously, by minimizing one global maximum-likelihood criterion. If the measured joint angles are free of noise this maximum-likelihood estimation simplifies significantly to a weighted least squares estimate:

$$\boldsymbol{\vartheta}_{wls} = \left(\boldsymbol{\phi}_{tot}{}^{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}_{tot}\right)^{-1}\boldsymbol{\phi}_{tot}{}^{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\tau}_{tot} \tag{4.44}$$

with the weighting function $\boldsymbol{\Sigma}^{-1}$ is the reciprocal of the variances of the measured torque values. The assumption of noise free joint position measures can be justified by the fact that the joint torque noise level is much higher than the noise level on position measurements.

As already mentioned, to improve the signal-to-noise ratio, the data sequences can be averaged over a certain number of periods S. The averaged values and variances are calculated according to the following formulas:

$$\bar{q}_i(k) = \frac{1}{S}\sum_{j=1}^{S} q_{ij}(k) \qquad \bar{\tau}_i(k) = \frac{1}{S}\sum_{j=1}^{S} q_{ij}(k) \tag{4.45}$$

$$\sigma_{q_i}^2 = \frac{1}{S-1}\frac{1}{S}\sum_{j=1}^{S}\left(q_{ij}(k) - \bar{q}_i(k)\right)^2 \qquad \sigma_{\tau_i}^2 = \frac{1}{S-1}\frac{1}{S}\sum_{j=1}^{S}\left(\tau_{ij}(k) - \bar{\tau}_i(k)\right)^2 \tag{4.46}$$

with P is equal to the number of samples per period $k \in [1 \dots P]$, j represents the excitation period, $\bar{q}_i(k)$ and $\bar{\tau}_i(k)$ are the averaged joint positions and torque measurements at time instance k, as well as $\sigma_{q_i}^2$ and $\sigma_{\tau_i}^2$ the variances of the joint positions and torque measurements respectively.

With this averaged data the final model parameter estimation can performed. To receive the not measurable joint velocities and accelerations, the averaged joint positions can be approximated by functions as splines and then analytical differentiated to receive the missing values. After this preprocessing the analytical functions can be sampled to extract the time samples for creation of the regressor matrix $\boldsymbol{\phi}_{tot}$. The averaged joint torques are directly used to form the torque vector $\boldsymbol{\tau}_{tot}$. The diagonal weighting matrix Σ^{-1} is created with the reciprocal of the torque variances $\sigma_{\tau_i}^2$:

$$\begin{aligned} \Sigma_k^{-1} =& \mathrm{diag}\left(1/\sigma_{\tau_1}^2, \dots, 1/\sigma_{\tau_n}^2\right) \\ \Sigma^{-1} =& \mathrm{diag}\left(\Sigma_1^{-1}, \dots, \Sigma_P^{-1}\right) \end{aligned} \tag{4.47}$$

Such a wighting scheme is also described in [45] and controls the influence of the variable. The larger the variance (uncertainty), the less this variable influences the least-squares solution relative to the other variables.

4.3 Parameter Identification on BioRob

The developed parameter estimation procedure is tested by simulation and experiment. Before trying the estimation on the real robot, the algorithm was evaluated with the proposed Simulink model of the 4-DoF BioRob-Arm in chapter 3.2.5. Aim is to determine all actuator side and load side parameters. Both dynamic models, on actuator and load sides were calculated as described in chapter 4.2.1 and chaper 4.2.2.

According to equation 4.4 for each joint, the created regressor form on actuator side contains a 4×16 regressor matrix, a 16×1 base parameter vector and 4×1 motor torque vector. As explained above this linear equation system only included single unknown parameters, so that all actuator parameters can be estimated without dependencies on other parameters. On load side, the modified Newton-Euler recursion and preceding model reduction produced a regressor form with a 4×22 regressor matrix, a 22×1 base parameter vector and 4×1 joint torque vector. In contrast to the actuator side, most of the base parameters were linear combinations of model parameters. After creation of both regression models the parameter identification algorithm can be evaluated.

4.3.1 Evaluation by Simulation

Before getting the excitation trajectory, some decisions according to the following questions have to be made. What kind of parameterized function is suitable for receiving an excitation trajectory? What degree of the chosen function is suitable for the problem? And finally, what is an appropriate observability measure? To answer these questions, a few optimization cycles were carried out in context of excitation trajectories for actuator side. To evaluate the quality of excitation trajectory the d-optimality criterion is used. The optimization is carried out for Fourier-series and polynomial functions with degree three, four and five. As quality measure (see [48]) the averaged relative error of the estimated parameters $\boldsymbol{\vartheta}_{est}$ in relation to the exact parameters $\boldsymbol{\vartheta}$ used in the simulation model:

$$\epsilon_{AV} = \frac{1}{16} \sum_{i=1}^{16} \left| \frac{\vartheta_i - \vartheta_{est}}{\vartheta_i} \right|, \tag{4.48}$$

the maximum relative error of the estimated parameters:

$$\epsilon_{MAX} = \frac{1}{16} \max_{i=1}^{16} \left| \frac{\vartheta_i - \vartheta_{est}}{\vartheta_i} \right|, \tag{4.49}$$

and the root mean square difference between the estimated $\boldsymbol{\tau}_{est}$ and measured $\boldsymbol{\tau}_i$ torques:

$$\epsilon_{RMS} = \sqrt{\frac{1}{P} \sum_{i=1}^{P} \left(\tau_i - \tau_{i,est} \right)^2} \tag{4.50}$$

are used.

For optimization Matlab provides the nonlinear constraint optimization routine "fmincon". This can be executed by different algorithms that are used during optimization. First of all the routine needs two functions, one with the objective function (calculating d-optimality or condition number in this case), as well as a nonlinear constraint function, that checks whether or not the constraints are violated for the actual trajectory parameters. To evaluate the joint and Cartesian constraints, the trajectories for all joints were sampled and the direct kinematic is used to get the end-effector position. The function returns the l_2-norm of the distance vector to the bounds, regarding all sample points that violate the constraints or the l_2-norm of the distance vector regarding all sample points scaled with the minimum distance (to realize a continuous change between trajectories with and without constraint violations). As Cartesian constraints, one can imagine that the robot arm is placed on a tables corner and should not collide with the tables surface behind it, as well as with its base. For this purpose a security distance around the first link with a certain radius and the distance from the tables surface are inspected. The resulting distance vector for all sample points is treated analogue to the joint constraints. Thus the complete vector returned from this function contains n entries for the evaluated joint constraints, and two entires for the Cartesian constrains.

The optimization algorithms provided for this routine are "interior-point", "trust-region-reflective" and "SQP". To determine the trajectory function, its degree and the optimization algorithm, the actuator side optimization results are examined. The optimization loop starts with a random set of trajectory parameters produces by the least squares approach described in chapter 4.2.3. This parameters do not obligatorily form trajectories that fulfill all joint and Cartesian constraints. So this step is repeated till a valid trajectory is achieved. To check the constraints and create the regressor matrix, the actual trajectory is sampled $P = 2000$ times per period ([37] used 1500 and [36] used 10.000 samples). As described above the averaged relative error, the maximum relative error and the root mean square difference between estimated parameters $\boldsymbol{\vartheta}_{est}$ and exact parameters $\boldsymbol{\vartheta}$ used in the simulation model are computed. The results of the optimization loop are shown in table A.1, A.2 and A.3. One can make four observations from the optimization results:

- For both, the polynomial function and Fourier-Series, and their various degree, the SQP algorithm needs the fewest iterations. The interior point algorithm only needs a few more iterations. Significantly more iterations are needed by the trust region reflective algorithm.
- Accept for degree four, the parameter estimated by the interior point algorithm achieved the smallest averaged and maximum relative error.
- The polynomial function with various degree always produced an estimation result with a bigger averaged and maximum relative error, in comparison with the corresponding Fourier-Series.
- The resulting root mean square error of the estimated elastic transmission torque are tendentially smaller for Fourier-Series.

Since the interior point algorithm produces equal or better results than the sqp algorithm, with only a few more iterations, this algorithm is used for parameter estimation. In addition the

Fourier-Series with degree five, seems to outperform all other function-degree combinations, so the interior point algorithm has to optimize this function, to produce an excitation trajectory.

	Joint 1			Joint 2		
	exact	estimated	error	exact	estimated	error
d_e $[\frac{Nm \cdot s}{rad}]$	0.20	0.156	$4.35 \cdot 10^{-2}$	0.20	0.20	$2.01 \cdot 10^{-4}$
$d_{v,m}$ $[\frac{Nm \cdot s}{rad}]$	$3.072 \cdot 10^{-4}$	$6.901 \cdot 10^{-4}$	$3.83 \cdot 10^{-4}$	$3.630 \cdot 10^{-3}$	$9.634 \cdot 10^{-4}$	$6.00 \cdot 10^{-4}$
k_e $[\frac{Nm}{rad}]$	17.5	17.403	$9.69 \cdot 10^{-2}$	10.0	9.980	$2.15 \cdot 10^{-2}$
$d_{C,m}$ $[Nm]$	0.016	0.0189	$2.85 \cdot 10^{-3}$	0.0209	0.0219	$1.06 \cdot 10^{-3}$
	Joint 3			**Joint 4**		
	exact	estimated	error	exact	estimated	error
d_e $[\frac{Nm \cdot s}{rad}]$	0.10	0.0827	$1.73 \cdot 10^{-2}$	0.10	0.0824	$1.76 \cdot 10^{-2}$
$d_{v,m}$ $[\frac{Nm \cdot s}{rad}]$	$2.041 \cdot 10^{-4}$	$-7.538 \cdot 10^{-5}$	$2.80 \cdot 10^{-4}$	$2.315 \cdot 10^{-4}$	$2.759 \cdot 10^{-4}$	$4.44 \cdot 10^{-5}$
k_e $[\frac{Nm}{rad}]$	6.0	5.985	$1.54 \cdot 10^{-2}$	6.0	6.013	$1.34 \cdot 10^{-2}$
$d_{C,m}$ $[Nm]$	0.0104	0.00996	$4.72 \cdot 10^{-4}$	0.011	0.0105	$6.10 \cdot 10^{-4}$

Table 4.2: Actuator side base parameters with the exact simulation value and the resulting estimation

The trajectory which achieves the best results in the optimization procedure estimated the actuator parameters listed in table 4.2. In a further step it has to be evaluated, if the estimated parameters generalizes well, or if they only describes the actuator dynamics in the special case of the exciting trajectory. For this purpose [37] used a validation trajectory that goes through 20 points randomly chosen in the workspace of the robot. Between these points the robot moves with maximum acceleration and deceleration and stops at each point. In the case of the BioRob-Arm the validation trajectory is constructed analogue, but with only six random chosen points. The best and worst motor torque estimation result in joint space, from the determined actuator model parameters, in case of the evaluation and excitation trajectory is depicted in figure A.1 and A.2. The corresponding quality measures for all joints are listed in table 4.3.

	trajectories		
	evaluation	excitation	random
ϵ_{RMS} Joint 1	$1.32 \cdot 10^{-1}$	$6.58 \cdot 10^{-2}$	$6.25 \cdot 10^{-2}$
ϵ_{RMS} Joint 2	$1.95 \cdot 10^{-1}$	$3.94 \cdot 10^{-2}$	$9.07 \cdot 10^{-2}$
ϵ_{RMS} Joint 3	$4.28 \cdot 10^{-2}$	$8.84 \cdot 10^{-2}$	$6.32 \cdot 10^{-2}$
ϵ_{RMS} Joint 4	$1.34 \cdot 10^{-1}$	$4.76 \cdot 10^{-2}$	$2.91 \cdot 10^{-1}$

Table 4.3: Error of motor torque estimation received from estimated actuator model

The root mean square error of the estimated motor torques in table 4.3 show that the estimation is more accurate for the excitation trajectory. But even the motor torques needed to track the evaluation trajectory are well estimated. One cause for the accuracy loss could be the discontinuous desired values of the evaluation trajectory forces the controller to produce high torque peaks and oscillation as shown in figure A.1. As neutral and every day trajectory, the model is also evaluated with a random trajectory, produced like the initial trajectories for the optimization process. Since this trajectory is continuous, the controller produces continuous desired values. This behavior is common to the behavior in a standard pick and place task and nearly produces the same accurate torque estimation than for excitation trajectory.

The load side parameters are estimated analogue to the actuator parameters. Additionally the optimization is done two times, with d-optimality and with the regressor's condition number as objective function. [37] and [45] suggest, that the d-optimality criterion is the most suitable observability index for parameter identification. In contrast to this statement [36] received a small improvement using the condition number as observability criterion. Because this matter of fact it seems to be advisable investigate the results of both measures in case of the BioRob-Arm. This is done during load side parameter estimation.

To facilitate the load side parameter estimation, the model parameters used in the Newton-Euler recursion to create the dynamic equations can be slightly reduced. Since the link coordinate frames are attached to the links center of mass, the cross products of inertia are identically zero, if the mass distribution is symmetric with respect to the center of mass (see [44]). This assumption holds for link one, three and four. At link two, only the x-z-plane and y-z-plane have symmetric mass distributions. As consequence, the following cross product of inertia can be set to zero: $I_{1,xy} = I_{1,xz} = I_{1,yz} = I_{2,xz} = I_{2,yz} = I_{3,xy} = I_{3,xz} = I_{3,yz} = I_{4,xy} = I_{4,xz} = I_{4,yz} = 0$. Because of the symmetric mass distribution also the possible center of mass positions can be reduced to the y-axis in link one, the x-y-plane in link two, as well as the x-axis in link three and four, which leads to : $r^1_{c_1,x} = r^1_{c_1,z} = r^2_{c_2,z} = r^3_{c_3,y} = r^3_{c_3,z} = r^4_{c_4,y} = r^4_{c_4,z} = 0$. To receive the regressor matrix fulfilling this simplifications, the columns to the corresponding parameters are set to zero before eliminating the linear dependent columns.

