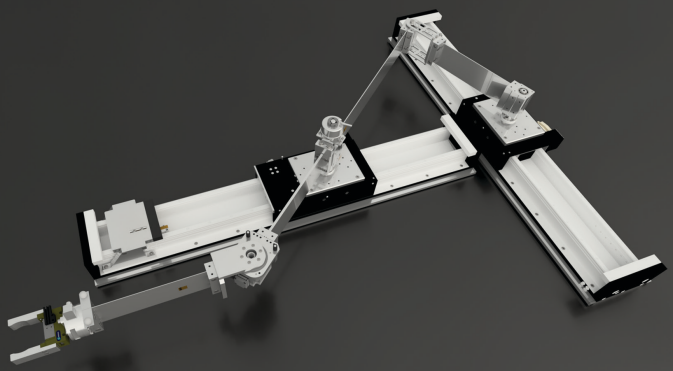
Control Strategies for Flexible Link Parallel Robots

Merlin Morlock



Control Strategies for Flexible Link Parallel Robots

Vom Promotionsausschuss der
Technischen Universität Hamburg
zur Erlangung des akademischen Grades
Doktor-Ingenieur (Dr.-Ing.)
genehmigte Dissertation

von
Merlin Morlock
aus Stuttgart

2023

1. Gutachter: Prof. Dr.-Ing. Robert Seifried
2. Gutachter: Prof. Dr.-Ing. Herbert Werner

Tag der mündlichen Prüfung: 10. Januar 2023

ORCID: 0000-0003-2574-1385
DOI: 10.15480/882.6203

To Helene.

Summary

With modern lightweight designs, driven by the need for material and energy efficiency, and the demand for inherent compliance within human-robot interaction an increasing amount of flexibility is introduced in robots nowadays. Through a reduction of the robot mass energy consumption is decreased and interactions are safer. Flexible link robots are a natural way to approach these topics. Nevertheless, the obtained compliance within flexible links introduces several challenges. These include amongst others underactuation, undesired structural oscillations and often an unstable internal dynamics when controlling the end-effector. Therefore, it is typically not possible to directly apply established methods from rigid body robotics but more sophisticated approaches are necessary.

The presented research deals with the challenges occurring for flexible link robots with serial and parallel parts, such that it is possible to eventually benefit from the variety of advantages which they offer. In this regard, the complete process is discussed from the development of a new modular flexible link parallel robot, over the compact as well as accurate modeling and model inversion, over state and output estimation, to end-effector trajectory tracking control and active oscillation damping. Since the literature on parallel robots with flexible links is very rare this newly developed robot called Flexor offers the possibility to further validate modeling and control concepts for such robots. For this system type, model order reduction and the floating frame of reference approach are combined with projections to transform the resulting differential-algebraic equations to ordinary differential equations in order to simplify the numerical solution. Together with the concept of servo constraints, this results in an accurate and computationally efficient inverse flexible multibody model. It even enables classical model inversion via forward time integration in real-time for Flexor if the internal dynamics is stable. To provide systematic and experimentally promising control concepts, it is further shown how several well-known control techniques can be adapted to flexible link parallel robots.

The main focus of this research is end-effector trajectory tracking, which can be regarded as the most difficult control task for flexible link robots. The reason is that using the end-effector as output typically results in a non-minimum phase system, i.e., the internal dynamics is unstable. Therefore, three concepts are developed to render the system minimum phase. All three concepts even result in a stable inverse model of Flexor for the considered scenarios. Here, the full dynamic flexible multibody model is inverted in real-time. This is a major

difference to the related literature, which is typically based on rigid body inverse kinematics or on offline computations for models including flexibility. Based on such a real-time model inversion the related rigid body literature on collision avoidance is then extended to flexible link robots. A dynamic flexible model inversion is also used within the framework of hybrid force/position control as opposed to the usual approaches within the related literature on flexible link robots. Moreover, input-output feedback linearization is adapted to parallel robots with flexible links. A systematic way via servo constraints is proposed for tracking a redefined output close to the exact end-effector. Finally, a linear-quadratic regulator is designed to track state trajectories of flexible link parallel robots and is used to damp oscillations such as after a trajectory tracking scenario or caused by disturbances, e.g., through contact to a human.

Within flexible link robotics many researches only rely on simulations since experiments are not straightforward to perform and feedback control with elastic measurements is difficult. Nevertheless, a major effort has been made to successfully apply all proposed control concepts to flexible link parallel robots within experiments. This is crucial to show the practical relevance of the concepts within realistic scenarios. The experiments also include human-robot interaction, which is very rarely considered in the literature of flexible link robots. The experiments confirm on the one hand the need for modern controllers based on flexible multibody models, as in most cases they significantly outperform controllers based on classical rigid multibody models. On the other hand, the applicability of flexible link parallel robots to typical robotic application scenarios is concluded. This thesis can thus serve as a basis for future research within model-based control designs and a transfer to industrial lightweight applications is also possible.

Contents

Chapter 1

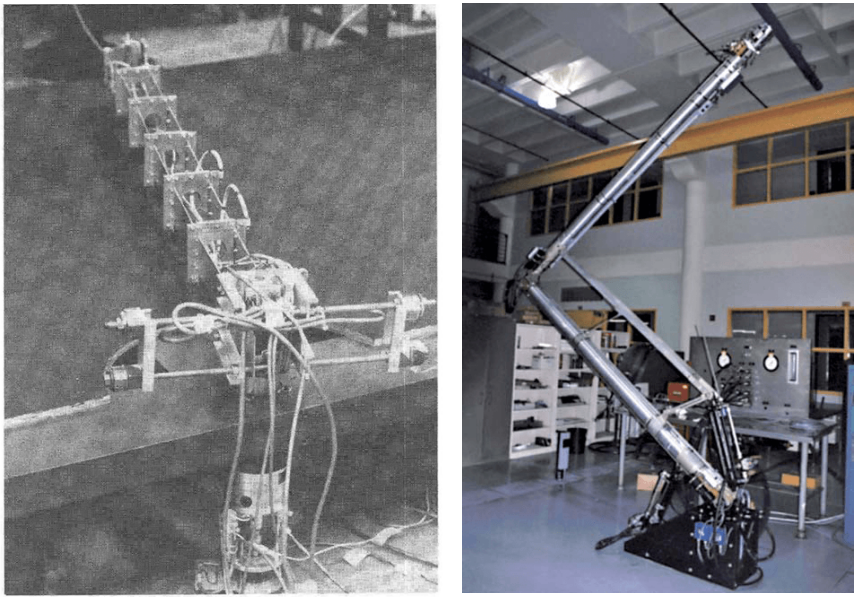
Introduction

1.1 Compliance in Robotics

Classical industrial robots are designed to be close to rigid in order to accurately perform manufacturing tasks. However, the resulting large mass and stiffness makes an interaction with human workers often dangerous and high forces can occur within scenarios with contact to the environment. Especially for such contact scenarios, efforts have been made to introduce a compliant behavior by control design. Well-known methods are compliance control and impedance control which date back to the 1970s and 1980s [1]. Nowadays, compliant controllers can be found in the portfolio of renowned companies such as ABBs SoftMove, which introduces robot compliance.
Actively introduced compliance can be effective but does not offer advantages such as the inherent safety aspects of passive compliance. Typical passively compliant elements in robotics are flexible joints, flexible links as well as soft and pneumatic components. Although compliance can reduce the control accuracy, compliant robots have gained increased interest in recent years amongst others due to the fact that they can ensure safe human-robot interaction (HRI). Thus, there tends to be a paradigm shift in the industry from reducing compliance to using compliance. Amongst others, Kuka has recently developed the LBR iiwa, which is a flexible joint lightweight robot that enables a collaboration between humans and robots without safety fences. The robot Sawyer from Rethink Robotics is also a relatively new collaborative flexible joint robot. Lately, Festo developed the BionicSoftArm which is a lightweight robot that uses pneumatic bellows segments and pneumatic actuators. Besides collaboration, compliance is beneficial within many application areas such as for surgeries, where the adaptable shape of compliant snake-like robots can be advantageous.
Significant passive compliance also occurs in large-scale machines since stiff designs are typically not feasible. Applications are amongst others long aerial ladders, large tower cranes or large space manipulators. Lightweight designs driven by the need for material and energy efficiency often also result in significant compliance. The resulting structural elasticities in the form of flexible links are the focus of this research. Although the demand for such lightweight designs is large the connected advantages come with significant control challenges. These are outlined in the following section.

1.2 Flexible Link Robots

Flexible link robots have been considered since the 1970s amongst others by Book et al. [2], while in the 1980s major research efforts began [3]. Early studies were closely connected to the need for lightweight space manipulators in order to reduce the launching cost [4, 5]. One of the first reported experiments on a flexible link robot dates back to 1984 by Cannon and Schmitz [6] who used the setup illustrated in Figure 1.1a). Other early experimental setups were investigated by Lee et al. [7], see Figure 1.1b), as well as by Naganathan and Soni [8].
For a detailed overview of modeling and control schemes it is directed to the following referenced review papers which summarize the important techniques investigated in the five decades of flexible link robots. In 1997, Shabana [10] reviewed flexible multibody dynamics developments which can be used to model flexible link robots. In 2004, Benosman and Le Vey [11] issued a survey paper with focus on control of flexible link robots. In 2006, Dwivedy and Eberhard [5] published a review paper on the dynamic analysis of flexible joint and flexible link robots. In 2015, Kiang et al. [3] reviewed besides control design also sensor systems for flexible joint and flexible link robots. In 2016, Sayahkarajy et al. [4] and Lochan et al. [12] published review papers with focus on flexible two-link



a) Experimental flexible arm [6]

b) Robotic arm large and flexible (RALF) [9]

Figure 1.1: Early flexible link robots.

manipulators. More specific literature reviews of the actually utilized modeling and control concepts are omitted here since they are reported in the corresponding sections throughout this thesis.
As indicated before, classical industrial robots are heavy in order to prevent undesired oscillations. This makes it necessary to use large actuators to ensure a fast operation, which typically consume a significant amount of energy. Additionally, the negligible compliance can make interactions with humans dangerous. The interest in solving these problems highly increased in the last decades. A possible solution are the considered flexible link robots since they provide promising advantages such as:

- Material efficiency
- Light weight
- Smaller actuators
- Faster motion
- Energy efficiency
- Smaller contact forces
- Safer HRI through inherent compliance

Nevertheless, the link flexibility comes at the cost of several disadvantages. Typical issues of flexible link robots are:

- Underactuation
- Non-collocation of input and output
- Unstable internal dynamics with end-effector as output
- Difficulty to measure the end-effector pose
- Undesired oscillations
- Static deflections
- Larger control errors

Due to the link flexibility, classical control concepts from rigid robotics can in most cases not be directly applied. For model-based control an accurate but computationally efficient flexible multibody model and adapted control designs are necessary. In this research, techniques are discussed to obtain such a model. Then, control approaches for flexible link robots are proposed which allow to cope with the mentioned disadvantages to eventually benefit from the large amount of advantages.

1.3 Main Contributions

The objective, i.e., the problem statement to be solved of the presented research is summarized as:

- A new modular flexible multi-link robot shall be built that enables the investigation of a variety of industrial robotic scenarios and problems such as pick and place and contact to the environment.
- Accurate but computationally efficient real-time capable models shall be developed and applied to model-based feedforward and feedback control with focus on end-effector trajectory tracking of flexible link robots.
- Experiments which also include HRI shall be conducted to validate the practical relevance of the proposed control concepts for flexible link robotics.

In the course of dealing with this general problem statement several contributions have been made which close various gaps in flexible link robotics research. The main contributions are:

- **Development of a new flexible link parallel robot**:
  A unique and modular flexible link robot called FLEXOR is developed to investigate complex nonlinear model-based controllers within a variety of different classical and modern robotic scenarios such as pick and place, force control, end-effector trajectory tracking and HRI. In contrast to the literature, which highly focuses on flexible single-link robots [4], the presented flexible robot has one rigid and two flexible links. Besides a serial part it also consists of a parallel part. Since the literature on parallel robots with flexible links is very rare [13, 14] this new robot offers the possibility to further validate modeling and control concepts for such systems.
- **Computationally efficient modeling**:
  Existing modeling methods such as the finite element method (FEM), model order reduction and the floating frame of reference approach are combined with projections to transform the resulting differential-algebraic equations (DAEs) to ordinary differential equations (ODEs). Together with the concept of servo constraints, this results in a real-time capable inverse flexible multibody model.
- **Real-time end-effector trajectory tracking**:
  The main focus of this research is end-effector trajectory tracking, which can be regarded as the most difficult control task for flexible link robots [11]. The reason is that using the end-effector as output typically results in a non-minimum phase system, i.e., the inverse system is unstable. Therefore, three ways are developed to render the system minimum phase. Firstly, output redefinition is considered. This has been done before, e.g., via a linearly combined output within simulations [15–17]. The presented research does not use a linear approximation of the output but directly weights the elastic deformations and rotations in the nonlinear output

function. Secondly, the mass distribution of the considered robot is adapted by using an additional counter weight. In contrast to [18, 19], this counter weight is not placed near the end-effector but close to the actuators on the robot base. Adding a small mass at the identified location already results in a minimum phase system and has only a minor influence on the link eigenmodes. This behavior is confirmed within experiments which has not been done within the cited references. Thirdly, a linear controller is applied which internally feeds back the state within the utilized model inversion. The idea is that instead of tracking the complete 2D pose of the end-effector its rotational degree of freedom (DOF) stays uncontrolled. This leads to kinematic redundancy which is used to stabilize the zero dynamics.
All three concepts result in a stable inverse model of FLEXOR for the considered scenarios. Here, the full dynamic flexible multibody model is inverted in real-time. This is a major difference to the related literature, which is typically based on rigid body inverse kinematics or on offline computations for models including flexibility.
Furthermore, experiments show that the end-effector trajectory tracking performance of the output redefinition approach is very close to cases which use the exact end-effector. This is amongst others confirmed for a scenario in 3D with the end-effector being in contact to the environment.

- **Real-time collision avoidance**:
  The output redefinition approach for end-effector trajectory tracking is combined with actuator limit avoidance and obstacle collision avoidance in real-time. This extends the related rigid body literature on collision avoidance to flexible link robots.
  Firstly, existing real-time concepts to avoid actuator limits usually directly adapt the desired actuator motion, which can excite significant oscillations in flexible link robots. Therefore, in this research the actuator motions are only adapted indirectly by changing the desired end-effector trajectory. A flexible model inversion then ensures effective end-effector trajectory tracking with simultaneous actuator limit avoidance.
  Secondly, obstacle collision avoidance is implemented which is very rarely found in the literature of flexible link robots. In contrast to the presented research, existing results only consider static obstacles as well as only serial robots and are not implemented within real-time experiments.
- **Hybrid force/position control**:
  A dynamic flexible model inversion is also used within the framework of hybrid force/position control as opposed to the usual approaches within the related literature on flexible link robots. This allows to also track the normal contact force at the end-effector besides its pose.
- **State and output estimation**:
  For nonlinear state estimation the well-known unscented Kalman filter (UKF) [20] is applied. Here, an adaption for flexible multibody systems with algebraic constraint equations, i.e., in DAE form, is implemented. The

presented approach uses minimal coordinates for the UKF similar to [21], but via a projection based on a QR decomposition while the constraint equations are satisfied with the Newton-Raphson method. This extends the related literature [21, 22].
The end-effector pose of a flexible link robot, being typically the output of interest, cannot be reconstructed only by motor encoders in contrast to rigid robots. Therefore, a marker-based reconstruction of the 3D end-effector pose is performed with a real-time capable motion tracking camera. Such systems are known from tracking human actors but are very rarely applied to flexible link robots.

- **State feedback control**:
  The previously discussed model inversion control concepts are essentially feedforward controllers, which are combined with end-effector and actuator feedback control but do not directly consider elastic measurements. Therefore, the elasticity is used within state feedback controllers to also correct for errors within the link deformations.
  Firstly, input-output feedback linearization is adapted to parallel robots with flexible links. A systematic way via servo constraints is proposed for tracking a redefined output close to the exact end-effector.
  Secondly, a linear-quadratic regulator (LQR) is designed to track state trajectories. In [23–25], LQRs are applied experimentally to end-effector trajectory tracking of serial robots which consist of two flexible links. The presented approach, however, considers parallel instead of serial robots, which complicates the control design.
- **Oscillation damping control**:
  Two systematic concepts are applied to damp occurring oscillations such as after a trajectory tracking scenario or caused by disturbances, e.g., through contact to a human. Firstly, the previously mentioned LQR is used. Secondly, the equations of motion of the structural flexibility are decoupled by a modal transformation. Subsequently, the modal damping is actively increased by the actuator motion. The main contribution is the successful implementation of both concepts on a flexible multi-link parallel robot.
- **Experimental validation**:
  Within flexible link robotics many researches only rely on simulations since experiments are not straightforward to perform and feedback control with elastic measurements is difficult. Nevertheless, a major effort has been made to successfully apply all presented control concepts to flexible link parallel robots within experiments. This is crucial to show the relevance of the concepts within realistic application scenarios. The experiments also include HRI, which is very rarely considered in the literature of flexible link robots. Investigated HRI scenarios are manual guidance, real-time end-effector tracking where the trajectory comes from a game pad and collision avoidance.

In summary, this research contributes to the field of flexible link parallel robots by showing how a variety of inherent problems of such robots can be solved. As the focus lies on systematic and experimentally promising techniques, a variety of well-established approaches are adapted to flexible link parallel robots. Mathematically, parallel robots described by DAEs can be regarded as an extension of serial robots described by ODEs. Therefore, the discussed techniques are also applicable to purely serial robots with flexible links. The proposed methods are validated within realistic experimental scenarios. These confirm on the one hand the need for modern controllers based on flexible multibody models, as in most cases they significantly outperform controllers based on classical rigid multibody models. On the other hand, the applicability of flexible link parallel robots to typical robotic application scenarios can be concluded.
Related publications by the author are [26–36], which partially overlap with the contents of this thesis.

1.4 Structure of the Thesis

In Chapter 2, the newly developed flexible link parallel robot Flexor is introduced. Its differences to existing robots are outlined and the selection of the utilized hardware as well as software components is explained. In Chapter 3, a computationally efficient flexible multibody model is established which serves as a basis for all model-based controllers. Here, the linear FEM and a subsequent model order reduction are utilized to describe the flexible links. The complete flexible multibody system is assembled within the framework of the floating frame of reference approach. Servo constraints are used to invert this flexible multibody model. Projections transform the resulting DAEs to ODEs which are easier to solve. Three different solution methods are discussed for the internal dynamics being classical inversion via forward integration of an initial value problem (IVP), trajectory optimization by solving a nonlinear programming problem and stable inversion by solving a two-point boundary value problem (BVP).
In Chapter 4, a UKF is applied to the underlying system of DAEs. Also, the end-effector pose is reconstructed with a motion tracking camera in real-time. In Chapter 5, end-effector trajectory tracking experiments are conducted. First, admittance control is implemented where a human manually guides the end-effector. Then, three approaches are presented to render the system minimum phase. This allows to use a dynamic model inversion in real-time as feedforward control. Such a controller is further utilized with feedback for collision avoidance and for hybrid force/position control. Afterwards, input-output feedback linearization is applied to end-effector trajectory tracking. Finally, comparisons are drawn to offline calculated approaches which track the exact end-effector. In Chapter 6, an LQR and a modal approach are used to actively damp the oscillations of flexible links. Chapter 7 concludes this thesis.

Chapter 2

Flexible Link Robots

Chapter Contents

This chapter gives an overview of different flexible link robots which have actually been built. Shortcomings and advantages of these systems are identified. To show the practical relevance of the later considered modeling and control algorithms, an appropriate experimental test rig is inevitable. Therefore, a new custom flexible link robot is developed which solves several of the identified shortcomings. The resulting highly adaptable research platform shall be used to investigate a variety of scenarios and control approaches from classical rigid body robotics but extended to flexible link robotics.

2.1 State of the Art

In this section, a gap analysis is performed based on existing flexible link robots in order to identify the design requirements for a new robot for research purposes.

2.1.1 Existing Flexible Link Robots

Flexible link robots are rarely found in industrial applications. Here, the focus lies on flexible joint robots such as Sawyer from Rethink Robotics or the LBR iiwa from Kuka. Flexible links occur typically only in special applications where large size renders stiff designs infeasible. This is often the case for space applications where a small weight is extremely important. See for example the 11.3 m long European Robotic Arm in Figure 2.1a). Also, since 1998 Magirus cooperates with the University of Stuttgart to actively damp oscillations in aerial ladders, see Figure 2.1b). Within a cooperation with Liebherr, the same research group considers structural flexibilities which occur in tower cranes, see Figure 2.1c). This shows that applications which use flexible links are still often closely connected to research. The reason is that for an accurate modeling and control typically modern and complex approaches are required. Nevertheless, such industrial large-scale machines are expensive, they cannot be easily adapted or operated and concepts such as HRI can hardly be investigated. Consequently, they are not further considered within the design process of a new flexible link robot.

Flexible link robots have been under investigation in research labs for several decades. The single-link flexible robot depicted in Figure 1.1a) has been utilized for experiments in 1984. Here, two flexible parts are used in parallel with a series of bridges between them to increase the torsional stiffness. Thus, torsion effects

a) The European Robotic Arm architecture [37]

b) Aerial ladder IVECO Magirus M60 L [38]

c) Simulator for tower cranes by Liebherr including structural dynamics [39]

Figure 2.1: Flexible links and structures in large-scale applications.

become negligibly small. Additionally, strain gauges are applied to the link, being a simple and popular measurement method to quantify the link deformation. The flexible two-link parallel robot under gravity in Figure 1.1b) has been built in the late 80s [7]. Due to the parallel part the second actuator does not need to be carried by the tip of the first link but is attached near the base. Such a design can reduce structural vibrations which occur in flexible link serial robots through carrying an additional actuator mass.

Further robots applied in research environments are depicted in Figure 2.2, which are divided into serial and parallel robots. Planar robots with two serial flexible links can suffer from significant torsion. In Figure 2.2a) this is solved by using an air-cushion pad. From the illustrated robots, ElRob in Figure 2.2c) is the only 3D robot. This feature allows the investigation of gravitational effects on flexible links such as static deformations.

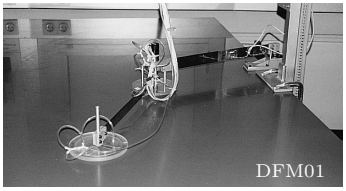
It should be noted that the distribution of serial and parallel robots in Figure 2.2 does not represent the distribution of the actual occurrences. The literature on flexible link robots highly focuses on single links [4] or serial robots, whereas results for flexible link parallel robots are rare [13, 14]. *Parallel robots* are understood as robots which have at least one parallel part, i.e., a kinematic loop. Such parallel parts are interesting as they often introduce significant nonlinearity and the mechanical model typically needs to be formulated with additional algebraic equations, i.e., DAEs instead of ODEs are obtained. This renders the solution process more difficult and thus well suited for research purposes. Introducing significant coupled oscillations, such as through an additional serial part, makes the problem even more interesting. The robots in Figure 2.2d), e), f) show rather complicated parallel mechanisms with several links. For the considered research, a more basic kinematic design is intended to focus on the investigation of dynamic effects. The robots in Figure 2.2h), i) and j) have the same simple parallel structure. However, as this setup has been thoroughly investigated, a different structure using the same simplicity is intended.

In Figure 2.2f), only the last serial link of the parallel robot is flexible. Including flexibility also in parallel parts would be even more interesting. Nevertheless, supported ends of flexible links within parallel parts can significantly increase the eigenfrequencies. This is the case for Figure 2.2g) with 92.5 Hz and for Figure 2.2h) with 23.3 Hz for the first eigenfrequencies. Such high frequencies are not suited for visualizations for the naked eye and they are difficult to simulate as small time steps are needed. An active oscillation damping would also require a very dynamic actuation. Thus, smaller eigenfrequencies, such as via a flexible link with a free end, are more desirable.

In robotics, the end-effector pose is typically the output of interest. In contrast to rigid robots, flexible link robots do not allow to measure this pose only via motor encoders. Therefore, Cannon and Schmitz [6] use an optical device to obtain the 2D position of a flexible single-link robot. A light bulb on the end-effector emits light being focused by a lens above it and a photodetector measures the analog 2D position. The advantage of such a direct measurement is the high

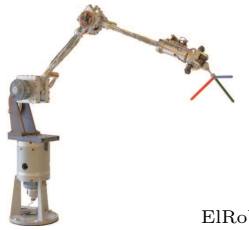
serial robots

a) [24]

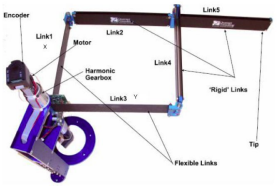


b) [40]

c) [41]

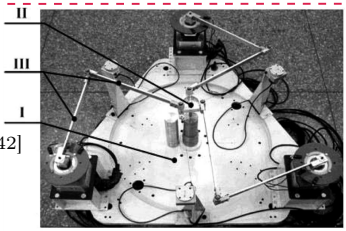


parallel robots

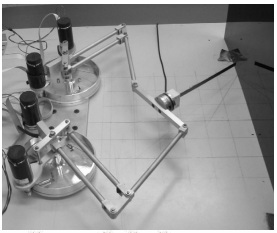


d) [14]

e) [42]

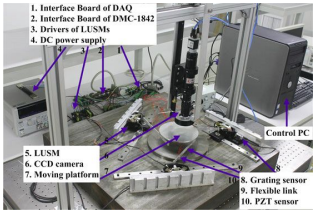


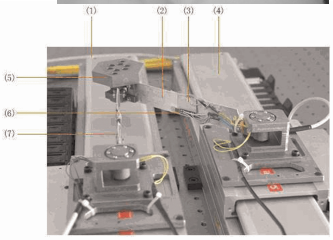
I. Frame platform II. Moving platform III. Kinetic branched chains



f) [43]

g) [13]

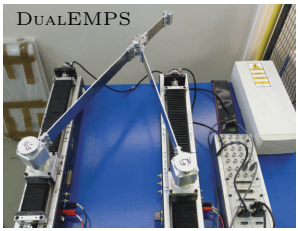


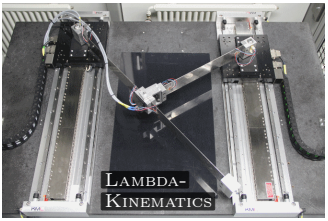


(1) Linear motor A; (2) Flexible link B; (3) PZT; (4) Linear motor B; (5) Payload; (6) Strain gauge; (7) Flexible link A

h) [44]

i) [45]





j) [46]

Figure 2.2: Overview of selected flexible link robots used in research.

accuracy being below 0.5 mm. However, a hood around the sensor is necessary to reduce noise from ambient light. Optical reconstruction of the end-effector position is also conducted for the Lambda-Kinematics in Figure 2.2j). Here, camera measurements of retroreflective foil are used which are however processed offline. For the robot in Figure 2.2e) three linear wire encoders are hinged to the end-effector to reconstruct its pose. This is not feasible for HRI as the wires could collide with a human and non-contact measurements are highly favored as they do not influence the system dynamics. For the robot in Figure 2.2f), Madani and Moallem reconstruct the end-effector position via the joint angles and the estimated elastic deflections. It should be noted that such reconstructions based on indirect measurements can be rather inaccurate as they include the modeling errors for instance of the link flexibility.
All of the presented research robots use either linear or rotary actuators but do not combine them. Also, they are typically designed for one setup, i.e., they are not modular in the sense that, e.g., the system structure or the functionality of the end-effector can be easily changed. Finally, most flexible link robots are used within rather academic scenarios without showing applications such as pick and place, obstacle collision avoidance or HRI.

2.1.2 Gap Analysis

Based on previously discussed existing flexible link robots in research labs, several advantages and disadvantages as well as needs are identified. A resulting gap within flexible link robotics is that there exists no robot that combines:

Structure:	Parallel and serial parts
	Parallel mechanism enabling actuators to be attached to the base instead of being carried along
	Small number of links to reduce complexity but still with multiple flexible links
	Reduction of torsion problems
	3D oscillations to investigate gravitational effects
	Free end of flexible links to obtain first bending eigenmodes with low frequencies and significant amplitudes to improve the visualization of the flexibilities and to ensure that the actuators are fast enough to control the deformations
Actuation:	Combination of linear and rotary actuators
	Two linear actuators which are not aligned collinearly
Size:	Appropriate to collaborate with humans in a sufficiently large workspace
Modularity:	Interchangeable links, purely 2D and 3D setups possible and end-effector applicable to gripping as well as contact tasks
Application:	Usable for realistic scenarios such as pick and place, end-effector contact, HRI, obstacle collision avoidance and end-effector trajectory tracking

Measurement:	Direct real-time measurements of the flexibilities and the end-effector pose with negligible influence on the system dynamics for validation or control purposes
Software:	Easily operable and adaptable software which quickly enables new users such as students to control the robot

2.2 Flexor - A New Flexible Link Parallel Robot

Based on the gap analysis from Section 2.1.2 the new flexible link parallel robot in Figure 2.3 has been developed. It is called Flexor, which originates from the latin word *flectere* meaning to bend. This name represents the main feature of the robot being the bending of its flexible links.
Flexor is a custom made flexible link research platform with a parallel and a serial part. Its system components are labeled in Figure 2.4, which are discussed throughout this section. The parallel part, also denoted as *kinematic loop*, is formed by only two thin metal sheets connecting the revolute joints on the two linear motors. This already results in a highly nonlinear working motion and prevents that one linear motor needs to carry the other one. Aligning these motors in a 90° angle further differentiates Flexor from the existing robots of Figure 2.2h), i) and j). Two additional metal sheets are connected by a rotary motor which introduces the serial part. Thus, a flexible three-link robot based on four metal sheets is formed which uses rotary and linear actuation.
The relevant flexibility of link 2 originates completely from the part 2_1. As it is supported at both ends and since the part 2_2 can be regarded as rigid body due

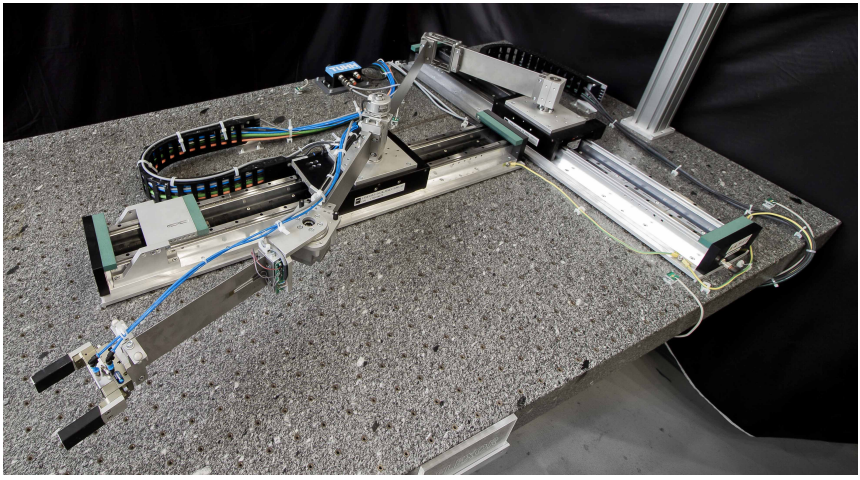


Figure 2.3: The developed flexible link parallel robot Flexor.

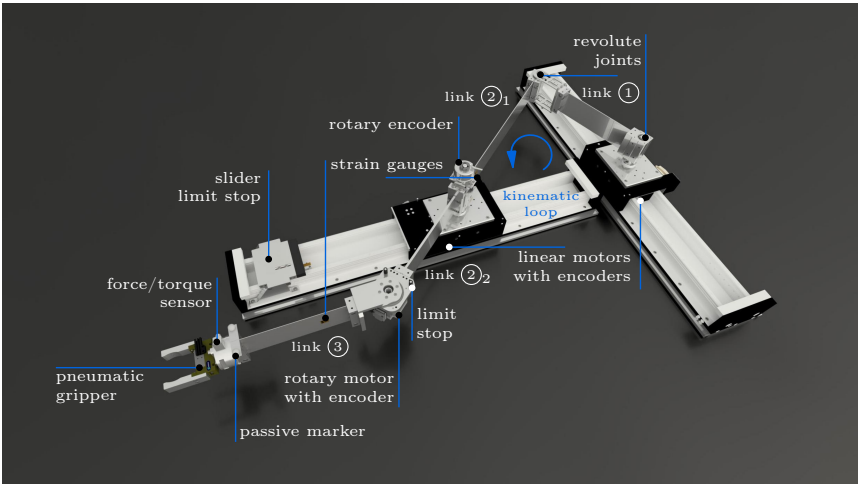


Figure 2.4: Components of the flexible link parallel robot Flexor.

to its thickness, no torsion problems occur in link 2. Because of its shortness and supported ends, link 1 can also be regarded as rigid body. Flexor is consequently a three-link robot with two flexible and one rigid link. Using multiple links allows to investigate more realistic scenarios. A drawback of classical parallel robots is their small workspace [47]. This is not the case for Flexor, where especially the serial part guarantees that the considered scenarios can cover a larger workspace. Besides, the rotary motor can be flipped by 90° transforming the planar robot to a 3D system which includes gravitational effects. This can be seen in Figure 2.5 on the right.
The robot is highly modular with easily interchangeable links, which can have a

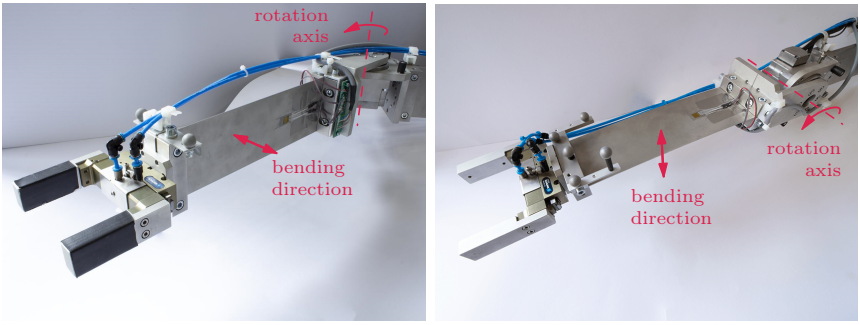


Figure 2.5: Horizontal and vertical link 3.

thickness between 1 mm and 8 mm. The standard thicknesses for link 2_1 is 2 mm and 1 mm for link 3. This leads to first bending eigenfrequencies below 2.25 Hz for the unconnected links 2 and 3. The small frequency of link 2 is directly related to its free end. When link 2 and link 3 are connected and in line, the first bending eigenfrequency of the complete system even reduces to 1.06 Hz. This can result in significant oscillation amplitudes of several centimeters when excited through normal operation.
Adapter plates allow to easily change the end-effector functionality. Different options are visualized in Figure 2.6. Besides a force/torque sensor and a gripper, a wheel-shaped contact element for 2D and a spherical contact element for 3D scenarios are available. The rolling motion of the wheel is used to reduce friction effects in the contact zone. Also, a particle damper can be attached to increase the passive system damping. For more details on this damper it is pointed to the author's publication [35], since it will not be further considered in this thesis.
For measurements of the flexibilities classical strain gauges but also a rotary encoder in the middle of link 2 are used. To obtain the end-effector pose an infrared (IR) motion tracking camera is utilized, which extracts marker centers in real-time. Problems with ambient light are consequently eliminated and this

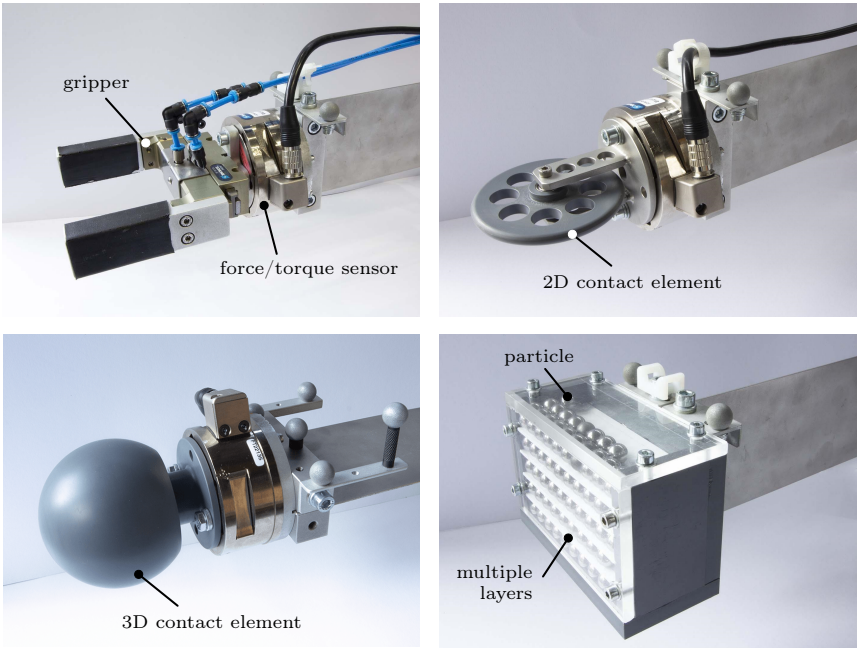


Figure 2.6: End-effector variants.

direct measurement method allows to accurately validate the later discussed model-based controllers. By attaching markers, e.g., to the hands of a user, also HRI scenarios such as collision avoidance can be investigated.
For the software, graphical programming is applied via MATLAB/SIMULINK and LABVIEW. These tools allow inexperienced users to quickly understand and adapt existing programs and help to reduce programming errors.