	D-Optimality	Condition Number
# interation	4	4
# func. eval.	279	267
obj. function	57.54	58.15
ϵ_{AV}	1.11	1.74
ϵ_{MAX}	6.48	10.87
ϵ_{RMS} Joint 1	$4.31 \cdot 10^{-3}$	$3.04 \cdot 10^{-3}$
ϵ_{RMS} Joint 2	$3.42 \cdot 10^{-3}$	$3.45 \cdot 10^{-3}$
ϵ_{RMS} Joint 3	$3.08 \cdot 10^{-3}$	$2.79 \cdot 10^{-3}$
ϵ_{RMS} Joint 4	$1.68 \cdot 10^{-3}$	$2.94 \cdot 10^{-2}$

Table 4.4: Trajectory optimization results and accuracy for load side

After this simplification the estimation with both trajectories received from optimization of d-optimality and condition number provided good estimation results. Further the root mean square error of the estimated elastic transmission torque during the evaluation trajectory for both parameter estimations are almost identical. This suggests that the d-optimality and condition number are suitable observability measures in this case. Since the d-optimality show a slightly improvement, regarding the averaged and maximum relative error, compared to the condition number, this observability measure is used for the experimental identification in the next section. The above described optimization and validation results are shown in table 4.4. To illustrate the load side estimation accuracy the best and worst elastic transmission torque estimation is depicted in figure A.3.

physical meaning	exact	estimation of ϑ	error
$I^1_{yy} + I^2_{yy} - I^2_{zz} + I^3_{yy} - I^3_{zz} + I^4_{yy} - I^4_{zz}$	0	-0.00176	$1.76 \cdot 10^{-3}$
d_1	0.50	0.497	$2.58 \cdot 10^{-3}$
$I^3_{zz}/l^2_3 - I^2_{zz}/l^2_2 + m_2$	-0.416	-0.420	$4.29 \cdot 10^{-3}$
$I^2_{zz}/l_2 + m_2 \cdot r_{c2,x}$	0.160	0.159	$5.53 \cdot 10^{-4}$
$m_2 \cdot r_{c2,y}$	0.0	$6.884 \cdot 10^{-8}$	$6.88 \cdot 10^{-8}$
$I^2_{xx} - I^2_{yy} + I^2_{zz}$	0.00888	0.00991	$1.03 \cdot 10^{-3}$
I^2_{xy}	0.0	$-6.134 \cdot 10^{-5}$	$6.13 \cdot 10^{-5}$
d_2	0.50	0.498	$1.56 \cdot 10^{-3}$
$I^4_{zz}/l^2_4 - I^3_{zz}/l^2_3 + m_3$	0.208	0.253	$4.56 \cdot 10^{-2}$
$I^3_{zz}/l_3 + m_3 \cdot r_{c3,x}$	0.0103	0.00821	$2.13 \cdot 10^{-3}$
$I^3_{xx} - I^3_{yy} + I^3_{zz}$	0.00320	0.00252	$6.81 \cdot 10^{-4}$
d_3	0.50	0.501	$6.95 \cdot 10^{-4}$
$m_4 - I^4_{zz}/l^2_4$	0.0292	-0.0101	$3.92 \cdot 10^{-2}$
$I^4_{zz}/l_4 + m_4 \cdot r_{c4,x}$	0.000601	0.00449	$3.89 \cdot 10^{-3}$
$I^4_{xx} - I^4_{yy} + I^4_{zz}$	$6.189 \cdot 10^{-5}$	$4.251 \cdot 10^{-4}$	$3.63 \cdot 10^{-4}$
d_4	0.50	0.501	$1.06 \cdot 10^{-3}$

Table 4.5: Load side base parameters with the exact simulation value and the resulting estimation

Table 4.5 show the base parameter set, their exact and estimated value. According to the high overlapping of the estimated and measures elastic transmission torques (see figure A.3) the base parameter estimation error is very small for most of them.

4.3.2 Experimental Identification

The optimized trajectories for actuator and load side, where both executed on the BioRob-Arm to estimate the real model parameters. Since Fourier-Series are used to describe the trajectories, they simply can be repeated S times and than averaged, to improve the signal-to-noise ratio. In the actual case the trajectories for actuator side and load side parameter estimation where repeated $S = 16$ times. Before performing parameter estimation with the BioRob-Arm, two problems have to mentioned. First, there are only motor and joint position available, and second no motor torque informations are present.

To create the motor together with joint velocities and accelerations one can use numerical differentiation. But this procedure has the drawback of amplifying high frequency noise, inherently existent because of sensor quantification (see. [48]). To avoid this, the position samples are piecewise fitted by splines which are then differentiated two times, generating smooth velocity and acceleration informations. The motor torque information can be produced by the following equation from [44]:

$$\boldsymbol{\tau}_m = \boldsymbol{k}_t \cdot \boldsymbol{i}_a, \tag{4.51}$$

with $\boldsymbol{\tau}_m$ the motor torque vector, $\boldsymbol{k}_t$ the torque constant vector provided by the motor manufacturer (each entry belongs to one actuator) and the corresponding armature current $\boldsymbol{i}_a$ at the actual time step. Since the motor current sensor only produces a 14 bit signal, one has to interpret this information and project it into its physical meaning. This is realized by measuring the actual motor current and sensor signal while the motor produces a certain torque. The recorded sensor signal is mapped to the current signal to determine the correct conversion factor. After this conversion the produced current represents the absolute value of the effectively used motor current. One way to identify the right current direction and thus the right motor torque direction, is to use the motor voltage and the following equation:

$$\boldsymbol{i}_a' = (\boldsymbol{u} - \boldsymbol{k}_v \cdot \dot{\boldsymbol{\theta}})/\boldsymbol{R}_a, \tag{4.52}$$

with $\boldsymbol{u}$ the actual motor voltage, $\boldsymbol{k}_v$ the motor speed constant, $\dot{\boldsymbol{\theta}}$ the motor velocity and $\boldsymbol{R}_a$ the terminal resistance. The calculated value $\boldsymbol{i}_a'$ represents the motor current, corresponding to the actual motor voltage without the influence of friction. The aim is to infer from the sign of $\boldsymbol{i}_a'$ to the sign of the real current $\boldsymbol{i}_a$. Since friction, not considered in $\boldsymbol{i}_a'$, can change the amplitude and the roots of the motor current, the sign of $\boldsymbol{i}_a'$ can not directly used as sign for the real current $\boldsymbol{i}_a$. But the roots of $\boldsymbol{i}_a'$ indicate a zero crossing in measured the motor current near this time instance. So if a zero crossing in $\boldsymbol{i}_a'$ occurs at time instance t, the minimum of $\boldsymbol{i}_a$ at the time window $t - \epsilon \leq t \leq t + \epsilon$ is defined as real zero crossing and the sign of $\boldsymbol{i}_a$ is switched (ϵ is set to 0.15 seconds). After this procedure the signed motor current is provided to calculate the signed motor torque. The intermediate stages of this procedure are shown in figure 4.1. The black lines represent the motor current without friction $\boldsymbol{i}_a'$ and its sign. The red line forms the raw motor current sensor corrected with the determined conversion factor. One can see, that for each zero crossing in $\boldsymbol{i}_a'$, the final zero crossing is determined by the minimum of $\boldsymbol{i}_{a,raw}$, resulting in the green curve $\boldsymbol{i}_a$. The final motor torque τ_m is than calculated with 4.51.

If the motor torques, as well as the motor and joint positions, velocities and acceleration are available, the estimation procedure can be executed as described for the simulation. The variances and averages are calculated as described with equation 4.45 and 4.46. To carry out the weighted least squares estimation the motor or rather the joint position, velocity and acceleration have to be free of noise. As listed in table 4.6 this variances are in order of magnitudes smaller than the motor torque variances.

The weighted least squares estimation provided the actuator side parameters shown in table 4.7. Since the spring stiffness $k_{e,1}$ and spring damping coefficient $d_{e,i}$ are negative, these parameters have to be set to a plausible value (mechanically negative damping or spring stiffness is not possible). The described Matlab simulation model is used to find the excitation trajectory. The simulation model parameters heavy differ from the real ones. This causes motor positions in the experiment that differ from the motor positions that are produced by the trajectory in

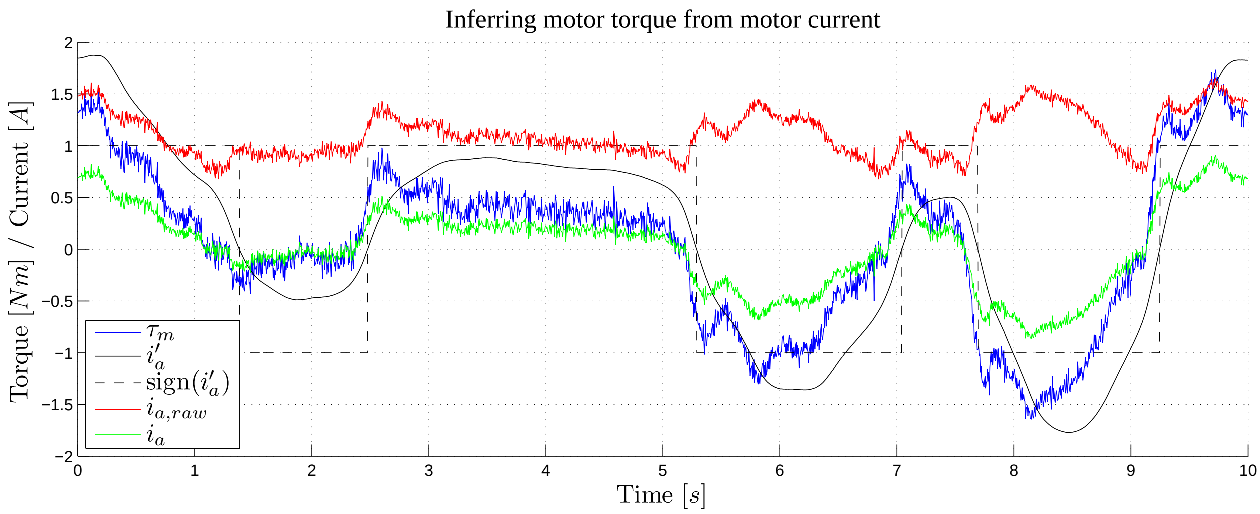


Figure 4.1: Example of motor torque calculation form the measured motor current

	$\sigma^2_{\tau_i}$	$\sigma^2_{q_i}$	$\sigma^2_{\theta_i}$
Joint 1	1.93	$3.14{\cdot}10^{-4}$	$1.59{\cdot}10^{-4}$
Joint 2	3.54	$4.61{\cdot}10^{-4}$	$3.37{\cdot}10^{-4}$
Joint 3	3.19	$1.06{\cdot}10^{-3}$	$7.98{\cdot}10^{-4}$
Joint 4	4.91	$6.24{\cdot}10^{-4}$	$1.05{\cdot}10^{-3}$

Table 4.6: Variances of actuator side parameter estimation

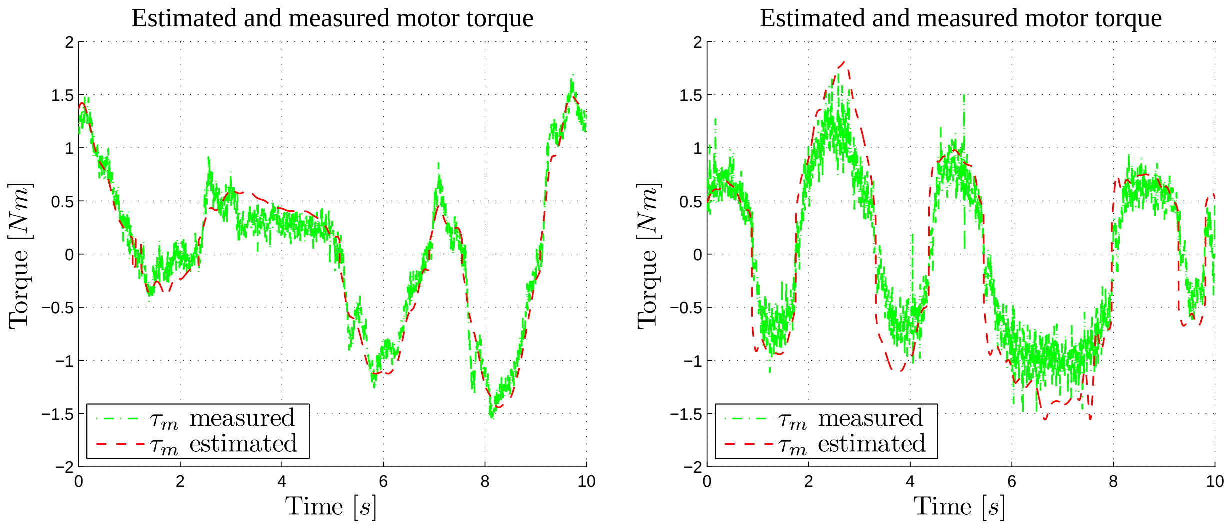
simulation. Thus, the created excitation trajectory in the simulation model must not be perfect for the real model. After the first identification try, the estimated parameters can be used in the simulation model to repeat the whole estimation loop, till the simulation model converges to the real model. The plausible values are set to the absolute value of the negative ones. This correction just marginally influences the quality of the estimation. The root mean square error per joint of the measured and estimated motor torque, as well as the error without the parameters absolute values are listed in table 4.8. The corresponding torque curves of the best and worst result are depicted in figure 4.2

		Joint 1	Joint 2	Joint 3	Joint 4
d_e	$[\frac{Nm{\cdot}s}{rad}]$	0.354	2.031	0.0194	$\lvert -2.401 \rvert$
$d_{v,m}$	$[\frac{Nm{\cdot}s}{rad}]$	1.716	1.814	0.580	0.516
k_e	$[\frac{Nm}{rad}]$	0.563	16.338	3.583	6.228
$d_{C,m}$	$[Nm]$	0.145	0.504	0.140	0.443

Table 4.7: Estimated actuator side base parameters after identification experiment

	estimated	corrected
ϵ_{RMS} Joint 1	0.172	0.171
ϵ_{RMS} Joint 2	0.329	0.329
ϵ_{RMS} Joint 3	0.300	0.300
ϵ_{RMS} Joint 4	0.300	0.374

Table 4.8: Error of motor torque estimation received from experimental actuator identification



(a) Best motor torque estimation for excitation trajectory

(b) Worst motor torque estimation for excitation trajectory

Figure 4.2: Measured and estimated motor torques in case of the excitation trajectory with corrected parameter values

After identification of the actuator model parameters, it is possible to calculate the elastic transmission torque and thus build up the equation system for the load side base parameter estimation. After recording the load side excitation trajectory, again the variances of the acting torques and the joint positions are calculated. These are listed in table 4.9. As for the actuator side the joint position variances are very small. Since the acting torques of the elastic transmission τ_{el} are calculated with the estimated parameters and the measured joint and motor positions, these are nearly free of noise and also produce a small variance.

	$\sigma^2_{\tau_{el}}$	$\sigma^2_{q_i}$
Joint 1	$9.27 \cdot 10^{-3}$	$1.69 \cdot 10^{-4}$
Joint 2	$2.76 \cdot 10^{-1}$	$7.74 \cdot 10^{-4}$
Joint 3	$4.65 \cdot 10^{-3}$	$3.94 \cdot 10^{-4}$
Joint 4	$2.31 \cdot 10^{-1}$	$1.14 \cdot 10^{-4}$

Table 4.9: Variances of load side parameter estimation

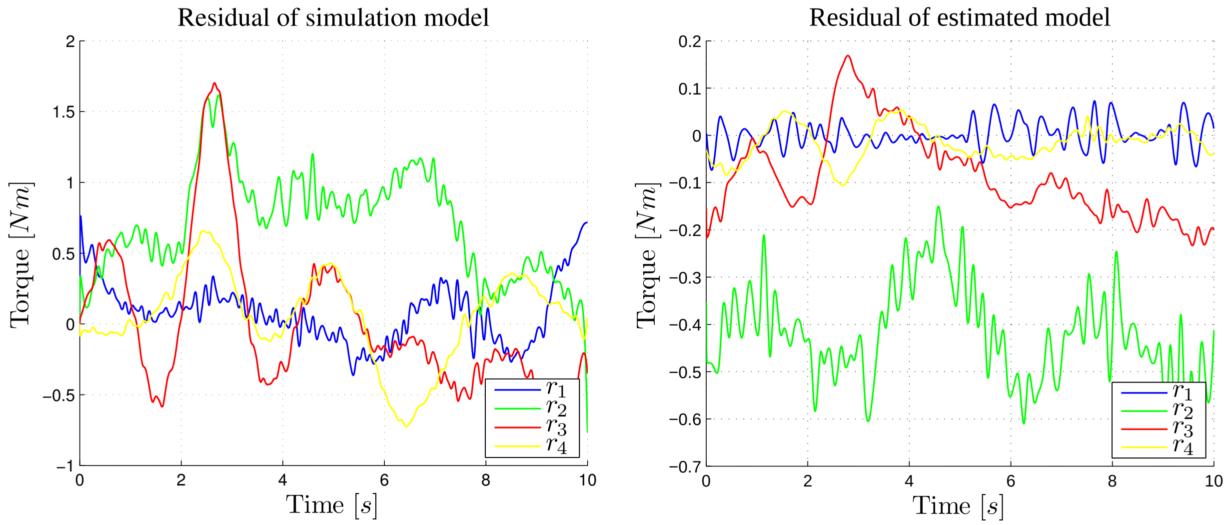
Again, the torques and joint position are averaged over the recorded period. The resulting equation system for the load side base parameters produced the values shown in table 4.10.

physical meaning	estimation of ϑ
$I^1_{yy} + I^2_{yy} - I^2_{zz} + I^3_{yy} - I^3_{zz} + I^4_{yy} - I^4_{zz}$	-0.00935
d_1	-0.0141
$I^3_{zz}/l_3^2 - I^2_{zz}/l_2^2 + m_2$	0.0981
$I^2_{zz}/l_2 + m_2 \cdot r_{c2,x}$	-0.0417
$m_2 \cdot r_{c2,y}$	-0.00126
$I^2_{xx} - I^2_{yy} + I^2_{zz}$	0.00869
I^2_{xy}	-0.000840
d_2	-0.0663
$I^4_{zz}/l_4^2 - I^3_{zz}/l_3^2 + m_3$	0.679
$I^3_{zz}/l_3 + m_3 \cdot r_{c3,x}$	-0.0345
$I^3_{xx} - I^3_{yy} + I^3_{zz}$	0.00554
d_3	0.0267
$m_4 - I^4_{zz}/l_4^2$	-0.484
$I^4_{zz}/l_4 + m_4 \cdot r_{c4,x}$	0.0469
$I^4_{xx} - I^4_{yy} + I^4_{zz}$	-0.00167
d_4	-0.0495

Table 4.10: Experimental estimation of load side base parameters

In contrast to load side base parameter estimation on the Matlab simulation model no joint torque sensors are available for the real robot. Thus it is not possible to directly compare the elastic transmission torque from the robot with the resulting torque from load side estimation. As shortly explained in chapter 2 a fault detection and isolation scheme tries to detect system faults comparing the system output with the output of an model-based observer. One possible observer will be introduced in chapter 5. The calculated residual of the system output and the observer output can not only be used to detect collisions, but also to determine if the model corresponds to the real system. This enables to evaluate the quality of the identified model. As already described, in most cases only linear combination of the single model parameters can be estimated. These base parameters $\boldsymbol{\vartheta}$ in combination with the base regressor $\boldsymbol{\phi}$ describe the whole actuator and load dynamics. Before using the observer, the dynamic matrices have to be extracted from the equation of motion $\boldsymbol{\tau} = \boldsymbol{\phi}\boldsymbol{\vartheta}$. This extraction is done as described in chapter 3.3 and produces the mass matrices $\boldsymbol{M}(\boldsymbol{q}, \boldsymbol{\vartheta})$, the Coriolis matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\vartheta})$, the friction matrix $\boldsymbol{F}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\vartheta})$ and the gravity vector $\boldsymbol{g}(\boldsymbol{q}, \boldsymbol{\vartheta})$ in dependence of the joint positions $\boldsymbol{q}$, joint velocities $\dot{\boldsymbol{q}}$ and the to be estimated base parameter vector $\boldsymbol{\vartheta} = (\vartheta_1, \vartheta_2, \ldots)^T$. After this step the residual can be computed using the joint positions and velocities of a recored trajectory, the estimated load side parameter vector $\boldsymbol{\vartheta}$ and the computed elastic transmission torque $\boldsymbol{\tau}_{el}$ using the estimated spring stiffness $\boldsymbol{k}_e$ and spring damping $\boldsymbol{d}_e$. To better appreciate the estimated model the evaluation is done two times, first with the original Matlab simulation model that is used for excitation trajectory estimation, and second with the estimated model parameters.

Comparing the produced residuals using both models, one can see that the original estimation was far away from the real model. Apart from joint one, every joint produces a failure torque that is bigger than 1.5 Nm. After identification the joint torque failure is reduced to at most 0.2 Nm except for joint two.



(a) Produced residual with original Matlab simulation model

(b) Produced residual with experimentally identified model

Figure 4.3: Evaluation of experimental identified model using a model-based observer

Since the excitation trajectories are optimized upon the Matlab simulation model that does not really represent the real robot (see figure 4.3(a)) it is likely that these trajectories are optimal for the simulation model, but not for the real robot. The model parameters, estimated after this fist iteration, have to be used to produce new excitation trajectories for a next estimation iteration. This loop has to be repeated till the model converges [1].

To use the estimated model parameters for the Matlab simulation model the single model parameters have to be extracted from the linear combined base parameters. This task is not trivial since many parameter values describe the same result in linear combination. One possible approach to find the parameter set, that describes the estimated base parameters, a optimization algorithm can be used that minimizes the least squares difference, between the estimated parameter values and the resulting value that originate from the parameters linear combination. This optimization problem can be formulated as:

$$\min_{\boldsymbol{p}\in\Re^{n_p}} \frac{1}{2}\sum_{i=1}^{n_p}\left\|\mathbf{baseParameters}(\boldsymbol{p})-\boldsymbol{\vartheta}_{est}\right\|_2^2 \qquad \text{s.t.} \quad \boldsymbol{a}(\boldsymbol{p})=0, \quad \boldsymbol{b}(\boldsymbol{p})\geq 0$$

with $\boldsymbol{p}$ the single model parameters to identify, $\mathbf{baseParameters}(\boldsymbol{p})$ the base parameter values with the current single parameter set $\boldsymbol{p}$ and $\boldsymbol{\vartheta}_{est}$ the estimated base parameters. To support the optimization algorithm, the value range of the single parameters can be reduced by inequality

[1] [37] also proposed a methodology that perform estimation procedures till the model converge

constraints $\boldsymbol{b}(\boldsymbol{p})$ or even set to a predefined value $\boldsymbol{a}(\boldsymbol{p})$. The initial values for optimization have to be carefully be chosen in order to find a solution that is near the real one.

Instead of using the Matlab simulation model for the next trajectory optimization loops, the model parameter dependent dynamic equations (actuator and load side) can directly be used to calculate the motor and joint Informations, that are necessary to compute the regressor matrices for trajectory evaluation.

4.4 Conclusion

The parameter identification framework presented in this chapter combined two approaches to enable model identification on a robot model with elastic joints. If only the load side estimation is considered, it is possible to use this for a standard rigid industrial robot, without high elasticities.

The parameters are estimated using a dynamic identification scheme that identifies the parameters with the robot in motion. To realize this a least squares approach is used:

$$\boldsymbol{\vartheta} = ({\boldsymbol{\phi}_{tot}}^T \boldsymbol{\phi}_{tot})^{-1} {\boldsymbol{\phi}_{tot}}^T \boldsymbol{\tau}_{tot}.$$

To solve the estimation equation system the regressor matrices for both sides have to be determined. First, the actuator side base regressor is theoretically created by just rearranging the dynamic equations and putting parameter independent expressions, unknown parameters or products with unknown parameters together.

The base regressor form of the load side dynamics model is received by using a modified Newton-Euler recursion. This recursion is introduced with the modification to express the inertia tensor in the link coordinate frame using the parallel-axis theorem. Since the generated regressor matrix does not have full rank, it has to be reduced by eliminating all linear dependent columns. For this purpose the fundamental functions of the dynamic equations are used, building up a matrix that represents the linear dependencies of the regressor. After transformation in to the Echelon from, this matrix can be used to eliminate all linear dependent regressor columns, as well as to create the base parameter set.

For dynamic Estimation an appropriate trajectory is necessary that excites the system in such a way, that all parameters can be well estimated. For this purpose polynomial trajectories and Fourier-Series are introduced. The trajectory parameters are detected by optimizing an observability index that evaluates if the resulting regressor matrix for the actual trajectory is suitable to solve the linear equation system. Using Fourier-Series as trajectory allows to average the recorded values (e.g. joint positions or torques) and calculate the torque variances. These variances are used to create a weighted least squares estimate:

$$\boldsymbol{\vartheta}_{wls} = \left({\boldsymbol{\phi}_{tot}}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\phi}_{tot}\right)^{-1} {\boldsymbol{\phi}_{tot}}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\tau}_{tot}$$

with the diagonal weighting matrix $\boldsymbol{\Sigma}^{-1}$ consisting of the reciprocal torque variances.

The whole estimation framework is carried out on the Matlab simulation model. For evaluation different quality measures are shortly introduced. To find the right trajectory function, optimization algorithm and observability measure, different estimations are accomplished and evaluated. It turns out that using the interior point optimization algorithm with Fourier-Series trajectory function and D-optimality as observability index produces the best identification results in case of the BioRob-Arm. In the next step the optimized trajectories are carried out on the real robot. But before the estimation process can be executed an approach for inferring the motor torques from motor current is described.

At last the estimation results are presented and evaluated using the observer-based fault detection and isolation scheme. The produced residual show a significant model improvement in comparison to the original model, used in the Matlab simulation. But even this improvement is not enough to realize a fast collision detection. This can be constituted by the used model during optimization of the excitation trajectories. Since this used model parameters are far away from the real ones, the produced trajectories are not optimal for the real robot. Thus the identification process has to be repeated with the new estimated model parameters, till it converges. Since the Matlab model used for trajectory optimization needs the single model parameters, these have to be extracted from the base parameter set. To realize this an optimization problem has been formulated. One way out of this is to use a on line optimization scheme, that directly tries out and evaluates the trajectory with its actual parameters. Alternatively the parameter dependent dynamic equations can be used during optimization to create the motor and joint information that are necessary for trajectory evaluation.

Another reason for a difference between the estimated and the real model is founded in the model itself. The presented model actual is build up as simple as possible. After successful identification this model has to be refined, taking into account that the elastic transmission consists of nonlinear spring characteristics with hysteresis. The assumed friction model is inexact and does not represent the real friction behavior. For this purpose a more complex model, as the Stribeck friction model should be embedded. In addition to the simple friction model, also no backlash effects are considered in the whole model.

5 Collision Detection and Reaction

5.1 Introduction

Collision detection and an appropriate reaction is the basis for robots and humans working safely together. One example for the need of robots and humans cooperating are small and medium enterprises (SMEs). As mentioned in [4], there are four key requirements for applications with an unstructured and shared environment: safety, flexibility, usability and performance.