It can be concluded that FLEXOR is an adaptive flexible link parallel robot which can be used within a variety of realistic scenarios. Thus, it is well suited to serve as a research platform for the experimental validation of modeling and control approaches for flexible link parallel robots. Further details on the utilized hardware and software components are explained in the following. This shall clarify important design decisions which can be helpful within the development of flexible link parallel robots.

2.2.1 Mechanical Parts

Initially, selected mechanical parts are described to highlight important features.

2.2.1.1 Flexible Links

Spring steel is selected for the thin link parts 1, 2_1 and 3. This is especially important for links 2_1 and 3 as they can be exposed to significant stresses. The quite common austenitic stainless spring steel X10CrNi18-8 is used, which comes from coils and needs to be flattened to be applicable as flexible link. For the thick link 2_2 no spring steel is used.

2.2.1.2 Slider Limit Stop

Due to the lengths of link 1 and link 2_1 within the kinematic loop, the sliders are able to pull on the joint between them when they are far enough apart. As this is undesired and since pure software limits are not safe enough, the custom slider limit stop depicted in Figure 2.7 ensures that the left slider is further limited in its range of motion. Through its variable mounting it is compatible with different link lengths within the kinematic loop of the robot.
Figures 2.8c) and d) confirm that the limit stop prevents the mentioned pulling on the joint connecting link 1 and link 2, since the angle between them is never close to 180°. Figure 2.8 also gives an impression of the nonlinear robot kinematics.

2.2.1.3 Contact Walls

For investigating contact scenarios a circular and a flat wood contact wall are available, see Figure 2.9. They are mounted on an aluminum base which can be clamped steplessly to the granite table to which the linear motors are attached.

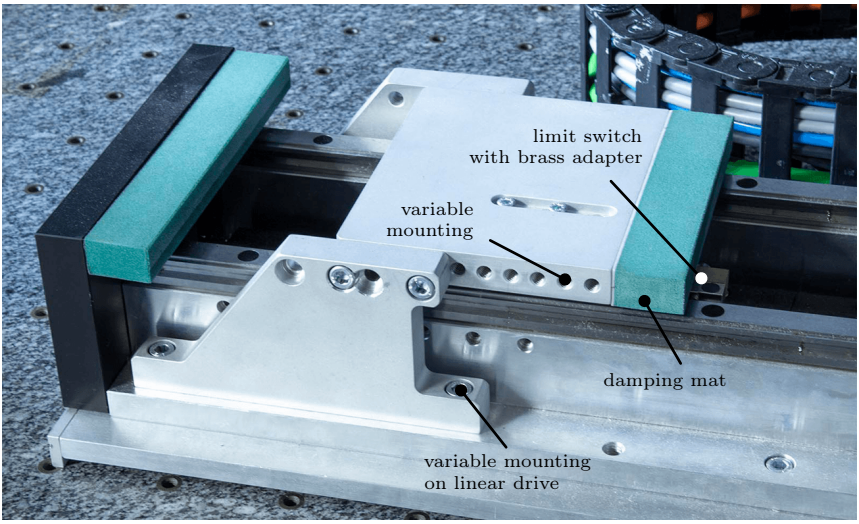


Figure 2.7: Slider limit stop.

2.2.2 Actuators

In many industrial applications only a precise positioning at the end location is important. The focus of the conducted research however are trajectory tracking scenarios. Therefore, a precise actuation of the robot is needed throughout the whole motion and not only at the end location. Such a precise actuation also enables an improved evaluation of the later applied model-based controllers. Therefore, special actuators need to be selected for the robot motion, which are discussed in the following together with the utilized actuators for gripping tasks.

2.2.2.1 Linear Drives

Several technologies exist to obtain a linear motion which in most cases rely on the transmission from a rotating motor. A comparison of different linear drive technologies is given in Table 2.2, with the ball screw having a rotating shaft instead of a rotating nut. As the linear motor stands out in the needed precision and is also very dynamic, this technique is selected. The high precision is related to the fact that linear motors are direct drives, i.e., they have amongst others no gearing which completely eliminates backlash effects. Despite the high acquisition cost this choice prevents investing a significant amount of research time and cost in tuning or compensating some of the errors of a less precise drive system.
Two LTSE165 profile rail slides from SKF are acquired which are driven by two iron core 3-phase linear synchronous motors 1FN3050-2K from Siemens, see Figure 2.10. These linear motors deliver up to 640 N. At a rated force of 265 N

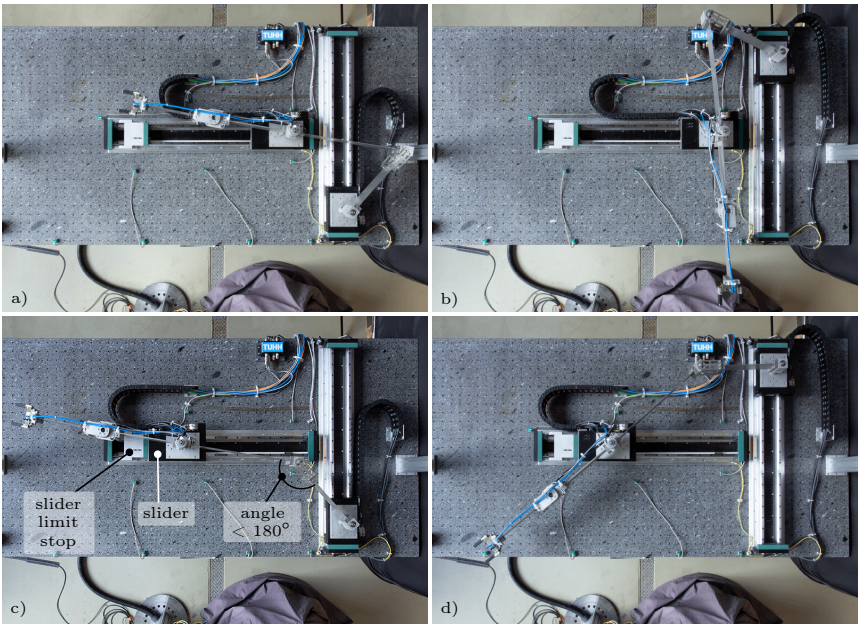


Figure 2.8: Visualization of the nonlinear kinematics.

Figure 2.9: Circular and flat wood contact walls for force/motion control.

a maximum velocity of roughly 6.5 m/s is possible, which enables high-speed scenarios. Because of the iron core the disturbances of the occurring cogging effect need to be compensated by the motion controller. The right drive has a

Table 2.2: Comparison of linear drives based on Wittel et al. [48].

criterion	toothed belt	rack & pinion	ball screw	linear motor	pneu-matics
precision	1	2	3	4	0
velocity	4	3	2	4	2
maintenance	4	1	2	4	1
stiffness	1	3	3	3	0
feed force	2	3	3	2	1
cost	3	1	2	0	4
legend	0 ≡ not suited	1 ≡ less suited	2 ≡ suited	3 ≡ well suited	4 ≡ very well suited

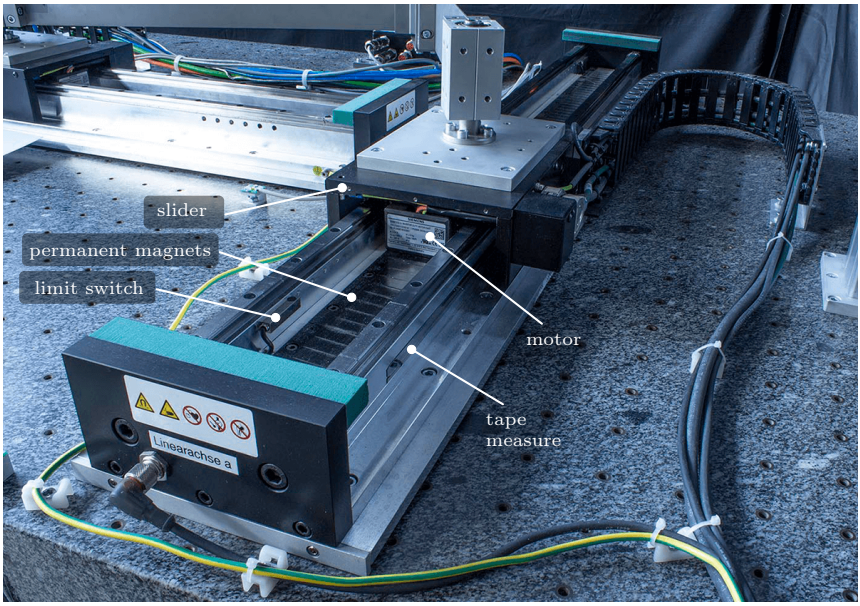


Figure 2.10: Linear motor with components.

stroke of 80 cm, while the left one uses a spring clamping element which reduces the stroke to 74 cm without the slider limit stop. Thus, the left drive can be reused in future vertical applications against gravity where the clamping element can keep the slider at its current position even in the case of a power outage. The incremental optical linear encoder LIA20 from Numeric Jena is used with a tape measure period of 20 µm for both drives. However, via processing such as interpolation of the analogue measurement a resolution of 50 nm can be obtained.

2.2.2.2 Rotary Motor

In robotics, high gear ratios are commonly used for rotary actuation as the speeds are usually drastically smaller than, e.g., in rotor applications. This also applies to the rotary motor of FLEXOR, where a gear is necessary to obtain reasonable velocities and torques. A detailed table comparing different gearing techniques is provided by Isermann [49], with a small excerpt shown in Table 2.3. Here, it is worth noting that parallel shaft and planetary gears have significantly larger backlash than harmonic drive gears. This is especially problematic for the necessary high gear ratio as parallel shaft and planetary gears need several stages and with each stage the overall backlash increases. Depending on the length of link 3, this can result in significant errors at the robot's end-effector. In contrast, only a single stage is needed when using a harmonic drive gear. Especially for the later considered applications which can include lots of direction reversals a harmonic drive gear with negligible backlash is very beneficial.
Based on these considerations, the AC synchronous servo motor FHA-14C from HARMONIC DRIVE is acquired, see Figure 2.11. This motor-gear combination is selected with a ratio of 100 for the single-stage harmonic drive gear, resulting

Table 2.3: Gear comparison based on Isermann [49].

	parallel shaft	**planetary**	**harmonic drive**
number of stages	1-15	1-5	1
gear ratio per stage	3-6	3-10	50-320
backlash	$<0.15°$	$<0.35°$	$<0.05°$

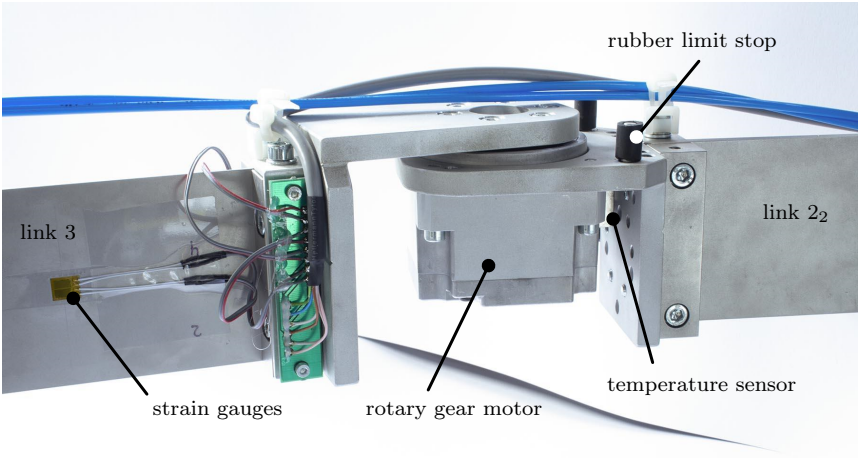


Figure 2.11: Rotary motor connected to the robot.

in a very compact design. At the output a maximum velocity of 2π rad/s and a maximum torque of 28 Nm are available. The high gear ratio further provides sufficient torque over a longer time for contact and 3D scenarios. To ensure that no overheating occurs in such scenarios, an external temperature sensor is used on the motor housing surface as no internal sensor is available. The included TTL incremental encoder has 2000 pulses per revolution on the motor side. The resolution is doubled via processing and together with the gear ratio a resolution on the output side of approximately $1^\circ \times 10^{-3}$ is obtained. For a straightforward referencing without additional sensor the rotary motor drives into contact with a rubber limit stop.

2.2.2.3 Gripper

To investigate pick and place scenarios, a gripper with a large enough stroke is needed for different objects sizes. Additionally, a small gripper weight is desired to reduce the influence on the bending eigenmodes. Pneumatic grippers are well suited as they are typically much lighter and more compact than electric grippers for the same gripping force. They also do not need additional energy for a static gripping force, are usually much cheaper and easier to use. The main downside is that the position of the gripper fingers cannot be controlled, which is however not needed for simple pick and place operations.
The pneumatic gripper KGG 80-30 from SCHUNK is selected, which is a 250 g light parallel gripper, see Figure 2.12. It allows basic gripping tasks with a relatively large stroke of 3 cm, i.e., 1.5 cm per finger. To ensure a safer handling rubber can be used to increase the friction coefficient and the contact area for round objects. A 5/2 bistable magnetic valve is used to operate the gripper, which ensures that the gripper stays closed even in the case of a power outage. Moreover, an exhaust air throttle reduces the closing speed of the gripper fingers to prevent that handling objects get damaged.
For the planar setup of FLEXOR a handling object needs to be either dropped at the target location or the target location needs to be vertically actuated. Here, the easier dropping is chosen in combination with a hole to clearly visualize if a placement is accurate. This concept is realized with the appliance depicted in Figure 2.13. To prevent wobbling of the handling object after it is successfully dropped, four coated metal fingers are used to keep it in place. Round and square shapes of the inner top part allow placement of different handling objects.
Within 3D scenarios the three motors of FLEXOR only allow to control the end-effector position to the target location but not the rotation. Therefore, a spherical handling object is used in the form of a miniature basketball with basket, which can be steplessly adjusted in height and rotation around the vertical axis. Besides object placement the basket has also been used for object throwing, see Figure 2.14. Object throwing can be used to overcome kinematic workspace limitations. Especially flexible links are interesting for such applications, since their spring-like behavior allows to notably increase the throwing distance. Still,

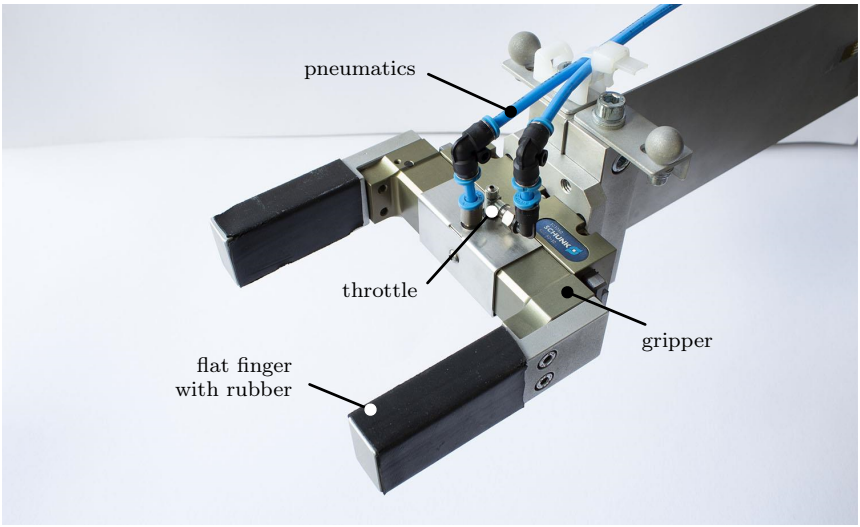


Figure 2.12: Pneumatic gripper.

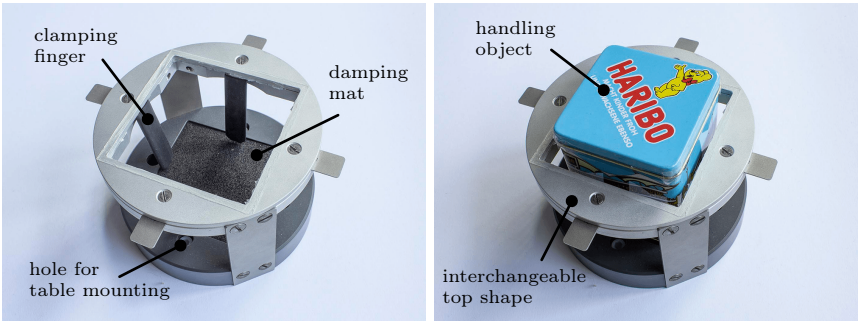


Figure 2.13: Object placement appliance.

existing contributions are very rare. Figure 2.14 is taken from the author's publication [34] where an iterative learning controller is utilized to learn a basketball free throw with a flexible link. Here, pre-learning based on simulations with a flexible link model is used to reduce the subsequent experimental training time. It is shown that with very few additional experimental trials, the learning controller achieves the required free throw accuracy. As this topic is out of the scope of this thesis it is pointed to [34] for details.

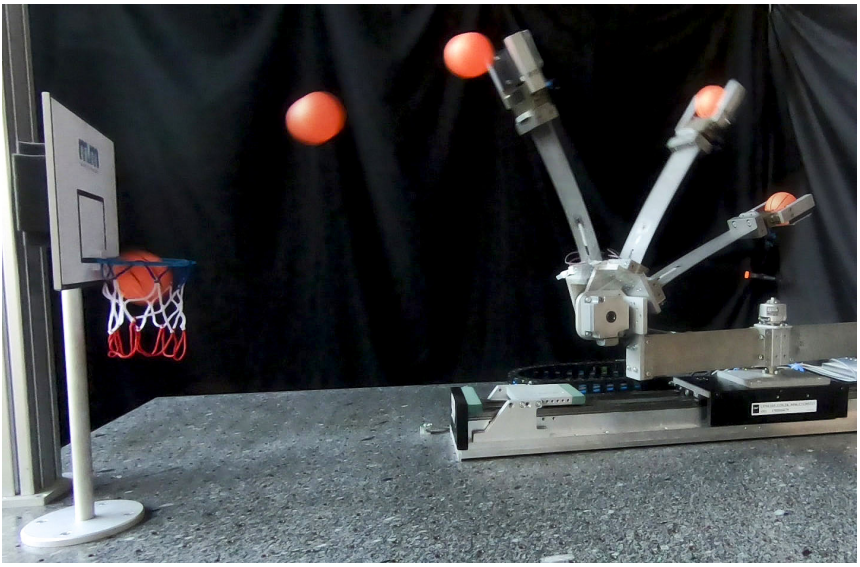


Figure 2.14: Model-based learning of a free throw with a flexible link robot.

2.2.2.4 Midi Servo

Since FLEXOR is planar in the standard case gripping and moving an object would always involve sliding the object on its base. This is typically undesired as the involved friction effects disturb the robot. Therefore, a base is built which can slightly move in vertical direction with low demands on accuracy. A cheap rotary midi servo is used for the actuation, which can be straightforwardly controlled via a micro controller such as the utilized ARDUINO Mega ADK. The base with the servo is depicted in Figure 2.15. With a small lever arm at the output the servo lifts all test objects without problems which weigh 630 g or less. At the lower position internal limit stops within the base hold the weight of the handling object to unload the servo.

2.2.3 Sensors

The previously discussed motors include encoders which are sufficient for typical control purposes for rigid robots. However, for flexible link robots further sensors are needed to quantify the link deformations and to measure the end-effector pose. These sensors are covered in the following as well as a sensor for contact scenarios.

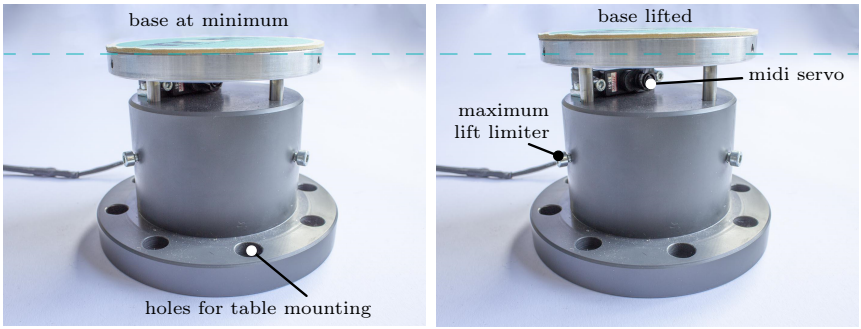


Figure 2.15: Base with vertical actuation for handling objects.

2.2.3.1 Strain Gauges

For validation and control, information is needed on the deflection of the flexible links. An effective and popular method is to measure the strain at selected locations on the flexible links. This could be done with optical fiber sensing based on the fiber Bragg grating (FBG) technology. Franke et al. [50] report a better signal-to-noise ratio of such optical sensors compared to classical electrical strain gauges for their flexible link robot. These sensors are insensitive to electromagnetic disturbances, mechanically robust and one measurement cable can connect many sensors. Nonetheless, for the later discussed control concepts only one measurement point per flexible link is needed, resulting in a high cost per measurement point for the FBG technique. Therefore, very affordable classical foil metal strain gauges are used. They are very well suited for dynamic real-time measurements especially as they are lightweight [51], i.e., the influence on the link oscillations is negligible. Their measurement principle relies on the resistance change due to geometric deformation of the resistor [52, 53].
Since only bending is significant for Flexor, the strain gauge model DY4 from HBM is selected which has two parallel measurement grids, see Figure 2.11. Strain gauges need to be placed at locations where substantial strain can be measured which is also not too large to damage the gauges. For the selected locations on link 2_1 and on link 3, the strains stay below 2‰ in very extreme bending scenarios which rarely occur. In most cases, the strains stay below the recommended 1‰. The relation between the measured output voltage U_o of the implemented Wheatstone bridge circuit and its supply, i.e., input voltage U_i follows according to Hoffmann [51] as

$$\frac{U_\mathrm{o}}{U_\mathrm{i}} = \frac{k}{4}(\epsilon_1 - \epsilon_2 + \epsilon_3 - \epsilon_4). \tag{2.1}$$

The gauge factor is here $k = 2.06$ and ϵ_1, ϵ_2, ϵ_3, ϵ_4 are the strains of the four strain gauges utilized within a full bridge configuration. The strains with negative sign

are related to gauges which are placed on the other side of the considered link. For the rectangular cross section of the links bending leads to the same strain on both link sides but with a different sign. Thus, the full bridge compensates to a high degree effects which influence both sides equally such as normal strains and thermal strains [51]. Also, torsion around the beam axis is not measured. Consequently, Equation (2.1) can be written as

$$\frac{U_\mathrm{o}}{U_\mathrm{i}} = k\epsilon \tag{2.2}$$

with ϵ being the resulting bending strain.
In order to relate the measured voltage U_o to the flexible multibody model introduced in Chapter 3, a relation between the strain and the elastic coordinates of the model is needed. The Timoshenko or Euler-Bernoulli beam theory [54] are popular approaches for the modeling of flexible links, which however should be only applied for a sufficiently large slenderness ratio. Since the Timoshenko beam theory includes first-order shear deformation a slenderness ratio

$$s_\mathrm{r} = GAL^2/(EI) \tag{2.3}$$

being larger than 30 is recommended [55]. Here, G is the shear modulus, A is the cross-section area, L is the overall beam length and EI is the flexural rigidity. The Euler-Bernoulli beam theory completely neglects shear deformation and should only be used for relatively long and thin beams [54]. A slenderness ratio larger than 1000 typically results in negligible differences to the Timoshenko beam theory. For Flexor even the thickest link 2_2 with 6 mm has a slenderness ratio of roughly 10000. Hence, both beam theories are valid. The Euler-Bernoulli beam theory assumes that there is a neutral beam axis which undergoes no extension or contraction and that plane and perpendicular cross sections in an undeformed state remain plane and perpendicular to the neutral beam axis in a deformed state. Based on this, one can relate the strain to the curvature κ as [54]

$$\epsilon = -d_\mathrm{n}\kappa. \tag{2.4}$$

Here, d_n denotes the signed distance from the neutral beam axis. Assuming a small elastic rotation expressed as $\boldsymbol{\psi}_z\boldsymbol{q}_\mathrm{e}$, the curvature over the beam length can be written as

$$\kappa(x,t) = \frac{\mathrm{d}\boldsymbol{\psi}_z(x)\boldsymbol{q}_\mathrm{e}(t)}{\mathrm{d}x} = \frac{\partial\boldsymbol{\psi}_z}{\partial x}\boldsymbol{q}_\mathrm{e}, \tag{2.5}$$

with $\boldsymbol{\psi}_z$ being the row vector of the rotational shape functions and $\boldsymbol{q}_\mathrm{e}$ being the elastic coordinates. The shape functions are computed from the later discussed finite element models, which are used within the flexible multibody model of Flexor. The curvatures at the strain gauge measurement points of link 2 and link 3 are denoted as κ_2 and κ_3 throughout this thesis and make up the vector of curvatures $\boldsymbol{\kappa} = [\kappa_2, \kappa_3]^\mathrm{T}$. With Equations (2.2), (2.4) and (2.5) the measured voltage U_o can be finally related to the elastic coordinates $\boldsymbol{q}_\mathrm{e}$ of the model as

$$\frac{\partial\boldsymbol{\psi}_z}{\partial x}\boldsymbol{q}_\mathrm{e} = -\frac{U_\mathrm{o}}{kd_\mathrm{n}U_\mathrm{i}}. \tag{2.6}$$

By measuring the strain on the surface, d_n becomes half the thickness of the considered links. If the neutral beam axis is extended or contracted against the assumptions made, Equation (2.6) can still be used since the full bridge compensates such normal strains which also do not affect the rotations in the utilized linear model. This means that both sides of Equation (2.6) do not account for such an effect and stay unchanged.

2.2.3.2 Rotary Joint Encoder

In its middle, link 2 is supported by a joint. Measuring the corresponding rotation is an interesting alternative approach to strain gauges for quantifying the link deformation. Based on a mechanical robot model the underlying rigid body rotation can be canceled to obtain the part of the rotation which is caused by an elastic deformation.
The absolute rotary optical encoder ECN425 from HEIDENHAIN is utilized which has a resolution of 25 bit per revolution. This corresponds approximately to a resolution of $1^\circ \times 10^{-5}$. The encoder can be conveniently attached to all revolute joints of the robot. Since the axles of these joints have a threaded hole they are simply elongated to mount the encoder, see Figure 2.16. Experimental tests show that the strain gauges and the rotary encoder are similarly qualified to quantify the link deformation. With an estimator, such as the later discussed UKF, fusion of both sensor signals may be realized to increase the measurement reliability.

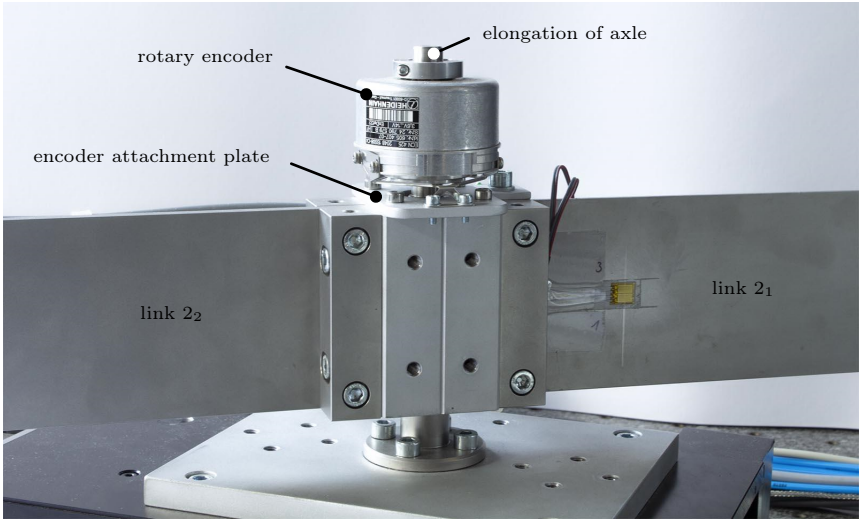


Figure 2.16: Rotary encoder attached to joint.

2.2.3.3 Force/Torque Sensor

For trajectory tracking of contact forces and manual guidance, a 6-axis force/torque sensor is needed. In such a guidance scenario, the measurement of torques allows an intuitive manual control of the end-effector rotation, see Section 5.1.2. For the considered applications, a measurement principle with small drift and fast dynamics is required to ensure an accurate measurement of static and dynamic forces. Thus, piezoelectricity is not recommended since it tends to result in significant drift [56]. Further principles are discussed in the book of Stefănescu [57]. According to this reference, resistive force sensors are the most common, they are reliable, simply constructed, have an adjustable resolution and use a maintenance-free technology. Electrical resistance can also be regarded as the easiest electrical property to be measured precisely over a wide measurement range at moderate cost.
This well-suited principle is selected in the form of the resistive monolithic 6-axis force/torque sensor FTD-GAMMA from ATI and SCHUNK, see Figure 2.6. Besides being appropriate for dynamic measurements, the drift of static force measurements is sufficiently small after a short warm-up period. For the sensor, the calibration SI-32-2.5 is used, i.e., up to 32 N can be measured for the lateral forces and 2.5 Nm for all torques. Also, a normal force up to 100 N is measurable. The 16 bit strain gauge data leads to a resolution of the normal force of 1/80 N and of the lateral forces of 1/160 N. The torques are resolved by 1/2000 Nm. Recalibration is possible if larger measurement ranges are needed. The sensor can be overloaded by more than 30 times its measurement range. This is especially advantageous for research environments to ensure that new controllers in their test phase do not damage the sensor in the case of a malfunctioning.
Metal strain gauges as used for the flexible links rely on a resistance change due to geometric changes of the resistor. The force/torque sensor however uses semiconductor silicium strain gauges. They rely on the piezoresistive effect where mechanical stress causes a change in resistivity of the material [52, 53]. A major advantage of semiconductor strain gauges is a much higher sensitivity compared to metal strain gauges. The resulting higher gauge factor allows to obtain a useful signal from a very stiff sensor [58], with a high first eigenfrequency that has an insignificant influence on the robot dynamics. The smallest eigenfrequency of the sensor is at 1.4 kHz, with typical deformations being in the lower micrometer range. As this is negligible compared to the link flexibilities, the sensor will later be modeled as a rigid body.

2.2.3.4 Motion Tracking Camera

Tracking Technology
An accurate measurement of the end-effector pose is needed to evaluate the later presented end-effector trajectory tracking controllers. Indirect methods using, e.g., the previously discussed strain gauges and encoders within a kinematic robot model are typically not very accurate as they include modeling errors of the

links. In addition, a validation method for model-based controllers should be available which is not based on the same models. Therefore, a direct measurement technique with an external device is recommended. This also allows to track, e.g., humans interacting with the robot, see Section 5.2.2. In order to use the measurements for control purposes they further need to be real-time capable.
A large variety of possible localization techniques exists. The end-effector pose could, e.g., be obtained via an inertial sensor, i.e., via accelerometers and gyroscopes by time integration. This can however lead to significant localization errors. Further localization techniques are discussed in the survey paper by Mainetti et al. [59]. Here, camera vision is reported as the most accurate technology compared to infrared radiation, ultrasound, Bluetooth, RFID and WLAN. With the increased interest of recent years in camera vision also lots of hardware and software solutions exist. Based on these considerations, camera vision is used in the following.
For this research, the motion tracking camera Prime 17W by OPTITRACK is selected, see Figure 2.17. Typically, several of these IR cameras are used at different positions for marker tracking on 3D objects such as human actors for animation purposes. Utilizing several cameras together with a graphical user interface (GUI) is also required by most companies for similar products. In contrast, OPTITRACK allows to use a single camera via a free C++ software development kit (SDK). The camera comes with onboard image processing to extract the 2D position of visible markers in pixels. This is done with up to 360 frames per second (FPS) at the full resolution of 1664 pixels × 1088 pixels with a latency of only 2.8 ms. Thus, the onbard processing is fast enough such that the extracted marker data can be used within the robot control. This is typically not possible with standard video cameras. Therefore, e.g., Burkhardt et al. [46] use the video material of a standard camera only within offline control validation. Also, Cannon and Schmitz [6] report problems with ambient light for their optical measurement of the end-effector position. This is not the case with the IR technology of the utilized camera, which is overall a well-suited device for tracking the end-effector pose of flexible link robots.