One way to increase safety is a lightweight design with compliant cable/spring transmission that decouples the reflected motor inertia off the impact force, like at the BioRob-Arm.

As introduced by [7], there are a lot of different contact scenarios where this is simply not enough. Figure 5.1 shows the different undesired contact scenarios between human beings and robots that could lead to injury of humans. They are classified in free impacts, clamping in the robot structure, constrained impacts, partially constrained impacts, and resulting secondary impacts.

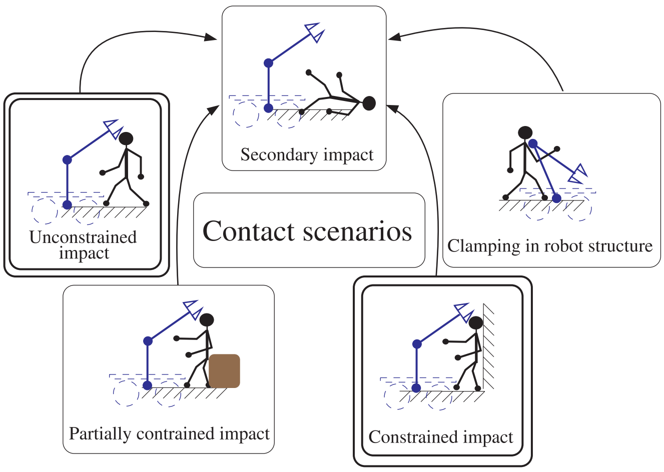


Figure 5.1: Classification of contact scenarios between human and robot (from [7])

For all this scenarios a collision detection can resolve a dangerous or at minimum unpleasant situation. A lot of research effort has been done by Haddadin et al. (see [7]) to evaluate safety requirements for robots. He investigated the effect of joint stiffness, the role of robot mass and velocity with non-constrained blunt impacts or constrained blunt impacts. One major conclusion ([8]) was that the HIC (Head Injury Criterion) index seems not to be suitable to evaluate the injury level of robots, since the operation speed is much to slow to harm people according to this criterion. This is why for future severity evaluation the force, needed to cause fractures, was investigated. Non-constrained blunt collision tests showed that only the link inertia is involved in the impact, and increasing the joint stiffness has no effect on hard impacts ([14], [7]). Before the joint torque starts to act into the collision, the collision force transmission caused by the link inertia is over. This leads to the conclusion that a collision detection and reaction mechanism is at maximum capable to reduce the impact forces caused by joint torques, but not by the link inertia. Decreasing the joint stiffness only decreases the spring force in magnitude and increases the duration. If in addition to a moderate joint stiffness the impact duration is increased a reliable collision detection can decrease the acting impact forces ([11]) caused by both, the joint torques and the link inertia. For example a collision with soft body parts result in a longer impact duration, than collisions with the head. Even for a constrained collision with a human head near a singular joint configuration, the used collision detection is able to prevent a facial fracture ([10]). Beside non-constraint or constraint collisions especially for a human-robot interaction it is necessary to know what injury risk is effected by a robot arm equipped with sharp tools. [49] created some fundamentals for soft-tissue injury evaluation, with various sharp tools. It was shown that the used collision detection provided a large benefit in reducing the penetration depth at stabbing or cutting experiments.

The results show that a reliable collision detection is reasonable to reduce injury risk in different all day work scenarios. To evaluate how suitable the BioRob-Arm is for a safe human-robot interaction, as was done for the DLR Lightweight III arm, the acting forces in critical situations have to be investigated and the influence of a collision detection on security tested in experiments. But before this ca be done, a reliable collision detection with an appropriate reaction strategy has to be implemented. As described in [3] the observer-based collision detection proposed by [23] fits for the BioRob-Arm since it only needs the elastic transmission torque $\boldsymbol{\tau}_{el}$, joint position $\boldsymbol{q}$ and velocity $\dot{\boldsymbol{q}}$. The elastic transmission torque can be simply computed using the spring stiffness $\boldsymbol{k}_e$ and damping $\boldsymbol{d}_e$, whereas the joint velocity has to be carefully determined from the joint position using noisy numerical differentiation.

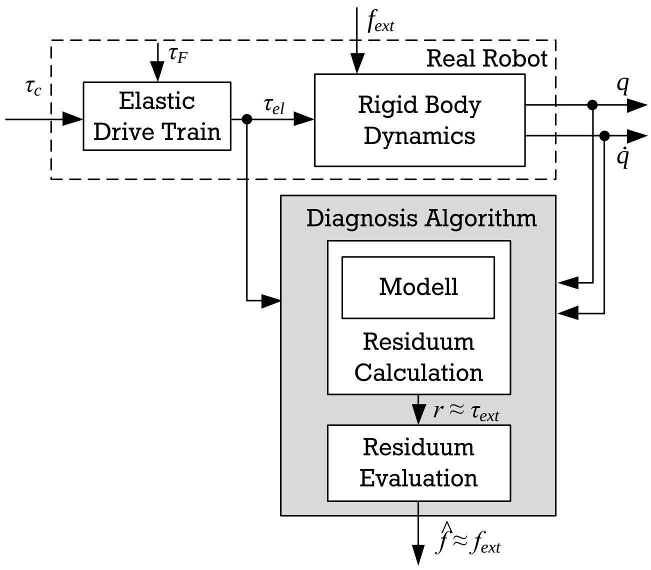


Figure 5.2: Structure of fault detection and isolation with disturbance observer (control variable τ_c, measured system output $\boldsymbol{q}, \dot{\boldsymbol{q}}$, residuum $\boldsymbol{r}$, estimated joint torque caused by external force $\boldsymbol{\tau}_{ext}$, external force $\boldsymbol{f}_{ext}$, motor friction $\boldsymbol{\tau}_F$, elastic transmission torque $\boldsymbol{\tau}_{el}$)

Figure 5.2 illustrates the structure of the realized fault detection and isolation using a disturbance observer. The control variable describes the desired motor torque. The transmitted torque to the rigid structure τ_{el}, produced from the motor under influence of motor friction torque τ_F, serves as one input for the observer. The measured joint position and velocity represents the system output and are also used as input for the observer. The external force f_{ext} that influences the system is estimated by the observer as joint torques τ_{ext} and can be converted into a force in Cartesian space $\hat{f}$ that approximates f_{ext}. The used observer from [23] uses the generalized momentum $p = M(p)\dot{q}$ to detect disturbance forces where the estimated joint torque caused by external forces r is a first-order filtered estimation of the real external joint torque τ_{ext}.

A collision detection based on this observer has been implemented at the Matlab simulation model of the BioRob-Arm ([3]). The simulation results showed that a reliable collision detection based on this observer is possible and also allows to avoid the torque caused by the reflected motor inertia to act into the collision. The proposed reaction strategies of [24] have also been investigated. A "admittance control reaction" strategy produced one of the fastest collision torque and joint velocity reduction. It uses the estimated collision torques to calculate a new desired joint velocity moving the robot arm out of the collision. If no collision is present any more, the residual and therefore the new desired velocity is zero, causing the robot arm to hold the position as soon as possible, reducing the likelihood of a second collision. This behavior can be described in the following formular:

$$q_d = q_{col} + \int K_R r, \quad \dot{q_d} = K_R r, \tag{5.1}$$

with q_{col} the joint position at collision detection, r the residual, K_R a diagonal matrix that converts the torques of r into a plausible value range for joint velocity, q_d and $\dot{q}_d$ the new desired joint position and velocity respectively. Another simulated result showed that it is not possible to cause facial fractures. To validate this results by experiment a reliable collision detection has to be implemented. The collision detection methodology and some first collision tests are described in this chapter.

5.2 Methodology of Collision Detection

This section shows the main steps to receive the first-order filtered joint torque, caused by an external force (as introduced by [23]). for additional information about the described equations please consult [23] or [3]. The generalized momentum of the robot is used to calculate a residual which represents the system disturbance. It is defined according to Newton's second law:

$$p = M(q)\dot{q} \tag{5.2}$$

with $p \in \Re^n$ the momentum, $M(q) \in \Re^{n\times n}$ the robots mass matrix and $\dot{q} \in \Re^n$ the vector of joint velocities. After differentiation one receives the dynamic equation that describes the momentum changes of the robot arm:

$$\dot{p} = \frac{d}{dt}\left(M(q)\right)\dot{q} + M(q)\dot{q} \tag{5.3}$$

In case of an external force influenceing the robot our in 3.2 described dynamic robot model has to be extended. The joint torques caused by the external force $\boldsymbol{\tau}_{ext}$ has to be added to the elastic transmission torque $\boldsymbol{\tau}_{el}$. The whole dynamic equation solved for the mass matrix results in:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} = \boldsymbol{\tau}_{el} + \boldsymbol{\tau}_{ext} - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q}) - \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \tag{5.4}$$

The skew symmetric mass matrix $\boldsymbol{M}(\boldsymbol{q})$ and the Coriolis matrix $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ (defined with the Christoffel symbols) can be combined to another skew symmetric term $\dot{\boldsymbol{M}}(\boldsymbol{q}) - 2\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$. From the skew symmetry of this term it follows that:

$$\dot{\boldsymbol{M}}(\boldsymbol{q}) = \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T \tag{5.5}$$

The proof of equation 5.5 is shown in [3]. After combining the equations 5.3, 5.4 and 5.5 the momentum derivative can be expressed as:

$$\begin{aligned}\dot{\boldsymbol{p}} &= \left(\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T\right)\dot{\boldsymbol{q}} + \boldsymbol{\tau}_{el} + \boldsymbol{\tau}_{ext} - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q}) - \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \\ &= \boldsymbol{\tau}_{el} + \boldsymbol{\tau}_{ext} - \underbrace{\left(-\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}\right)}_{\boldsymbol{\alpha}(\boldsymbol{q},\dot{\boldsymbol{q}})} \\ &\Rightarrow \dot{\boldsymbol{p}} = \boldsymbol{\tau}_{el} + \boldsymbol{\tau}_{ext} - \boldsymbol{\alpha}(\boldsymbol{q},\dot{\boldsymbol{q}})\end{aligned} \tag{5.6}$$

Solving equation 5.6 for $\boldsymbol{\tau}_{ext}$ delivers an expression that represents the torques caused by external forces without calculation of the joint acceleration:

$$\boldsymbol{\tau}_{ext} = \dot{\boldsymbol{p}} - \boldsymbol{\tau}_{el} - \boldsymbol{\alpha}(\boldsymbol{q},\dot{\boldsymbol{q}}) \tag{5.7}$$

Equation 5.7 could be used as residual for fault detection, if it is robust against noisy sensors. Since this is not the case, a further step has to be done. To reduce the influence of noise a first-order low pass filter is used which smooths high frequent signals. One way to realize a low pass filtered signal is to use a first order lag element as described by the following equation in time domain:

$$\begin{aligned}T_i \cdot \dot{r}_i + r_i &= \tau_{est,i} \\ \dot{r}_i &= -\frac{1}{T_i} r_i + \frac{1}{T_i}\tau_{ext,i}\end{aligned} \tag{5.8}$$

with $\tau_{ext,i}$ as input value, r_i and $\dot{r}_i$ the smoothed output and its derivate respectively as well as T_i the time constant which describes the reaction rate at which the filter react on a step in the input signal. Since the residual is a $n \times 1$ vector, each vector entry has to be filtered. Writing equation 5.8 as matrix vector equation and substituting $\boldsymbol{\tau}_{ext}$ by 5.7 defines a first ordered filtered derivative of the residual:

$$\begin{aligned}\dot{\boldsymbol{r}} &= -\boldsymbol{K}_I \boldsymbol{r} + \boldsymbol{K}_I \boldsymbol{\tau}_{ext} \\ &= \boldsymbol{K}_I\left(\dot{\boldsymbol{p}} - \boldsymbol{\tau}_{el} - \boldsymbol{\alpha}(\boldsymbol{q},\dot{\boldsymbol{q}}) - \boldsymbol{r}\right)\end{aligned} \tag{5.9}$$

with $\boldsymbol{K_I}$ as diagonal matrix representing the filters time constant defined by the reciprocal $1/T_i$. The final step to receive the residual is to time integrate equation 5.9:

$$
\begin{aligned}
\boldsymbol{r} &= \int_0^t \left(\boldsymbol{K_I}\left(\dot{\boldsymbol{p}} - \boldsymbol{\tau}_{el} - \boldsymbol{\alpha}(\boldsymbol{q},\dot{\boldsymbol{q}}) - \boldsymbol{r}\right)\right) dt \\
&= \int_0^t \left(\boldsymbol{K_I}\left(\dot{\boldsymbol{p}} - \boldsymbol{\tau}_{el} - \boldsymbol{C}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)^T \dot{\boldsymbol{q}} + \boldsymbol{g}\left(\boldsymbol{q}\right) + \boldsymbol{F}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} - \boldsymbol{r}\right)\right) dt \\
&= \boldsymbol{K_I}\left(\boldsymbol{p}(t) - \int_0^t \left(\boldsymbol{\tau}_{el} + \boldsymbol{C}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)^T \dot{\boldsymbol{q}} - \boldsymbol{g}\left(\boldsymbol{q}\right) - \boldsymbol{F}\left(\boldsymbol{q},\dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}} + \boldsymbol{r}\right)\right) dt - \boldsymbol{p}(0)
\end{aligned}
\tag{5.10}
$$

In equation 5.10 $\boldsymbol{r}(0) = 0$ defines the start value of the residual, $\boldsymbol{K_I} > 0$ the diagonal matrix of the filters time constants and $\boldsymbol{p}(t)$ the generalized momentum at time $t \geq 0$.

As already explained, the time constant T_i defines the filters reaction speed on a step in the input signal. A small time constant results in a nearly unfiltered output and hence closely represents the real acting joint torque $\boldsymbol{r} \approx \boldsymbol{\tau_{ext}}$. The trade-off between filtering and estimated torque accuracy has to be considered when choosing the filters time constant. The smaller the time constant, the higher the estimation accuracy. But if chosen to small, noise distorts the estimation. Since $\boldsymbol{K_I}$ contains the reciprocal time constants on its diagonal one has to choose $\boldsymbol{K_I}$ as high as possible regarding the influence of noise. Another issue with a very small time constant is that the built reciprocal causes numerical problems.

Before using the residual it is important to know how it is built and at which conditions it is able to detect external forces. One distinguishes between external forces that arise in static or dynamic cases. In static cases ($\dot{\boldsymbol{q}} = 0$) it is possible to formulate a relation between the external force and the resulting joint torques using the virtual work. The work produced by an external force $\boldsymbol{F}$, causing a virtual displacement $\delta\boldsymbol{X}$ of the end-effector and the joints $\delta\boldsymbol{q}$, and the corresponding joint torque $\boldsymbol{\tau}$ has to be the same ([50]):

$$\boldsymbol{\tau}^T \delta\boldsymbol{q} = \boldsymbol{F}^T \delta\boldsymbol{X} \tag{5.11}$$

The virtual displacement of the Cartesian and joint space are related to each other with the Jacobian matrix ([44]):

$$\delta\boldsymbol{X} = \boldsymbol{J}(\boldsymbol{q})\delta\boldsymbol{q} \tag{5.12}$$

Substituting 5.12 in equation 5.11, one gets a direct relation between the at the end-effector acting force and the resulting joint torque:

$$\boldsymbol{\tau} = \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{F} \tag{5.13}$$

Since only linear forces at the end-effector are important, one further just need the linear part of the jacobian, that is calculated as follows:

$${}^0\boldsymbol{J}_i(\boldsymbol{q}) = \frac{\partial\, {}^0 r_i(\boldsymbol{q})}{\partial \boldsymbol{q}} \tag{5.14}$$

The vector ${}^0\boldsymbol{r}_i$ that describes the position of i-th link coordinate frame with respect to the world coordinate frame is created using the transformation matrix, that expresses the orientation and position of the link coordinate frame with respect to the world coordinate system ${}^0\boldsymbol{T}_i(\boldsymbol{q})$:

$$
{}^0\boldsymbol{T}_1(q_1) \cdot {}^1\boldsymbol{T}_2(q_2) \cdot \ldots \cdot {}^{i-1}\boldsymbol{T}_i(q_i) = {}^0\boldsymbol{T}_i(\boldsymbol{q}) = \begin{pmatrix} {}^0\boldsymbol{R}_i(\boldsymbol{q}) & {}^0\boldsymbol{r}_i \\ \boldsymbol{0}^T & 1 \end{pmatrix} \tag{5.15}
$$

Equation 5.15 constitutes that ${}^0\boldsymbol{r}_i$ only depends on the first i joint positions and causes the last $N - i$ columns of the Jacobian matrix to be zero:

$$
{}^0\boldsymbol{J}_i(\boldsymbol{q}) = \left[\frac{\partial {}^0 r_i(\boldsymbol{q})}{\partial q_1}, \frac{\partial {}^0 r_i(\boldsymbol{q})}{\partial q_2}, \ldots, \frac{\partial {}^0 r_i(\boldsymbol{q})}{\partial q_i}, 0, \ldots, 0 \right] \tag{5.16}
$$

If now a external force acts on link i the residual estimated this force $\boldsymbol{r} \approx \tau_{ext}$. With the relation 5.13 and how the Jacobian matrix is built, it follows that $\boldsymbol{r} \approx \boldsymbol{\tau_{ext}} = {}^0\boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{F_{ext}}$ and only the first i columns of residual are different from zero ($*$):

$$
\begin{array}{l} \boldsymbol{r} = [* \;\; \ldots \;\; * \;\; * \;\; 0 \;\; \ldots \;\; 0]^T \\ \qquad\qquad\qquad\quad\;\; \uparrow \qquad\quad \uparrow \\ \qquad\qquad\qquad\;\; i+1 \ldots N \end{array} \tag{5.17}
$$

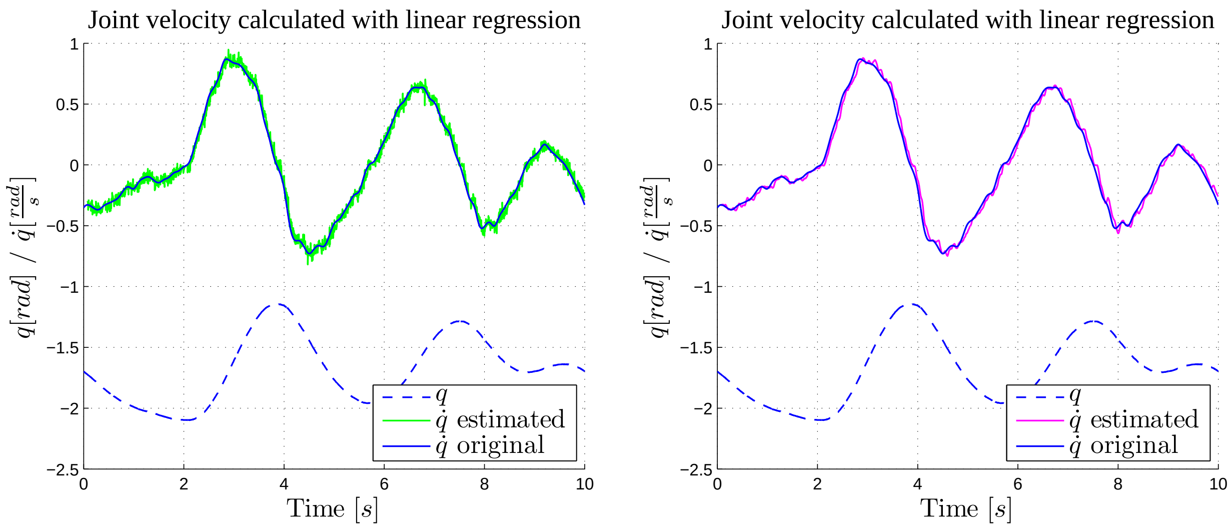
This shows that in static cases, it is possible to detect external forces and even isolate at which link they are acting. That this property also holds at the dynamic case can be illustrated with the residuals dynamic equation 5.9. Only the external torque $\boldsymbol{\tau_{ext}}$ influences the residual which indicates that the behaviour in dynamic cases will not differ greatly from the static behavior.

For a reliable fault detection and isolation the residual has to be zero as long as no fault is present. That this is fulfilled can also be explained with the residuals dynamic 5.9. If no external torque is present $\boldsymbol{\tau_{ext}} = \boldsymbol{0}$ the residual does not change and keeps its initial value. In case of a collision $\boldsymbol{\tau_{ext}}$ will differ from zero and the residual will approximate this torque corresponding to its time constant. If the collision is resolved the residual will again be zero.

5.3 Implementation and Experiments

As described in section 5.2 the residual for detecting external forces needs the joint torque, determined with the elastic transmission torque $\boldsymbol{\tau_{el}}$, the joint positions $\boldsymbol{q}$, and velocities $\dot{\boldsymbol{q}}$. The elastic transmission torque is computed with the estimated spring stiffness $\boldsymbol{k_e}$ and damping $\boldsymbol{d_e}$. $\boldsymbol{q}$ is directly available from the joints rotary encoder but $\dot{\boldsymbol{q}}$ has to be carefully computed. This implementation issue and how the residual is used in the BioRobApp will be discussed in this section. Additionally, the program's first collision tests are performed on the BioRobApp simulation model.

The easiest way to compute the joint velocity from the joint positions is to use numerical differentiation, e.g. with the backward difference quotient. Since the joint position sensor values are quantified, the resulting joint velocities are very noisy. Figure 5.3(a) shows the resulting joint velocity of the backward difference quotient that has been low pass filtered. A weighted linear regression of the last n joint positions to approximate the slope produces significantly less noise (see figure 5.3(b)). The time information of all considered joint positions is used to create a exponential decreasing weight during the regression. Thus the joint positions near to the actual time instance have higher influence on the slope estimation than the older ones. Similar to the first order filter this method has the drawback of a delayed result. The delay is determined by the number of data points that are used during the regression. In addition the estimated joint velocity shown in figure 5.3(b) has also been slightly filtered.



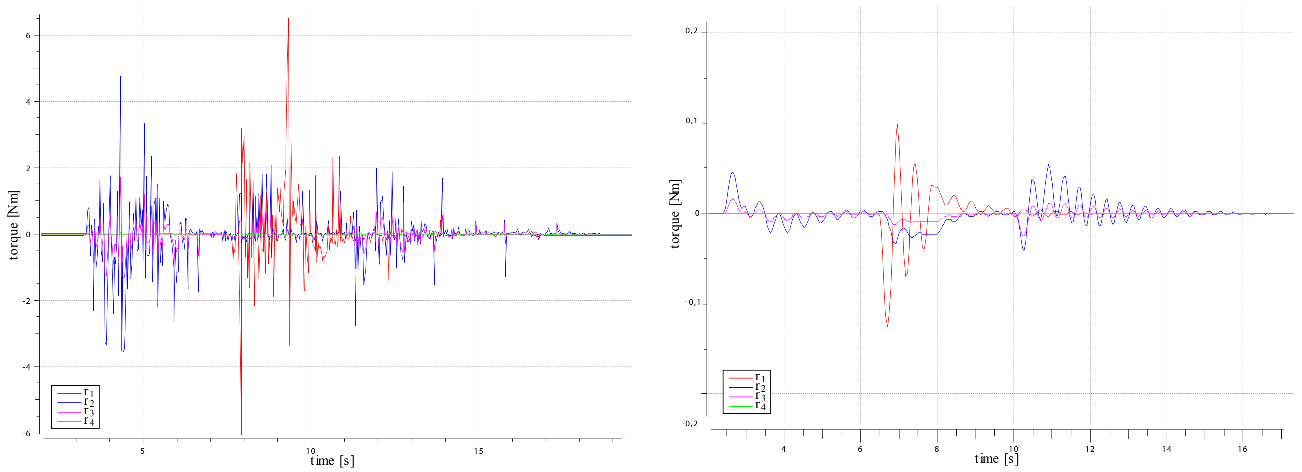
(a) Joint velocity calculated with backward difference at each time instance t_i

(b) Joint velocity calculated with linear regression of last n positions at each time instance t_i

Figure 5.3: Comparing joint velocity calculation of backward difference and weighted linear regression regarding noise

To evaluate the effect of the joint velocity, calculated with backward difference and linear regression the robot's joint are rotated one after another by 90 degrees. As one can see in figure 5.4(a) the backward difference calculation produces a very noisy residual so that no collision detection can be realized with this computation method. In contrast, the generated residual using the linear regression generates a smooth signal (see figure 5.4(b)). Whereas the noisy residual produces values that even exceed 6 Nm, the noise free signals only show a smooth oscillation that does not produce values bigger then 0.1 Nm. The oscillation is caused by the produced delay when calculating the joint velocity and controller parameters that do not perfectly fit to the simulation model. But even under this circumstances a collision detection can be realized with a collision threshold of e.g. 0.2 Nm. Since the linear regression method seems to be working well, it will be used to calculate the joint velocities in the following collision test.

Before performing a collision test with the BioRobApp, an obstacle with an appropriate contact model has to be created. To realize a compliant contact model a simple linear spring-damper



(a) Highly disturbed residual, caused by joint velocities calculated with backward difference

(b) Nearly noise free residual, produced by joint velocities calculated with linear regression

Figure 5.4: Effect of a noisy joint velocity signal on the residual calculation

model can be used. This is computationally simple but has some weaknesses. These weaknesses and a more appropriate nonlinear contact model are presented in [51] and will be described in the following paragraph. The linear spring-damper contact model can be described by the equation:

$$f = -d\dot{x} - kx, \tag{5.18}$$

with f the resulting force, d the spring damping coefficient, k the spring stiffness coefficient, x and $\dot{x}$ the penetration depth and velocity. The first weakness of this model is the discontinuity at the moment of impact. In fact, the force produced by the spring is zero, but the acting damping force steps from zero to $-dv_i$ with v_i the impact velocity. The second weakness is that this approach tends to hold the collided object together just before separation. So not only compression forces are simulated also tensile forces are produced just before separation. The tensile forces are produced, since x tends to be zero and $\dot{x}$ will be negative and nonzero at the moment just before separation. This results in a force $-dv_0$ ($v_0 < 0$) that holds the objects together. To overcome this weaknesses Hunt and Crossley proposed a nonlinear spring damper model:

$$f = -(\lambda x^n)\dot{x} - kx^n, \tag{5.19}$$

with the power n depends on the surface geometry. In this model the produced damping force not only depends on the impact velocity, but on the penetration depth. This results in a contact force that evolves continuously from zero upon contact and returns to zero as separation approaches. The coefficient of restitution e represents the ratio of speed after and before an impact. That means that objects, solid or elastic, can be calculated by equation 5.20

$$e = 1 - \alpha v_i, \tag{5.20}$$

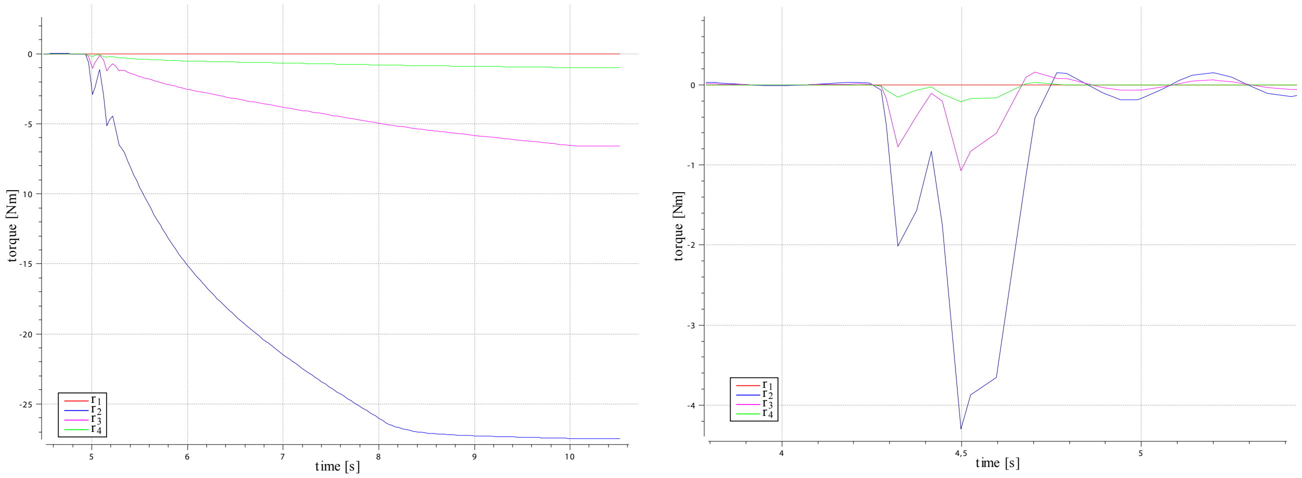
if λ is chosen with equation 5.21

$$\lambda = \frac{3}{2}\alpha k, \tag{5.21}$$

for sufficiently small α. The contact model used in the simulation uses $n = 1$, which corresponds to an impact on a flat surface, $e = 0.8$, $k = 10kN/m$, and $\alpha = 0.04$.