Figure 2.17: Motion tracking camera Prime 17W with active LED ring.

Markers

The utilized camera can be applied with passive retroreflective markers and with active IR light-emitting diode (LED) markers. For the passive markers the LED ring around the camera lens is used to illuminate the scene with IR light, while the active markers emit their own light. To reduce reflections when illuminating the scene, shot peening is used for most metal parts. Setting the search area close to the expected marker positions further excludes remaining disturbances from reflections.

In this research, passive markers are used for tracking the robot's end-effector and humans interacting with the robot, as no long moving cables or batteries are necessary. For a planar motion two markers are sufficient, see Figure 2.18. Here, the rotation angle around the vertical axis is calculated from the positions of these markers. A reasonable distance between them is required to ensure an accurate reconstruction of the rotation. For the utilized 3D reconstruction at least four markers are necessary. An asymmetrical shape of the marker pattern should be used to ensure a unique result and occlusions need to be prevented.

To identify the intrinsic parameters of the camera a large number of markers is beneficial. For such cases, retroreflective foil in the form of dots is used as it is significantly cheaper than the spheres on the end-effector or active markers. Nevertheless, the calibration board utilized for intrinsic parameter identification also allows to use spherical markers via threaded holes, see Figure 2.19. This level and stiff aluminum board enables an accurate calibration. Since the detection of foil markers is much more angle dependent than of spherical markers, they are only used for calibration purposes where the offline processing can easily cope with a missed marker. Handles on the board ensure that the user does not occlude markers while calibrating. Figure 2.20 shows the different passive markers via a standard camera on the left and via the IR camera on the right. The disturbances result from reflections of the camera light.

Active IR LED markers as shown in Figure 2.21 are attached to static positions on the granite table. These markers are used as reference positions for extrinsic

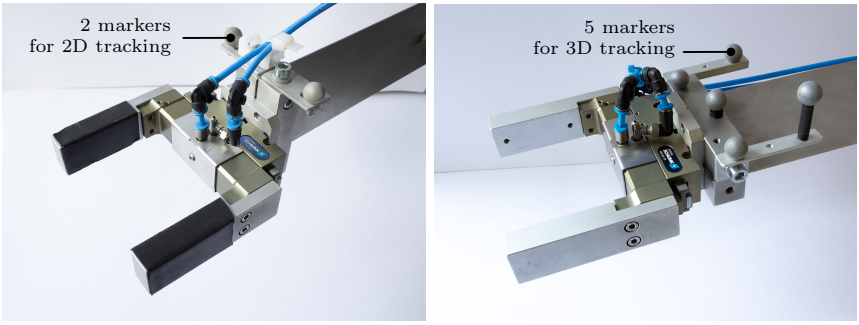


Figure 2.18: Marker patterns for 2D and 3D motion tracking.

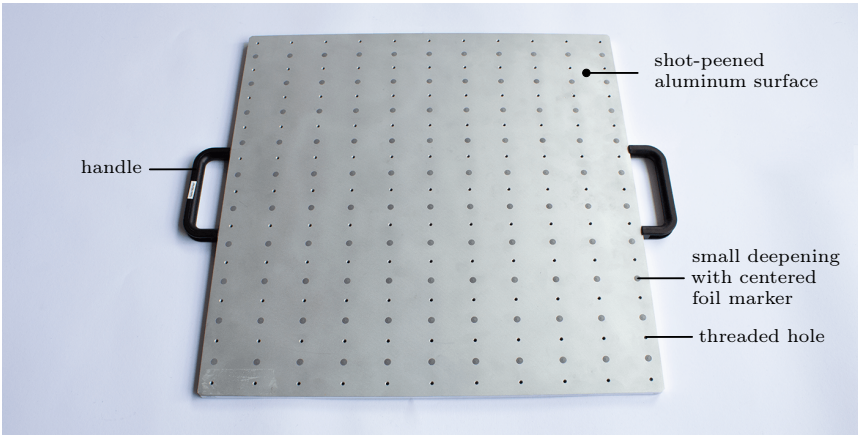


Figure 2.19: Calibration board for intrinsic parameter identification.

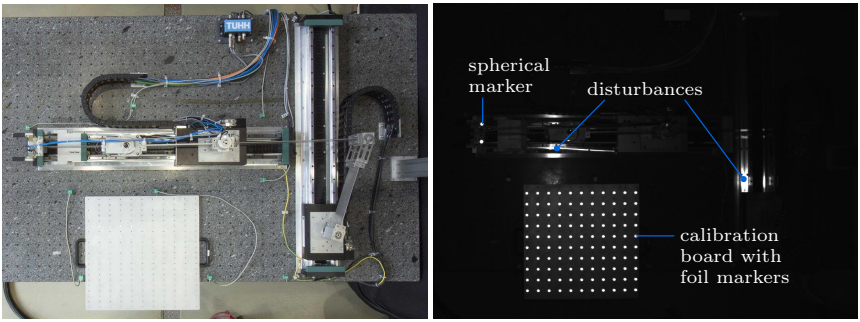


Figure 2.20: Spherical markers and the calibration board taken with a standard and the IR camera.

calibrations, i.e., to determine the pose of the camera relative to the robot. The utilized marker pattern can be seen in Figure 2.22. In an uncalibrated case it could be difficult to find passive markers when reflections are close, since search areas for the markers in the camera image cannot be defined yet. Active markers are here advantageous as no problems with reflections occur. Turning the active markers on successively also directly provides a correct labeling by the camera.

Camera Mounting

To ensure accurate measurements the camera needs an appropriate mounting. The realized solution is a simple design with 3 aluminum profiles from ITEM, see Figure 2.23. Two columns are fixed on the table to prevent a relative motion between the camera mounting and the robot base. These columns are thin

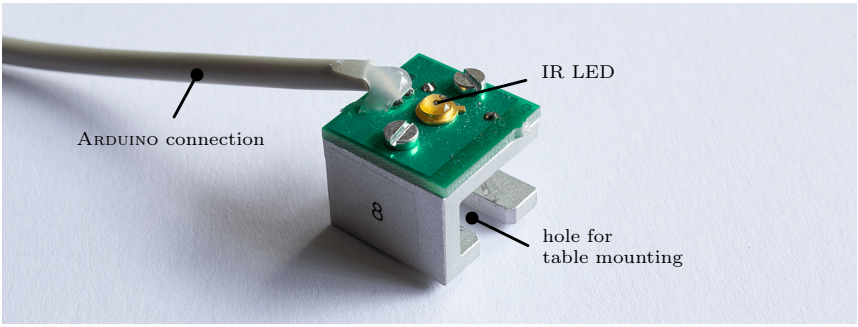


Figure 2.21: Active IR LED marker for extrinsic parameter identification.

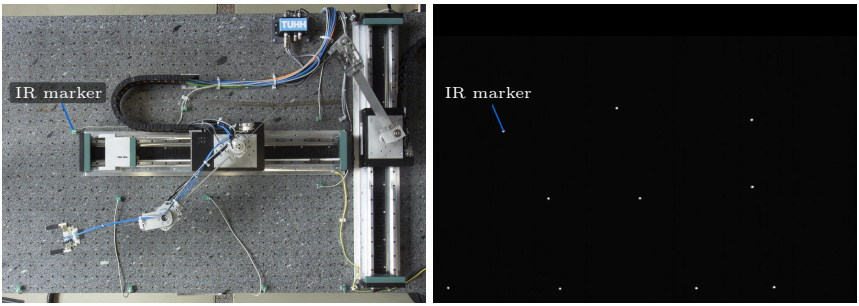


Figure 2.22: Active IR LED markers taken with a standard and the IR camera.

enough to ensure that no collisions with the robot occur. The columns support a lightweight crossbar which can be easily moved. As the camera has no zoom lens, this might be desired in scenarios with a small range of motion in order to exploit a larger part of the camera resolution for the actual measurement. The first eigenfrequency of the mounting with camera is approximately at 11 Hz in the worst case, i.e., with the crossbar at maximum height. For this case, a maximum relative motion between a static marker on the table and the camera of 0.2 mm has been measured for an excitation through significant robot motion. This is sufficiently small to not disturb the later conducted measurements as, e.g., the tracking errors of the end-effector are in the millimeter and centimeter range.

2.2.4 Infrastructure

Since Flexor shall be applied to a variety of scenarios, several industrial, consumer and laboratory components are involved. To provide a better understanding of the relation between these components, the resulting software and

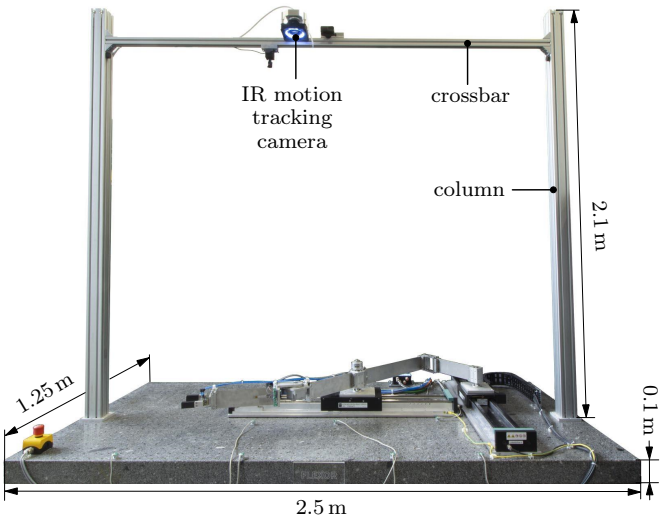


Figure 2.23: Camera mounting.

hardware infrastructure, shown in Figure 2.24, is discussed in the following. This infrastructure has been more compactly introduced by the author in [32].

2.2.4.1 Nondeterministic Layer

A distributed system is used for FLEXOR, which can be divided into a deterministic and a nondeterministic layer. The nondeterministic layer consists mainly of the human-robot interface and the motion tracking of the robot.
The WINDOWS host runs a LABVIEW GUI which allows to control the functions and parameters of the robot, visualizes signals online and collects measurement data within the Hierarchical Data Format (HDF). The desired end-effector trajectory can be applied via this GUI or by a standard USB game pad, which gives the user vibration feedback via the XInput API. With this game pad the user may also perform a gripping task.
The later designed trajectory tracking controllers are validated with the marker-based IR tracking camera running at 360 FPS. The camera extracts the marker centers in pixels onboard and in real-time. To access this data, the CAMERASDK from OPTITRACK needs to be used which has to run on a WINDOWS operating system. It is called within a SIMULINK C++ S-Function since MATLAB/SIMULINK allows an easy processing and visualization of the marker pixel data. This data needs to be transformed into a physical pose with respect to the inertial system. This is realized via additional SIMULINK C++ S-Functions, which call functions from the computer vision library OPENCV. The complete SIMULINK camera tracking program is built with the NATIONAL INSTRUMENTS (NI) VeriStand

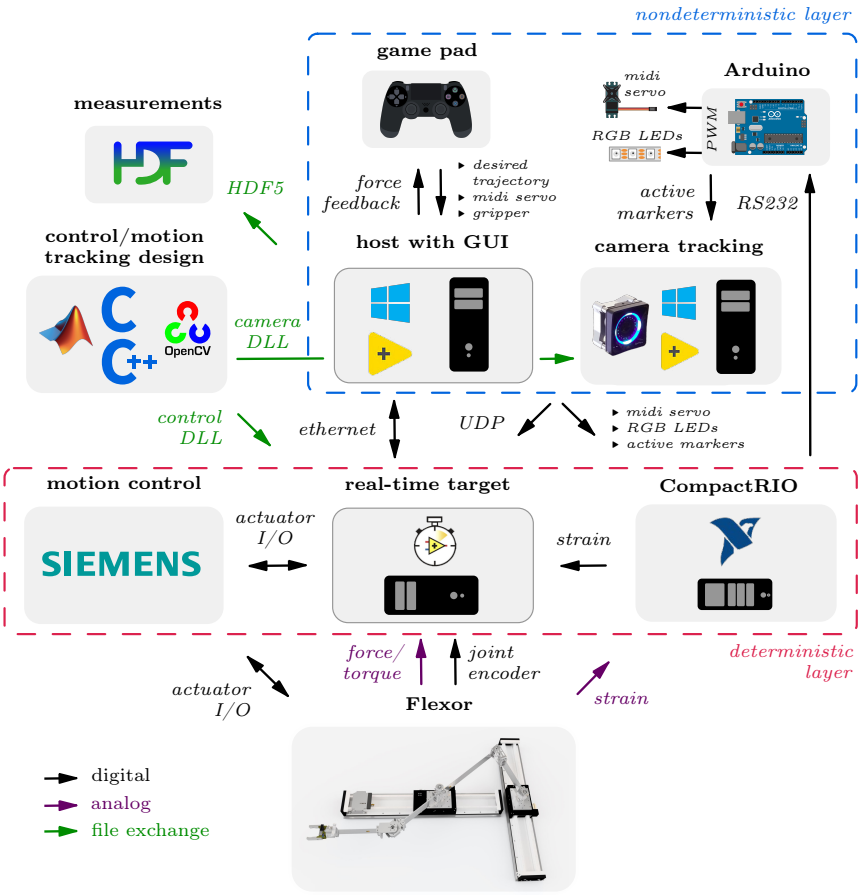


Figure 2.24: Infrastructure of Flexor (with logos from Mathworks, OpenCV, HDF Group, Microsoft Windows, National Instruments and Siemens).

Model Framework into a camera dynamic-link library (DLL) for Windows targets. This export and by using a dedicated computer for the camera tracking program with real-time priority yield a behavior close to deterministic, although Windows is no real-time operating system. The processed data, such as the physical end-effector pose, is sent via UDP to the real-time target for measurement and control purposes. Compared to TCP, UDP offers typically smaller delay and delay jitter [60] being especially important for real-time control. The complete latency of the camera tracking system until the data is received on the real-time target

is roughly only 3 ms and highly constant. This value is obtained by measurement comparison with an encoder of a linear motor assuming that it has no noteworthy latency.
Besides, an ARDUINO Mega ADK is used to control the midi servo for gripping tasks, RGB LEDs to provide visual feedback of the robot state and to turn on the active IR markers. As a deterministic behavior is not required here, delays are used for activating the markers successively to obtain the desired labeling by the camera.

2.2.4.2 Deterministic Layer

The deterministic layer consists of the low-level control of current and speed for the linear and rotary motors, of an embedded measurement device from NI called CompactRIO and of a PC-based real-time target as the high-level controller.
The low-level control is done by the industrial motion controller Simotion D435-2 and the Sinamics S120 variable-speed drive from SIEMENS. The high-level control is performed on a LabVIEW real-time target running on an INTEL CORE i7 CPU at 4 GHz with 4 cores. Here, the Phar Lap ETS real-time operating system is used. The target runs the state machine of the robot, executes the deterministic and nondeterministic communication between the different controllers, processes measurement signals and runs the model-based control. The model-based control consists of standard graphical SIMULINK components and is discussed in detail in Section 2.2.5. The dynamic parts of the model-based control are generated with the MATLAB-based flexible multibody system toolbox NEWEUL-M^2, which also provides an export to C code. This C code is included within SIMULINK C S-Function blocks. The complete SIMULINK model is built with the NI VeriStand Model Framework into a DLL. This DLL is then embedded into the controller software running on the real-time target. Since the scientifically relevant controllers are all implemented in SIMULINK, researchers and students can directly and automatically build their successfully simulated controllers and run them on the hardware. This can significantly reduce research time.
The loop sample time of the low-level motion controllers for velocity and current is 125 µs. The deterministic loop on the real-time target, which contains the model-based control, runs at a sample time of $T = 1$ ms. The communication between the target and the Simotion is done also at a rate of 1 kHz via Isochronous Real-Time (IRT) PROFINET, where the Simotion is the master and the target is the slave. Data acquisition is performed on an NI reconfigurable DAQ FPGA board at higher rates in order to enable signal processing such as digital filtering between two control time steps. The joint encoder from HEIDENHAIN with its digital EnDat interface is connected to this FPGA as well as the force/torque sensor. Since the CompactRIO can be used with the available NI 9237 simultaneous bridge module the strain gauges are connected here. The converted data is transferred via an 8 bit parallel data link implemented between the FPGAs of the CompactRIO and the real-time target. This ensures deterministic data transfer at rates significantly higher than 1 kHz.

2.2.5 Overall Control Structure

A variety of control concepts will be discussed throughout this thesis. To clarify the big picture, the overall model-based control structure is explained in the following.

2.2.5.1 Overview

The overall control structure is shown in Figure 2.25. All visualized functions are implemented in Simulink and run on the real-time target. Only the velocity and current loop of the cascade control run on Siemens components, which realize the desired actuator motion. In addition, the output functions such as data logging and force feedback are performed within a LabVIEW environment.
The control structure in Figure 2.25 is divided into blocks which combine functions of the same type. Several of these blocks include alternative control approaches from which the user has to select one. The data transfer between the blocks is implemented via a data bus. Hence, all signals are available in each block by their names and the visual layout is very clean, which prevents programming errors.
From the left of Figure 2.25 input signals such as measurements arrive. These measurements are used for the subsequent estimation. Next, different ways of trajectory generation are possible, e.g., via a game pad or an offline predefined motion. Afterwards, the trajectory can be adapted, e.g., to prevent collisions with the actuator limits or with external objects. The object collision avoidance is the only controller which needs feedback of a subsequent block. Here, control point positions on the robot are taken from a corresponding model inversion of a rigid or a flexible robot model. Such model inversions are only meaningful if trajectories are provided for the tracking output being typically the end-effector. If no model inversion is selected the actuators are controlled directly. Finally, active oscillation damping might be added. All listed controllers are explained and applied throughout this thesis.
Before using the controllers within experiments, their real-time capability needs to be confirmed. The computational load of the most complex model-based controllers and estimators is depicted in Table 2.4. Here, link 1 is always modeled as rigid body and the flexible links include only the first bending eigenmode. In Table 2.4, the basic load is independent of the selected link setup. It includes the data processing and communication within LabVIEW as well as basic processing and signal filtering within the Simulink model which also runs inside LabVIEW. The UKF uses the forward Euler method for time integration and the model inversion with flexible links is integrated via the explicit fourth-order Runge–Kutta method. Both use a step size of 1 ms and lead to a significant load. As the LQR and the feedback linearization do not contain a time integration of the robot model, the computational load is much smaller. Besides, model inversion with only rigid links is purely based on kinematics and causes therefore negligible loads.

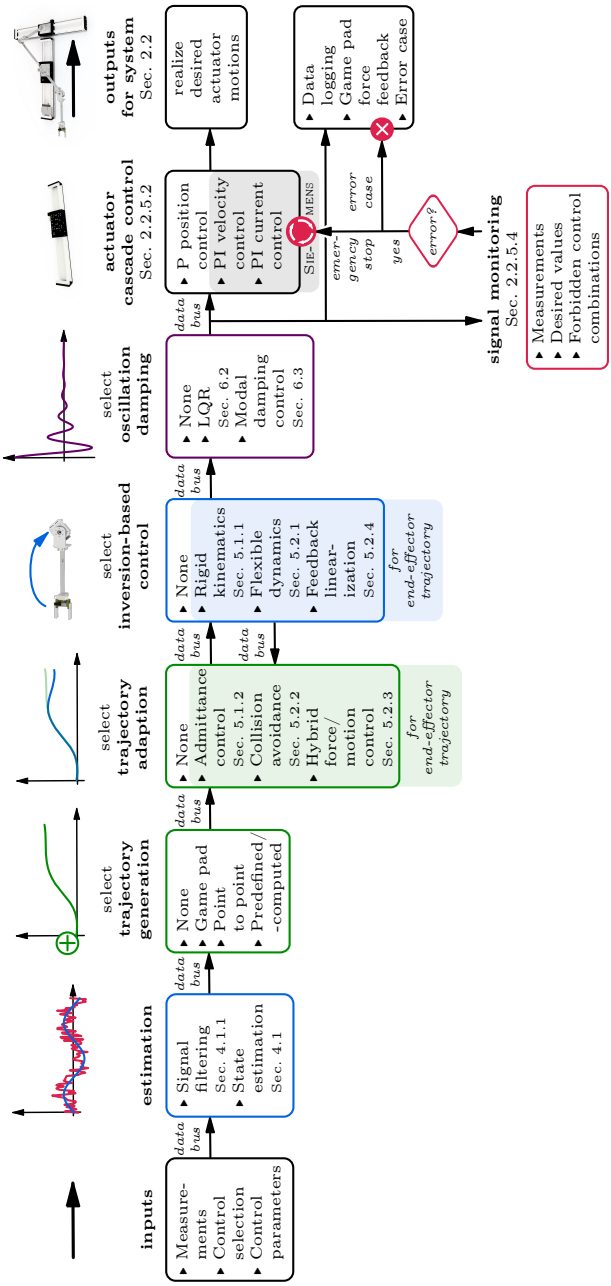


Figure 2.25: Overall control structure of FLEXOR.

Table 2.4: Computational loads of a single 4 GHz i7 core of the real-time target.

link setup	*without link 3*		*with link 3*	
in 2D	**all rigid**	**link 2 flexible**	**all rigid**	**links 2/3 flexible**
basic load	20%	20%	20%	20%
UKF	–	19%	–	65%
model inversion	<1%	19%	≈ 1%	42%
LQR	-	5%	-	21%
feedback linearization	-	5%	-	10%

Table 2.4 clearly shows that the computational complexity rises significantly when attaching a flexible link 3. As a result, the UKF and the flexible model inversion cannot be used on the same core. Therefore, the UKF runs on a separate core and sends its results in each time step to the main model-based control loop. The resulting time delay of 1-2 ms is unproblematic.
Based on Table 2.4, it can be concluded that the utilized hardware and the later discussed modeling, estimation and control approaches provide real-time capability.

2.2.5.2 Actuator Cascade Control

The basis of all experimentally applied control concepts is the actuator cascade controller depicted in Figure 2.26. Here, q_a is the position or rotation of the considered actuator and $\dot{q}_\mathrm{a}$ is the corresponding velocity. The related desired values are $q_\mathrm{a,d}$ and $\dot{q}_\mathrm{a,d}$. The current is denoted as i and u_v is the voltage. The cascaded structure can improve disturbance rejection and allows to implement desired trajectories on position and velocity level instead of only forces or torques. This helps to compensate to a large extent effects of, e.g., unmodeled friction in the actuators. The desired motion can consequently be realized with only small deviations.
According to Figure 2.26, the desired velocity is being adapted by the position controller within SIMULINK and is then sent to the SIEMENS velocity controller. In

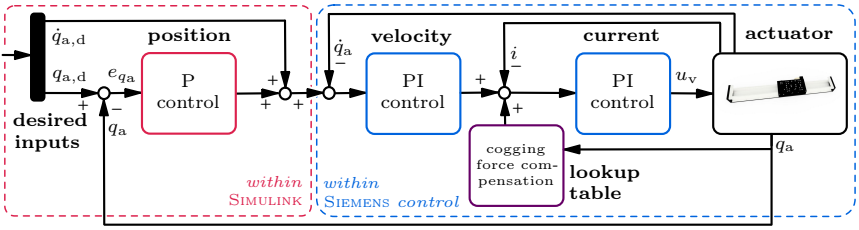


Figure 2.26: Cascade control for a single actuator.

the subsequent current loop an additional cogging force compensation is used for the linear motors. The TTL encoder of the rotary motor is not suited for cogging torque compensation. Not illustrated are filters and safety functions which saturate the maximum currents and kinematic actuator values. The electrical dynamics of the actuators is also not considered as it is much faster than the mechanical dynamics. The AC motors are here controlled like DC motors since the necessary transformations are done internally by SIEMENS components.
The loops of the cascade control are tuned from inside to outside since the inner loop is the fastest. The current loop gains are provided by SIEMENS and typically do not need to be adapted. The proportional gain of the velocity loop should be tuned first and afterwards the integral part. All PI controllers have the form

$$Pe(t) + I \int_0^t e(\tau)\mathrm{d}\tau, \tag{2.7}$$

with e being the corresponding control error. The tuned proportional and integral gains P and I of the cascade control are listed in Table 2.5. It should be noted that in contrast to the linear motors the rotary motor has a transmission with a gear ratio of 100. The SIEMENS motion controller controls the motor side being the reason for the small velocity gains of the rotary motor which would be higher by a factor of 10000 on the output side. Also, the results of the velocity control can be transformed to ampere via the motor constants. With the utilized gains the maximum absolute errors of the actuators on the output side typically stay below 3 mm and 1° even for a very dynamic motion. Still, the control gains are not very stiff to ensure a safe operation in all scenarios especially for HRI. The positioning accuracy at the start and end points of a motion is in the range of the measurement resolution of the corresponding actuator.

2.2.5.3 Human-Robot Interaction

Since flexible link robots are compliant and typically much lighter than classical industrial robots, they can notably decrease the potential of damage and injuries. Therefore, such robots are predestined for HRI applications which have gained increased interest in recent years. Still, contributions are very rare for HRI within flexible link robotics. Because of the high relevance a compact outlook is given here on the later presented HRI applications for FLEXOR. These are summarized in Figure 2.27. A GUI is implemented in LABVIEW with a structure analog to

Table 2.5: Actuator cascade control gains.

	linear motors		*rotary motor*	
loop	**P gain**	**I gain**	**P gain**	**I gain**
position	$15\,\mathrm{s}^{-1}$	-	$15\,\mathrm{s}^{-1}$	-
velocity	7000 Ns/m	3500000 N/m	0.045 Nms/rad	9 Nm/rad
current	100 V/A	50000 V/As	8.5 V/A	4250 V/As

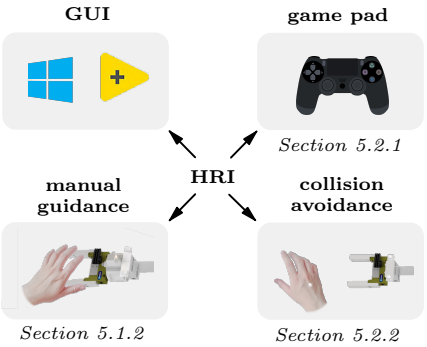


Figure 2.27: Applications of human-robot interaction (HRI) for Flexor (with logos from Microsoft Windows and National Instruments).

the control structure of Figure 2.25. Besides this standard way of interaction, the concepts discussed in Section 5.2.1 allow to accurately control the end-effector pose of Flexor in real-time via a game pad. With the settings of Figure 2.28 a human user could perform, e.g., a pick and place task. Since in such a case the user will focus on the end-effector, force feedback via the XInput API informs the user when an actuator gets close to its limit stop.
In Section 5.1.2, manual guidance is investigated which can be used to collaborate with the robot or to manually teach a desired end-effector trajectory. Finally, in Section 5.2.2 marker-based camera tracking is used to prevent collisions between human hands and Flexor.

Figure 2.28: Utilized game pad with two vibration motors and XInput support.

2.2.5.4 Safety Concepts

As Flexor is built to investigate amongst others HRI no safety fences are used. Since additionally new controllers are tested, it is likely that unexpected behavior occurs. Therefore, standard manual emergency stops and limitations within the motion control are not sufficient. In order to realize a safer operation, comprehensive signal monitoring is conducted, see Figure 2.25. Here, measurements and desired signals are checked for a limit crossing, if signals become imaginary or not a number and if the selected control combination is feasible. Monitoring of the link deflections, e.g., via strain measurements, is extremely important in flexible link robotics since a slow actuator motion can still excite significant oscillations. Moreover, checking desired values enables to immediately detect if, e.g., a dynamic inversion of a flexible multibody model becomes unstable.

Visual feedback by RGB LEDs informs the user via traffic light colors if, e.g., a control scenario is currently running, see Figure 2.29. An additional blinking effect attracts the attention of humans nearby to reduce the risk that somebody accidentally walks into the robot's workspace.

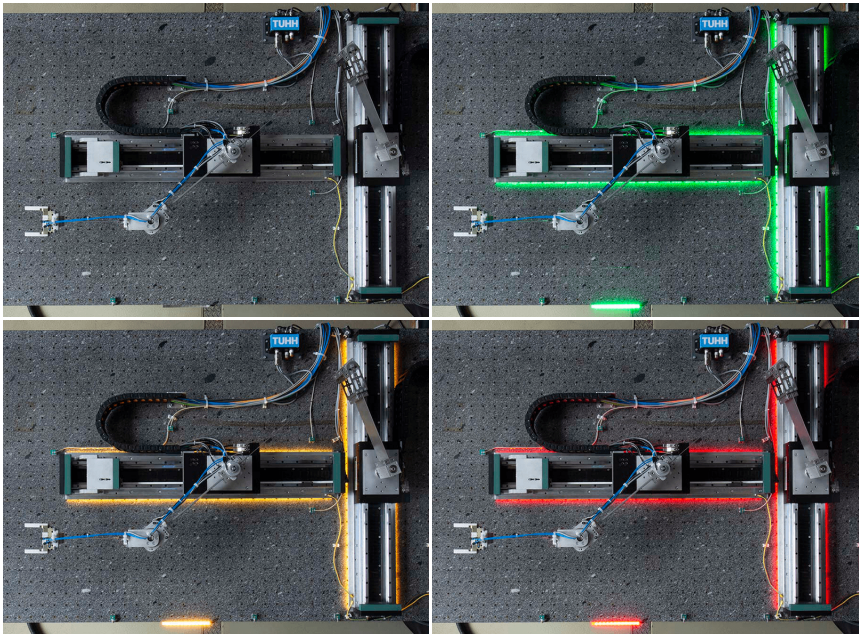


Figure 2.29: Visual feedback via traffic light colors.

Chapter 3

Modeling, Inversion and Solution

Chapter Contents

Since the focus of this research is model-based control of flexible link parallel robots, an accurate but computationally efficient model is required. The electrical time constant of actuators is usually much smaller than the mechanical time constant, see Spong et al. [61, p. 235]. Therefore, only the mechanical model is considered in this research. The main steps utilized in this chapter to obtain such a model as well as the inversion and the solution are outlined in Figure 3.1. Initially, the linear FEM is used to describe the flexible links. This enables a straightforward representation of arbitrary geometries and with a large number of finite elements a high model accuracy is provided. A subsequent modal model order reduction (MOR) ensures computational efficiency by reducing the number of elastic degrees of freedom and retaining only the significant eigenmodes. After a conversion to the standard input data (SID) format [62] the flexible bodies are combined with the rigid bodies within the framework of the floating frame of reference (FFOR) [63]. Here, the elastic deformations are described with respect to such reference frames which ensures that using linear finite element models is adequate. The assembly of the different bodies results in a flexible multibody system (FMBS) where kinematic loops are virtually cut open. In this research, this leads to a system in chain/serial structure. The kinematic loops are closed again via geometric constraints, which gives a system of DAEs that represents the forward dynamics in loop/parallel structure. Afterwards, the model is inverted by additional servo constraints, which incorporate the desired trajectory of the tracking output on position level. The DAEs are then transformed to ODEs by projections to simplify the numerical solution of the internal dynamics, i.e., to further ensure computational efficiency.

The applicable solution method depends on the stability of the internal dynamics. If it is stable classical inversion can be used to obtain a bounded solution online. The term *online* denotes here a case where the complete desired trajectory is not known in advance but it is obtained and processed successively. For sufficient computational power, an online case might even be computed in *real-time*. In contrast, when using the exact end-effector of flexible link robots as output the internal dynamics is usually unstable. Then, trajectory optimization or stable inversion can be used to obtain a bounded solution offline. The term *offline* analogously denotes a case where the complete desired trajectory is known in advance.

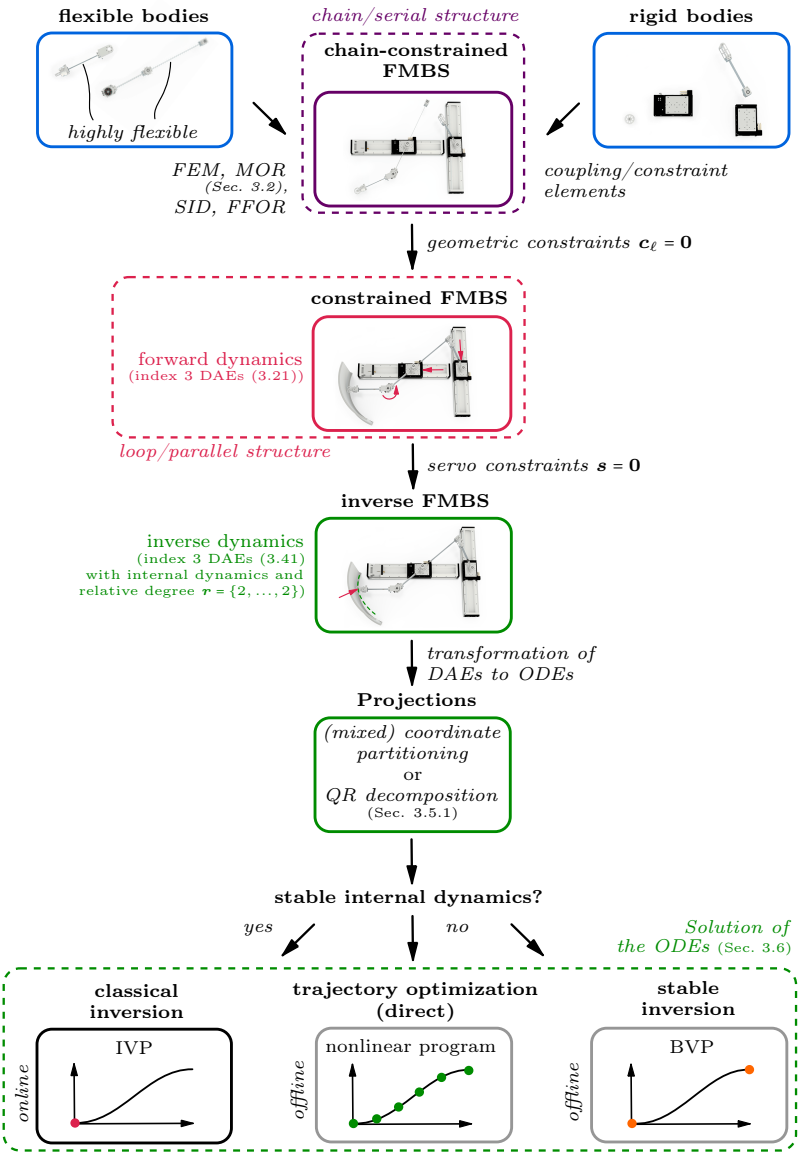


Figure 3.1: Overview of the modeling process with inversion and solution.