The collision test is carried out with an obstacle that is modeled using equation 5.19. The robot arm starts in the vertical position and joint two is controlled to reach 120 degrees. The obstacle is placed in the workspace that the collision occurs when the end-effector reaches the height of l_1 ($q_2 = 90°$). Since the penetration depth is calculated $x = l_1 - e_z$, with e_z being the z-coordinate of the end-effector, the simulated collision represents a constrained one.

During the movement of the BioRob-Arm, the control loop calculates the residual in each step relative to the controller frequency and checks whether the norm exceeds the collision threshold of $0.1Nm$. If this is the case the collision reaction strategy 5.1 is activated and the new desired values are calculated accordingly. As described above, the reaction strategy uses the residual information to calculate the velocity, that moves the robot arm out of the collision with the conversion factor $\boldsymbol{K_R} = \boldsymbol{E} \cdot 0.03$, $\boldsymbol{E} \in \Re^{4\times 4}$ identity matrix. First a collision without reaction is carried out (see figure 5.5(a)). Here, the detected joint torque increases till the maximum torque is reached. In contrast to this behavior, the collision strategy successfully reduces the detected joint torques after collision detection (see figure 5.5(b)). Besides the reaction of collision forces, figure 5.5(b) shows that the time till the collision is dissipated takes approximately one second, which is far to long to avoid that the reflected motor inertias act into the collision. After knowing that the implemented collision detection works it should be tested with the real robot. It is advisable to investigate the acting collision forces and their reduction with appropriate force sensors. After this, the real collision characteristics can be analyzed and it can be evaluated how the collision detection can improve safety.



(a) No reaction to collision (b) Moving out of collision after detection

Figure 5.5: Residual of the collision test, with and without reaction strategy

5.4 Conclusion

This chapter first introduced a collision scenario classification. All these scenarios can occur if robots and human beings work together in the same environment. A short overview is given how the impact type, joint stiffness, and the collision duration influence the possibility of a reliable collision detection. The results of the presented works show that a collision detection is reasonable to reduce injury risk. For this purpose the used fault detection and isolation scheme using a disturbance observer is introduced. As result of a previous work (see [3]) an active collision reaction strategy is shown:

$$\boldsymbol{q}_d = \boldsymbol{q}_{col} + \int \boldsymbol{K}_R \boldsymbol{r}, \quad \dot{\boldsymbol{q}}_d = \boldsymbol{K}_R \boldsymbol{r},$$

that moves the robot arm out of the collision, according to the observed collision forces. If the collision force disappears the arm holds its current position. This reduces the likelihood for a second collision.

The residual which differs from zero only when a collision occurs is based on the generalized momentum

$$\boldsymbol{p} = \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}}.$$

The determined joint torque $\boldsymbol{\tau}_{el}$ caused by an external force depends only on the elastic transmission torque, as well as the joint position and velocity. Thus, no joint acceleration has to be computed which is not a trivial task. Additionally, the observed joint torques are low pass filtered to reduce the influence of noise which results in the residual

$$\boldsymbol{r} = \boldsymbol{K}_I \left(\boldsymbol{p}(t) - \int_0^t \left(\boldsymbol{\tau}_{el} + \boldsymbol{C}\left(\boldsymbol{q}, \dot{\boldsymbol{q}}\right)^T \dot{\boldsymbol{q}} - \boldsymbol{g}\left(\boldsymbol{q}\right) - \boldsymbol{F}\left(\boldsymbol{q}, \dot{\boldsymbol{q}}\right) \dot{\boldsymbol{q}} + \boldsymbol{r} \right) \right) dt - \boldsymbol{p}(0).$$

Each entry of the constructed residual has the property of only differing from zero when the corresponding joint is involved in the collision. Thus, it is possible not only to identify a collision but also to isolate where the collision takes place.

After the introduction of the residual, its implementation in the BioRobApp is evaluated by execution of a collision. Since calculating the joint velocities using backward difference results in a highly disturbed residual an alternative approach is presented. This approach tries to approximate the joint position derivative by calculating the slope at the actual time step with a weighted linear regression. This regression uses the time information of previous joint positions to exponentially reduce their influence into the equation system. The resulting residual is nearly free of noise and can be used for collision detection.

To test the collision detection with the BioRobApp simulation model, an obstacle with an appropriate contact model is introduced. The collision test shows that the robot arm moves out of the collision as intended. In a next step the real collision characteristics, including the acting forces and their time response, has to be evaluated with the real robot. Since the collision test in the application has been successful, the program can be directly used for a test with the real robot. With such a real collision test, one is able to evaluate how the collision detection is able to reduce collision forces whether it is fast enough and which injury risks can be assumed for the BioRob-Arm.

6 Conclusion and Further Work

This work considered the important role of robots and humans that cooperate together to fulfill tasks. The role of physical human-robot interaction increases since robots are suited to help humans with routine work or collaborate with each other during production. Small and medium enterprises represent an example where safe physical human-robot interaction can be used to increase their cost efficiency. To evaluate safety some metrics are shortly introduced. Subject of research in this thesis is the human inspired BioRob-Arm. After investigation of safety requirements e.g. the ISO 10218 and collision tests, the acting collision torques have been described. Besides some design choices a reliable collision detection is able to reduce the injury risk for people working with robots. Since the most collision detection schemes need additional sensors, which are not applicable for die BioRob-Arm, it is advisable to use a fault detection and isolation scheme with a model-based observer. The used observer does not need joint acceleration and further low pass filters the residual to reduce the influence of noise. Before using such a scheme a model has to be created and then the parameters have to be identified.

To build the model of a robot with elastic joints two assumptions are made. These assumptions facilitate the model. Based on the Denavit-Hartenberg convention the kinematic model of the BioRob-Arm is described. To create the actuator side model, all parts are modeled separately and than combined to the actuator drive train. The rigid dynamics model has been calculated using the Newton-Euler recursion. Since the robot arm has to be controlled the controller is shortly introduced using a simplified inverse dynamics to calculate the motor position from the joint position. Now the whole model of an elastic joint robot arm is described and can be used for collision detection.

The model parameter estimation is done by using two approaches. On actuator side the linear least squares estimation is realized by creating the regressor matrix only by rearranging the different types of parameters. For load side a modified Newton-Euler approach is used to create a dynamics model, which only depends linearly on the model parameters. After model reduction (elimination of linear dependent columns of the regressor matrix) this model is used similar to the actuator side, to estimate the parameters. The least squares estimation is done by a dynamic estimation scheme, which tries to estimate all parameters with the data recorded from a trajectory. Since not all trajectories produce good results of the least squares estimation, these are optimized according to a observability measure. The carried out optimizations and parameter estimations showed that the D-optimality observability measure in combination with trajectories based on Fourier-Series are working well for the BioRob-Arm. The created models have been evaluated using the disturbance observer, which is used for collision detection. The resulting models are not accurate enough to use them for collision detection, so they have to be refined repeating the identification iteration.

As collision detection, a disturbance observer based on the generalized momentum has been implemented into the BioRobApp. The described residual is calculated only by the elastic transmission torque, joint position and velocity. To be more robust against noise the estimated external joint torque is low pass filtered. Since numerical differentiation of the joint positions causes high distortion in the residual, an alternative approach using weighted least squares has been presented to calculate the joint velocities. A first collision test between the robot's end-effector and an obstacle has been successfully carried out in simulation. The used reaction strategy moved the robot arm out of the collision area by using the residual information.

The proposed model identification framework is suitable to produce good parameter estimation results, as seen in the evaluation by simulation. The experimental estimated model is not accurate enough to be used for reliable collision detection. This leads to the work that needs to be done. Since the used model for trajectory optimization does not match the real one, the produces trajectories are not optimal for identification. Thus, more estimation iterations has to be carried out till the model converges. A faster way to directly create optimal trajectories is to use an on-line optimization scheme. Actual the ROS (Robot Operation System) connection is under construction. This provides an interface to directly send trajectories to the robot. After the trajectories have been performed on the robot, the results can be directly evaluated. Another accuracy limit is set by the model precision. Up to now only a simple friction model is used, linear spring characteristics and also no backlash are assumed. To increase the identification results this model parts should be refined. Another problem for identification is determined by the lightweight design. Even at very high velocities only a little range of the elastic transmission characteristic is observed. Furthermore the friction parameters can significantly change when the robot has grabbed a workpiece. So the loaded state has to be considered during the estimation process.

If a model with a certain accuracy has been determined, the collision test should be carried out on the real robot. For example with a force plate the acting collision forces and the joint torque characteristics can be evaluated with high accuracy. Only with this information it is possible to evaluate the benefit of the implemented collision detection. Further, the real injury risk arising by the BioRob-Arm can be investigated.

Bibliography

[1] B. Siciliano and O. Khatib, “Introduction,” in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer Berlin Heidelberg, 2008, pp. 1–4.

[2] A. Bicchi, M. A. Peshkin, and J. E. Colgate, “Safety for physical human-robot interaction,” in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds., 2008, pp. 1335–1348.

[3] J. Kirchhoff, “Kollisionserkennung am gelenkelastischen manipulator biorob mittels residuen basierend auf dem verallgemeinerten impuls,” 2008.

[4] T. Lens, J. Kunz, C. Trommer, A. Karguth, and O. von Stryk, “Biorob-arm: A quickly deployable and intrinsically safe, light- weight robot arm for service robotics applications,” in *41st International Symposium on Robotics (ISR 2010) / 6th German Conference on Robotics (ROBOTIK 2010)*, Munich, Germany, Jun 2010, pp. 905–910.

[5] Webpage. SMErobot - The European Robot Initiative for Strengthening the Competitiveness of SMEs in Manufacturing. http://www.smerobot.org/.

[6] *ISO 10218-1:2006, Robots for industrial environments - Safety requirements - Part 1*, Std.

[7] S. Haddadin, A. Albu-Schäffer, and G. Hirzinger, “Requirements for safe robots: Measurements, analysis and new insights,” 2009, pp. 1507 – 1527.

[8] ——, “Safety evaluation of physical human-robot interaction via crash-testing,” in *In Robotics: Science and Systems Conf. (RSS2007*, 2007, pp. 217–224.

[9] ——, “Approaching asimov’s 1st law: The impact of the robot’s weight class,” 2007.

[10] ——, “Safe physical human-robot interaction: Measurements, analysis and new insights,” 2007.

[11] S. Haddadin, A. Albu-Schaffer, A. De Luca, and G. Hirzinger, “Collision detection and reaction: A contribution to safe physical human-robot interaction,” in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, 2008, pp. 3356 – 3363.

[12] ——, “Evaluation of Collision Detection and Reaction for a Human-Friendly Robot on Biological Tissues.” [Online]. Available: http://www.phriends.eu/IARP_08c.pdf

[13] Hauger, Werner; Schnell, Walter; Gross, Dietmar; Schröder, Jörg, *Technische Mechanik 3 - Kinetik*, 8th ed. Springer, 2005, [Kapitel 3] Bewegung eines starren Körpers.

[14] A. Bicchi and G. Tonietti, “Fast and "soft-arm" tactics [robot arm design],” *Robotics Automation Magazine, IEEE*, vol. 11, no. 2, pp. 22 – 33, 2004.

[15] G. Pratt and M. Williamson, “Series elastic actuators,” in *Intelligent Robots and Systems 95. ’Human Robot Interaction and Cooperative Robots’, Proceedings. 1995 IEEE/RSJ International Conference on*, vol. 1, Aug. 1995, pp. 399 –406 vol.1.

[16] M. Zinn, B. Roth, O. Khatib, and J. K. Salisbury, "A New Actuation Approach for Human Friendly Robot Design," *The International Journal of Robotics Research*, vol. 23, no. 4-5, pp. 379–398, 2004.

[17] V. Duchaine, N. Lauzier, M. Baril, M.-A. Lacasse, and C. Gosselin, "A flexible robot skin for safe physical human robot interaction," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, May 2009, pp. 3676 –3681.

[18] D. Ebert and D. Henrich, "Safe human-robot-cooperation: image-based collision detection for industrial robots," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 2, 2002, pp. 1826 – 1831 vol.2.