3.1 Classification

A classification of selected underactuated multibody system types is now discussed in order to clarify important system properties of flexible link robots. An *underactuated system* can be understood as a system where not all degrees of freedom can be controlled independently. This occurs when a system has less control inputs than degrees of freedom. Underactuation is also possible for the same number of inputs and degrees of freedom when a redundant actuation occurs, which will however not be considered in this research.
Using the basic robot model of Figure 3.2 as illustrative example, the close connection between selected underactuated multibody system types is now demonstrated but also fundamentally different system properties are highlighted. In the first place, the example system represents a robot with a passive joint. This joint is supported by a linear spring and a linear damper being characterized via the constants $c > 0$ and $d \geq 0$, respectively. In addition, such passive joints are a simple approach to model flexible links being denoted as lumped parameter method. As a result, the system of Figure 3.2 also represents a rudimentary model of a flexible link robot with one elastic degree of freedom. Furthermore, when setting $\ell_1 = 0$ one obtains a basic flexible joint robot model which can be

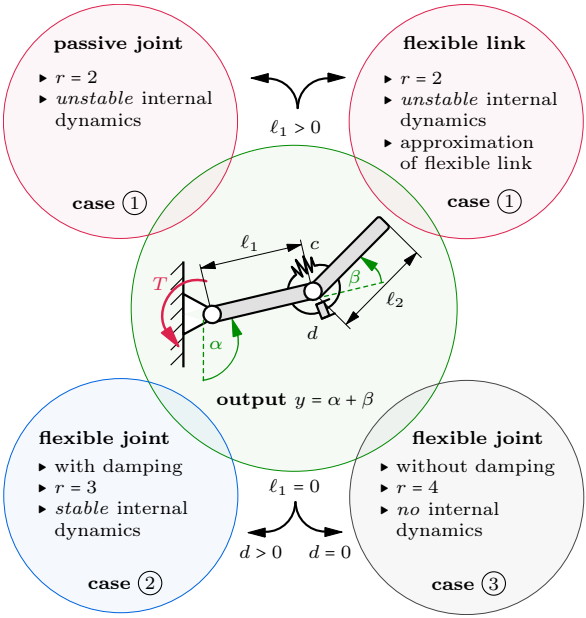


Figure 3.2: Relation between selected underactuated multibody systems.

damped or undamped. The different system properties outlined in Figure 3.2 are discussed in the following.

The equations of motion of the example system from Figure 3.2 read

$$\underbrace{\begin{bmatrix} I_1 + I_2 + m_1\ell_1^2/4 + m_2(\ell_1^2 + \ell_2^2/4) + m_2\ell_1\ell_2\cos\beta & \text{symmetrical} \\ I_2 + m_2\ell_2^2/4 + m_2\ell_1\ell_2\cos(\beta)/2 & I_2 + m_2\ell_2^2/4 \end{bmatrix}}_{\boldsymbol{M}} \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix}$$
$$+ \begin{bmatrix} -m_2\ell_1\ell_2\dot{\beta}(\dot{\alpha} + \dot{\beta}/2)\sin\beta \\ m_2\ell_1\ell_2\dot{\alpha}^2\sin(\beta)/2 \end{bmatrix} = \begin{bmatrix} 0 \\ -c\beta - d\dot{\beta} \end{bmatrix} + \underbrace{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{\boldsymbol{B}} T. \tag{3.1}$$

Here, m_1 and m_2 are the masses and I_1 and I_2 are the moments of inertia with respect to the center of gravity of the two rigid homogeneous links. The absolute angle of the second link is used as output

$$y = \alpha + \beta. \tag{3.2}$$

The inverse model is obtained by constraining the output y to a desired trajectory $y_\mathrm{d}(t)$. The dynamics of the inverse model corresponds to the so-called *internal dynamics* which is a common term within feedback linearization, compare Isidori [64] and Slotine and Li [65]. The trajectory y_d is then set equal to Equation (3.2), which gives $\alpha = y_\mathrm{d} - \beta$. Incorporating this relation and the corresponding time derivatives into Equation (3.1) yields the required torque input

$$\begin{aligned} T = {} & \left(I_1 + I_2 + m_1\ell_1^2/4 + m_2(\ell_1^2 + \ell_2^2/4) + m_2\ell_1\ell_2\cos\beta\right)\ddot{y}_\mathrm{d} \\ & - \left(I_1 + m_1\ell_1^2/4 + m_2\ell_1^2 + m_2\ell_1\ell_2\cos(\beta)/2\right)\ddot{\beta} - m_2\ell_1\ell_2\dot{\beta}(\dot{y}_\mathrm{d} - \dot{\beta}/2)\sin\beta \end{aligned} \tag{3.3}$$

from the first row of Equation (3.1). The unknown β and its time derivatives are obtained from the remaining dynamics being expressed by the second row of Equation (3.1) as

$$\begin{aligned} & -(m_2\ell_1\ell_2\cos(\beta)/2)\,\ddot{\beta} + d\dot{\beta} + c\beta \\ = {} & -\left(I_2 + m_2\ell_2^2/4 + m_2\ell_1\ell_2\cos(\beta)/2\right)\ddot{y}_\mathrm{d} - m_2\ell_1\ell_2(\dot{y}_\mathrm{d} - \dot{\beta})^2\sin(\beta)/2. \end{aligned} \tag{3.4}$$

This is the mentioned internal dynamics, which is unobservable from the output and is driven by the desired output. Since investigating the stability of such a nonlinear time-variant system can be complex often stability analysis is done just for the so-called *zero dynamics*. The zero dynamics corresponds to the internal dynamics when keeping the output constantly at zero. This can be relaxed by demanding the output only to be constant, which is with small abuse of notation also termed zero dynamics in this thesis. By setting the time derivatives of the desired output y_d to zero within Equation (3.4) one obtains the zero dynamics

$$-(m_2\ell_1\ell_2\cos(\beta)/2)\,\ddot{\beta} + d\dot{\beta} + c\beta = -m_2\ell_1\ell_2\dot{\beta}^2\sin(\beta)/2. \tag{3.5}$$

To further simplify the stability analysis it is done locally via a linearization of Equation (3.5) around the equilibrium at $\beta = 0$, giving

$$-\left(m_2 \ell_1 \ell_2 / 2\right) \ddot{\beta} + d\dot{\beta} + c\beta = 0. \tag{3.6}$$

The internal dynamics and the (linearized) zero dynamics are then used to deduce the system properties summarized in Figure 3.2. The corresponding underactuated multibody systems are classified via the following cases:

- **Case 1**: For $\ell_1 > 0$, the linearized zero dynamics (3.6) is unstable and consequently also the internal dynamics. The internal dynamics (3.4) is of order $n_{\mathrm{id}} = 2$. Since the state space of the system is of dimension $n = 4$ this results in a relative degree of $r = n - n_{\mathrm{id}} = 2$. The *relative degree* r is the number of time derivatives of the output until the input shows up explicitly for the first time. It is thus usually obtained by time differentiation of the output y.
  Solving the internal dynamics (3.4) in state-space representation yields β and $\dot{\beta}$. Inserting them again into Equation (3.4) gives an algebraic expression for $\ddot{\beta}$. Thus, all unknowns to calculate the input T (3.3) are directly obtained from the internal dynamics which relies on the desired output up to its second time derivative $\ddot{y}_{\mathrm{d}}$.

- **Case 2**: Setting $\ell_1 = 0$ results in a basic flexible joint robot. The zero dynamics (3.5) then reduces to $d\dot{\beta} + c\beta = 0$, being a stable dynamics for nonzero damping. A linear or nonlinear system with a stable zero dynamics is also called *minimum phase*. If the zero dynamics is unstable the system is said to be *non-minimum phase*. The internal dynamics (3.4) reduces to

$$d\dot{\beta} + c\beta = -\left(I_2 + m_2 \ell_2^2 / 4\right) \ddot{y}_{\mathrm{d}}, \tag{3.7}$$

  which also corresponds to a stable system but with a time-varying excitation due to $\ddot{y}_{\mathrm{d}}$. Equation (3.7) is only an ODE of order $n_{\mathrm{id}} = 1$ leading to a relative degree of $r = n - n_{\mathrm{id}} = 3$. Its solution gives β which can again be inserted into the internal dynamics (3.7) to algebraically compute $\dot{\beta}$. Differentiating Equation (3.7) with respect to time leads to

$$d\ddot{\beta} + c\dot{\beta} = -\left(I_2 + m_2 \ell_2^2 / 4\right) \dddot{y}_{\mathrm{d}}, \tag{3.8}$$

  where the known $\dot{\beta}$ can be plugged in to compute $\ddot{\beta}$. Equations (3.7) and (3.8) may also be used to replace $\dot{\beta}$ and $\ddot{\beta}$ within the input equation (3.3), which gives

$$\begin{aligned} T = {} & \left(I_1 + I_2 + m_2 \ell_2^2 / 4\right) \ddot{y}_{\mathrm{d}} \\ & + \frac{I_1}{d} \left(\left(I_2 + m_2 \ell_2^2 / 4\right) \dddot{y}_{\mathrm{d}} - \frac{c}{d} \left(\left(I_2 + m_2 \ell_2^2 / 4\right) \ddot{y}_{\mathrm{d}} + c\beta \right) \right). \end{aligned} \tag{3.9}$$

  This demonstrates that the input T is related to the desired output up to its third time derivative $\dddot{y}_{\mathrm{d}}$.

- **Case 3**: If additionally to $\ell_1 = 0$ the damping is also set to zero, i.e., $d = 0$, there is no zero dynamics left but only the algebraic expression $c\beta = 0$. This can be confirmed via the internal dynamics (3.4) which reduces to the algebraic equation

$$c\beta = -\left(I_2 + m_2\ell_2^2/4\right)\ddot{y}_\mathrm{d}. \tag{3.10}$$

The relative degree is consequently $r = n = 4$. Further differentiations of Equation (3.10) with respect to time yield

$$c\dot{\beta} = -\left(I_2 + m_2\ell_2^2/4\right)\dddot{y}_\mathrm{d} \tag{3.11}$$

and

$$c\ddot{\beta} = -\left(I_2 + m_2\ell_2^2/4\right)\ddddot{y}_\mathrm{d}. \tag{3.12}$$

From these algebraic relations the unknowns β, $\dot{\beta}$ and $\ddot{\beta}$ can be computed. Also, with Equation (3.12) the input (3.3) follows as

$$T = \left(I_1 + I_2 + m_2\ell_2^2/4\right)\ddot{y}_\mathrm{d} + \frac{I_1}{c}\left(I_2 + m_2\ell_2^2/4\right)\ddddot{y}_\mathrm{d}, \tag{3.13}$$

which uses the fourth time derivative of y_d. It is concluded that all unknowns depend purely algebraically on the desired output y_d and its time derivatives. This property is called *differential flatness*, see Fliess et al. [66].

Alternatively, for case 1 the relative degree of $r = 2$ may be confirmed with the so-called *decoupling matrix* occurring after two time differentiations of the output y, which yields

$$\ddot{y} = \ddot{\alpha} + \ddot{\beta} = \underbrace{\begin{bmatrix}1 & 1\end{bmatrix}}_{\boldsymbol{H}}\begin{bmatrix}\ddot{\alpha}\\ \ddot{\beta}\end{bmatrix}. \tag{3.14}$$

Equation (3.1) is solved for the accelerations $\ddot{\alpha}$ and $\ddot{\beta}$, which are then inserted into Equation (3.14). This gives the decoupling matrix $\bar{\boldsymbol{\Delta}}$, which here degenerates to a decoupling scalar

$$\bar{\boldsymbol{\Delta}} = \boldsymbol{H}\boldsymbol{M}^{-1}\boldsymbol{B} = -\frac{m_2\ell_1\ell_2\cos(\beta)}{2\det(\boldsymbol{M})} \tag{3.15}$$

as a factor in front of the input T. This scalar needs to be nonzero to invert it for the computation of T. With the mass matrix $\boldsymbol{M}$ being positive definite the decoupling scalar is nonzero for $\ell_1 > 0$ and $\cos(\beta) \neq 0$, confirming $r = 2$ for case 1. Due to $\ell_1 = 0$ for case 2 and 3 the decoupling scalar equals zero, i.e., it is non-invertible, which confirms that $r \neq 2$.

For the later considered multiple-input multiple-output systems the concept of relative degree is extended to the so-called *vector relative degree*. It is the number of time derivatives of each output until at least one input shows up explicitly for the first time. Additionally, the corresponding decoupling matrix needs to be invertible, assuming an identical number of inputs and outputs. See, e.g., [67, p. 252] or [68, p. 93] for a formal definition.

With the example of Figure 3.2 and based on Seifried [68, p. 124], selected types of underactuated multibody systems are classified by means of the internal dynamics, see Figure 3.3. It should be noted that the internal dynamics depends on the specific system under consideration. Thus, the classification of Figure 3.3 shall only give an overview of typical system properties. A much more detailed analysis of the inverse dynamics of flexible joint robots has been carried out by De Luca and Book [9]. Moreover, Seifried [69] provides an in-depth discussion of the inverse model of robots with passive joints. In the present thesis, the focus lies on flexible link robots with Flexor being the main application example. Its internal dynamics is unstable when using the exact end-effector pose as output, which is a typical property of flexible link robots. This is usually undesired since then the solution of the internal dynamics has to be obtained offline and is much more complex to compute. Therefore, in Section 5.2.1 three approaches are proposed to gain a stable internal dynamics. These allow online model inversion, which can even be performed in real-time for Flexor. As a result, flexible link robots,

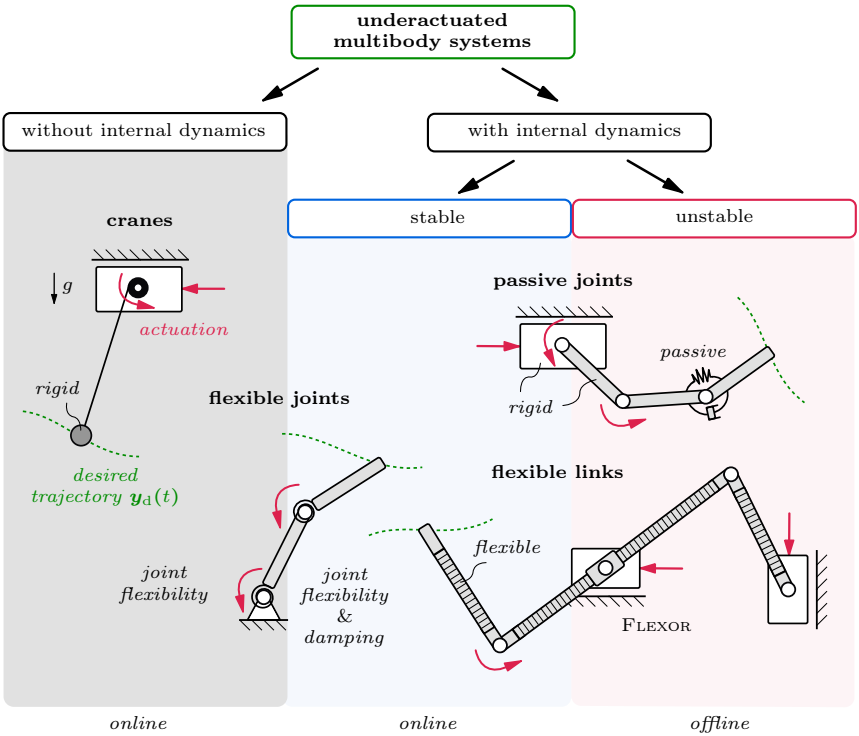


Figure 3.3: Classification of selected types of underactuated multibody systems based on Seifried [68].

such as FLEXOR, are classified within two categories of Figure 3.3.
The need for solving an unstable internal dynamics offline is demonstrated by means of the example system from Figure 3.2 with $\ell_1 > 0$. For a smooth change of the desired output angle $y_\mathrm{d}(t)$ the behavior illustrated schematically in Figure 3.4 occurs. It can be seen that even before point 2 on the time axis, at which y_d starts to change the output y, the internal dynamics needs to be excited to reach the required pre-deflection to fulfill the y_d-trajectory. For the inverse system y_d is actually the input and the resulting output is the actuation T. Thus, in the so-called *pre-actuation phase* the output, i.e., the actuation needs to occur before the input, i.e., y_d changes. This is an *anticausal* characteristics of the inverse system due to the unstable part. Since the y_d-trajectory needs to be known in advance this inverse system is inherently not usable when y_d shall be applied online. At point 4 the change of the y_d-trajectory ends and the *post-actuation phase* starts. Here, the input y_d is constant whereas the output T and the internal dynamics slowly come to rest, which is a *causal* behavior. Such a combination of a causal and an anticausal characteristics is called *noncausal* [70]. A noncausal behavior often occurs for flexible link robots and also holds for FLEXOR when using the exact end-effector pose as tracking output.

In this section, selected types of underactuated multibody systems and their relation as well as their different properties are discussed. A variety of technical terms is introduced to provide a basis for the following investigations. For more detailed background information it is referred to Isidori [64], Sastry [71], Seifried [68], Slotine and Li [65] and De Luca and Book [9].

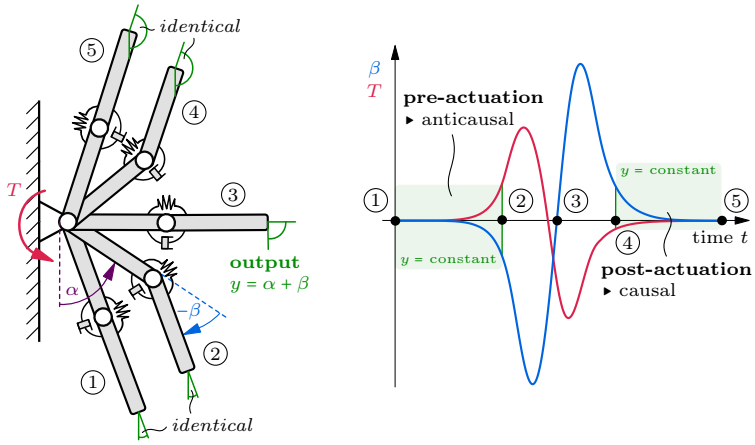


Figure 3.4: Schematic motion of a robot with a passive joint with pre-actuation and post-actuation phases for a desired smooth change of the tip output angle.

3.2 Finite Element Model

Modeling flexible bodies as continua results in partial differential equations. Since such partial differential equations can often only be solved exactly for basic problems, the continua shall be approximated by finite dimensional models. The most popular approaches to obtain finite dimensional models for flexible links are the [12]:

- Lumped parameter method
- Assumed modes method
- Finite element method (FEM)

The lumped parameter method is simple but tends to be not very accurate [5, 12]. Also, according to Junkins and Kim [72] for complex geometries it is very difficult to establish the global shape functions for the flexible bodies via the assumed modes method. Therefore, the FEM is applied in this research which discretizes flexible bodies into small elements which are described by simple local shape functions. Their assembly gives the global shape functions of the complete flexible bodies even for very complex geometries and boundary conditions. Here, the Ritz method [73] is used to approximate the elastic displacement of the j-th flexible body as

$$\boldsymbol{u}_j(\boldsymbol{R}_{\mathrm{O}_j\mathrm{P}}, t) \approx \boldsymbol{\Phi}_j(\boldsymbol{R}_{\mathrm{O}_j\mathrm{P}})\boldsymbol{q}_{\mathrm{e},j}(t) \tag{3.16}$$

and the elastic rotation, if structural elements such as beams are considered, as

$$\boldsymbol{\chi}_j(\boldsymbol{R}_{\mathrm{O}_j\mathrm{P}}, t) \approx \boldsymbol{\Psi}_j(\boldsymbol{R}_{\mathrm{O}_j\mathrm{P}})\boldsymbol{q}_{\mathrm{e},j}(t). \tag{3.17}$$

The Ritz method allows to approximate the partial differential equations of a continuum by ODEs for the unknowns $\boldsymbol{q}_{\mathrm{e},j}$ [74]. The matrices $\boldsymbol{\Phi}_j \in \mathbb{R}^{3\times f_{\mathrm{e},j}}$ and $\boldsymbol{\Psi}_j \in \mathbb{R}^{3\times f_{\mathrm{e},j}}$ contain the global shape functions of the displacements and the rotations, respectively. They depend on the vector $\boldsymbol{R}_{\mathrm{O}_j\mathrm{P}}$, which denotes the relative position in reference configuration, compare Seifried [68, p. 33]. The elastic coordinates $\boldsymbol{q}_{\mathrm{e},j} \in \mathbb{R}^{f_{\mathrm{e},j}}$ correspond here to the nodal degrees of freedom of the finite element model of the j-th flexible body.
In this research, the linear FEM is applied to obtain the linear equations of motion of the flexible bodies. This means that a linear elastic behavior is assumed without material and geometric nonlinearities. By describing the elastic deformation with respect to so-called *floating frames of reference* [63] the linear FEM is in many cases a reasonable approximation.

The general theory of the FEM is treated in detail in the literature, see, e.g., Bathe [73]. Therefore, system specific details of Flexor are discussed instead. This application example shall provide a closer insight into the different components which can be used to model flexible links. The finite element models implemented within Ansys are illustrated in Figure 3.5, which rely on the parameters of Table 3.1. Three individual finite element models are used, i.e., each link represents

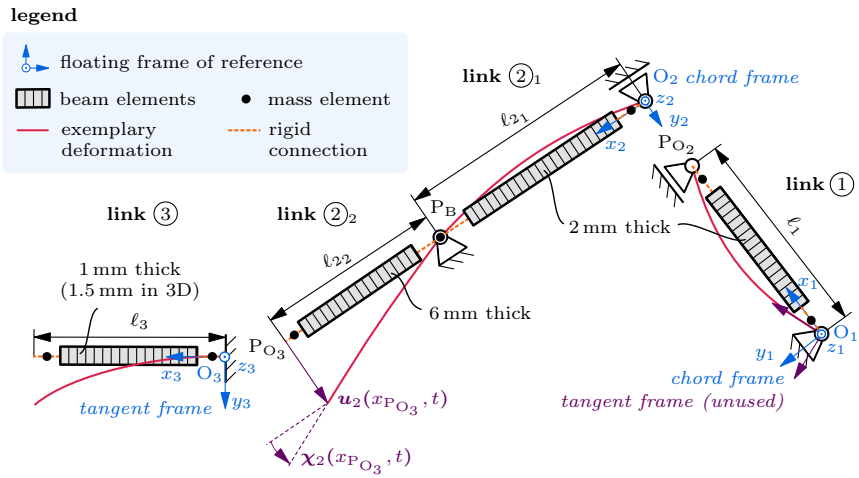


Figure 3.5: Individual finite element models of the three links with corresponding support.

Table 3.1: Link parameters.

description	parameter	value
length link 1	ℓ_1	42 cm
length link 2_1	ℓ_{2_1}	60 cm
length link 2_2	ℓ_{2_2}	40 cm
length link 3	ℓ_3	40 cm to 52 cm
beam elements per link part	n_ℓ	100 or 1000
density	ϱ_ℓ	7900 kg/m^3
Poisson's ratio	ν_ℓ	0.3
link height	h_ℓ	8 cm
Young's modulus links 1, 2_1, 3	E_{sp}	180 GPa
Young's modulus link 2_2	E_{st}	200 GPa

a separate flexible body which however also include rigid parts. The flexible link parts are described by BEAM188 elements being based on the Timoshenko beam theory. As noted in Section 2.2.3.1, the difference to the Euler-Bernoulli beam theory, which neglects shear deformation, is insignificant for the considered thin beams and consequently both theories are applicable. Due to their high stiffness, all other parts are modeled as rigid mass elements with corresponding inertia and rigid connections.

With deformations in 2D being the focus, the translational degrees of freedom in the corresponding z_j-direction and the rotational degrees of freedom around the corresponding x_j-axis and y_j-axis of all finite element model nodes are constrained.

An exception are the nodes of link 3 other than at O_3 where the rotation around the x_3-axis is unconstrained since a torsion mode with low frequency occurs here. To reflect the physical supports of the robot appropriate reference frames are selected which will serve later as floating frames of reference. The revolute joints in Figure 3.5 represent chord reference frames for link 1 and 2 and a fixed support forms the tangent reference frame utilized for link 3. The corresponding boundary conditions in 2D are listed in Table 3.2. Exemplary deformations on the beam axes that comply with these boundary conditions are included in Figure 3.5. Also, a tangent reference frame is added to link 1 for illustration purposes of the different behavior. This would result in a much larger deformation when measured with respect to the tangent frame, which is undesired when using linear models. Further details on reference frames are provided by Schwertassek et al. [75].

The normalized shape functions related to the first eigenmodes of the highly flexible links 2 and 3 are shown in Figure 3.6. These result from the finite element models for the standard 2D configuration of FLEXOR with free end-effector. One

Table 3.2: Boundary conditions of the finite element models in 2D.

local displacement	*link 1*		*link 2*		*link 3*
and rotation	O_1	P_{O_2}	O_2	P_B	O_3
displacement in x	0			0	0
displacement in y	0	0	0	0	0
rotation around z					0

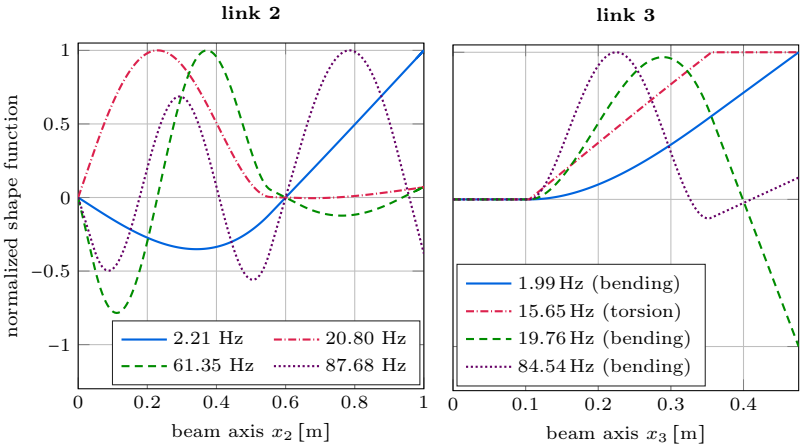


Figure 3.6: Normalized shape functions of the displacement of the first four bending eigenmodes of link 2 and of the displacement of the first three bending eigenmodes as well as of the rotation of the first torsional eigenmode of link 3.

can clearly identify the rigid parts at the revolute joint at 0.6 m of link 2 as well as at the first 10 cm of link 3. At around 35 cm there is no additional torsion at the end-effector since it is also a rigid part. Finally, it should be noted that the bending shape functions only provide deformations in the local y-directions due to the underlying linear modeling approach.

3.2.1 Model Validation

Since FLEXOR in its 2D configuration with a free end-effector is the main application focus of this research, it shall now serve as example for a compact validation of the effectiveness of the linear FEM to model flexible links. In this regard, the frequency and damping of the first four eigenmodes of the highly flexible links 2 and 3 are experimentally measured and compared to the corresponding models. To enable a meaningful comparison of experimental data to the finite element models, the boundary conditions need to be in line with Figure 3.5. This is realized by demounting link 3 and clamping the links within the utilized measurement setup according to Figure 3.7. Here, a laser Doppler vibrometer from

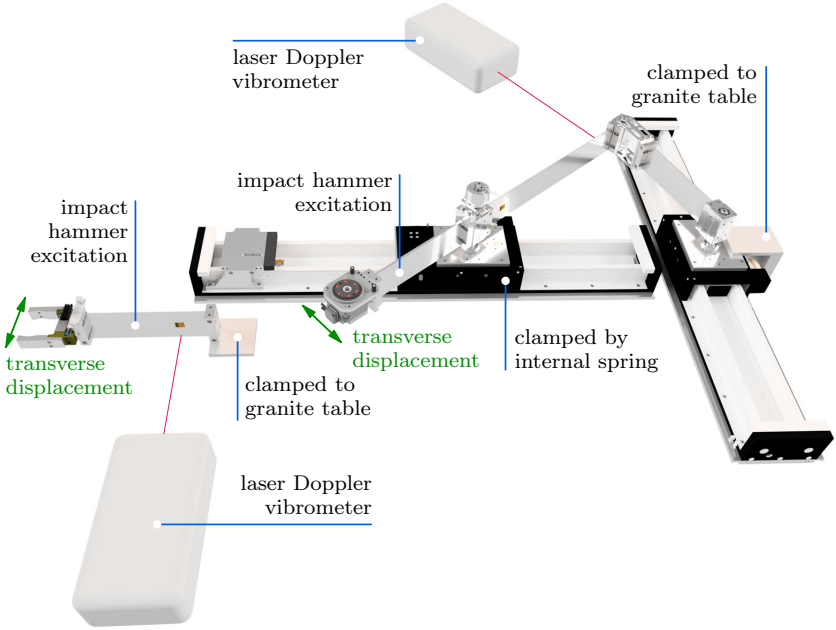


Figure 3.7: Sketch of the utilized measurement setup for the eigenfrequencies and damping of FLEXOR.

Polytec is used to conduct the measurements. The free eigenmodes of interest are then described by decoupled equations of the form

$$\ddot{q}_{\mathrm{m},i} + 2\delta_i \dot{q}_{\mathrm{m},i} + \omega_{0,i}^2 q_{\mathrm{m},i} = 0. \tag{3.18}$$

The symbol $q_{\mathrm{m},i}$ denotes the i-th modal coordinate, δ_i is the linear damping parameter and $\omega_{0,i} = \sqrt{\omega_i^2 + \delta_i^2}$ is the undamped angular eigenfrequency, with ω_i being the damped angular eigenfrequency. The corresponding frequencies in hertz are obtained as usual via $f_i = \omega_i/(2\pi)$ and $f_{0,i} = \omega_{0,i}/(2\pi)$. The general solution of Equation (3.18) for a here considered weakly damped case, i.e., $\delta_i^2 < \omega_{0,i}^2$, reads

$$q_{\mathrm{m},i}(t) = A_i \mathrm{e}^{-\delta_i t} \sin(\omega_i t - \zeta_i) \tag{3.19}$$

with the amplitude A_i and the phase shift ζ_i. Usually, the shape of the first bending eigenmode of a supported flexible link with a free end basically corresponds to a static mode, which can be excited by applying a transverse displacement at the free end, compare Figure 3.7. This allows to fit the resulting measured oscillations to the time domain model of Equation (3.19) to obtain the measured frequency and damping of the first bending eigenmode. Alternatively, an impact hammer can be used to excite multiple modes. The eigenfrequencies and the corresponding damping can then be identified within frequency domain, i.e., based on a fast Fourier transform (FFT). Within frequency domain the peaks represent the damped eigenfrequencies and the damping is identified with the so-called *half-power bandwidth* method [76, p. 83]. In this research, the time domain approach is used to measure the first bending eigenmode of both links, since the resulting oscillations closely match typical operation. All other eigenmodes are identified via the frequency domain approach.
The results of the first four eigenmodes are shown in Tables 3.3 and 3.4, where f and δ represent f_i and δ_i of the different modes. The finite element models are not tuned to match the eigenfrequencies of the experiments but they come essentially directly from computer-aided design (CAD) and finite element data. The damping values in Tables 3.3 and 3.4 clearly show the weakly damped system behavior, which leads to negligible differences between the damped frequencies f and the undamped frequencies f_0. Thus, both tables show that the FEM can deliver quite accurate undamped eigenfrequencies even for scenarios where a physical system might not be available for identification purposes. Especially

Table 3.3: Eigenmodes of link 2.

	measured		**modeled**	
type	f	2δ	f	2δ
bending	2.22 Hz	$0.25\,\mathrm{s}^{-1}$	2.21 Hz	$0.25\,\mathrm{s}^{-1}$
bending	20.68 Hz	$2.39\,\mathrm{s}^{-1}$	20.80 Hz	$1.27\,\mathrm{s}^{-1}$
bending	61.99 Hz	$8.72\,\mathrm{s}^{-1}$	61.35 Hz	$9.22\,\mathrm{s}^{-1}$
bending	83.45 Hz	$17.17\,\mathrm{s}^{-1}$	87.68 Hz	$18.59\,\mathrm{s}^{-1}$

Table 3.4: Eigenmodes of link 3.

	measured		modeled	
type	f	2δ	f	2δ
bending	2.04 Hz	$0.41\,\mathrm{s}^{-1}$	1.99 Hz	$0.41\,\mathrm{s}^{-1}$
torsion	13.71 Hz	$2.14\,\mathrm{s}^{-1}$	15.65 Hz	$0.84\,\mathrm{s}^{-1}$
bending	18.52 Hz	$2.14\,\mathrm{s}^{-1}$	19.76 Hz	$1.10\,\mathrm{s}^{-1}$
bending	78.27 Hz	$11.28\,\mathrm{s}^{-1}$	84.54 Hz	$13.21\,\mathrm{s}^{-1}$

the frequencies of the dominant first bending eigenmodes match quite well. It is important to note that the damping, however, usually needs to be identified on a real system. For a basic linear damping description the so-called *Rayleigh damping* model [77] is applied. The exemplarily modeled damping within Tables 3.3 and 3.4 comes from such a Rayleigh damping model written as

$$2\delta_i = r_\alpha + r_\beta \omega_{0,i}^2. \tag{3.20}$$

The real parameters r_α and r_β are identified for each link via a least-squares fit to the measured data, see Figure 3.8. Since the first bending eigenmode of each link is clearly dominant in typical operation, the Rayleigh damping is chosen to perfectly match the measured damping of this first mode. Figure 3.8 illustrates that such a simple model can often only roughly describe the real damping properties. Also, since all damping parameters are measurable in this example they could also be implemented directly within the finite element models instead of using Rayleigh damping. Nevertheless, the Rayleigh model is discussed here to point out a convenient method which most importantly allows to describe the damping of modes of which no measurement data is available.