[19] J. Heinzmann and A. Zelinsky, "Quantitative Safety Guarantees for Physical Human-Robot Interaction," *The International Journal of Robotics Research*, vol. 22, no. 7-8, pp. 479–504, 2003.

[20] Craig, John C., *Introduction to Robotics - Mechanics and Control*, 3rd ed. Pearson Prentice Hall, 2005, [Kapitel 6.10, S.185ff] Formulating Manipulator Dynamics in Cartesian Space.

[21] H.-W. Je, J.-Y. Baek, and M.-C. Lee, "A study of the collision detection of robot manipulator without torque sensor," in *ICCAS-SICE*, 2009, pp. 4468 –4471.

[22] Lunze, J., *Automatisierungstechnik*. Oldenbourg Wissenschaftsverlag, 2003, [Kapitel 10.2.1, S266f], Fehlererkennung mit einem Zustandsbeobachter - Diagnoseaufgabe.

[23] A. De Luca and R. Mattone, "Actuator failure detection and isolation using generalized momenta," *International Conference on Intelligent Robots and Systems*, pp. 634–639, 2003.

[24] A. De Luca, A. Albu-Schäffer, S. Haddadin, and G. Hirzinger, "Collision detection and safe reaction with the DLR-III lightweight manipulator arm," *International Conference on Intelligent Robots and Systems*, pp. 634–639, 2003.

[25] Z. Lu, K. B. Shimoga, and A. A. Goldenberg, "Experimental determination of dynamic parameters of robotic arms," *Journal of Robotic Systems*, vol. 10, pp. 1097–4563, 1993.

[26] B. Armstrong, O. Khatib, and J. Burdick, "The explicit dynamic model and inertial parameters of the puma 560 arm," in *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, vol. 3, Apr. 1986, pp. 510 – 518.

[27] A. Mukerjee and D. Ballard, "Self-calibration in robot manipulators," in *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, vol. 2, Mar. 1985, pp. 1050 – 1057.

[28] H. Olsen and G. Bekey, "Identification of robot dynamics," in *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, vol. 3, Apr. 1986, pp. 1004 – 1010.

[29] M. Gautier and W. Khalil, "On the identification of the inertial parameters of robots," in *Decision and Control, 1988., Proceedings of the 27th IEEE Conference on*, Dec. 1988, pp. 2264 –2269 vol.3.

[30] ——, “Direct calculation of minimum set of inertial parameters of serial robots,” *Robotics and Automation, IEEE Transactions on*, vol. 6, no. 3, pp. 368 –373, June 1990.

[31] M. Gautier, A. Janot, and P. Vandanjon, “Didim: A new method for the dynamic identification of robots from only torque data,” in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, May 2008, pp. 2122 –2127.

[32] A. Janot, P. Vandanjon, and M. Gautier, “Identification of robots dynamics with the instrumental variable method,” may. 2009, pp. 1762 –1767.

[33] C. H. An, C. G. Atkeson, and J. M. Hollerbach, “Estimation of inertial parameters of rigid body links of manipulators,” in *Decision and Control, 1985 24th IEEE Conference on*, vol. 24, 1985, pp. 990 –995.

[34] P. K. Khosla and T. Kanade, “Parameter identification of robot dynamics,” in *Decision and Control, 1985 24th IEEE Conference on*, vol. 24, 1985, pp. 1754 –1760.

[35] P. Khosla, “Categorization of parameters in the dynamic robot model,” *Robotics and Automation, IEEE Transactions on*, vol. 5, no. 3, pp. 261 –268, June 1989.

[36] K. Radkhah, D. Kulic, and E. Croft, “Dynamic parameter identification for the crs a460 robot,” in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, Oct. 29 - Nov. 2 2007, pp. 3842 –3847.

[37] J. Swevers, C. Ganseman, D. Tukel, J. de Schutter, and H. Van Brussel, “Optimal robot excitation and identification,” *Robotics and Automation, IEEE Transactions on*, vol. 13, no. 5, pp. 730 –740, Oct. 1997.

[38] P. v. Zutven, D. Kostic, and H. Nijmeijer, “Parameter identification of robotic systems with series elastic actuators,” in *8th IFAC Symposium Nonlinear Control Systems (NOLCOS 2010), Bologna (I)*, Sep. 01-03 2010, pp. 350–355.

[39] A. De Luca and W. Book, “Robots with flexible elements,” in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer Berlin Heidelberg, 2008, pp. 287–319.

[40] T. Lens, J. Kunz, and O. von Stryk, “Dynamic modeling of the 4 dof biorob series elastic robot arm for simulation and control,” in *Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR 2010)*, ser. Lecture Notes in Artificial Intelligence. Springer, 2010, pp. 411–422.

[41] O. von Stryk, “Robotik 1 Skriptum,” 2010.

[42] Webpage. Fachgebiet Simulation, Systemoptimierung und Robotik, Fachbereich Informatik, Technische Universität Darmstadt. http://www.biorob.de/.

[43] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, “Dynamics,” in *Robotics: modelling, planning and control*, ser. Advanced Textbooks in Control and Signal Processing, M. J. Grimble and M. A. Johnson, Eds. Springer London, 2009, pp. 247–302.

[44] Spong, Mark W. and Hutchinson, Seth and Vidyasagar, M., *Robot Modeling and Control*. Wiley, 2006, [Chapter 4.10, S148ff] Static Force/Torque Relationships, [Chapter 6.1, S205ff] Actuator Dynamics, [Chapter 7.1, S240ff] The Euler-Lagrange Equations, [Chapter 7.2.1, S251ff] The Inertia Tensor, [Chapter 7.5.3, S270f] Linearity in the Parameters, [Chapter 7.6, S271ff] Newton-Euler Formulation.

[45] J. Hollerbach, W. Khalil, and M. Gautier, “Model identification,” in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer Berlin Heidelberg, 2008, pp. 321–344.

[46] H. Kawasaki, T. Shimizu, and K. Kanzaki, “Symbolic analysis of the base parameters for closed-chain robots based on the completion procedure,” in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 2, Apr. 1996, pp. 1781 –1786 vol.2.

[47] Y. Sun and J. Hollerbach, “Observability index selection for robot calibration,” in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, May 2008, pp. 831 –836.

[48] M. M. Olsen, J. Swevers, and W. Verdonck, “Maximum likelihood identification of a dynamic robot model: Implementation issues,” *The International Journal of Robotics Research*, vol. 21, no. 2, pp. 89–96, 2002.

[49] S. Haddadin, A. Albu-Scha? andffer, and G. Hirzinger, “Soft-tissue injury in robotics,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, May 2010, pp. 3426 –3433.

[50] Hauger, Werner; Schnell, Walter; Gross, Dietmar; Schröder, Jörg, *Technische Mechanik 1 - Kinetik*, 10th ed. Springer, 2008, [Kapitel 8.2, S211f] Der Arbeitssatz.

[51] D. Marhefka and D. Orin, “Simulation of contact using a nonlinear damping model,” in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 2, Apr. 1996, pp. 1662 –1668 vol.2.

Symbols

$\boldsymbol{A}$		Trajectory coefficient matrix for P time instances
$\boldsymbol{\alpha}_c(\boldsymbol{q})$	$[rad]$	Correction vector of the motor position to receive the equilibrium position
$\boldsymbol{B}$		Matrix of unknown system parameter vectors for each regressor entry
$\boldsymbol{B_E}$		Matrix $\boldsymbol{B}$ in Echelon from
$\boldsymbol{B_E u}$		Non zero rows of $\boldsymbol{B_E}$
$\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$		Matrix of the centrifugal and Coriolis forces
$\boldsymbol{d}$	$[\frac{Nm\ s}{rad}]$	Vector of the joints viscous damping coefficients
$\boldsymbol{d_e}$	$[\frac{Nm\ s}{rad}]$	Vector of spring damping coefficients
$d_{v,m}$	$[\frac{Nm\ s}{rad}]$	Reflected viscous damping coefficient of the motor and gearbox
$\boldsymbol{D_{v,m}}$	$[\frac{Nm\ s}{rad}]$	Diagonal matrix of the reflected viscous damping coefficient
$d_{C,m}$	$[\frac{Nm\ s}{rad}]$	Reflected Coulomb friction coefficient of the motor and gearbox
$\boldsymbol{D_{C,m}}$	$[\frac{Nm\ s}{rad}]$	Diagonal matrix of the reflected Coulomb friction coefficient
$\boldsymbol{D}^{i}_{i+1}$		Pseudo-rotation matrix between link i and $i+1$
$\boldsymbol{\delta_i}$		Trajectory parameter vector for joint i
$\delta\boldsymbol{q}$	$[rad]$	Virtual displacement of the joints
$\delta\boldsymbol{X}$	$[rad]$	Virtual displacement of the end-effector
ϵ_{AV}		Relative error of the estimated parameters
ϵ_{MAX}		Maximum relative error of the estimated parameters
ϵ_{RMS}		Root mean square difference between measured and estimated joint torques
f_i	$[kg]$	Force exerted on link i by link $i-1$
F_i	$[kg]$	Net force exerted on link i
$\boldsymbol{g}(\boldsymbol{q})$		Gravity torque vector
$\boldsymbol{\gamma_i}$	$[kg, Nm]$	Vector that combines the force and torque exerted on link i by link $i-1$
$\boldsymbol{\Gamma_i}$	$[kg, Nm]$	Vector that combines the net force and torque exerted on link i
$\boldsymbol{i_a}$	$[A]$	Vector of the motors armature current
$\boldsymbol{i'_a}$	$[A]$	Vector of the motors armature current calculated with the actual motor voltage without influence of friction
$\boldsymbol{I}$	$[kg\ m^2]$	Link inertia tensor
I^{ci}_i	$[kg\ m^2]$	Inertia tensor of link i expressed about the center of mass of link i
I'_i	$[kg\ m^2]$	Inertia tensor of link i expressed about the links coordinate frame
$\boldsymbol{I_g}$	$[kg\ m^2]$	Vector of the gearbox' inertias with respect to the motor axis
$\boldsymbol{I_r}$	$[kg\ m^2]$	Vector of the motors rotor inertia
$\boldsymbol{J}(\boldsymbol{q})$		The robot's Jacobian matrix
$\boldsymbol{K}$	$[\frac{Nm}{rad}]$	Diagonal spring stiffness matrix

$\boldsymbol{K_i}$		The matrix that describes the kinematic structure of link i
$\boldsymbol{K_I}$		Diagonal time constant matrix of the residual
$\boldsymbol{k_e}$	$[\frac{Nm}{rad}]$	Vector of spring stiffness coefficients
$\boldsymbol{K_R}$		Diagonal gain matrix for the collision reaction strategy
$\boldsymbol{k_t}$	$\frac{Nm}{A}$	The motors torque constant vector
$\boldsymbol{k_v}$	$\frac{Vs}{rad}$	The motors speed constant vector
$\boldsymbol{l}$	$[m]$	Vector of the links lengths
L	$[Nm]$	Angular momentum of a rigid body (e.g. the rotor)
m_i	$[kg]$	Total mass of link i
$\boldsymbol{M(q)}$		Mass matrix
$\boldsymbol{n_g}$		Vector of gearbox ratios
n_i	$[Nm]$	Torque exerted on link i by link $i-1$
N_i	$[Nm]$	Net torque exerted on link i
$\boldsymbol{n_p}$		Vector of elastic transmission ratios
ω_i	$[\frac{rad}{s}]$	Angular velocity of the i-th coordinate frame S_i
$\dot{\omega}_i$	$[\frac{rad}{s^2}]$	Angular acceleration of the i-th coordinate frame S_i
$\boldsymbol{p}$		Generalized momentum
P		Number of measurements used for parameter identification
$\boldsymbol{\phi}$		Coefficient matrix of the linear dynamic model
$\boldsymbol{\phi_{tot}}$		Stacked coefficient matrix for P measurements
$\boldsymbol{q}$	$[rad]$	Vector of joint positions
$\bar{q}_i(k)$	$[rad]$	Mean joint position over S periods at time instance k of joint i
$\dot{\boldsymbol{q}}$	$[\frac{rad}{s}]$	Vector of joint velocities
$\ddot{\boldsymbol{q}}$	$[\frac{rad}{s^2}]$	Vector of joint accelerations
$\boldsymbol{r}$	$[Nm]$	Residual vector
$\boldsymbol{r^i_{c_i}}$		Vector that determines the center of mass position relative to the i-th link frame S_i
R^{i-1}_i		Orthogonal rotation matrix, which transforms a vector in the i-th coordinate frame to a coordinate frame, which is parallel to the $(i-1)$-th coordinate frame, for $i = 1, 2, \ldots, n$, where $R^n_{n+1} = I$
S_i		Coordinate frame of link i
$\boldsymbol{\Sigma}^{-1}$		Diagonal matrix containing the reciprocal of the measured torque variances
$\sigma^2_{q_i}$	$[rad^2]$	Variance of the joint positions for S periods at joint i
$\sigma^2_{\tau_i}$	$[Nm^2]$	Variance of the joint torques for S periods at joint i
T_i	$[s]$	Time constant of the low pass filter for joint i
$\boldsymbol{\tau}$	$[Nm]$	Vector of joint torques
$\boldsymbol{\tau_{tot}}$	$[Nm]$	Stacked joint torques for P measurements
τ_i	$[Nm]$	Joint torque at joint i
$\hat{\boldsymbol{\tau}}$	$[Nm]$	Vector of joint torques, which are estimated using the robot dynamics model
$\bar{\tau}_i(k)$	$[Nm]$	Mean joint torque over S periods at time instance k of joint i

$\boldsymbol{\tau_{el}}$	$[Nm]$	Vector of the elastic transmission torques
$\boldsymbol{\tau_{ext}}$	$[Nm]$	Vector of torques caused by an external force
$\boldsymbol{\tau_m}$	$[Nm]$	Vector of motor torques
τ_r	$[Nm]$	The motors rotor torque
$\boldsymbol{\theta}$	$[rad]$	Vector of motor positions
$\dot{\boldsymbol{\theta}}$	$[\frac{rad}{s}]$	Vector of motor velocities
$\ddot{\boldsymbol{\theta}}$	$[\frac{rad}{s^2}]$	Vector of motor accelerations
$\boldsymbol{\vartheta}$		Dynamic parameter vector
$\boldsymbol{u}$	$[V]$	Vector of the motors voltage
v_i	$[\frac{m}{s}]$	Linear velocity of the i-th coordinate frame S_i
$\dot{v}_i$	$[\frac{m}{s^2}]$	Linear acceleration of the i-th coordinate frame S_i
v_{c_i}	$[\frac{m}{s}]$	Linear velocity of center of mass of link i
$\dot{v}_{c_i}$	$[\frac{m}{s^2}]$	Linear acceleration of center of mass of link i
x	$[m]$	Penetration depth of the end-effector into the collided obstacle
$\dot{x}$	$[\frac{m}{s}]$	Collision velocity at the end-effector
z		The allover transmission ratio of the drive train ($z = n_g \cdot n_p$)

List of Figures

List of Tables

A Additional Information Regarding Parameter Identification

The tables A.1, A.2 and A.3 show the optimization results of the simulation model with different optimization algorithms, trajectory functions and varying function degree.

	Frouries-Series Degree 3			**Polynom Degree 3**		
	IP	TRR	SQP	IP	TRR	SQP
# interation	2	42	1	3	17	1
# func. eval.	101	1772	82	89	725	70
obj. function	22.27	24.42	25.05	20.84	14.33	21.51
ϵ_{AV}	7.95	29.82	61.71	45.04	55.67	64.41
ϵ_{MAX}	77.54	252.00	720.59	33.37	287.58	408.37
ϵ_{RMS} Joint 1	$1.53{\cdot}10^{-1}$	$1.54{\cdot}10^{-1}$	$1.48{\cdot}10^{-1}$	$3.04{\cdot}10^{-1}$	$1.77{\cdot}10^{-1}$	$1.98{\cdot}10^{-1}$
ϵ_{RMS} Joint 2	$1.49{\cdot}10^{-1}$	$1.38{\cdot}10^{-1}$	$1.75{\cdot}10^{-1}$	$9.36{\cdot}10^{-1}$	$1.95{\cdot}10^{-1}$	$1.85{\cdot}10^{-1}$
ϵ_{RMS} Joint 3	$5.23{\cdot}10^{-2}$	$6.64{\cdot}10^{-2}$	$5.24{\cdot}10^{-2}$	$5.20{\cdot}10^{-2}$	$6.32{\cdot}10^{-2}$	$7.73{\cdot}10^{-2}$
ϵ_{RMS} Joint 4	$1.11{\cdot}10^{-1}$	$1.10{\cdot}10^{-1}$	$1.09{\cdot}10^{-1}$	$1.01{\cdot}10^{-1}$	$1.03{\cdot}10^{-1}$	$1.00{\cdot}10^{-1}$

Table A.1: Trajectory optimization results for actuator side with function of degree three

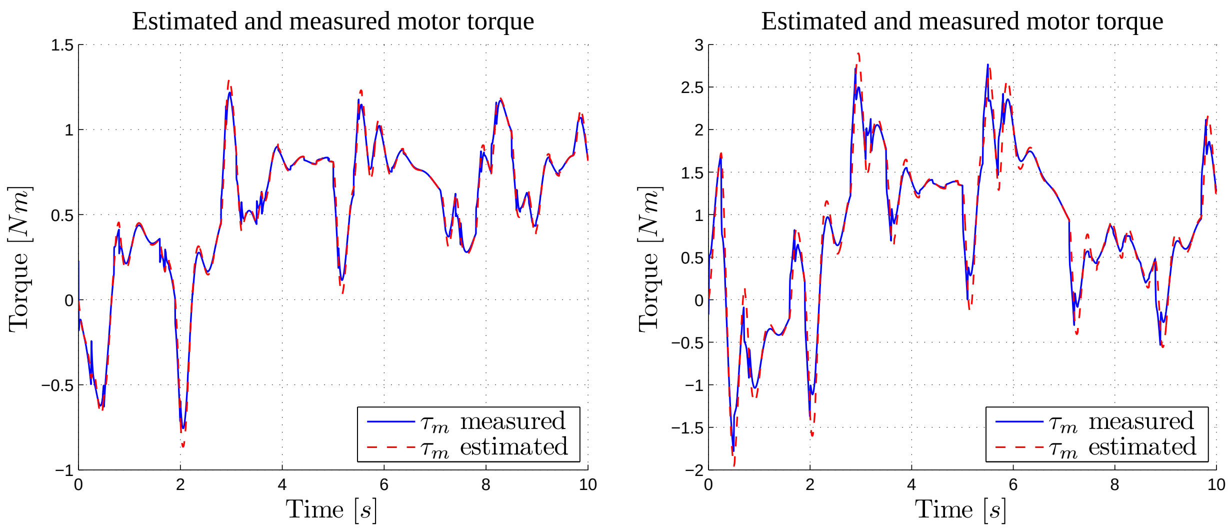
	Frouries-Series Degree 4			**Polynom Degree 4**		
	IP	TRR	SQP	IP	TRR	SQP
# interation	10	57	2	3	33	2
# func. eval.	444	2833	178	132	1309	140
obj. function	25.40	26.36	30.08	24.94	24.89	21.12
ϵ_{AV}	25.09	9.82	11.38	56.51	54.99	29.74
ϵ_{MAX}	221.53	127.02	86.55	522.75	659.33	152.31
ϵ_{RMS} Joint 1	$1.46{\cdot}10^{-2}$	$3.03{\cdot}10^{-1}$	$1.52{\cdot}10^{-1}$	$2.01{\cdot}10^{-1}$	$1.63{\cdot}10^{-1}$	$1.47{\cdot}10^{-1}$
ϵ_{RMS} Joint 2	$1.47{\cdot}10^{-2}$	$1.40{\cdot}10^{-1}$	$1.50{\cdot}10^{-1}$	$1.94{\cdot}10^{-1}$	$1.87{\cdot}10^{-1}$	$1.38{\cdot}10^{-1}$
ϵ_{RMS} Joint 3	$5.23{\cdot}10^{-2}$	$5.31{\cdot}10^{-2}$	$5.51{\cdot}10^{-2}$	$5.25{\cdot}10^{-2}$	$5.31{\cdot}10^{-2}$	$8.12{\cdot}10^{-2}$
ϵ_{RMS} Joint 4	$1.11{\cdot}10^{-2}$	$1.11{\cdot}10^{-1}$	$1.12{\cdot}10^{-1}$	$1.04{\cdot}10^{-1}$	$1.03{\cdot}10^{-1}$	$1.01{\cdot}10^{-1}$

Table A.2: Trajectory optimization results for actuator side with function of degree four

	Frouries-Series Degree 5			Polynom Degree 5		
	IP	TRR	SQP	IP	TRR	SQP
# interation	8	42	2	5	25	2
# func. eval.	468	2457	194	206	1347	151
obj. function	33.32	33.30	37.09	28.77	24.61	29.87
ϵ_{AV}	3.56	97.35	87.55	90.75	28.46	97.08
ϵ_{MAX}	22.25	128.35	109.27	101.09	277.57	1165.61
ϵ_{RMS} Joint 1	$1.15{\cdot}10^{-1}$	$1.44{\cdot}10^{-1}$	$1.50{\cdot}10^{-1}$	$2.39{\cdot}10^{-1}$	$1.04{\cdot}10^{-1}$	$1.81{\cdot}10^{-1}$
ϵ_{RMS} Joint 2	$1.42{\cdot}10^{-1}$	$1.39{\cdot}10^{-1}$	$1.49{\cdot}10^{-1}$	$1.67{\cdot}10^{-1}$	$9.36{\cdot}10^{-1}$	$1.57{\cdot}10^{-1}$
ϵ_{RMS} Joint 3	$5.27{\cdot}10^{-2}$	$6.08{\cdot}10^{-2}$	$5.50{\cdot}10^{-2}$	$5.70{\cdot}10^{-2}$	$5.20{\cdot}10^{-2}$	$9.90{\cdot}10^{-2}$
ϵ_{RMS} Joint 4	$1.12{\cdot}10^{-2}$	$1.09{\cdot}10^{-2}$	$1.14{\cdot}10^{-1}$	$1.06{\cdot}10^{-1}$	$2.36{\cdot}10^{-1}$	$1.08{\cdot}10^{-1}$

Table A.3: Trajectory optimization results for actuator side with function of degree five

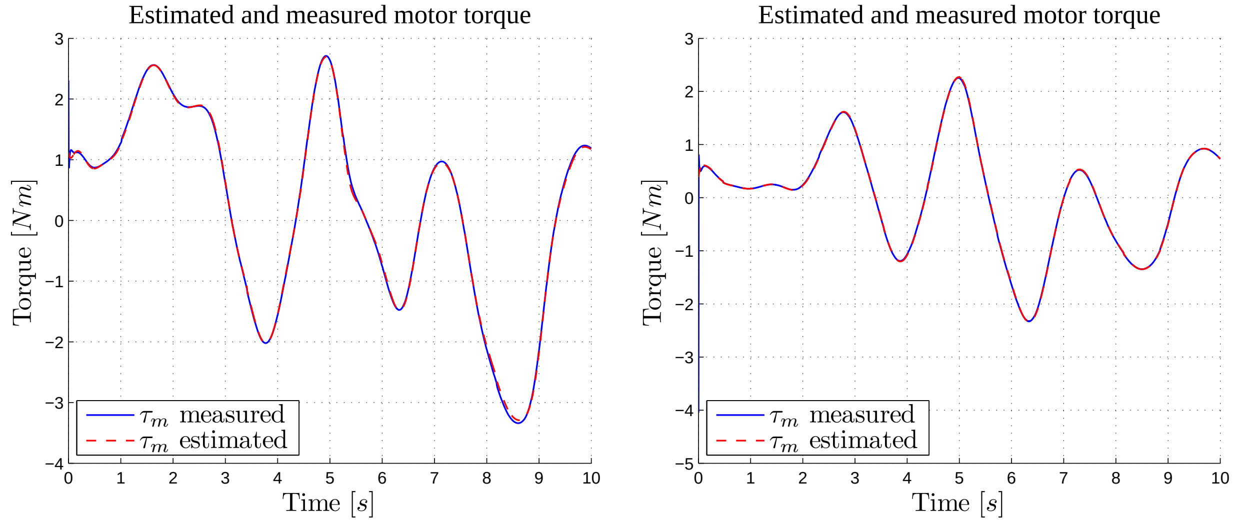
The plots in figure A.1 and A.2 show the best and worst torque estimation from the determined actuator model parameters in case of the evaluation and excitation trajectory.



(a) Best motor torque estimation for evaluation trajectory

(b) Worst motor torque estimation for evaluation trajectory

Figure A.1: Motor torques calculated from the estimated actuator parameters in case of the evaluation trajectory

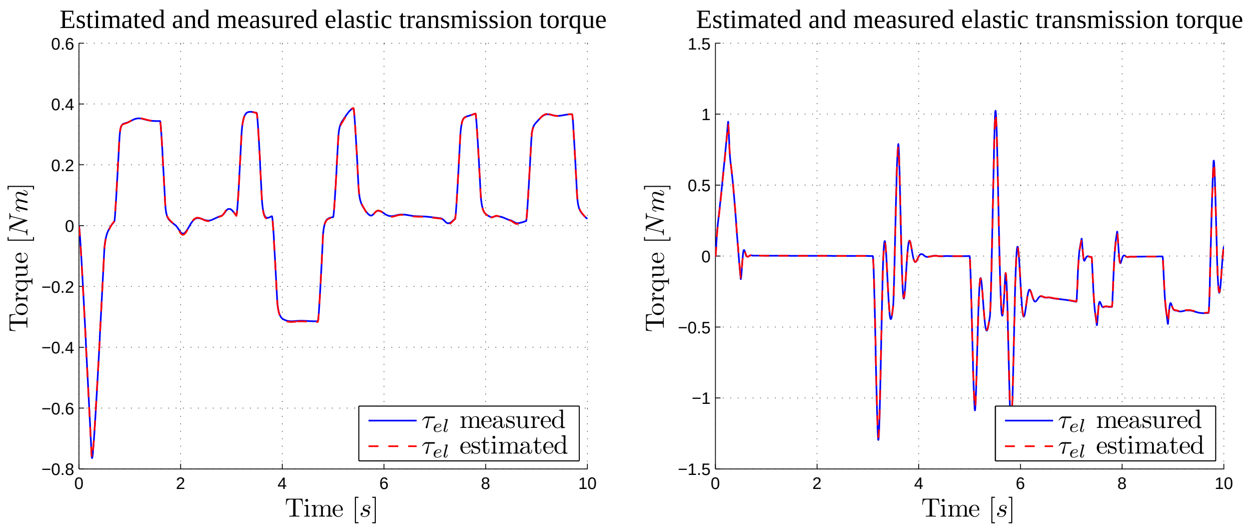


(a) Best motor torque estimation for excitation trajectory

(b) Worst motor torque estimation for excitation trajectory

Figure A.2: Motor torques calculated from the estimated actuator parameters in case of the excitation trajectory

After the actuator side, the load side parameters are estimated. To show the estimation quality, the best and worst elastic transmission torque is depicted in figure A.3.



(a) Best elastic transmission torque estimation for evaluation trajectory

(b) Worst elastic transmission torque estimation for evaluation trajectory

Figure A.3: Elastic transmission torques calculated from the estimated load parameters in case of the evaluation trajectory