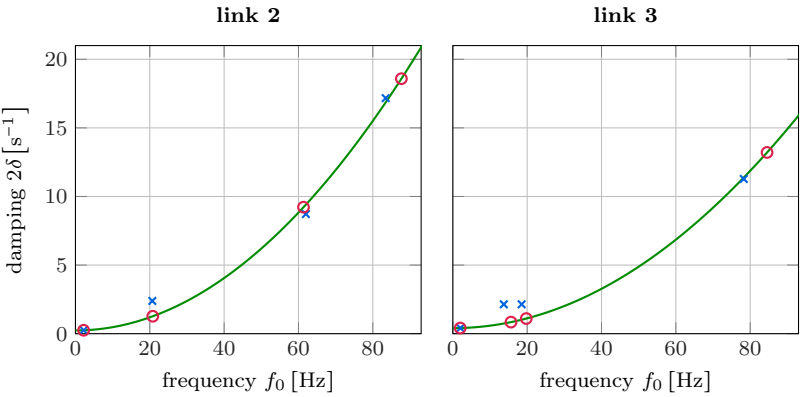


Figure 3.8: Rayleigh damping models (——) based on the measured (×) and applied to the modeled (○) first four eigenfrequencies of link 2 and link 3.

In conclusion, the utilized finite element modeling approach is compactly validated within experiments. Identification and modeling approaches of the modal damping are discussed, which can be used to obtain a basic description of the real damping. Finally, the experiments conducted throughout this thesis which use a flexible multibody model, being based on the linear FEM, further validate the implemented modeling approach.

3.2.2 Model Order Reduction

Using a large number of elastic, i.e., nodal degrees of freedom within finite element models of flexible links ensures a fine spatial discretization and high accuracy of the eigenfrequencies. This, however, leads to a large computational load. Therefore, a model order reduction shall be applied to reduce the number of elastic degrees of freedom while keeping the significant dynamics. The aspired computational efficiency is required to enable, e.g., model-based estimation and control based on model inversion in real-time.
Since the finite element models considered in this research are linear, linear model order reduction techniques can be directly applied. Here, exclusively the so-called *modal reduction* technique is applied, which is also denoted as *modal truncation*. The idea is that the eigenmodes with the highest frequencies are truncated, while the remaining reduced number of elastic degrees of freedom corresponds to modal coordinates of the remaining eigenmodes. Since this is a well-known technique, it is not further explained here. Instead, it is pointed to the literature [78, 79], where more information on modal reduction and on more complex model order reduction techniques can be found.

The following serves to show that the classical modal reduction technique can be highly effective in obtaining very small but accurate link models for flexible link robots. In this regard, the complete flexible multibody model of Flexor in its 2D configuration is considered within an exemplary scenario, which represents a typical operation. Here, all three actuators start at their zero position from which the sliders move by 15 cm and the rotary motor by 45° on the output side within 1 s in order to excite significant bending. Since end-effector trajectory tracking is the main focus of this research, the errors of the exact end-effector pose are used to evaluate how many elastic degrees of freedom $f_{\mathrm{e},j}$, i.e., eigenmodes per link are necessary for an accurate description. The maximum Euclidean norm of the end-effector position error $e_{\mathrm{p,max}}$ and the maximum absolute end-effector rotation error $e_{\mathrm{r,max}}$ between a considered model and the reference model are visualized in Figure 3.9. The reference model uses 30 elastic degrees of freedom, i.e., the first 30 eigenmodes, for each of the three links. To ensure an accurate discretization 1000 beam elements are used for each of the four link parts 1, 2_1, 2_2 and 3.
Figures 3.9a) and b) show the significant accuracy improvement from a completely rigid multibody model with zero elastic degrees of freedom to a flexible

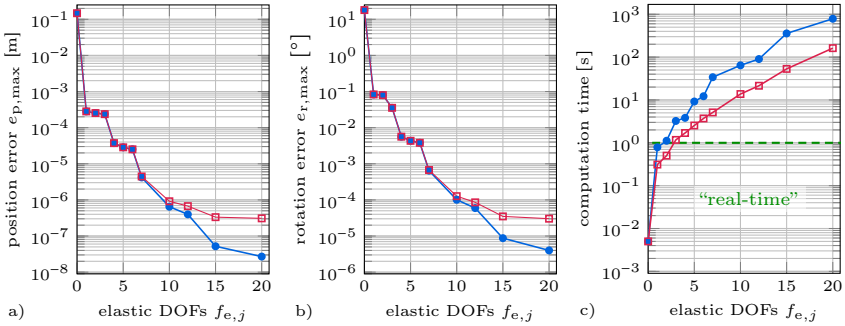


Figure 3.9: Comparison of the a), b) end-effector errors and c) computation times for an excitation of the flexible multibody model with all three links being flexible (—●—) as well as with only link 2 and link 3 being flexible (—⊟—) using a modal reduction.

multibody model with only one elastic degree of freedom for link 2 and link 3, respectively. Addition of further elastic degrees of freedom then only leads to insignificant changes compared to typical experimental errors within measurement and actuation. This means that almost only the first two bending eigenmodes are excited in usual scenarios. As a result, link 1 can be modeled as a rigid body since it exhibits negligible oscillations being related to its supported ends.
Within Figure 3.9c), it can be seen that the computation times rise quickly with an increasing number of elastic degrees of freedom. Besides the larger system size, this is also connected to the rising frequencies of the flexible cases which require smaller integration time steps. The case with only rigid links is obtained purely based on kinematic calculations and is therefore very efficient. All other flexible cases are computed using the later discussed formulation with kinematic inputs of Figure 3.21. Only for very small models the computation time is less than the considered simulation time span of 1 s, which is here sloppily denoted as "real-time". Nevertheless, it should be noted that these simulations have not been done on the real-time target but on an Intel Xeon CPU at 3.60 GHz with 4 cores. Also, the simulations including flexibilities are using Matlab's variable-step solver `ode45` which can be slower than real-time at certain time instances. Nevertheless, a rough impression of the computational cost of integrating the flexible multibody system can be obtained, which apparently demands a small number of elastic degrees of freedom for real-time applications.

In summary, the modal reduction can significantly improve the computation times for flexible link robots while still ensuring high accuracy. In regard of the simulations, all following investigations for Flexor use a rigid body model for link 1 and link 2 as well as link 3 are described by only one elastic degree of freedom, which corresponds to their first bending eigenmode. This drastic reduction to only two elastic degrees of freedom overall will eventually enable the

exact inversion of the flexible multibody model in real-time within experiments, see, e.g., Section 5.2.1.4. For such small models the number of beam elements can also be decreased. Using in the following 100 beam elements for link 2_1, for link 2_2 as well as for link 3 still guarantees a high accuracy of the retained eigenmodes.

3.3 Flexible Multibody Model

When one or more deformable, i.e., flexible bodies are added to a rigid multibody system one obtains a so-called *flexible multibody system*. In this research, it is assumed that only the links contain flexibility and that the resulting underactuation is exclusively due to this link flexibility. To obtain the equations of motion of a flexible multibody system the floating frame of reference approach is used which is covered, e.g., in the books of Shabana [63], Seifried [68] and Schwertassek and Wallrapp [74]. With the floating frame of reference approach the motion of a flexible body can be separated into a potentially large nonlinearly described reference frame motion and possibly small elastic deformations described with respect to this reference frame. In many applications it is then accurate enough to use linear models for the elastic deformation. As discussed in Section 3.2, these linear models come from the linear FEM in this research. This allows to use linear model order reduction techniques, such as modal truncation, to obtain computationally efficient models being necessary for control purposes.
Linear finite element models are also an acceptable approximation for the highly flexible links of the application examples investigated in this research. For controllers based on a flexible multibody model the deformations stay typically small to moderate. Then, the influence of nonlinearities, such as the foreshortening effect due to geometric nonlinearity, is not substantial and is therefore not considered in this research.
For the implementation of the reduced flexible bodies into the multibody system they are expressed within the SID format [62, 74]. Here, only terms up to the first order within the elastic coordinates are included via a linearization. This allows to obtain the integrals over the considered flexible body within a pre-processing step to ensure computational efficiency. The details of the floating frame of reference approach and of the integration of flexible bodies into a multibody system are well documented in the literature [63, 68, 74]. Therefore, a significantly shortened discussion of the involved modeling steps is carried out in the following.

3.3.1 Flexible Multibody Model in Chain Coordinates

To ensure computational efficiency it is desirable to describe a multibody system with a minimal set of coordinates. However, it is usually not possible to directly describe parallel robots, i.e., multibody systems with kinematic loops, with

minimal coordinates. Therefore, the idea is to dissolve existing kinematic loops by virtually cutting them open at selected supports. This leads to a system in chain or tree structure, which both are treated analogously. In this research, it is assumed that a chain structure occurs, i.e., serial parts are obtained which can be directly described with chain coordinates. The notion of *chain coordinates* is introduced here to denote a minimal set of coordinates needed to describe a system in chain structure. The cut kinematic loops are closed again via n_c algebraic geometric constraints $\boldsymbol{c} = \boldsymbol{0}$ and $c_{ef} = 0$ for an additional contact at the end-effector, which together with the system dynamics describe the forward model in its original loop, i.e., parallel structure.
These modeling steps are illustrated exemplarily for FLEXOR in Figure 3.10. The parallel robot is thus transformed to two serial subsystems. Each serial subsystem can then be modeled with standard methods for serial flexible link robots and are described with the chain coordinates $\boldsymbol{q} \in \mathbb{R}^f$. Afterwards, the subsystems and the environment are connected by geometric loop closing constraints $\boldsymbol{c} = \boldsymbol{0}$ and $c_{ef} = 0$, which are enforced by the Lagrange multipliers λ_x, λ_y and λ_{ef}.
The resulting general formulation of the forward dynamics of flexible link parallel robots with contact at the end-effector reads

$$\underbrace{\begin{bmatrix} \boldsymbol{M}_{ee} & \boldsymbol{M}_{ea} & \boldsymbol{M}_{eu} \\ \boldsymbol{M}_{ea}^T & \boldsymbol{M}_{aa} & \boldsymbol{M}_{au} \\ \boldsymbol{M}_{eu}^T & \boldsymbol{M}_{au}^T & \boldsymbol{M}_{uu} \end{bmatrix}}_{\boldsymbol{M}(\boldsymbol{q})} \underbrace{\begin{bmatrix} \ddot{\boldsymbol{q}}_e \\ \ddot{\boldsymbol{q}}_a \\ \ddot{\boldsymbol{q}}_u \end{bmatrix}}_{\ddot{\boldsymbol{q}}} = \underbrace{\begin{bmatrix} \boldsymbol{f}_e \\ \boldsymbol{f}_a \\ \boldsymbol{f}_u \end{bmatrix}}_{\boldsymbol{f}(\boldsymbol{q},\dot{\boldsymbol{q}})} + \underbrace{\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{B}_a \\ \boldsymbol{0} \end{bmatrix}}_{\boldsymbol{B}(\boldsymbol{q})} \boldsymbol{u} + \underbrace{\begin{bmatrix} \boldsymbol{C}_e^T \\ \boldsymbol{C}_a^T \\ \boldsymbol{C}_u^T \end{bmatrix}}_{\boldsymbol{C}^T(\boldsymbol{q})} \boldsymbol{\lambda} + \underbrace{\begin{bmatrix} \boldsymbol{c}_{ef,e}^T \\ \boldsymbol{c}_{ef,a}^T \\ \boldsymbol{c}_{ef,u}^T \end{bmatrix}}_{\boldsymbol{c}_{ef}^T(\boldsymbol{q})} \lambda_{ef}, \tag{3.21a}$$

$$\boldsymbol{c}_\ell(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{c}(\boldsymbol{q}) \\ c_{ef}(\boldsymbol{q}) \end{bmatrix} = \boldsymbol{0}, \tag{3.21b}$$

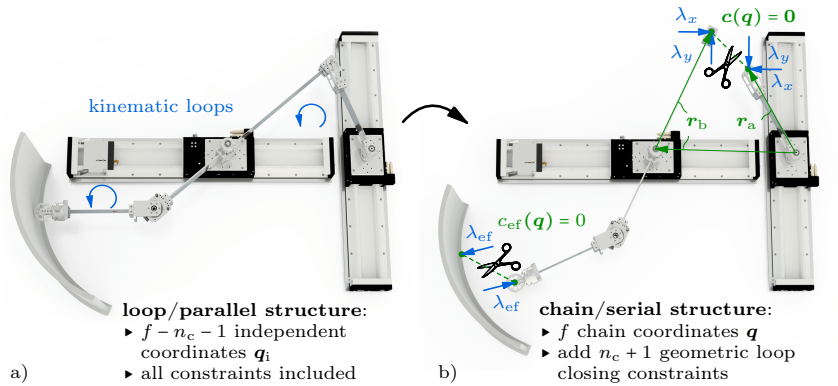


Figure 3.10: FLEXOR a) in loop/parallel structure and b) in chain/serial structure with reactions to enforce the geometric constraints.

which is a set of DAEs with differentiation index 3. According to Campbell [80], the differentiation index is a measure of how singular a DAE is and a larger index implies a higher complexity. An ODE has a differentiation index of 0. In Equation (3.21), $\boldsymbol{M} \in \mathbb{R}^{f \times f}$ is the symmetric and positive definite mass matrix. The input matrix $\boldsymbol{B} \in \mathbb{R}^{f \times f_\mathrm{a}}$ of the input-affine system is multiplied with the control inputs $\boldsymbol{u} \in \mathbb{R}^{f_\mathrm{a}}$. It is assumed that the coordinates are chosen such that $\boldsymbol{B}$ only has entries in the rows of the actuated accelerations $\ddot{\boldsymbol{q}}_\mathrm{a}$. This form of $\boldsymbol{B}$ is needed when using kinematic inputs, see Section 3.5.4. The vector $\boldsymbol{f} \in \mathbb{R}^{f}$ contains the contributions of the Coriolis, the centrifugal, the gyroscopic and of the elastic inner forces as well as of the applied forces which are not related to the control inputs $\boldsymbol{u}$. Moreover, $\boldsymbol{C} = \partial \boldsymbol{c} / \partial \boldsymbol{q} \in \mathbb{R}^{n_\mathrm{c} \times f}$ and $\boldsymbol{c}_\mathrm{ef} = \partial c_\mathrm{ef} / \partial \boldsymbol{q} \in \mathbb{R}^{f}$ are the Jacobians of the geometric constraints and $\boldsymbol{\lambda} \in \mathbb{R}^{n_\mathrm{c}}$ as well as λ_ef are the corresponding Lagrange multipliers which represent the reactions to enforce these constraints. For a case with a free end-effector the equations of motion (3.21) of the forward dynamics reduce to

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} = \boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{B}(\boldsymbol{q})\boldsymbol{u} + \boldsymbol{C}^\mathrm{T}(\boldsymbol{q})\boldsymbol{\lambda}, \tag{3.22a}$$

$$\boldsymbol{c}(\boldsymbol{q}) = \boldsymbol{0}. \tag{3.22b}$$

The notation and further details for Flexor are visualized in Figure 3.11. Here, the position $\boldsymbol{r}_\mathrm{ef}$ and rotation ϑ_ef describe the so-called *exact end-effector*. The term *exact* is used to differentiate the end-effector from approximations such as an equivalent *rigid end-effector* formulation. The utilized $f = 7$ chain coordinates $\boldsymbol{q}$ consist of the $f_\mathrm{e} = 2$ elastic coordinates

$$\boldsymbol{q}_\mathrm{e} = \begin{bmatrix} q_{\mathrm{e},2} & q_{\mathrm{e},3} \end{bmatrix}^\mathrm{T}, \tag{3.23}$$

which come from a modal reduction and correspond to the first bending eigenmode of link 2 and 3, respectively. The elastic coordinates and the corresponding curvatures κ_2 and κ_3 at the strain gauge measurement points of link 2 and 3 are defined to be positive for the exemplary bending deformation illustrated in Figure 3.11. The chain coordinates further contain the $f_\mathrm{a} = 3$ actuator, i.e., actuated coordinates

$$\boldsymbol{q}_\mathrm{a} = \begin{bmatrix} a & b & \gamma \end{bmatrix}^\mathrm{T}. \tag{3.24}$$

This vector is also called *actuator positions*, which for Flexor includes linear and angular positions. The chain coordinates are completed by $f_\mathrm{u} = 2$ additional unactuated coordinates

$$\boldsymbol{q}_\mathrm{u} = \begin{bmatrix} \alpha & \beta \end{bmatrix}^\mathrm{T}. \tag{3.25}$$

These represent angles of link 1 and 2. The angle β, which includes an elastic rotation, corresponds to the rotation of the revolute joint on slider B. This enables a direct measurement of β via the attached rotary encoder. In contrast, the angle β_R corresponds to β but for an equivalent rigid system, which is also the

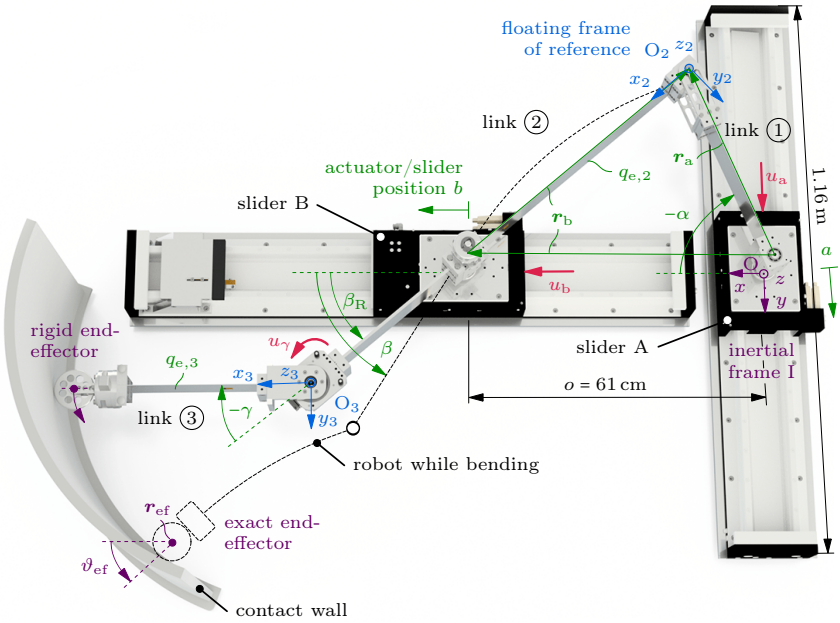


Figure 3.11: Planar Flexor with notation.

rotation of the floating frame of reference of link 2. The part of $\boldsymbol{B}$ related to the actuated coordinates contains the motor constants and reads

$$\boldsymbol{B}_a = \begin{bmatrix} K_T & 0 & 0 \\ 0 & K_T & 0 \\ 0 & 0 & K_R \end{bmatrix}, \tag{3.26}$$

with $K_T = 80\,\mathrm{N/A}$ for the linear direct drives and $K_R = 15\,\mathrm{Nm/A}$ on the output side of the gearing of the rotary motor. Consequently, $\boldsymbol{B}$ is constant for Flexor and the inputs

$$\boldsymbol{u} = \begin{bmatrix} u_a & u_b & u_\gamma \end{bmatrix}^T \tag{3.27}$$

correspond to motor currents.

For the geometric constraints $\boldsymbol{c} = \boldsymbol{0}$ the position of the point O_2, where the cut loop shall be closed, is described with respect to the two serial subsystems via the 2D position vectors $\boldsymbol{r}_a$ and $\boldsymbol{r}_b$. Thus, the $n_c = 2$ constraints can be written as the difference

$$\boldsymbol{c}(\boldsymbol{q}) = \boldsymbol{r}_{a|I}(\boldsymbol{q}) - \boldsymbol{r}_{b|I}(\boldsymbol{q}) = \boldsymbol{0}, \tag{3.28}$$

where the vectors are represented in the inertial frame I with respect to the origin O. This leads to the Lagrange multipliers

$$\boldsymbol{\lambda} = \begin{bmatrix} \lambda_x & \lambda_y \end{bmatrix}^{\mathrm{T}} \tag{3.29}$$

in x- and y-direction of the inertial frame.

3.3.2 Lagrange Multipliers

To compute the Lagrange multipliers $\boldsymbol{\lambda}$ and λ_{ef} within Equation (3.21), the loop closing constraint equation (3.21b) is differentiated twice with respect to time. This results in

$$\dot{\boldsymbol{c}}_\ell = \frac{\partial \boldsymbol{c}_\ell}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{c}_\ell}{\partial t} = \boldsymbol{C}_\ell \dot{\boldsymbol{q}} + \boldsymbol{c}'_\ell = \boldsymbol{0}, \tag{3.30a}$$

$$\ddot{\boldsymbol{c}}_\ell = \boldsymbol{C}_\ell \ddot{\boldsymbol{q}} + \dot{\boldsymbol{C}}_\ell \dot{\boldsymbol{q}} + \dot{\boldsymbol{c}}'_\ell = \boldsymbol{C}_\ell \ddot{\boldsymbol{q}} + \boldsymbol{c}''_\ell = \boldsymbol{0} \tag{3.30b}$$

with

$$\boldsymbol{C}_\ell = \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{c}_{\mathrm{ef}} \end{bmatrix}. \tag{3.31}$$

By solving Equation (3.21a) for $\ddot{\boldsymbol{q}}$, plugging the result into Equation (3.30b) and solving for the Lagrange multipliers one obtains

$$\boldsymbol{\lambda}_\ell = \begin{bmatrix} \boldsymbol{\lambda} \\ \lambda_{\mathrm{ef}} \end{bmatrix} = \left(\boldsymbol{C}_\ell \boldsymbol{M}^{-1} \boldsymbol{C}_\ell^{\mathrm{T}} \right)^{-1} \left(-\boldsymbol{c}''_\ell - \boldsymbol{C}_\ell \boldsymbol{M}^{-1} (\boldsymbol{f} + \boldsymbol{B}\boldsymbol{u}) \right). \tag{3.32}$$

The inversion is possible when the constraints $\boldsymbol{c}_\ell = \boldsymbol{0}$ are independent, i.e., when the rows of $\boldsymbol{C}_\ell$ are linearly independent [81, p. 464], which is typically the case and is assumed here. Then the differentiation index is 3 as mentioned before. To evaluate the algebraic equation (3.32) the input $\boldsymbol{u}$ needs to be given and the state $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ must be known, e.g., from a time integration of the equations of motion.

3.3.3 Contact at the End-Effector

Since within contact scenarios the reactions at the end-effector are of major interest, the previously introduced contact constraint $c_{\mathrm{ef}} = 0$ is now further highlighted and the modeling assumptions of the contact are justified. These assumptions describe the contact with the following properties:

- Without deformation in the contact zone
- Permanent
- Point contact
- Without friction
- Geometric equality constraint

Detailed effects in the contact zone such as local deformations are typically complex processes, especially when considering impact scenarios, see Tschigg [82] and Seifried [83]. Since such local deformations are in addition not of interest here they are not further considered. Also, impacts or bouncing effects are extremely difficult to account for within feedback control as their dynamics is very fast and highly nonlinear. Furthermore, they typically only occur when the robot starts or ends a contact scenario during which a permanent contact is aspired. Therefore, in this research the transition between a free and a contact case is not considered but a permanent contact is assumed. In experiments, predefined desired actuator motions are applied to realize the required transition.
Moreover, a simple point contact model is adequate to obtain the normal contact force being the value of interest. Additional friction effects in the contact zone are neglected since they are difficult to model and can often be significantly reduced by design. As pointed out in the book of Flores and Lankarani [84, p. 3], the models for the contact response can be generally classified into contact force based methods, also referred to as penalty or compliant methods, and methods using geometric constraints. Within the former, the appropriate selection, e.g., of the contact stiffness is often difficult. Also, contact force based models tend to require small integration time steps leading to large computation times. As a result, contact force based models are usually not appropriate for real-time applications and are in consequence not further considered.
Therefore, a geometric contact constraint $c_{\mathrm{ef}} = 0$ is used for the assumed permanent point contact. This is especially advantageous for parallel robots which are typically also described with geometric constraints such that an additional contact constraint does not change the problem structure. The resulting DAEs can then be transformed to ODEs analogously to the steps discussed later in Section 3.5, which highly simplifies the numerical solution. The implemented contact constraint

$$c_{\mathrm{ef}}(\boldsymbol{q}) = c_{\mathrm{ef}}(\boldsymbol{r}_{\mathrm{ef|I}}(\boldsymbol{q})) = 0 \tag{3.33}$$

ensures that the Cartesian position of the exact end-effector $\boldsymbol{r}_{\mathrm{ef|I}}$ represented in the inertial frame I with respect to its origin O is constrained to the environment, being here a contact wall. Thus, the constraint describes the shape of this contact wall.
It is worth noting that in this research not the actual contact point is constrained since it is not a fixed point on the robot. Instead, $\boldsymbol{r}_{\mathrm{ef|I}}$ corresponds to the center of a circular or spherical contact element, which has an offset to the wall of the contact element radius r_{ce} when being in contact. The utilized contact element shape ensures that the normal contact force always points to the element center. This allows that the element center, i.e., a fixed point on the robot, can be used for the contact constraint, which simplifies the modeling.

For simulative force control design the physical normal contact force due to the constraint $c_{\mathrm{ef}} = 0$ needs to be computed based on the corresponding Lagrange

multiplier λ_{ef}. This Lagrange multiplier may be obtained via Equation (3.32). It can be shown that λ_{ef} is related to the physical normal contact force $\boldsymbol{f}_{\mathrm{c|I}}$ at the exact end-effector within the inertial frame I via

$$\boldsymbol{f}_{\mathrm{c|I}} = \left(\frac{\partial c_{\mathrm{ef}}}{\partial \boldsymbol{r}_{\mathrm{ef|I}}} \right)^{\mathrm{T}} \lambda_{\mathrm{ef}}, \tag{3.34}$$

compare, e.g., Woernle [85]. For a normalized direction, the resulting Lagrange multiplier corresponds to the signed magnitude of the physical normal contact force F_{c}. This allows to directly use the Lagrange multiplier, e.g., within a force feedback controller. When assuming that the Euclidean norm $\| \partial c_{\mathrm{ef}} / \partial \boldsymbol{r}_{\mathrm{ef|I}} \|$ is constant, then a normalization may be implemented by using a contact constraint of the form

$$\frac{c_{\mathrm{ef}}}{\| \partial c_{\mathrm{ef}} / \partial \boldsymbol{r}_{\mathrm{ef|I}} \|} = 0. \tag{3.35}$$

Equation (3.34) is then reformulated as

$$\boldsymbol{f}_{\mathrm{c|I}} = \frac{1}{\| \partial c_{\mathrm{ef}} / \partial \boldsymbol{r}_{\mathrm{ef|I}} \|} \left(\frac{\partial c_{\mathrm{ef}}}{\partial \boldsymbol{r}_{\mathrm{ef|I}}} \right)^{\mathrm{T}} \bar{\lambda}_{\mathrm{ef}}. \tag{3.36}$$

The terms in front of $\bar{\lambda}_{\mathrm{ef}}$ represent obviously a unit vector. As a result, $\bar{\lambda}_{\mathrm{ef}}$ corresponds exactly to the physical normal contact force F_{c} without further post-processing. In this research, flat and circular contact wall shapes are considered, see Figure 2.9. For these shapes the discussed normalization of the direction can be applied since $\| \partial c_{\mathrm{ef}} / \partial \boldsymbol{r}_{\mathrm{ef|I}} \|$ is constant.

Example: circular contact wall

In the case of a contact wall describing a circle in the xy-plane, the constraint equation for the exact end-effector $\boldsymbol{r}_{\mathrm{ef|I}} = [x_{\mathrm{ef}},\, y_{\mathrm{ef}},\, z_{\mathrm{ef}}]^{\mathrm{T}}$ can be written as

$$c_{\mathrm{ef}}(\boldsymbol{r}_{\mathrm{ef|I}}(\boldsymbol{q})) = R^2 - (x_{\mathrm{ef}}(\boldsymbol{q}) - o_x)^2 - (y_{\mathrm{ef}}(\boldsymbol{q}) - o_y)^2 = 0. \tag{3.37}$$

Here, R is the radius of the circle and its origin is located at $x = o_x$ and $y = o_y$. The magnitude

$$\left\| \frac{\partial c_{\mathrm{ef}}}{\partial \boldsymbol{r}_{\mathrm{ef|I}}} \right\| = \left\| \begin{bmatrix} -2(x_{\mathrm{ef}}(\boldsymbol{q}) - o_x) & -2(y_{\mathrm{ef}}(\boldsymbol{q}) - o_y) & 0 \end{bmatrix} \right\| \overset{(3.37)}{=} 2R \tag{3.38}$$

is indeed constant. Dividing Equation (3.37) by $2R$ then ensures that the corresponding Lagrange multiplier $\bar{\lambda}_{\mathrm{ef}}$ equals the physical normal contact force F_{c}.

3.3.4 Note on Gearing

It is well-known that the load inertia from the output side of rotary gear drives is reflected to the motor by the factor $1/N_{\mathrm{g}}^2$ [86], where N_{g} is the gear ratio. This

also means that the motor inertia I_m around the motor rotation axis is reflected to the output side as $I_\mathrm{m} N_\mathrm{g}^2$. However, there is no reflected inertia when the whole gear drive is rotated by an external motion. This behavior can be modeled straightforwardly by adding a rigid motor body to the flexible multibody system, which has an inertia of I_m and rotates with the angle

$$\gamma_\mathrm{m} = N_\mathrm{g} \gamma_\mathrm{o}, \tag{3.39}$$

where γ_o is the output angle. With the description of the complete flexible multibody system using, e.g., D'Alembert's principle and with the implementation of the kinematics to obtain a system in chain coordinates, a term of the form $\boldsymbol{J}_{\mathrm{R,m}}^\mathrm{T} \boldsymbol{I}_\mathrm{m} \boldsymbol{J}_{\mathrm{R,m}}$ occurs within the mass matrix $\boldsymbol{M}$ [68]. Here, $\boldsymbol{J}_{\mathrm{R,m}}$ is the Jacobian matrix of the motor rotation which includes the gear ratio N_g when assuming γ_o belongs to the vector of chain coordinates. Also, $\boldsymbol{I}_\mathrm{m}$ is the motor inertia tensor which includes I_m. The evaluation of the mass matrix will eventually result in the reflected inertia $I_\mathrm{m} N_\mathrm{g}^2$ at the output side for the diagonal entry of $\boldsymbol{M}$ related to $\ddot{\gamma}_\mathrm{o}$.
Including this reflected inertia is crucial for the LQR in Section 6.2 in order to obtain reasonable desired actuator accelerations.

3.4 Inverse Flexible Multibody Model

The flexible multibody model of Section 3.3 allows to design model-based controllers such as a feedforward control, which can be used for end-effector trajectory tracking. Using for instance the quasi-static deformation compensation [87] for this purpose does not lead to exact trajectory tracking by design as it relies on approximations. Since in this research theoretically exact trajectory tracking for the end-effector is desired, exact model inversion of the full dynamic flexible multibody model is considered. This model inversion is discussed in the following and is the basis for the later implemented feedforward controllers, see Chapter 5.

The model inversion is performed via so-called *servo constraints* [88]. These servo constraints are additional algebraic equations which restrict the system's tracking output $\boldsymbol{y}_\mathrm{o} \in \mathbb{R}^{f_\mathrm{o}}$ to a desired trajectory $\boldsymbol{y}_\mathrm{d}(t) \in \mathbb{R}^{f_\mathrm{o}}$. Since for the inverse model of a contact scenario a desired trajectory shall be tracked at the end-effector, the contact constraint $c_\mathrm{ef} = 0$ in Equation (3.21) needs to be removed and replaced by servo constraints. These servo constraints have to replicate the wall shape, i.e., the shape of the environment, in order to track a trajectory on the wall. Furthermore, the contact reaction force F_c shall be tracked, which is normal to the wall. This is the only considered contact force since friction in the contact zone is neglected in this research. Nevertheless, by removing $c_\mathrm{ef} = 0$ also the corresponding reaction force F_c due to the Lagrange multiplier λ_ef has been removed. For the inverse model this force needs to be reintroduced but it is now

constrained to a desired force trajectory according to

$$F_\mathrm{c} = F_\mathrm{c,d}(t). \tag{3.40}$$

As opposed to geometric and servo constraints, the simple force constraint of Equation (3.40) can be directly incorporated into the system dynamics. Here, it is included within the term $\boldsymbol{f}$ which is therefore now explicitly time-dependent. Thus, F_c is a reaction force in the forward model but it corresponds to an applied force in the inverse model. The general formulation of the inverse dynamics is then obtained by appending n_s servo constraints $\boldsymbol{s} = \boldsymbol{0}$ to the formulation of Equation (3.22). This gives

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} = \boldsymbol{f}(t, \boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{B}(\boldsymbol{q})\boldsymbol{u} + \boldsymbol{C}^\mathrm{T}(\boldsymbol{q})\boldsymbol{\lambda}, \tag{3.41a}$$

$$\boldsymbol{c}(\boldsymbol{q}) = \boldsymbol{0}, \tag{3.41b}$$

$$\boldsymbol{s}(t, \boldsymbol{q}) = \boldsymbol{y}_\mathrm{o}(\boldsymbol{q}) - \boldsymbol{y}_\mathrm{d}(t) = \boldsymbol{0}. \tag{3.41c}$$

This set of DAEs holds for cases with and without contact at the end-effector. The only difference is that for a free end-effector the term $\boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is not explicitly time-dependent since the incorporated desired normal contact force is constant zero, i.e., $F_\mathrm{c,d} = 0$. The servo constraints represent rheonomic holonomic constraints and are here given on position level. Whereas the Lagrange multipliers $\boldsymbol{\lambda}$ enforce $\boldsymbol{c} = \boldsymbol{0}$, the control inputs $\boldsymbol{u} \in \mathbb{R}^{f_\mathrm{a}}$ shall enforce $\boldsymbol{s} = \boldsymbol{0}$, assuming $f_\mathrm{a} = n_\mathrm{s}$. Thus, the control inputs represent applied forces in the forward model but can be regarded as reactions in the inverse model. The analogy between geometric constraints and servo constraints is visualized in Figure 3.12 for Flexor. In this example, e.g., the 2D pose of the exact end-effector could be the output of interest, leading to $f_\mathrm{o} = 3$ output components and accordingly $n_\mathrm{s} = 3$ servo constraints. It can be

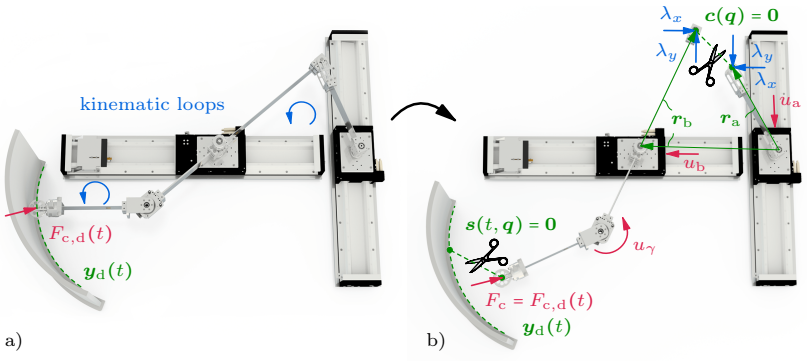


Figure 3.12: Flexor a) in loop/parallel structure with desired trajectories and b) in chain/serial structure with force constraint and reactions to enforce the geometric as well as the servo constraints.

seen that the control inputs and the servo constraints are non-collocated which is usually the case.

Overall, the concept of servo constraints is especially appealing for model inversion of parallel robots since they are usually already in DAE form. For the considered system type where geometric constraints lead to DAEs with differentiation index 3, the inversion via servo constraints does not even change the index. The solution of Equation (3.41) is a major focus of this research to enable end-effector trajectory tracking. In this regard, it represents the considered problem class being input-affine flexible multibody systems with holonomic geometric and servo constraints with a vector relative degree of $\boldsymbol{r} = \{2, ..., 2\}$. For details on the relation between the differentiation index and the relative degree it is referred to Campbell [80].

3.4.1 Lagrange Multipliers and Inputs

Based on the inverse model of Equation (3.41) the Lagrange multipliers $\boldsymbol{\lambda}$ and the control inputs $\boldsymbol{u}$ shall now be determined. These have been required, e.g., in simulations for FLEXOR to dimension the actuators and they are also needed for the input-output feedback linearization presented in Section 5.2.4.
Initially, the constraint equations (3.41b) and (3.41c) are differentiated twice with respect to time which gives

$$\dot{\boldsymbol{c}} = \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{c}}{\partial t} = \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{c}' = \boldsymbol{0}, \tag{3.42a}$$

$$\ddot{\boldsymbol{c}} = \boldsymbol{C}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{C}}\dot{\boldsymbol{q}} + \dot{\boldsymbol{c}}' = \boldsymbol{C}\ddot{\boldsymbol{q}} + \boldsymbol{c}'' = \boldsymbol{0}, \tag{3.42b}$$

$$\dot{\boldsymbol{s}} = \frac{\partial \boldsymbol{y}_\mathrm{o}}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{y}_\mathrm{o}}{\partial t} - \dot{\boldsymbol{y}}_\mathrm{d} = \boldsymbol{H}\dot{\boldsymbol{q}} + \boldsymbol{h}' - \dot{\boldsymbol{y}}_\mathrm{d} = \boldsymbol{0}, \tag{3.42c}$$

$$\ddot{\boldsymbol{s}} = \boldsymbol{H}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{H}}\dot{\boldsymbol{q}} + \dot{\boldsymbol{h}}' - \ddot{\boldsymbol{y}}_\mathrm{d} = \boldsymbol{H}\ddot{\boldsymbol{q}} + \boldsymbol{h}'' - \ddot{\boldsymbol{y}}_\mathrm{d} = \boldsymbol{0}. \tag{3.42d}$$

Despite the terms $\boldsymbol{c}'$ and $\boldsymbol{h}'$ vanish for the considered constraints, they are kept for generality. The Jacobian matrices of the constraints and the input matrix are summarized as

$$\boldsymbol{\Gamma}^\mathrm{T} = \begin{bmatrix} \boldsymbol{C}^\mathrm{T} & \boldsymbol{H}^\mathrm{T} \end{bmatrix}, \tag{3.43a}$$

$$\boldsymbol{G}^\mathrm{T} = \begin{bmatrix} \boldsymbol{C}^\mathrm{T} & \boldsymbol{B} \end{bmatrix}. \tag{3.43b}$$

Here, both matrices $\boldsymbol{\Gamma}^\mathrm{T} \in \mathbb{R}^{f \times (n_\mathrm{c}+n_\mathrm{s})}$ and $\boldsymbol{G}^\mathrm{T} \in \mathbb{R}^{f \times (n_\mathrm{c}+n_\mathrm{s})}$ have the same size, being important for the following computations. Thus, the number of servo constraints needs to match the number of utilized inputs. If more inputs are available, a submatrix of the input matrix $\boldsymbol{B}$ would be used within $\boldsymbol{G}^\mathrm{T}$. This special case of kinematic redundancy is discussed in Section 3.5.2.
For underactuated systems the matrix $\boldsymbol{G}^\mathrm{T}$ cannot be inverted to solve Equation (3.41a) directly for $\boldsymbol{\lambda}$ and $\boldsymbol{u}$. Nevertheless, they can be obtained in a form which is independent of $\ddot{\boldsymbol{q}}$. These accelerations are eliminated by combining

the system dynamics (3.41a) with the second time derivatives of the constraint equations (3.42b) and (3.42d). This gives the intermediate result

$$\boldsymbol{C}\boldsymbol{M}^{-1}\left(\boldsymbol{f}+\boldsymbol{B}\boldsymbol{u}+\boldsymbol{C}^{\mathrm{T}}\boldsymbol{\lambda}\right)+\boldsymbol{c}''=\boldsymbol{0}, \tag{3.44a}$$

$$\boldsymbol{H}\boldsymbol{M}^{-1}\left(\boldsymbol{f}+\boldsymbol{B}\boldsymbol{u}+\boldsymbol{C}^{\mathrm{T}}\boldsymbol{\lambda}\right)+\boldsymbol{h}''-\ddot{\boldsymbol{y}}_{\mathrm{d}}=\boldsymbol{0}. \tag{3.44b}$$

Analogously to [46], solving for $\boldsymbol{\lambda}$ and $\boldsymbol{u}$ leads to

$$\begin{bmatrix}\boldsymbol{\lambda}\\ \boldsymbol{u}\end{bmatrix}=\underbrace{\begin{bmatrix}\boldsymbol{C}\boldsymbol{M}^{-1}\boldsymbol{C}^{\mathrm{T}} & \boldsymbol{C}\boldsymbol{M}^{-1}\boldsymbol{B}\\ \boldsymbol{H}\boldsymbol{M}^{-1}\boldsymbol{C}^{\mathrm{T}} & \boldsymbol{H}\boldsymbol{M}^{-1}\boldsymbol{B}\end{bmatrix}}_{\boldsymbol{\Delta}}{}^{-1}\begin{bmatrix}-\boldsymbol{c}''-\boldsymbol{C}\boldsymbol{M}^{-1}\boldsymbol{f}\\ \ddot{\boldsymbol{y}}_{\mathrm{d}}-\boldsymbol{h}''-\boldsymbol{H}\boldsymbol{M}^{-1}\boldsymbol{f}\end{bmatrix}. \tag{3.45}$$

This is an algebraic equation to be evaluated with the system state $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ coming, e.g., from a time integration of the equations of motion of the inverse model. Details on how to obtain the state are discussed in the following Sections 3.5 and 3.6. The matrix $\boldsymbol{\Delta}=\boldsymbol{\Gamma}\boldsymbol{M}^{-1}\boldsymbol{G}^{\mathrm{T}}$ needs to be regular and is here denoted as *extended decoupling matrix*. The reason is that it can be regarded as an extension of the usual decoupling matrix $\bar{\boldsymbol{\Delta}}=\boldsymbol{H}\boldsymbol{M}^{-1}\boldsymbol{B}$, compare [68].

3.4.2 Singularities

In the previous Section 3.4.1, it is indicated that the extended decoupling matrix $\boldsymbol{\Delta}$ needs to be nonsingular. However, a major drawback of parallel robots are possible singularities [47]. With Flexor being the main application focus of this research, it shall now serve as example to identify such singularities and their relevance for experiments.

For the major part of Flexor's configuration space $\boldsymbol{\Delta}$ of Equation (3.45) is regular when using the end-effector as output. However, there are configurations close to the slider limits where $\boldsymbol{\Delta}$ becomes singular due to the parallel part. Two such cases are illustrated in Figure 3.13. Here, singularities occur at $\alpha=0^\circ$ and $\beta=90^\circ$ in steady state. For the configuration in Figure 3.13a), a motion of the right slider A has no influence on the motion of the point O_2 and therefore also has no influence on the end-effector output. Thus, at this configuration only two of the three actuators can control the 2D end-effector pose. Additionally, when slider A is mirrored at an $\alpha=0^\circ$ configuration, one obtains an identical end-effector pose. This means that when considering besides $\alpha<0^\circ$ also $\alpha>0^\circ$ configurations one cannot relate the end-effector pose to a unique slider position a anymore. In Figure 3.13b), the motion of both sliders results in a purely horizontal translation at the rotary motor. Again, this causes that one degree of freedom of the end-effector pose cannot be controlled independently anymore.

The slider positions a and b which result in such singular configurations are obtained with the geometric constraint equations of the parallel part. In steady state the constraints at O_2 can be written as

$$\boldsymbol{c}=\begin{bmatrix}(\ell_1\cos(-\alpha))-(o+b-\ell_{2_1}\cos\beta)\\ (a+\ell_1\sin\alpha)-(-\ell_{2_1}\sin\beta)\end{bmatrix}=\boldsymbol{0}. \tag{3.46}$$

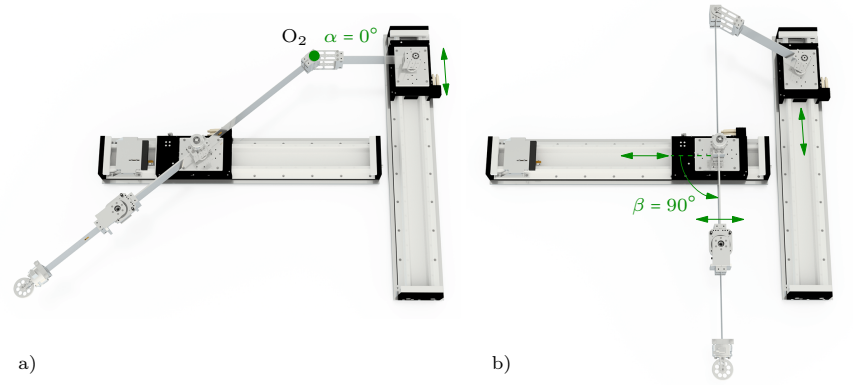


Figure 3.13: Exemplary configurations with singular extended decoupling matrix $\boldsymbol{\Delta}$.

By setting $\alpha = 0°$ Equation (3.46) is reformulated as a circular equation

$$\ell_{2_1}^2 = a^2 + (o - \ell_1 + b)^2 \tag{3.47}$$

and analogously for $\beta = 90°$ Equation (3.46) is rewritten as

$$\ell_1^2 = (\ell_{2_1} + a)^2 + (o + b)^2. \tag{3.48}$$

In Figure 3.14, both circular equations (3.47) and (3.48) are now compared to the Euclidean norm *condition number* of $\boldsymbol{\Delta}$, which is calculated for feasible slider positions. This condition number is the ratio of the largest singular value of $\boldsymbol{\Delta}$ to its smallest singular value and can be obtained, e.g., with MATLAB's `cond` command. A high condition number indicates an ill-conditioned matrix, i.e., a matrix which is close to singular. The color bar is limited to capture most of the values but not the single extreme values going over 10^{10} to ensure better readability. Also, for $\boldsymbol{\Delta}$ an equivalent rigid end-effector $\boldsymbol{y}_\mathrm{r}$ is used as output $\boldsymbol{y}_\mathrm{o}$, i.e., the elastic deformations are neglected in the 2D end-effector pose. The fact that the circles go through the regions with the highest condition numbers confirms that the singularities within $\boldsymbol{\Delta}$ related to the parallel part are indeed due to the $\alpha = 0°$ and $\beta = 90°$ configurations. In the upper right of the surface plot the condition number is small since the circle implies a case with $\alpha = 0°$ for which link 1 and link 2_1 are folded downwards at their connection. This is however not possible due to the slider limit stop shown in Figure 2.7. This means that not all configurations where the circles intersect the surface plot are singular.

In summary, FLEXOR has singular configurations in its workspace which is a common property of parallel robots. As these occur however very close to the slider limits they are not relevant for typical application scenarios. Therefore,

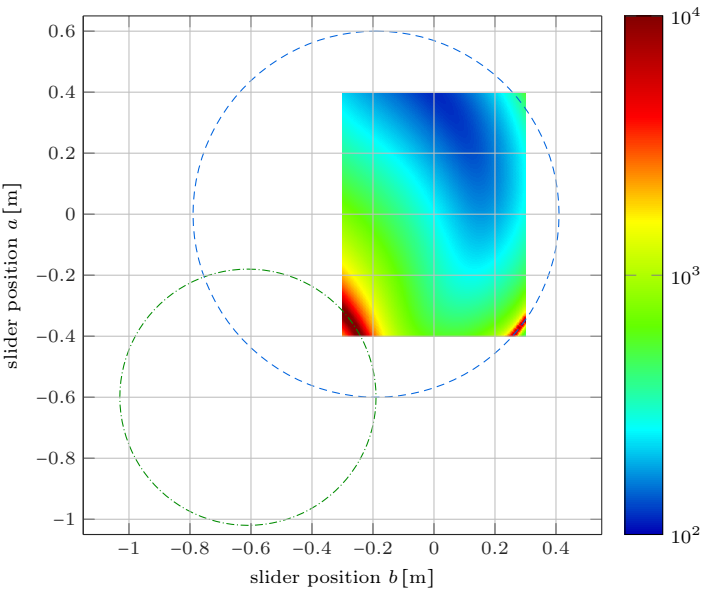


Figure 3.14: Condition number of $\boldsymbol{\Delta}$ ( ) for an equivalent rigid end-effector at steady state with $\gamma = 0°$ and the circles for $\alpha = 0°$ (- - - -) and $\beta = 90°$ (-·-·-).

within all following investigations it is assumed that $\boldsymbol{\Delta}$ is regular, resulting in a differentiation index of 3 and a vector relative degree of $\boldsymbol{r} = \{2, ..., 2\}$ as mentioned before. Further details on singularities within parallel robots are given, e.g., by Merlet and Gosselin [47] as well as by Gosselin and Angeles [89].

3.5 Transformation to ODEs

The forward dynamics of parallel robots typically needs to be formulated as a set of DAEs and the inversion via servo constraints also introduces algebraic equations. For flexible link robots, the resulting DAEs usually have differentiation index 3. Such DAEs tend to be difficult to solve, which renders index reductions attractive. Kunkel and Mehrmann [90] as well as Altmann et al. [91] discuss the concept of minimal extension in order to reduce the index of DAEs by introducing new variables. Altmann et al. [91] compare their approach to the projection-based approach of Blajer and Kołodziejczyk [88]. In this research also a projection approach is used. Still, for the case of model inversion with the same number of servo constraints and inputs, these control inputs are canceled from the system dynamics via a projection, which is one difference to the approach of Blajer and

Kołodziejczyk [88]. The presented approach is mainly based on Seifried et al. [15], which allows to reduce the differentiation index of the occurring DAEs from 3 to 0, i.e., to obtain ODEs. This highly simplifies the numerical solution. The resulting formulation via ODEs might even be computationally efficient enough to enable a forward integration in real-time. This is shown, e.g., in Section 5.2.1.4 for a stable internal dynamics where a real-time model inversion is used as feedforward control for FLEXOR within experiments.

As discussed in Section 3.4, for model inversion with end-effector contact the contact constraint $c_{\mathrm{ef}} = 0$ is replaced by servo constraints, which leads to an identical formulation as for a free end-effector. Thus, Equation (3.41) represents the considered inverse model in all cases.
For the forward model, the formulation of Equation (3.21) with end-effector contact is only used for simulative test purposes in this research, where $c_{\mathrm{ef}} = 0$ can be normalized according to Section 3.3.3. Here, the transformation to ODEs can be performed, e.g., analogously to the later presented Section 3.5.3. However, since within experiments the real system replaces the forward model with contact this model will not be further considered. Instead, in the following only the free end-effector formulation of Equation (3.22), i.e., without $c_{\mathrm{ef}} = 0$, will be of interest such as for estimator and control design.
Since the inverse model of Equation (3.41) can be regarded as an extension of the forward model of Equation (3.22) by adding n_{s} servo constraints, the transformation of the inverse model from DAEs to ODEs is discussed first. The transformation of the forward model to ODEs follows then by neglecting the terms related to servo constraints. Furthermore, for the inverse model a formulation with kinematic redundancy is discussed, which allows to pursue additional objectives besides trajectory tracking. Finally, for the forward model an alternative formulation with kinematic inputs is introduced, which neglects the actuator dynamics to simplify, e.g., the estimator and control design.

3.5.1 Inverse Model

Initially, the standard inverse flexible multibody model is considered with an equal number of inputs $\boldsymbol{u}$ and servo constraints, i.e., $f_{\mathrm{a}} = n_{\mathrm{s}}$. Three alternative approaches are discussed to arrive at the ODEs. They all use a projection with a null space matrix which cancels the unknown Lagrange multipliers $\boldsymbol{\lambda}$ and the inputs $\boldsymbol{u}$ from the system dynamics (3.41a). Therefore, this can be denoted as *null space method*.

3.5.1.1 Coordinate Partitioning

Firstly, the coordinate partitioning approach [68] is being extended to the case with servo constraints. Here, the chain coordinates $\boldsymbol{q} \in \mathbb{R}^{f}$ are manually split up in independent coordinates $\boldsymbol{q}_{\mathrm{i}} \in \mathbb{R}^{f_{\mathrm{i}}}$ with $f_{\mathrm{i}} = f - n_{\mathrm{c}} - n_{\mathrm{s}}$ and in dependent

coordinates $\boldsymbol{q}_\mathrm{d} \in \mathbb{R}^{f_\mathrm{d}}$ with $f_\mathrm{d} = n_\mathrm{c} + n_\mathrm{s}$. Thus, the chain coordinates can be written as $\boldsymbol{q} = [\boldsymbol{q}_\mathrm{i}^\mathrm{T}, \boldsymbol{q}_\mathrm{d}^\mathrm{T}]^\mathrm{T}$. It should be noted that this coordinate partitioning is needed for the subsequent steps but it does not imply that the first f_i coordinates within $\boldsymbol{q}$ need to be selected as independent. Instead, the independent coordinates can be chosen freely with a subsequent internal reordering within $\boldsymbol{q}$. For systems where the underactuation is due to the link flexibility, the elastic coordinates are a possible choice of the independent coordinates.
Based on the partitioning of the coordinates $\boldsymbol{q}$ and with Equation (3.43a), Equations (3.42a) and (3.42c) are rewritten as

$$\begin{bmatrix} \dot{\boldsymbol{c}} \\ \dot{\boldsymbol{s}} \end{bmatrix} = \boldsymbol{\Gamma}\dot{\boldsymbol{q}} + \boldsymbol{\gamma}' = \boldsymbol{\Gamma}_\mathrm{i}\dot{\boldsymbol{q}}_\mathrm{i} + \boldsymbol{\Gamma}_\mathrm{d}\dot{\boldsymbol{q}}_\mathrm{d} + \boldsymbol{\gamma}' = \boldsymbol{0}, \tag{3.49}$$

where $\boldsymbol{\gamma}' = [(\boldsymbol{c}')^\mathrm{T}, (\boldsymbol{h}' - \dot{\boldsymbol{y}}_\mathrm{d})^\mathrm{T}]^\mathrm{T}$. Here, the matrices

$$\boldsymbol{\Gamma} = \frac{\partial \boldsymbol{\gamma}}{\partial \boldsymbol{q}}, \quad \boldsymbol{\Gamma}_\mathrm{i} = \frac{\partial \boldsymbol{\gamma}}{\partial \boldsymbol{q}_\mathrm{i}}, \quad \boldsymbol{\Gamma}_\mathrm{d} = \frac{\partial \boldsymbol{\gamma}}{\partial \boldsymbol{q}_\mathrm{d}} \tag{3.50}$$

are Jacobians with $\boldsymbol{\gamma} = [\boldsymbol{c}^\mathrm{T}, \boldsymbol{s}^\mathrm{T}]^\mathrm{T} \in \mathbb{R}^{n_\mathrm{c}+n_\mathrm{s}}$. Solving Equation (3.49) for $\dot{\boldsymbol{q}}_\mathrm{d}$ gives

$$\dot{\boldsymbol{q}} = \begin{bmatrix} \dot{\boldsymbol{q}}_\mathrm{i} \\ \dot{\boldsymbol{q}}_\mathrm{d} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{I}_{f_\mathrm{i}} \\ -\boldsymbol{\Gamma}_\mathrm{d}^{-1}\boldsymbol{\Gamma}_\mathrm{i} \end{bmatrix}}_{\boldsymbol{J}_\mathrm{r}} \dot{\boldsymbol{q}}_\mathrm{i} + \underbrace{\begin{bmatrix} \boldsymbol{0} \\ -\boldsymbol{\Gamma}_\mathrm{d}^{-1}\boldsymbol{\gamma}' \end{bmatrix}}_{\boldsymbol{\theta}'}, \tag{3.51}$$

with $\mathbf{I}_{f_\mathrm{i}}$ being the identity matrix of dimension $f_\mathrm{i} \times f_\mathrm{i}$ and $\boldsymbol{J}_\mathrm{r}$ being of dimension $f \times f_\mathrm{i}$. Similarly, with Equations (3.42b) and (3.42d) it follows on acceleration level

$$\ddot{\boldsymbol{q}} = \begin{bmatrix} \ddot{\boldsymbol{q}}_\mathrm{i} \\ \ddot{\boldsymbol{q}}_\mathrm{d} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{I}_{f_\mathrm{i}} \\ -\boldsymbol{\Gamma}_\mathrm{d}^{-1}\boldsymbol{\Gamma}_\mathrm{i} \end{bmatrix}}_{\boldsymbol{J}_\mathrm{r}} \ddot{\boldsymbol{q}}_\mathrm{i} + \underbrace{\begin{bmatrix} \boldsymbol{0} \\ -\boldsymbol{\Gamma}_\mathrm{d}^{-1}\boldsymbol{\gamma}'' \end{bmatrix}}_{\boldsymbol{\theta}''}, \tag{3.52}$$

with $\boldsymbol{\gamma}'' = [(\boldsymbol{c}'')^\mathrm{T}, (\boldsymbol{h}'' - \ddot{\boldsymbol{y}}_\mathrm{d})^\mathrm{T}]^\mathrm{T}$. In order to transform the DAEs (3.41) to ODEs a projection matrix $\boldsymbol{J}_\ell \in \mathbb{R}^{f \times f_\mathrm{i}}$ needs to be found to cancel $\boldsymbol{G}$, i.e., the Lagrange multipliers $\boldsymbol{\lambda}$ and the inputs $\boldsymbol{u}$ from Equation (3.41a). A possible choice is

$$\boldsymbol{G}\boldsymbol{J}_\ell = \begin{bmatrix} \boldsymbol{G}_\mathrm{i} & \boldsymbol{G}_\mathrm{d} \end{bmatrix} \underbrace{\begin{bmatrix} \mathbf{I}_{f_\mathrm{i}} \\ -\boldsymbol{G}_\mathrm{d}^{-1}\boldsymbol{G}_\mathrm{i} \end{bmatrix}}_{\boldsymbol{J}_\ell} = \boldsymbol{0}. \tag{3.53}$$

Here, $\boldsymbol{G}_\mathrm{i} \in \mathbb{R}^{(n_\mathrm{c}+n_\mathrm{s}) \times f_\mathrm{i}}$ and $\boldsymbol{G}_\mathrm{d} \in \mathbb{R}^{(n_\mathrm{c}+n_\mathrm{s}) \times f_\mathrm{d}}$ can be obtained by a partitioning of $\boldsymbol{G}$ analogously to $\boldsymbol{\Gamma}$ in Equation (3.49). The transposed projection matrix $\boldsymbol{J}_\ell^\mathrm{T}$ is now multiplied from the left to the system dynamics (3.41a), where also the accelerations $\ddot{\boldsymbol{q}}$ are replaced by Equation (3.52), yielding

$$\boldsymbol{J}_\ell^\mathrm{T}\boldsymbol{M}\left(\boldsymbol{J}_\mathrm{r}\ddot{\boldsymbol{q}}_\mathrm{i} + \boldsymbol{\theta}''\right) = \boldsymbol{J}_\ell^\mathrm{T}\boldsymbol{f} + \underbrace{\boldsymbol{J}_\ell^\mathrm{T}\boldsymbol{G}^\mathrm{T}}_{\boldsymbol{0}} \begin{bmatrix} \boldsymbol{\lambda} \\ \boldsymbol{u} \end{bmatrix}. \tag{3.54}$$

The equation of motion (3.54) in ODE as well as in minimal form, i.e., for the independent accelerations, is then rearranged to

$$\ddot{\boldsymbol{q}}_\mathrm{i} = \left(\boldsymbol{J}_\ell^\mathrm{T} \boldsymbol{M} \boldsymbol{J}_\mathrm{r}\right)^{-1} \boldsymbol{J}_\ell^\mathrm{T} (\boldsymbol{f} - \boldsymbol{M}\boldsymbol{\theta}''). \tag{3.55}$$

Now, solving Equation (3.51) for $\dot{\boldsymbol{q}}_\mathrm{i}$ via a left multiplication with the transposed non-square matrix $\boldsymbol{J}_\mathrm{r}$ yields the independent velocities

$$\dot{\boldsymbol{q}}_\mathrm{i} = \left(\boldsymbol{J}_\mathrm{r}^\mathrm{T} \boldsymbol{J}_\mathrm{r}\right)^{-1} \boldsymbol{J}_\mathrm{r}^\mathrm{T} (\dot{\boldsymbol{q}} - \boldsymbol{\theta}'). \tag{3.56}$$

Plugging Equations (3.55) and (3.56) back into Equations (3.52) and (3.51) gives the equations of motion in chain coordinates

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_\mathrm{r} \left(\boldsymbol{J}_\mathrm{r}^\mathrm{T} \boldsymbol{J}_\mathrm{r}\right)^{-1} \boldsymbol{J}_\mathrm{r}^\mathrm{T} \left(\dot{\boldsymbol{q}} - \boldsymbol{\theta}'\right) + \boldsymbol{\theta}', \tag{3.57a}$$

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}_\mathrm{r} \left(\boldsymbol{J}_\ell^\mathrm{T} \boldsymbol{M} \boldsymbol{J}_\mathrm{r}\right)^{-1} \boldsymbol{J}_\ell^\mathrm{T} \left(\boldsymbol{f} - \boldsymbol{M}\boldsymbol{\theta}''\right) + \boldsymbol{\theta}''. \tag{3.57b}$$

These ODEs in state-space representation, for the state $\boldsymbol{x} = [\boldsymbol{q}^\mathrm{T}, \dot{\boldsymbol{q}}^\mathrm{T}]^\mathrm{T}$, are in a form which can be directly used within an integrator. Here, the right-hand side is evaluated leading to the left-hand side needed by such an integrator. It is worth noting that $\dot{\boldsymbol{q}}$ appears on both sides of Equation (3.57a). This, however, does not indicate that it needs to be solved for $\dot{\boldsymbol{q}}$ but by inserting and projecting it with $\boldsymbol{J}_\mathrm{r}$ it is ensured that the obtained $\dot{\boldsymbol{q}}$ on the left side complies with the constraints on velocity level. Consequently, the drift through integration is reduced to being only linear within the constraint equations (3.41b) and (3.41c) on position level.

3.5.1.2 QR Decomposition

A manual selection of the independent and dependent coordinates as in the preceding coordinate partitioning approach might lead to singularities within $\boldsymbol{\Gamma}_\mathrm{d}$ and also $\boldsymbol{G}_\mathrm{d}$ can be singular, which will cause the algorithm to fail. Therefore, an alternative approach is now presented which uses amongst others an automatic selection of the independent and dependent coordinates. Based on [15, 92], this approach relies on the QR decompositions

$$\boldsymbol{\Gamma}^\mathrm{T} = \begin{bmatrix} \boldsymbol{C}^\mathrm{T} & \boldsymbol{H}^\mathrm{T} \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{Q}_\mathrm{r} & \boldsymbol{J}_{\mathrm{r,q}} \end{bmatrix}}_{\boldsymbol{Q}_\Gamma} \begin{bmatrix} \boldsymbol{R}_\mathrm{r} \\ \boldsymbol{0} \end{bmatrix} = \boldsymbol{Q}_\mathrm{r} \boldsymbol{R}_\mathrm{r}, \tag{3.58a}$$

$$\boldsymbol{G}^\mathrm{T} = \begin{bmatrix} \boldsymbol{C}^\mathrm{T} & \boldsymbol{B} \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{Q}_\ell & \boldsymbol{J}_{\ell,\mathrm{q}} \end{bmatrix}}_{\boldsymbol{Q}_\mathrm{G}} \begin{bmatrix} \boldsymbol{R}_\ell \\ \boldsymbol{0} \end{bmatrix} = \boldsymbol{Q}_\ell \boldsymbol{R}_\ell, \tag{3.58b}$$

which yield the matrices $\boldsymbol{J}_{\mathrm{r,q}}$ and $\boldsymbol{J}_{\ell,\mathrm{q}}$. They have the same dimension and purpose as $\boldsymbol{J}_\mathrm{r}$ and $\boldsymbol{J}_\ell$ from the coordinate partitioning approach but contain

different values. With $\boldsymbol{Q}_\Gamma \in \mathbb{R}^{f\times f}$ and $\boldsymbol{Q}_\mathrm{G} \in \mathbb{R}^{f\times f}$ being orthogonal matrices, it follows

$$\boldsymbol{\Gamma}\boldsymbol{J}_{\mathrm{r,q}} = \boldsymbol{R}_\mathrm{r}^\mathrm{T} \underbrace{\boldsymbol{Q}_\mathrm{r}^\mathrm{T}\boldsymbol{J}_{\mathrm{r,q}}}_{\mathbf{0}} = \mathbf{0}, \tag{3.59a}$$

$$\boldsymbol{G}\boldsymbol{J}_{\ell,\mathrm{q}} = \boldsymbol{R}_\ell^\mathrm{T} \underbrace{\boldsymbol{Q}_\ell^\mathrm{T}\boldsymbol{J}_{\ell,\mathrm{q}}}_{\mathbf{0}} = \mathbf{0}. \tag{3.59b}$$

Thus, analogously to the coordinate partitioning approach the columns of $\boldsymbol{J}_{\mathrm{r,q}}$ and $\boldsymbol{J}_{\ell,\mathrm{q}}$ span the null space of $\boldsymbol{\Gamma}$ and $\boldsymbol{G}$, respectively. Based on the QR decomposition, a new set of independent and dependent coordinates is introduced according to

$$\boldsymbol{q} = \boldsymbol{Q}_\Gamma \begin{bmatrix} \boldsymbol{q}_{\mathrm{d,q}} \\ \boldsymbol{q}_{\mathrm{i,q}} \end{bmatrix} = \boldsymbol{J}_{\mathrm{r,q}}\boldsymbol{q}_{\mathrm{i,q}} + \boldsymbol{Q}_\mathrm{r}\boldsymbol{q}_{\mathrm{d,q}}, \tag{3.60}$$

with $\boldsymbol{q}_{\mathrm{d,q}} \in \mathbb{R}^{f_\mathrm{d}}$ and $\boldsymbol{q}_{\mathrm{i,q}} \in \mathbb{R}^{f_\mathrm{i}}$ having the same dimension as in the coordinate partitioning approach. This relates the independent coordinates to an orthonormal basis of the constraint tangent plane and the dependent coordinates are related to an orthonormal basis of the row space of the constraint Jacobian $\boldsymbol{\Gamma}$. These coordinates are in general not single elements of $\boldsymbol{q}$, i.e., they have no direct physical meaning anymore. Also, the selection changes automatically in each time step depending on $\boldsymbol{q}$. Its time derivatives can be written as

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_{\mathrm{r,q}}\dot{\boldsymbol{q}}_{\mathrm{i,q}} + \boldsymbol{Q}_\mathrm{r}\boldsymbol{q}'_{\mathrm{d,q}} = \boldsymbol{J}_{\mathrm{r,q}}\dot{\boldsymbol{q}}_{\mathrm{i,q}} + \boldsymbol{\theta}'_\mathrm{q}, \tag{3.61a}$$

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}_{\mathrm{r,q}}\ddot{\boldsymbol{q}}_{\mathrm{i,q}} + \boldsymbol{Q}_\mathrm{r}\boldsymbol{q}''_{\mathrm{d,q}} = \boldsymbol{J}_{\mathrm{r,q}}\ddot{\boldsymbol{q}}_{\mathrm{i,q}} + \boldsymbol{\theta}''_\mathrm{q}, \tag{3.61b}$$

with $\boldsymbol{q}'_{\mathrm{d,q}}$ and $\boldsymbol{q}''_{\mathrm{d,q}}$ to be determined. Inserting Equations (3.58a) and (3.61a) into the constraint equation (3.49) on velocity level gives

$$\begin{bmatrix} \dot{\boldsymbol{c}} \\ \dot{\boldsymbol{s}} \end{bmatrix} = \boldsymbol{\Gamma}\dot{\boldsymbol{q}} + \boldsymbol{\gamma}' = \boldsymbol{R}_\mathrm{r}^\mathrm{T} \underbrace{\boldsymbol{Q}_\mathrm{r}^\mathrm{T}\boldsymbol{J}_{\mathrm{r,q}}}_{\mathbf{0}} \dot{\boldsymbol{q}}_{\mathrm{i,q}} + \boldsymbol{R}_\mathrm{r}^\mathrm{T} \underbrace{\boldsymbol{Q}_\mathrm{r}^\mathrm{T}\boldsymbol{Q}_\mathrm{r}}_{\mathbf{I}_{f_\mathrm{d}}} \boldsymbol{q}'_{\mathrm{d,q}} + \boldsymbol{\gamma}' = \mathbf{0}. \tag{3.62}$$

This yields $\boldsymbol{q}'_{\mathrm{d,q}} = -\boldsymbol{R}_\mathrm{r}^{-\mathrm{T}}\boldsymbol{\gamma}'$ where $\boldsymbol{R}_\mathrm{r} \in \mathbb{R}^{(n_\mathrm{c}+n_\mathrm{s})\times(n_\mathrm{c}+n_\mathrm{s})}$ is a square matrix. Analogously, on acceleration level it follows $\boldsymbol{q}''_{\mathrm{d,q}} = -\boldsymbol{R}_\mathrm{r}^{-\mathrm{T}}\boldsymbol{\gamma}''$. Based on Equation (3.59b), the Lagrange multipliers $\boldsymbol{\lambda}$ and the control inputs $\boldsymbol{u}$ are canceled by a left multiplication with $\boldsymbol{J}_{\ell,\mathrm{q}}^\mathrm{T}$, compare Equation (3.54). With analog steps to arrive at Equation (3.57), the equations of motion in chain coordinates follow as

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_{\mathrm{r,q}}\boldsymbol{J}_{\mathrm{r,q}}^\mathrm{T}\left(\dot{\boldsymbol{q}} - \boldsymbol{\theta}'_\mathrm{q}\right) + \boldsymbol{\theta}'_\mathrm{q}, \tag{3.63a}$$

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}_{\mathrm{r,q}}\left(\boldsymbol{J}_{\ell,\mathrm{q}}^\mathrm{T}\boldsymbol{M}\boldsymbol{J}_{\mathrm{r,q}}\right)^{-1}\boldsymbol{J}_{\ell,\mathrm{q}}^\mathrm{T}\left(\boldsymbol{f} - \boldsymbol{M}\boldsymbol{\theta}''_\mathrm{q}\right) + \boldsymbol{\theta}''_\mathrm{q}. \tag{3.63b}$$

These are again ODEs in state-space representation. The equation structure is the same as for the coordinate partitioning. The only difference is that for

the QR decomposition the product $\boldsymbol{J}_{\mathrm{r,q}}^{\mathrm{T}}\boldsymbol{J}_{\mathrm{r,q}}$ yields the identity matrix and has therefore been neglected within Equation (3.63a).
The model inversion in chain coordinates with the state $\boldsymbol{x} = [\boldsymbol{q}^{\mathrm{T}}, \dot{\boldsymbol{q}}^{\mathrm{T}}]^{\mathrm{T}}$ is summarized in Figure 3.15. Here, the superscript "iv" in the function symbol $\boldsymbol{f}_{\mathrm{c}}^{\mathrm{iv}}$

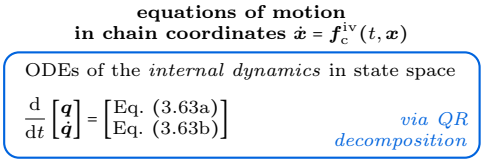


Figure 3.15: Model inversion for flexible multibody systems with geometric and servo constraints in chain coordinates.

denotes that an inverse model is utilized and the subscript "c" represents the formulation in chain coordinates. Due to the prevention of singularities when computing the projection matrices, exclusively the QR decomposition is used in this research for this and further formulations in chain coordinates. For the same reason, the formulation of Figure 3.15 is also of major interest for real-time model inversion. This is implemented experimentally, amongst others, within collision avoidance in Section 5.2.2 and within hybrid force/position control in Section 5.2.3. There, the prevention of singularities enables a safe operation and no potentially computationally expensive algorithms are needed to catch or treat singularities.

3.5.1.3 Mixed Coordinate Partitioning

It is worth noting that it is possible to replace $\boldsymbol{J}_{\ell}$ with $\boldsymbol{J}_{\ell,\mathrm{q}}$ within the coordinate partitioning approach in order to prevent singularities, which might appear in $\boldsymbol{G}_{\mathrm{d}}$, while still keeping the manually selected independent coordinates. This means that a mix of both approaches from Sections 3.5.1.1 and 3.5.1.2 is possible to benefit from the advantages of both methods. This mix is here denoted as *mixed coordinate partitioning*. Based on Equation (3.55), the equation of motion in minimal form then follows as

$$\ddot{\boldsymbol{q}}_{\mathrm{i}} = \left(\boldsymbol{J}_{\ell,\mathrm{q}}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{J}_{\mathrm{r}}\right)^{-1}\boldsymbol{J}_{\ell,\mathrm{q}}^{\mathrm{T}}(\boldsymbol{f} - \boldsymbol{M}\boldsymbol{\theta}''). \tag{3.64}$$

The model inversion with the independent state $\boldsymbol{x}_{\mathrm{i}} = [\boldsymbol{q}_{\mathrm{i}}^{\mathrm{T}}, \dot{\boldsymbol{q}}_{\mathrm{i}}^{\mathrm{T}}]^{\mathrm{T}}$ is summarized in Figure 3.16. As depicted in step 1), it is required to internally solve for the dependent coordinates $\boldsymbol{q}_{\mathrm{d}}$. Thus, there is no drift of the constraint equations (3.41b) and (3.41c) on position level. Also, the manual selection of the independent coordinates $\boldsymbol{q}_{\mathrm{i}}$ ensures that the coordinate selection does not change throughout operation. Since this often simplifies the implementation, a manual coordinate selection is typically utilized for minimal forms in this research.
The formulation of Figure 3.16 is well suited for offline calculations where possible singularities through a manual coordinate partitioning can be caught. This

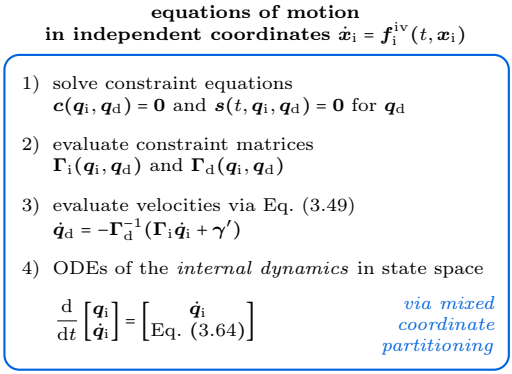


Figure 3.16: Model inversion for flexible multibody systems with geometric and servo constraints in independent coordinates (minimal form).

renders it attractive when the solution is noncausal, which needs to be obtained offline anyways. Such a case occurs in Section 5.3.1 where the model inversion is formulated as a BVP and as a trajectory optimization problem. These problems also benefit from the compactness of the minimal form and from the prevention of a drift of the constraint equations.

3.5.2 Inverse Model with Kinematic Redundancy

So-called *kinematic redundancy* for underactuated systems occurs when more independent control inputs are available than outputs to track. This means that the dimension of $\boldsymbol{u}$ is larger than the number of servo constraints, i.e., $f_\mathrm{a} > n_\mathrm{s}$. The difference of such a redundant case compared to Section 3.5.1 is that not all elements of $\boldsymbol{u}$ are needed to enforce the servo constraints. Therefore, kinematic redundancy allows to perform secondary tasks besides the primary task of end-effector, i.e., output trajectory tracking.
Without loss of generality, it is now assumed that the kinematic redundancy occurs through a reduction of the servo constraints. Hence, the servo constraints $\boldsymbol{s} = \boldsymbol{0}$ are here renamed to *reduced servo constraints* $\bar{\boldsymbol{s}} = \boldsymbol{0}$ and the constraints $\boldsymbol{\gamma} = \boldsymbol{0}$ to *reduced constraints* $\bar{\boldsymbol{\gamma}} = \boldsymbol{0}$. This means that in this context the optional bar symbols serve to highlight a difference to the non-redundant case from Section 3.5.1. Initially, n_s inputs $\bar{\boldsymbol{u}}_\ell$ are manually selected from all elements of $\boldsymbol{u}$ to enforce the n_s servo constraints. Thus, $\boldsymbol{u}$ is partitioned according to

$$\boldsymbol{u} = \begin{bmatrix}\bar{\boldsymbol{u}}_\ell\\ \bar{\boldsymbol{u}}_\mathrm{r}\end{bmatrix}. \tag{3.65}$$

The additional inputs $\bar{\boldsymbol{u}}_\mathrm{r}$ due to the redundancy can then be freely used to realize further control tasks. It should be noted that the partitioning of the inputs in

Equation (3.65) does not indicate that the first elements need to make up $\bar{\boldsymbol{u}}_\ell$ but rather an internal reordering may be done. After partitioning the inputs $\boldsymbol{u}$ according to Equation (3.65), the inverse flexible multibody model can then be transformed to ODEs analogously to Section 3.5.1.

3.5.2.1 QR Decomposition

The QR decompositions of the reduced constraint and input matrices

$$\boldsymbol{\Gamma}^{\mathrm{T}} = \overline{\boldsymbol{\Gamma}}^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{C}^{\mathrm{T}} & \overline{\boldsymbol{H}}^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \bar{\boldsymbol{Q}}_{\mathrm{r}} & \bar{\boldsymbol{J}}_{\mathrm{r,q}} \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{R}}_{\mathrm{r}} \\ \boldsymbol{0} \end{bmatrix} = \bar{\boldsymbol{Q}}_{\mathrm{r}} \bar{\boldsymbol{R}}_{\mathrm{r}}, \tag{3.66a}$$

$$\boldsymbol{G}^{\mathrm{T}} = \bar{\boldsymbol{G}}^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{C}^{\mathrm{T}} & \bar{\boldsymbol{B}}_\ell \end{bmatrix} = \begin{bmatrix} \bar{\boldsymbol{Q}}_\ell & \bar{\boldsymbol{J}}_{\ell,\mathrm{q}} \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{R}}_\ell \\ \boldsymbol{0} \end{bmatrix} = \bar{\boldsymbol{Q}}_\ell \bar{\boldsymbol{R}}_\ell, \tag{3.66b}$$

give the projection matrices $\bar{\boldsymbol{J}}_{\mathrm{r,q}}$ and $\bar{\boldsymbol{J}}_{\ell,\mathrm{q}}$. Here, $\overline{\boldsymbol{H}}$ corresponds to $\boldsymbol{H}$ for the reduced servo constraints and $\bar{\boldsymbol{B}}_\ell$ is the part of $\boldsymbol{B}$ related to $\bar{\boldsymbol{u}}_\ell$. Analogously to Section 3.5.1.2, the equations of motion in chain coordinates then follow as

$$\dot{\boldsymbol{q}} = \bar{\boldsymbol{J}}_{\mathrm{r,q}} \bar{\boldsymbol{J}}_{\mathrm{r,q}}^{\mathrm{T}} \left(\dot{\boldsymbol{q}} - \bar{\boldsymbol{\theta}}_{\mathrm{q}}' \right) + \bar{\boldsymbol{\theta}}_{\mathrm{q}}', \tag{3.67a}$$

$$\ddot{\boldsymbol{q}} = \bar{\boldsymbol{J}}_{\mathrm{r,q}} \left(\bar{\boldsymbol{J}}_{\ell,\mathrm{q}}^{\mathrm{T}} \boldsymbol{M} \bar{\boldsymbol{J}}_{\mathrm{r,q}} \right)^{-1} \bar{\boldsymbol{J}}_{\ell,\mathrm{q}}^{\mathrm{T}} \left(\boldsymbol{f} + \bar{\boldsymbol{B}}_{\mathrm{r}} \bar{\boldsymbol{u}}_{\mathrm{r}} - \boldsymbol{M} \bar{\boldsymbol{\theta}}_{\mathrm{q}}'' \right) + \bar{\boldsymbol{\theta}}_{\mathrm{q}}''. \tag{3.67b}$$

Here, $\bar{\boldsymbol{\theta}}_{\mathrm{q}}' = -\bar{\boldsymbol{Q}}_{\mathrm{r}} \bar{\boldsymbol{R}}_{\mathrm{r}}^{-\mathrm{T}} \bar{\boldsymbol{\gamma}}'$ and $\bar{\boldsymbol{\theta}}_{\mathrm{q}}'' = -\bar{\boldsymbol{Q}}_{\mathrm{r}} \bar{\boldsymbol{R}}_{\mathrm{r}}^{-\mathrm{T}} \bar{\boldsymbol{\gamma}}''$ hold where $\bar{\boldsymbol{\gamma}}'$ and $\bar{\boldsymbol{\gamma}}''$ correspond to $\boldsymbol{\gamma}'$ and $\boldsymbol{\gamma}''$ for the reduced constraints. In contrast to Equation (3.63), the part $\bar{\boldsymbol{B}}_{\mathrm{r}} \bar{\boldsymbol{u}}_{\mathrm{r}}$ of $\boldsymbol{B}\boldsymbol{u}$ is still available in Equation (3.67) to control this internal dynamics. This model inversion is summarized in Figure 3.17. The prevention of singularities

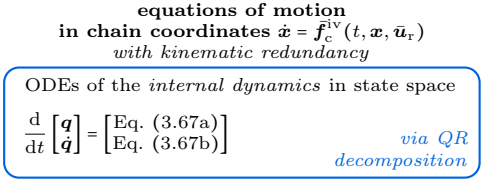


Figure 3.17: Model inversion for flexible multibody systems with geometric and servo constraints in chain coordinates with kinematic redundancy.

when computing the projection matrices due to the QR decomposition renders this formulation well suited for real-time model inversion. It is therefore implemented within the state control with kinematic redundancy in Section 5.2.1.4. In the corresponding experiments, the rotation of the end-effector of Flexor is not tracked but only its position. The resulting redundancy allows then to use the input $\bar{\boldsymbol{u}}_{\mathrm{r}} = u_\gamma$ of the rotary motor as a design degree of freedom to achieve a stable internal dynamics.

3.5.2.2 Mixed Coordinate Partitioning

With the reduced constraints and the coordinate partitioning approach one obtains the relation

$$\ddot{\boldsymbol{q}} = \bar{\boldsymbol{J}}_\mathrm{r}\ddot{\boldsymbol{q}}_\mathrm{i} + \bar{\boldsymbol{\theta}}'' \tag{3.68}$$

analogously to Equation (3.52). It should be noted that for consistency with other formulations the optional bar symbols are dropped for the independent and dependent coordinates $\boldsymbol{q}_\mathrm{i}$ and $\boldsymbol{q}_\mathrm{d}$ as well as for their time derivatives. With Equation (3.68) and $\bar{\boldsymbol{J}}_{\ell,\mathrm{q}}$ of the QR decomposition (3.66b) as well as with the system dynamics (3.41a), the equation of motion in minimal form for the mixed coordinate partitioning then follows as

$$\ddot{\boldsymbol{q}}_\mathrm{i} = \left(\bar{\boldsymbol{J}}_{\ell,\mathrm{q}}^\mathrm{T}\boldsymbol{M}\bar{\boldsymbol{J}}_\mathrm{r}\right)^{-1}\bar{\boldsymbol{J}}_{\ell,\mathrm{q}}^\mathrm{T}(\boldsymbol{f} + \bar{\boldsymbol{B}}_\mathrm{r}\bar{\boldsymbol{u}}_\mathrm{r} - \boldsymbol{M}\bar{\boldsymbol{\theta}}''). \tag{3.69}$$

In contrast to the non-redundant case of Equation (3.64), the part $\bar{\boldsymbol{u}}_\mathrm{r}$ of the input $\boldsymbol{u}$ is still available to control this internal dynamics. The model inversion is shown in Figure 3.18, where the matrices $\overline{\boldsymbol{\Gamma}}_\mathrm{i}$ and $\overline{\boldsymbol{\Gamma}}_\mathrm{d}$ correspond to $\boldsymbol{\Gamma}_\mathrm{i}$ and $\boldsymbol{\Gamma}_\mathrm{d}$

equations of motion
in independent coordinates $\dot{\boldsymbol{x}}_\mathrm{i} = \bar{\boldsymbol{f}}_\mathrm{i}^\mathrm{iv}(t, \boldsymbol{x}_\mathrm{i}, \bar{\boldsymbol{u}}_\mathrm{r})$
with kinematic redundancy

1) solve constraint equations
   $\boldsymbol{c}(\boldsymbol{q}_\mathrm{i}, \boldsymbol{q}_\mathrm{d}) = \boldsymbol{0}$ and $\bar{\boldsymbol{s}}(t, \boldsymbol{q}_\mathrm{i}, \boldsymbol{q}_\mathrm{d}) = \boldsymbol{0}$ for $\boldsymbol{q}_\mathrm{d}$
2) evaluate constraint matrices
   $\overline{\boldsymbol{\Gamma}}_\mathrm{i}(\boldsymbol{q}_\mathrm{i}, \boldsymbol{q}_\mathrm{d})$ and $\overline{\boldsymbol{\Gamma}}_\mathrm{d}(\boldsymbol{q}_\mathrm{i}, \boldsymbol{q}_\mathrm{d})$
3) evaluate velocities based on $\dot{\bar{\boldsymbol{\gamma}}} = \boldsymbol{0}$ via
   $\dot{\boldsymbol{q}}_\mathrm{d} = -\overline{\boldsymbol{\Gamma}}_\mathrm{d}^{-1}(\overline{\boldsymbol{\Gamma}}_\mathrm{i}\dot{\boldsymbol{q}}_\mathrm{i} + \bar{\boldsymbol{\gamma}}')$
4) ODEs of the *internal dynamics* in state space
   $\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\boldsymbol{q}_\mathrm{i}\\ \dot{\boldsymbol{q}}_\mathrm{i}\end{bmatrix} = \begin{bmatrix}\dot{\boldsymbol{q}}_\mathrm{i}\\ \text{Eq. (3.69)}\end{bmatrix}$ *via mixed coordinate partitioning*

Figure 3.18: Model inversion for flexible multibody systems with geometric and servo constraints in independent coordinates (minimal form) with kinematic redundancy.

for the reduced constraints. This minimal form is utilized within a trajectory optimization in Section 5.3.1. There, the redundancy obtained by not tracking the end-effector rotation allows to reduce the link deformations while tracking the end-effector position.

3.5.3 Forward Model

The considered forward model of Equation (3.22) simply corresponds to the inverse model of Equation (3.41) for a free end-effector but without servo constraints, i.e., with $n_\mathrm{s} = 0$. This means that the equations introduced in Section 3.5.1 can be analogously applied when omitting the terms coming from servo constraints.

3.5.3.1 Coordinate Partitioning

The matrices

$$\boldsymbol{J}_\ell = \boldsymbol{J}_\mathrm{r} = \begin{bmatrix} \mathbf{I}_{f_\mathrm{i}} \\ -\boldsymbol{C}_\mathrm{d}^{-1}\boldsymbol{C}_\mathrm{i} \end{bmatrix} = \boldsymbol{J}_\mathrm{c} \tag{3.70}$$

of dimension $f \times f_\mathrm{i}$ are now identical and are denoted as $\boldsymbol{J}_\mathrm{c}$. Here, only the Jacobians of the geometric constraints

$$\boldsymbol{C}_\mathrm{i} = \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}_\mathrm{i}}, \quad \boldsymbol{C}_\mathrm{d} = \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}_\mathrm{d}} \tag{3.71}$$

are needed which have the dimension $n_\mathrm{c} \times f_\mathrm{i}$ and $n_\mathrm{c} \times f_\mathrm{d}$, respectively. With the projection matrix $\boldsymbol{J}_\mathrm{c}$ the unknown Lagrange multipliers $\boldsymbol{\lambda}$ are canceled from the system dynamics (3.22a). Based on a derivation analogously to Equation (3.55), the equation of motion in minimal form follows as

$$\ddot{\boldsymbol{q}}_\mathrm{i} = \left(\boldsymbol{J}_\mathrm{c}^\mathrm{T}\boldsymbol{M}\boldsymbol{J}_\mathrm{c}\right)^{-1}\boldsymbol{J}_\mathrm{c}^\mathrm{T}\left(\boldsymbol{f} + \boldsymbol{B}\boldsymbol{u} - \boldsymbol{M}\boldsymbol{\theta}_\mathrm{c}''\right). \tag{3.72}$$

Here, the control input $\boldsymbol{u}$ still occurs since a forward model is considered in contrast to Equation (3.55). Also, the terms related to servo constraints within $\boldsymbol{\theta}''$ vanish, which gives

$$\boldsymbol{\theta}'' = \begin{bmatrix} \mathbf{0} \\ -\boldsymbol{C}_\mathrm{d}^{-1}\boldsymbol{c}'' \end{bmatrix} = \boldsymbol{\theta}_\mathrm{c}''. \tag{3.73}$$

With Equation (3.72), the function for the equations of motion in independent coordinates is established analogously to Figure 3.16, but without servo constraints. Thus, the system is now time-invariant and depends on the input $\boldsymbol{u}$. The involved steps of the resulting function $\dot{\boldsymbol{x}}_\mathrm{i} = \boldsymbol{f}_\mathrm{i}^\mathrm{fw}(\boldsymbol{x}_\mathrm{i}, \boldsymbol{u}) \in \mathbb{R}^{2f_\mathrm{i}}$ are shown in Figure 3.19. Here, "fw" denotes that a forward model is considered.

equations of motion
in independent coordinates $\dot{\boldsymbol{x}}_\mathrm{i} = \boldsymbol{f}_\mathrm{i}^\mathrm{fw}(\boldsymbol{x}_\mathrm{i}, \boldsymbol{u})$

1) solve constraint equations $\boldsymbol{c}(\boldsymbol{q}_\mathrm{i}, \boldsymbol{q}_\mathrm{d}) = \mathbf{0}$ for $\boldsymbol{q}_\mathrm{d}$
2) evaluate constraint matrices $\boldsymbol{C}_\mathrm{i}(\boldsymbol{q}_\mathrm{i}, \boldsymbol{q}_\mathrm{d})$ and $\boldsymbol{C}_\mathrm{d}(\boldsymbol{q}_\mathrm{i}, \boldsymbol{q}_\mathrm{d})$
3) evaluate velocities based on $\dot{\boldsymbol{c}} = \mathbf{0}$ via $\dot{\boldsymbol{q}}_\mathrm{d} = -\boldsymbol{C}_\mathrm{d}^{-1}(\boldsymbol{C}_\mathrm{i}\dot{\boldsymbol{q}}_\mathrm{i} + \boldsymbol{c}')$
4) ODEs of the *forward dynamics* in state space

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\boldsymbol{q}_\mathrm{i} \\ \dot{\boldsymbol{q}}_\mathrm{i}\end{bmatrix} = \begin{bmatrix}\dot{\boldsymbol{q}}_\mathrm{i} \\ \text{Eq. (3.72)}\end{bmatrix}$$

via coordinate partitioning

Figure 3.19: Forward model for flexible multibody systems with geometric constraints in independent coordinates (minimal form).

In Section 6.2, this formulation is used to design an LQR, which is applied to Flexor for oscillation damping. There, the minimal form ensures that the system is stabilizable.

3.5.3.2 QR Decomposition

With the matrices from Equation (3.43), which reduce to

$$\boldsymbol{G} = \boldsymbol{\Gamma} = \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}} = \boldsymbol{C} \tag{3.74}$$

of dimension $n_\mathrm{c} \times f$ for the forward model, the QR decomposition follows as

$$\boldsymbol{C}^\mathrm{T} = \begin{bmatrix} \boldsymbol{Q}_\mathrm{c} & \boldsymbol{J}_{\mathrm{c,q}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}_\mathrm{c} \\ \boldsymbol{0} \end{bmatrix} = \boldsymbol{Q}_\mathrm{c} \boldsymbol{R}_\mathrm{c}. \tag{3.75}$$

Besides $\boldsymbol{J}_{\mathrm{c,q}} \in \mathbb{R}^{f \times f_\mathrm{i}}$, this also gives $\boldsymbol{Q}_\mathrm{c} \in \mathbb{R}^{f \times n_\mathrm{c}}$ and the square matrix $\boldsymbol{R}_\mathrm{c} \in \mathbb{R}^{n_\mathrm{c} \times n_\mathrm{c}}$. Moreover, the independent accelerations $\ddot{\boldsymbol{q}}_{\mathrm{i,q}}$, which have no direct physical meaning, need to be transformed back to the accelerations $\ddot{\boldsymbol{q}}$ in chain coordinates. Since here

$$\boldsymbol{J}_{\ell,\mathrm{q}} = \boldsymbol{J}_{\mathrm{r,q}} = \boldsymbol{J}_{\mathrm{c,q}} \tag{3.76}$$

holds one obtains in a similar way as for Equation (3.63b) the relation in chain coordinates

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}_{\mathrm{c,q}} \left(\boldsymbol{J}_{\mathrm{c,q}}^\mathrm{T} \boldsymbol{M} \boldsymbol{J}_{\mathrm{c,q}} \right)^{-1} \boldsymbol{J}_{\mathrm{c,q}}^\mathrm{T} \left(\boldsymbol{f} + \boldsymbol{B}\boldsymbol{u} - \boldsymbol{M}\boldsymbol{\theta}''_{\mathrm{c,q}} \right) + \boldsymbol{\theta}''_{\mathrm{c,q}}, \tag{3.77}$$

with $\boldsymbol{\theta}''_{\mathrm{c,q}} = -\boldsymbol{Q}_\mathrm{c} \boldsymbol{R}_\mathrm{c}^{-\mathrm{T}} \boldsymbol{c}''$. Equation (3.77) is used to transform the control inputs of the LQR in Section 6.2 from currents or forces to actuator motions. Via a subsequent actuator cascade control this allows to reduce the influence of disturbances such as friction. Since the transformation needs to be done in real-time a QR decomposition is recommended. It prevents possible singularities within $\boldsymbol{C}_\mathrm{d}$, which could occur through manually selecting the independent coordinates.

3.5.4 Forward Model with Kinematic Inputs

An alternative formulation of the forward dynamics follows when the actuator dynamics is neglected and the actuator kinematics $\boldsymbol{q}_\mathrm{a}$, $\dot{\boldsymbol{q}}_\mathrm{a}$ and $\ddot{\boldsymbol{q}}_\mathrm{a}$ is used as control input. This formulation is here denoted as *kinematic inputs*. Since then the actuated coordinates $\boldsymbol{q}_\mathrm{a}$ are no degrees of freedom anymore, the equations of motion (3.22) for the free end-effector case reduce to

$$\underbrace{\begin{bmatrix} \boldsymbol{M}_\mathrm{ee} & \boldsymbol{M}_\mathrm{eu} \\ \boldsymbol{M}_\mathrm{eu}^\mathrm{T} & \boldsymbol{M}_\mathrm{uu} \end{bmatrix}}_{\boldsymbol{M}_\mathrm{k}(\boldsymbol{q})} \underbrace{\begin{bmatrix} \ddot{\boldsymbol{q}}_\mathrm{e} \\ \ddot{\boldsymbol{q}}_\mathrm{u} \end{bmatrix}}_{\ddot{\boldsymbol{q}}_\mathrm{k}} = \underbrace{\begin{bmatrix} \boldsymbol{f}_\mathrm{e} \\ \boldsymbol{f}_\mathrm{u} \end{bmatrix}}_{\boldsymbol{f}_\mathrm{k}(\boldsymbol{q},\dot{\boldsymbol{q}})} + \underbrace{\begin{bmatrix} -\boldsymbol{M}_\mathrm{ea} \\ -\boldsymbol{M}_\mathrm{au}^\mathrm{T} \end{bmatrix}}_{\boldsymbol{B}_\mathrm{k}(\boldsymbol{q})} \ddot{\boldsymbol{q}}_\mathrm{a} + \underbrace{\begin{bmatrix} \boldsymbol{C}_\mathrm{e}^\mathrm{T} \\ \boldsymbol{C}_\mathrm{u}^\mathrm{T} \end{bmatrix}}_{\boldsymbol{C}_\mathrm{k}^\mathrm{T}(\boldsymbol{q})} \boldsymbol{\lambda}, \tag{3.78a}$$

$$\boldsymbol{c}(\boldsymbol{q}) = \boldsymbol{0}. \tag{3.78b}$$

Here, it is assumed that the coordinates are chosen such that the matrix $\boldsymbol{B}$ of Equation (3.22a) only has non-zero entries in the rows of the actuated accelerations $\ddot{\boldsymbol{q}}_\mathrm{a}$. This means that the control input $\boldsymbol{u}$ on force level completely vanishes

within Equation (3.78a) and instead the kinematics $\boldsymbol{q}_\mathrm{a}$, $\dot{\boldsymbol{q}}_\mathrm{a}$ and $\ddot{\boldsymbol{q}}_\mathrm{a}$ is treated as input. Thus, the system dynamics (3.78a) is represented only by $f_\mathrm{k} = f - f_\mathrm{a}$ instead of f equations. These are given in the coordinates $\boldsymbol{q}_\mathrm{k} \in \mathbb{R}^{f_\mathrm{k}}$.
The fact that Equation (3.78) does not need a model of the actuator dynamics can be highly advantageous since it is often difficult to describe this dynamics accurately, e.g., due to friction effects. Consequently, to compare the system responses of a real system and a forward model with kinematic inputs one can simply apply the experimentally measured actuator motions to the forward model instead of forces and torques. This can significantly improve the match of both responses.

Since the coordinates $\boldsymbol{q}_\mathrm{a}$ are no degrees of freedom anymore, the time derivatives of the n_c geometric constraints $\boldsymbol{c} = \boldsymbol{0}$ are rearranged according to

$$\dot{\boldsymbol{c}} = \underbrace{\frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}_\mathrm{k}}}_{\boldsymbol{C}_\mathrm{k}} \dot{\boldsymbol{q}}_\mathrm{k} + \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}_\mathrm{a}} \dot{\boldsymbol{q}}_\mathrm{a} + \frac{\partial \boldsymbol{c}}{\partial t} = \boldsymbol{C}_\mathrm{k} \dot{\boldsymbol{q}}_\mathrm{k} + \boldsymbol{c}'_\mathrm{k} = \boldsymbol{0}, \tag{3.79a}$$

$$\ddot{\boldsymbol{c}} = \boldsymbol{C}_\mathrm{k} \ddot{\boldsymbol{q}}_\mathrm{k} + \dot{\boldsymbol{C}}_\mathrm{k} \dot{\boldsymbol{q}}_\mathrm{k} + \dot{\boldsymbol{c}}'_\mathrm{k} = \boldsymbol{C}_\mathrm{k} \ddot{\boldsymbol{q}}_\mathrm{k} + \boldsymbol{c}''_\mathrm{k} = \boldsymbol{0}. \tag{3.79b}$$

The actuator velocities $\dot{\boldsymbol{q}}_\mathrm{a}$ are now included within $\boldsymbol{c}'_\mathrm{k}$. Therefore, $\boldsymbol{c}'_\mathrm{k} \neq \boldsymbol{0}$ as opposed to Equation (3.42a) where $\boldsymbol{c}' = \boldsymbol{0}$. Similarly, the actuator accelerations $\ddot{\boldsymbol{q}}_\mathrm{a}$ are incorporated within $\boldsymbol{c}''_\mathrm{k}$.

3.5.4.1 Coordinate Partitioning

The independent coordinates $\boldsymbol{q}_\mathrm{i} \in \mathbb{R}^{f_\mathrm{i}}$ and the dependent coordinates $\boldsymbol{q}_\mathrm{d} \in \mathbb{R}^{f_\mathrm{d}}$ partition now $\boldsymbol{q}_\mathrm{k}$ and are therefore of dimension $f_\mathrm{i} = f_\mathrm{k} - n_\mathrm{c}$ and $f_\mathrm{d} = n_\mathrm{c}$, respectively. Equation (3.79b) can then be solved for the dependent accelerations $\ddot{\boldsymbol{q}}_\mathrm{d}$. Similar to Equation (3.52), this yields

$$\ddot{\boldsymbol{q}}_\mathrm{k} = \begin{bmatrix} \ddot{\boldsymbol{q}}_\mathrm{i} \\ \ddot{\boldsymbol{q}}_\mathrm{d} \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{I}_{f_\mathrm{i}} \\ -\boldsymbol{C}_{\mathrm{k,d}}^{-1} \boldsymbol{C}_{\mathrm{k,i}} \end{bmatrix}}_{\boldsymbol{J}_\mathrm{k}} \ddot{\boldsymbol{q}}_\mathrm{i} + \underbrace{\begin{bmatrix} \boldsymbol{0} \\ -\boldsymbol{C}_{\mathrm{k,d}}^{-1} \boldsymbol{c}''_\mathrm{k} \end{bmatrix}}_{\boldsymbol{\theta}''_\mathrm{k}}. \tag{3.80}$$

The matrices $\boldsymbol{C}_{\mathrm{k,i}}$ and $\boldsymbol{C}_{\mathrm{k,d}}$ are the Jacobians of $\boldsymbol{c}$ with respect to $\boldsymbol{q}_\mathrm{i}$ and $\boldsymbol{q}_\mathrm{d}$, compare Equation (3.71), which are here related to the coordinates $\boldsymbol{q}_\mathrm{k}$. A left multiplication of Equation (3.78a) with the transposed projection matrix $\boldsymbol{J}_\mathrm{k} \in \mathbb{R}^{f_\mathrm{k} \times f_\mathrm{i}}$ and replacing $\ddot{\boldsymbol{q}}_\mathrm{k}$ with the right side of Equation (3.80) yields the equation of motion in minimal form as

$$\boldsymbol{J}_\mathrm{k}^\mathrm{T} \boldsymbol{M}_\mathrm{k} \boldsymbol{J}_\mathrm{k} \ddot{\boldsymbol{q}}_\mathrm{i} = \boldsymbol{J}_\mathrm{k}^\mathrm{T} \left(\boldsymbol{f}_\mathrm{k} + \boldsymbol{B}_\mathrm{k} \ddot{\boldsymbol{q}}_\mathrm{a} - \boldsymbol{M}_\mathrm{k} \boldsymbol{\theta}''_\mathrm{k} \right). \tag{3.81}$$

The resulting forward model is shown in Figure 3.20. For systems where the elastic coordinates $\boldsymbol{q}_\mathrm{e}$ can be selected as independent coordinates $\boldsymbol{q}_\mathrm{i}$, which is usually the case for flexible link robots, this formulation allows to consider only

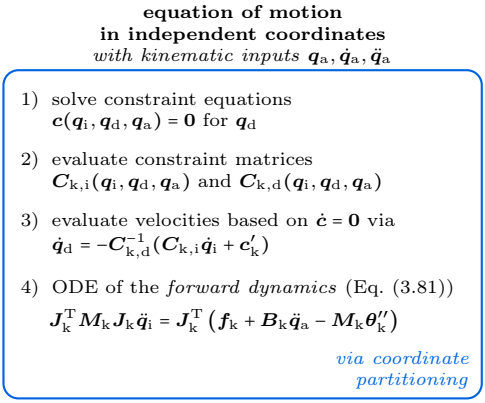


Figure 3.20: Forward model for flexible multibody systems with geometric constraints in independent coordinates (minimal form) with kinematic inputs.

the deformation dynamics. This is especially advantageous for the modal damping controller introduced in Section 6.3 as it enables to focus on the eigenmodes due to the link elasticity to actively increase their damping.

3.5.4.2 QR Decomposition

For kinematic inputs the QR decomposition is applied to the transposed Jacobian matrix $\boldsymbol{C}_\mathrm{k} \in \mathbb{R}^{n_\mathrm{c} \times f_\mathrm{k}}$ of the geometric constraints, which results in

$$\boldsymbol{C}_\mathrm{k}^\mathrm{T} = \begin{bmatrix} \boldsymbol{Q}_\mathrm{k} & \boldsymbol{J}_{\mathrm{k,q}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}_\mathrm{k} \\ \boldsymbol{0} \end{bmatrix} = \boldsymbol{Q}_\mathrm{k}\boldsymbol{R}_\mathrm{k}. \tag{3.82}$$

This gives $\boldsymbol{Q}_\mathrm{k} \in \mathbb{R}^{f_\mathrm{k} \times n_\mathrm{c}}$ and $\boldsymbol{R}_\mathrm{k} \in \mathbb{R}^{n_\mathrm{c} \times n_\mathrm{c}}$ as well as the projection matrix $\boldsymbol{J}_{\mathrm{k,q}} \in \mathbb{R}^{f_\mathrm{k} \times f_\mathrm{i}}$. Analogously to Section 3.5.1, the equations of motion in minimal form follow as

$$\dot{\boldsymbol{q}}_{\mathrm{i,q}} = \boldsymbol{J}_{\mathrm{k,q}}^\mathrm{T}\big(\dot{\boldsymbol{q}}_\mathrm{k} + \underbrace{\boldsymbol{Q}_\mathrm{k}\boldsymbol{R}_\mathrm{k}^{-\mathrm{T}}\boldsymbol{c}_\mathrm{k}'}_{-\boldsymbol{\theta}_{\mathrm{k,q}}'}\big), \tag{3.83a}$$

$$\ddot{\boldsymbol{q}}_{\mathrm{i,q}} = \left(\boldsymbol{J}_{\mathrm{k,q}}^\mathrm{T}\boldsymbol{M}_\mathrm{k}\boldsymbol{J}_{\mathrm{k,q}}\right)^{-1}\boldsymbol{J}_{\mathrm{k,q}}^\mathrm{T}\big(\boldsymbol{f}_\mathrm{k} + \boldsymbol{B}_\mathrm{k}\ddot{\boldsymbol{q}}_\mathrm{a} + \boldsymbol{M}_\mathrm{k}\underbrace{\boldsymbol{Q}_\mathrm{k}\boldsymbol{R}_\mathrm{k}^{-\mathrm{T}}\boldsymbol{c}_\mathrm{k}''}_{-\boldsymbol{\theta}_{\mathrm{k,q}}''}\big). \tag{3.83b}$$

This formulation is used for the UKF in Section 4.1.3. Due to the kinematic inputs no model of the actuator friction is needed within this UKF. This formulation also reduces the model size, which is beneficial for state estimation being usually required in real-time. Additionally, the QR decomposition is advantageous for such real-time calculations since it prevents singularities when computing the projection matrix $\boldsymbol{J}_{\mathrm{k,q}}$.

Transforming Equation (3.83) back with a formulation analog to Equation (3.61) yields the equations of motion

$$\dot{\boldsymbol{q}}_\mathrm{k} = \boldsymbol{J}_{\mathrm{k,q}} \boldsymbol{J}_{\mathrm{k,q}}^\mathrm{T} \left(\dot{\boldsymbol{q}}_\mathrm{k} - \boldsymbol{\theta}'_{\mathrm{k,q}} \right) + \boldsymbol{\theta}'_{\mathrm{k,q}}, \tag{3.84a}$$

$$\ddot{\boldsymbol{q}}_\mathrm{k} = \boldsymbol{J}_{\mathrm{k,q}} \left(\boldsymbol{J}_{\mathrm{k,q}}^\mathrm{T} \boldsymbol{M}_\mathrm{k} \boldsymbol{J}_{\mathrm{k,q}} \right)^{-1} \boldsymbol{J}_{\mathrm{k,q}}^\mathrm{T} \left(\boldsymbol{f}_\mathrm{k} + \boldsymbol{B}_\mathrm{k} \ddot{\boldsymbol{q}}_\mathrm{a} - \boldsymbol{M}_\mathrm{k} \boldsymbol{\theta}''_{\mathrm{k,q}} \right) + \boldsymbol{\theta}''_{\mathrm{k,q}}. \tag{3.84b}$$

This formulation of the forward model is summarized in Figure 3.21, where $\boldsymbol{x}_\mathrm{k} = [\boldsymbol{q}_\mathrm{k}^\mathrm{T}, \dot{\boldsymbol{q}}_\mathrm{k}^\mathrm{T}]^\mathrm{T}$ holds. It is used for the validation of the model order reduction

$$\textbf{equations of motion } \dot{\boldsymbol{x}}_\mathrm{k} = \boldsymbol{f}_\mathrm{k}^\mathrm{fw}(\boldsymbol{x}_\mathrm{k}, \overbrace{\boldsymbol{q}_\mathrm{a}, \dot{\boldsymbol{q}}_\mathrm{a}, \ddot{\boldsymbol{q}}_\mathrm{a}}^{\textit{kinematic inputs}})$$

ODEs of the *forward dynamics* in state space

$$\frac{\mathrm{d}}{\mathrm{d}t} \begin{bmatrix} \boldsymbol{q}_\mathrm{k} \\ \dot{\boldsymbol{q}}_\mathrm{k} \end{bmatrix} = \begin{bmatrix} \text{Eq. (3.84a)} \\ \text{Eq. (3.84b)} \end{bmatrix}$$

via QR decomposition

Figure 3.21: Forward model for flexible multibody systems with geometric constraints in $\boldsymbol{q}_\mathrm{k}$-coordinates with kinematic inputs.

in Figure 3.9 and for the parameter estimation in Figure 4.9. Especially for the parameter estimation using kinematic inputs ensures a close correspondence between experiments and simulations, since, e.g., no actuator friction needs to be considered.

3.6 Solution of the Inverse Flexible Multibody Model

Solving the various ODE formulations of the forward model introduced in Sections 3.5.3 and 3.5.4 is usually straightforward since the model is stable. This allows to use standard ODE integrators to solve an IVP via forward time integration. Consequently, this section only focuses on the solution of the inverse flexible multibody model.

Such an inverse model typically has a dynamics, which corresponds to the internal dynamics being commonly known from feedback linearization [64]. The existence and stability of the internal dynamics determine how the various ODE formulations of Sections 3.5.1 and 3.5.2 can be solved.

When assuming that the underactuation is due to link flexibility neglecting this flexibility leads to an equivalent rigid system which is fully actuated. A *fully actuated system* can be understood as a system which has an equal number of control inputs and degrees of freedom, where each degree of freedom can be controlled independently. Here, it is also assumed that the same number of tracking outputs is available. Such an equivalent rigid and fully actuated system is typically differentially flat. Then, the inverse is algebraic, i.e., no internal dynamics exists and the system state can be obtained via inverse kinematics, see Section 3.6.1.

If the link flexibilities are not neglected, then the inverse model is usually a dynamical system itself, i.e., the system is not differentially flat. If this internal dynamics is stable classical inversion can be used to obtain bounded results. Here, an IVP needs to be solved via forward time integration, see Section 3.6.2. In Chapter 5, it is shown that both approaches, inverse kinematics and classical inversion, can even be computed in real-time for FLEXOR within experiments.
Nevertheless, if the internal dynamics is unstable classical inversion leads to unbounded results. This is typically the case when tracking the exact end-effector of flexible link robots, i.e., when using $\boldsymbol{y}_{\mathrm{ef}}$ as output $\boldsymbol{y}_{\mathrm{o}}$. The solution process to obtain bounded results is clearly more complicated if the internal dynamics is unstable. For flexible link robots, the unstable internal dynamics usually also includes a stable part. In such a case, trajectory optimization or stable inversion can be used to obtain a bounded but noncausal solution, with pre- and post-actuation phases, which needs to be computed offline. In this research, trajectory optimization is utilized via direct collocation, see Section 3.6.3. Then, the trajectory optimization problem is often easier to construct and to solve than the stable inversion problem, which relies on solving a two-point BVP as explained in Section 3.6.4. To make it more likely that a solution is found for the BVP, the trajectory optimization can be used to provide a suitable initial guess. It should be noted that both methods, trajectory optimization and stable inversion, can also be used for model inversion in the case of a stable internal dynamics. This is, however, usually undesired since both methods are much more involved than classical inversion via forward integration.

3.6.1 Rigid Inverse Kinematics

For a system where the underactuation is due to flexibilities, neglecting these flexibilities yields an equivalent rigid system which is fully actuated. In such a case, typically no internal dynamics exists and the inverse is purely algebraic. Then, the projected parts of, e.g., Equation (3.57) vanish since no independent coordinates $\boldsymbol{q}_{\mathrm{i}}$ exist, which gives $\dot{\boldsymbol{q}} = \boldsymbol{\theta}'$ and $\ddot{\boldsymbol{q}} = \boldsymbol{\theta}''$. Instead of time integration the model inversion problem can also be solved by root finding of the constraint equations (3.41b) and (3.41c) on position level as well as with the corresponding first time derivatives (3.49). This corresponds to inverse kinematics, where the rigid system state $\boldsymbol{x}_{\mathrm{r}}$, i.e., $\boldsymbol{x}$ without elastic terms, depends only on the desired output $\boldsymbol{y}_{\mathrm{d}}(t)$ and its first time derivative for an output on position level.
The model inversion via inverse kinematics is summarized in Figure 3.22. Here, $\boldsymbol{f}_{\mathrm{r}}^{\mathrm{iv}}$ indicates that a rigid inverse model is considered. This formulation represents the classical rigid model inversion approach, which will be used mainly as reference throughout Chapter 5. Neglecting the flexibilities in the model inversion usually performs unsatisfyingly when being utilized to control flexible link robots. Nevertheless, in Section 5.1.2 it is shown that a rigid model inversion can be advantageous when manually guiding a flexible link robot.

state $\boldsymbol{x}_\mathrm{r} = \boldsymbol{f}_\mathrm{r}^\mathrm{iv}(t)$ of the inverse kinematics of the fully actuated equivalent rigid system

1) solve constraint equations $\boldsymbol{c}(\boldsymbol{q}_\mathrm{d}) = \boldsymbol{0}$ and $\boldsymbol{s}(t, \boldsymbol{q}_\mathrm{d}) = \boldsymbol{0}$ for $\boldsymbol{q}_\mathrm{d}$

2) evaluate constraint matrix $\boldsymbol{\Gamma}_\mathrm{d}(\boldsymbol{q}_\mathrm{d})$

3) evaluate velocities based on $\dot{\boldsymbol{\gamma}} = \boldsymbol{0}$ via $\dot{\boldsymbol{q}}_\mathrm{d} = -\boldsymbol{\Gamma}_\mathrm{d}^{-1}\boldsymbol{\gamma}'$

4) assemble state of the *inverse kinematics* of the fully actuated system

$$\boldsymbol{x}_\mathrm{r} = \begin{bmatrix} \boldsymbol{q}_\mathrm{d} \\ \dot{\boldsymbol{q}}_\mathrm{d} \end{bmatrix}$$

Figure 3.22: Inverse kinematics for fully actuated multibody systems with geometric and servo constraints.

3.6.2 Classical Inversion

When the flexibilities are not neglected, the inverse model of flexible link robots usually has a dynamics. If this internal dynamics is stable, classical inversion, which originates from Hirschorn [93, 94], can be used to obtain a bounded solution. A definition of the classical inversion problem can be found in [68, p. 79]. It basically corresponds to an IVP for the inverse model to be solved by forward time integration. This leads to a so-called *causal solution*, which is a solution that does not have a pre-actuation phase. If the internal dynamics is unstable, classical inversion still yields a causal solution, which results, however, in an unbounded state $\boldsymbol{x}$ and input $\boldsymbol{u}$. Thus, it cannot be meaningfully utilized.
Since for flexible link robots the internal dynamics is typically unstable when tracking the exact end-effector $\boldsymbol{y}_\mathrm{ef}$, such as for FLEXOR, three approaches are discussed in Section 5.2.1 to render the internal dynamics stable. These approaches allow to apply classical inversion to obtain a bounded and causal solution online via forward integration of an IVP.
In this research, classical inversion is utilized for the ODE formulations in chain coordinates $\dot{\boldsymbol{x}} = \boldsymbol{f}_\mathrm{c}^\mathrm{iv}(t, \boldsymbol{x})$ and $\dot{\boldsymbol{x}} = \bar{\boldsymbol{f}}_\mathrm{c}^\mathrm{iv}(t, \boldsymbol{x}, \bar{\boldsymbol{u}}_\mathrm{r})$ introduced in Sections 3.5.1.2 and 3.5.2.1. In the case of considering only the first two bending eigenmodes of FLEXOR, these formulations are computationally efficient enough to even enable model inversion in real-time, see, e.g., Section 5.2.1.4. There, details can be found on the utilized integration method for classical inversion, which is exclusively the explicit fourth-order Runge–Kutta method within experiments.

3.6.3 Inversion via Trajectory Optimization

As mentioned before, tracking of the exact end-effector $\boldsymbol{y}_\mathrm{ef}$ of a flexible link robot usually results in an unstable internal dynamics. Consequently, classical inver-

sion via forward integration leads to unbounded solutions. Such a problem can however be solved with bounded results by trajectory optimization techniques. According to Kelly [95], most trajectory optimization techniques can be classified into direct or indirect methods. The indirect methods optimize first and then discretize, while the direct methods discretize first and then optimize. Although indirect methods tend to be more accurate, they are typically more difficult to construct and solve, e.g., a more accurate initialization might be necessary. Therefore, only direct methods are considered here, since on the one hand an easier construction is preferred to ensure a straightforward adaptation to new problem formulations. On the other hand, solutions based on an inaccurate initial guess shall be found, e.g., with zero elastic deformation if a planar case without contact is considered, since often almost no information exists on the trajectories of the elastic coordinates.

Trajectory optimization problems for flexible and underactuated rigid systems, solved with direct methods, have been considered before in the literature. For instance, Springer et al. [96] investigate time-optimal trajectory planning for a flexible link robot with flexible joints. They use a reduced lumped parameter model with a flat output and conduct experiments. As opposed to this, in the presented research trajectory optimization with fixed time is considered in order to solve end-effector trajectory tracking problems with internal dynamics. This has also been discussed by Seifried et al. [97] for an underactuated rigid manipulator with two passive joints. Bastos et al. [18] deal as well with end-effector trajectory tracking for underactuated rigid manipulators based on optimization via a direct method and compare it to stable inversion via a two-point BVP. Likewise, Lismonde et al. [98] treat an optimization problem for end-effector trajectory tracking of a 3D flexible link robot, being solved via a direct method. In contrast to [18, 97, 98], which consider serial robots without experiments, the presented research applies end-effector trajectory tracking via trajectory optimization to the flexible link parallel robot FLEXOR within experiments. Here, geometric constraints and servo constraints are incorporated within transformations to ODEs. By using only the remaining internal dynamics for the independent state $\boldsymbol{x}_\mathrm{i}$ within the optimization, the number of design variables is significantly reduced. Besides, when only tracking the end-effector position but not its rotation, it is shown that the optimization via a direct method can be used to obtain a causal solution with reduced elastic link deflections.

This section gives a compact overview on the utilized optimization method. For a more detailed discussion on trajectory optimization it is pointed to the review paper by Kelly [95]. The experimental application to FLEXOR is then discussed in Section 5.3.1.

3.6.3.1 Model Inversion

First, model inversion for end-effector trajectory tracking is considered for the non-redundant case having the same number of control inputs and servo constraints, i.e., $f_\mathrm{a} = n_\mathrm{s}$. This allows to cancel all control inputs to obtain the ODE formulation of the inverse model from Figure 3.16. Thus, the system dynamics of the inverse model, which corresponds to the internal dynamics, has the form

$$\dot{\boldsymbol{x}}_\mathrm{i} = \boldsymbol{f}_\mathrm{i}^\mathrm{iv}(t, \boldsymbol{x}_\mathrm{i}). \tag{3.85}$$

Assuming that the underactuation is due to link flexibility, the independent state can be chosen as $\boldsymbol{x}_\mathrm{i} = [\boldsymbol{q}_\mathrm{i}^\mathrm{T}, \dot{\boldsymbol{q}}_\mathrm{i}^\mathrm{T}]^\mathrm{T} = [\boldsymbol{q}_\mathrm{e}^\mathrm{T}, \dot{\boldsymbol{q}}_\mathrm{e}^\mathrm{T}]^\mathrm{T}$. The considered trajectory optimization problem is then stated as

$$\min_{\boldsymbol{x}_\mathrm{i}} J(\boldsymbol{x}_\mathrm{i}) = \min_{\boldsymbol{x}_\mathrm{i}} \int_{t_0}^{t_\mathrm{F}} \underbrace{\boldsymbol{q}_\mathrm{e}^\mathrm{T}(t) \boldsymbol{W}_\mathrm{e} \boldsymbol{q}_\mathrm{e}(t)}_{w(t)} \, \mathrm{d}t \tag{3.86a}$$

$$\text{subject to } \dot{\boldsymbol{x}}_\mathrm{i} = \boldsymbol{f}_\mathrm{i}^\mathrm{iv}(t, \boldsymbol{x}_\mathrm{i}). \tag{3.86b}$$

Thus, within the objective function J only the so-called *Lagrange term* is used. Similar to [97], $\boldsymbol{W}_\mathrm{e} \in \mathbb{R}^{f_\mathrm{e} \times f_\mathrm{e}}$ is a diagonal matrix which here weights the elastic coordinates $\boldsymbol{q}_\mathrm{e}$. Since all actuators are needed to enforce the servo constraints, no additional degree of freedom exists to actually minimize the elastic deformations within J. This objective function is in fact only aimed at finding a bounded solution. Then, no additional bounds or boundary conditions need to be enforced on the states. This can be regarded as implementation advantage compared to stable inversion via solving a BVP, since no complicated boundary conditions need to be established [18].
In Section 5.3.1, the trajectory optimization problem of Equation (3.86) is applied to the standard 2D case of Flexor with three control inputs and three servo constraints, i.e., $f_\mathrm{a} = n_\mathrm{s} = 3$. In this case, the independent state $\boldsymbol{x}_\mathrm{i} = [q_{\mathrm{e},2}, q_{\mathrm{e},3}, \dot{q}_{\mathrm{e},2}, \dot{q}_{\mathrm{e},3}]^\mathrm{T}$ relies on the elastic coordinates of the first bending eigenmode of link 2 and link 3, respectively. Within the objective function, they are weighted equally according to

$$w(t) = q_{\mathrm{e},2}^2(t) + q_{\mathrm{e},3}^2(t). \tag{3.87}$$

In this research, the Hermite–Simpson direct collocation method is used to discretize trajectory optimization problems. Thus, Equation (3.86) is converted into a nonlinear programming problem. The problem is discretized in time into N segments. This leads to $2N + 1$ collocation points, i.e., grid points as each segment midpoint is considered within the Hermite–Simpson method. As a result, the time points $t^{(0)}, t^{(0+1/2)}, \ldots, t^{(N)}$ are obtained as well as a finite set of design

variables being summarized in the vector

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{x}_\mathrm{i}^{(0)} \\ \boldsymbol{x}_\mathrm{i}^{(0+1/2)} \\ \vdots \\ \boldsymbol{x}_\mathrm{i}^{(N)} \end{bmatrix}, \tag{3.88}$$

which here consists of the discretized independent state. Since a fixed initial time t_0 and a fixed final time t_F are used, they do not belong to the design variables. The design variables are used to set up the so-called *collocation constraints* for the system dynamics

$$\boldsymbol{x}_\mathrm{i}^{(k+1)} - \boldsymbol{x}_\mathrm{i}^{(k)} = \frac{h^{(k)}}{6}\left(\boldsymbol{f}_\mathrm{i}^{\mathrm{iv}(k)} + 4\boldsymbol{f}_\mathrm{i}^{\mathrm{iv}(k+1/2)} + \boldsymbol{f}_\mathrm{i}^{\mathrm{iv}(k+1)}\right), \tag{3.89}$$

with $h^{(k)} = t^{(k+1)} - t^{(k)}$ and $k = 0, 1, \ldots, (N-1)$. Since the collocation constraints are used here in the so-called *separated form*, the introduced design variables $\boldsymbol{x}_\mathrm{i}^{(k+1/2)}$ occur within $\boldsymbol{f}_\mathrm{i}^{\mathrm{iv}(k+1/2)}$ at the segment midpoints. These midpoints are included by interpolation leading to

$$\boldsymbol{x}_\mathrm{i}^{(k+1/2)} = \frac{1}{2}\left(\boldsymbol{x}_\mathrm{i}^{(k)} + \boldsymbol{x}_\mathrm{i}^{(k+1)}\right) + \frac{h^{(k)}}{8}\left(\boldsymbol{f}_\mathrm{i}^{\mathrm{iv}(k)} - \boldsymbol{f}_\mathrm{i}^{\mathrm{iv}(k+1)}\right). \tag{3.90}$$

The collocation constraints of Equations (3.89) and (3.90) are enforced on each trajectory segment. Similarly, the objective function is discretized according to the formula

$$J = \int_{t_0}^{t_\mathrm{F}} w(t)\,\mathrm{d}t \approx \sum_{k=0}^{N-1} \frac{h^{(k)}}{6}\left(w^{(k)} + 4w^{(k+1/2)} + w^{(k+1)}\right). \tag{3.91}$$

Further details on the utilized Hermite–Simpson collocation method can be found in [95, 99].

To solve the discretized trajectory optimization problem MATLAB's nonlinear programming solver `fmincon` is utilized with the interior-point algorithm. Here, first a coarse mesh with a small number of collocation points is used to obtain an easier problem but also an inaccurate solution. Then, the mesh is refined within multiple steps to obtain a more accurate solution. This refinement strategy can increase the efficiency of the solution process. Due to the considered unstable internal dynamics, the solution is noncausal with a pre-actuation and a post-actuation phase. To account for these phases, the initial time t_0 and the final time t_F need to be selected accordingly.

3.6.3.2 Model Inversion with Kinematic Redundancy

When obtaining a bounded solution to the considered end-effector trajectory tracking problem with unstable internal dynamics, pre-actuation and post-actuation phases occur as mentioned in Section 3.6.3.1. This might however be undesired

within an application. Therefore, a method is now presented to obtain a bounded and causal solution which means that no pre-actuation occurs and additionally no post-actuation shall arise.
Graichen et al. [100] also discuss a technique to obtain causal transitions between stationary set points for an unstable internal dynamics by using free parameters to design the desired output trajectory. This is formulated as a two-point BVP. Seifried [68, chapter 4.3] investigates the same idea of parameterizing the output trajectory and solving a two-point BVP, but applies it to a manipulator with a passive joint and kinematic redundancy. The redundancy is used to introduce free design parameters to obtain a bounded and causal solution without post-actuation. Nevertheless, solving a BVP can be difficult. Also, suitable free design parameters need to be found which render the BVP solvable. Thus, applying these concepts to a completely different system is typically very involved.
Consequently, trajectory optimization via direct collocation is used in the following, which is typically easier to formulate and solve. Here, the kinematic redundancy due to applying less servo constraints than available independent control inputs, i.e., $f_\mathrm{a} > n_\mathrm{s}$, is used as a design degree of freedom to obtain a causal solution. Additionally, this solution shall have no post-actuation. Therefore, steady-state boundary conditions $\boldsymbol{b}_\mathrm{c} = \boldsymbol{0}$ are enforced and the trajectory optimization problem formulation from Equation (3.86) changes to

$$\min_{\boldsymbol{x}_\mathrm{i},\bar{\boldsymbol{u}}_\mathrm{r}} J(\boldsymbol{x}_\mathrm{i},\bar{\boldsymbol{u}}_\mathrm{r}) = \min_{\boldsymbol{x}_\mathrm{i},\bar{\boldsymbol{u}}_\mathrm{r}} \int_{t_0}^{t_\mathrm{F}} \underbrace{\boldsymbol{q}_\mathrm{e}^\mathrm{T}(t)\boldsymbol{W}_\mathrm{e}\boldsymbol{q}_\mathrm{e}(t) + \bar{\boldsymbol{u}}_\mathrm{r}^\mathrm{T}(t)\boldsymbol{W}_\mathrm{u}\bar{\boldsymbol{u}}_\mathrm{r}(t)}_{w(t)} \,\mathrm{d}t \tag{3.92a}$$

$$\text{subject to } \dot{\boldsymbol{x}}_\mathrm{i} = \bar{\boldsymbol{f}}_\mathrm{i}^\mathrm{iv}(t,\boldsymbol{x}_\mathrm{i},\bar{\boldsymbol{u}}_\mathrm{r}), \tag{3.92b}$$

$$\boldsymbol{b}_\mathrm{c}(\boldsymbol{x}_\mathrm{i}(t_0),\boldsymbol{x}_\mathrm{i}(t_\mathrm{F})) = \boldsymbol{0}, \tag{3.92c}$$

$$\boldsymbol{b}_\ell \le \boldsymbol{x}_\mathrm{i}(t) \le \boldsymbol{b}_\mathrm{u}. \tag{3.92d}$$

Here, $\boldsymbol{W}_\mathrm{u} \in \mathbb{R}^{(f_\mathrm{a}-n_\mathrm{s})\times(f_\mathrm{a}-n_\mathrm{s})}$ is also a diagonal weighting matrix. For this redundant case, the inverse model $\dot{\boldsymbol{x}}_\mathrm{i} = \bar{\boldsymbol{f}}_\mathrm{i}^\mathrm{iv}(t,\boldsymbol{x}_\mathrm{i},\bar{\boldsymbol{u}}_\mathrm{r})$ in ODE form is obtained as described in Figure 3.18.
In Section 5.3.1, Equation (3.92) is applied to a 2D scenario without contact in which the end-effector rotation of Flexor is not tracked. This means that only two servo constraints are enforced although three actuators are available. This could be utilized to handle rotational symmetric objects where the end-effector rotation is not of interest but only its position. The independent state is then selected as $\boldsymbol{x}_\mathrm{i} = [q_{\mathrm{e},2},\, q_{\mathrm{e},3},\, \gamma,\, \dot{q}_{\mathrm{e},2},\, \dot{q}_{\mathrm{e},3},\, \dot{\gamma}]^\mathrm{T}$ where the rotary motor angle γ is now also treated as independent coordinate. The internal dynamics further depends on $\bar{\boldsymbol{u}}_\mathrm{r} = u_\gamma$ since the utilized transformations to ODEs based on two servo constraints only allow to cancel two control inputs being here u_a and u_b. This means that an optimal control problem is considered. The boundary conditions $\boldsymbol{b}_\mathrm{c} = \boldsymbol{0}$ ensure that $q_{\mathrm{e},2}$, $q_{\mathrm{e},3}$, $\dot{q}_{\mathrm{e},2}$, $\dot{q}_{\mathrm{e},3}$ and $\dot{\gamma}$ are zero at initial time t_0 as well as at final time t_F. Here, these times match exactly the start and end time of the interval in which the desired end-effector trajectory changes. The lower and upper path

bounds in Equation (3.92d) can additionally support that a feasible solution is found. For FLEXOR the bounds

$$-\pi/2 \leq \gamma(t) \leq \pi/2 \tag{3.93}$$

are used to guarantee that the rotary motor angle γ stays in a reasonable range. It turns out that by not further restricting the start and end values of γ, smoother solutions are possible. Compared to the non-redundant case of Section 3.6.3.1, the objective function J of FLEXOR is extended by the motor current u_γ according to

$$w(t) = q_{\mathrm{e},2}^2(t) + q_{\mathrm{e},3}^2(t) + 0.5u_\gamma^2(t), \tag{3.94}$$

in order to yield reasonable inputs. Most importantly, due to the introduced redundancy the elastic deformations within J will now in fact be minimized. Thus, besides obtaining a causal solution without post-actuation for the considered end-effector trajectory tracking problem, the link deformations are reduced simultaneously.
Next, the Hermite-Simpson direct collocation method is used again for discretization, see Section 3.6.3.1. The vector of design variables now also includes the additional control inputs due to the redundancy and reads

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{x}_{\mathrm{i}}^{(0)} \\ \boldsymbol{x}_{\mathrm{i}}^{(0+1/2)} \\ \vdots \\ \boldsymbol{x}_{\mathrm{i}}^{(N)} \\ \bar{\boldsymbol{u}}_{\mathrm{r}}^{(0)} \\ \bar{\boldsymbol{u}}_{\mathrm{r}}^{(0+1/2)} \\ \vdots \\ \bar{\boldsymbol{u}}_{\mathrm{r}}^{(N)} \end{bmatrix}. \tag{3.95}$$

Experimental results are presented in Section 5.3.1, which show the successful realization of the proposed approach.

In summary, trajectory optimization via direct collocation can be applied to a variety of interesting problems within flexible link robotics with only small additional implementation effort. Still, since the computational cost of such an optimization can be relatively high, it is especially advantageous for repetitive tasks.

3.6.4 Stable Inversion

Alternatively to trajectory optimization, the so-called *stable inversion*, originating from Chen and Paden [101] and Devasia et al. [102], can be utilized to obtain a noncausal but bounded solution in the case of an unstable internal dynamics which also includes a stable part. A definition of stable inversion is given, e.g., in [101], which does not demand that the initial values are met but the results need

to be bounded. Thus, no IVP needs to be solved. Instead, stable inversion requires that a typically much more complicated two-point BVP is solved. This means that one cannot predefine the initial state but for the considered system type a pre-actuation is needed to drive the system to the required state while keeping the output constant. Additionally, a post-actuation occurs at the trajectory end. For the stable inversion approach it is necessary that the equilibrium points of the internal dynamics are hyperbolic, i.e., the linearization has no eigenvalues on the imaginary axis. Also, the boundary conditions of the BVP demand that the internal dynamics starts on an unstable manifold and ends on a stable manifold. These stable and unstable invariant manifolds are the nonlinear analogs of the corresponding stable and unstable eigenspaces, which are tangent to these manifolds at the considered equilibrium [71]. Since these manifolds are typically difficult to handle they are now locally approximated by the eigenspaces which has also been done by Zhao and Chen [103]. The eigenspaces are obtained by a linearization of the internal dynamics around the corresponding hyperbolic equilibrium points at the trajectory start and end, where the output is held constant via servo constraints. This linearization of the zero dynamics, i.e., the internal dynamics with constant output $\boldsymbol{y}_\mathrm{o}$, can be performed, e.g., by applying finite differences to the state-space representation $\dot{\boldsymbol{x}}_\mathrm{i} = \boldsymbol{f}_\mathrm{i}^\mathrm{iv}(\boldsymbol{x}_\mathrm{i})$ or $\dot{\boldsymbol{x}} = \boldsymbol{f}_\mathrm{c}^\mathrm{iv}(\boldsymbol{x})$ introduced in Section 3.5.1. This allows to obtain the partial derivative with respect to $\boldsymbol{x}_\mathrm{i}$ or $\boldsymbol{x}$ giving the corresponding constant system matrix $\boldsymbol{A}_\eta$ of the linearized zero dynamics. It can be consequently written as autonomous system

$$\dot{\boldsymbol{\eta}} = \boldsymbol{A}_\eta \boldsymbol{\eta}. \tag{3.96}$$

Then, the eigenvectors and eigenvalues of $\boldsymbol{A}_\eta$ are calculated. Subsequently, the possibly complex diagonal form with the diagonal matrix of the complex eigenvalues is transformed to real block-diagonal form, e.g., via MATLAB's `cdf2rdf` command. This yields the real block-diagonal matrix $\boldsymbol{D}_\eta$, which has the same eigenvalues as $\boldsymbol{A}_\eta$. The corresponding real transformation matrix $\boldsymbol{T}$ fulfills

$$\boldsymbol{A}_\eta = \boldsymbol{T}\boldsymbol{D}_\eta\boldsymbol{T}^{-1}. \tag{3.97}$$

The $\boldsymbol{\eta}$-coordinates of the linearized zero dynamics can then be transformed to new coordinates $\boldsymbol{\varphi} = \boldsymbol{T}^{-1}\boldsymbol{\eta}$, leading to

$$\dot{\boldsymbol{\varphi}} = \boldsymbol{D}_\eta \boldsymbol{\varphi}. \tag{3.98}$$

Due to the block-diagonal form of $\boldsymbol{D}_\eta$, the stable and unstable dynamics are decoupled in this equation. With $\boldsymbol{T}_\mathrm{s}$ being the rows of $\boldsymbol{T}^{-1}$ which are related to the stable eigenvalues and $\boldsymbol{T}_\mathrm{u}$ being the rows related to the unstable eigenvalues, one can construct the boundary conditions. At the trajectory start time $t = t_0$ the boundary condition

$$\boldsymbol{T}_\mathrm{s}(t_0)\boldsymbol{\eta}(t_0) = \boldsymbol{0} \tag{3.99}$$

keeps the internal dynamics on the approximated unstable manifold and

$$\boldsymbol{T}_\mathrm{u}(t_\mathrm{F})\boldsymbol{\eta}(t_\mathrm{F}) = \boldsymbol{0} \tag{3.100}$$

is used for the approximated stable manifold at the trajectory end time $t = t_\mathrm{F}$ [68, 71, 104]. As a result of the real block-diagonal form these boundary conditions are expressed with real values.
As mentioned before, the eigenvalues of the linearized zero dynamics at the equilibrium points need to be off the imaginary axis. If one applies the discussed approach to a formulation in chain coordinates the rows of $\boldsymbol{T}^{-1}$ related to the additional $2f_\mathrm{d}$ zero eigenvalues, introduced through the constraint equations, need to be removed. To make up for the now missing boundary conditions, e.g., $\boldsymbol{c} = \boldsymbol{0}$ and $\boldsymbol{s} = \boldsymbol{0}$ might be used at one boundary and $\dot{\boldsymbol{c}} = \boldsymbol{0}$ and $\dot{\boldsymbol{s}} = \boldsymbol{0}$ at the other boundary. Nevertheless, in this research exclusively the formulation in independent coordinates $\dot{\boldsymbol{x}}_\mathrm{i} = \boldsymbol{f}_\mathrm{i}^\mathrm{iv}(t, \boldsymbol{x}_\mathrm{i})$ is used for stable inversion, which makes the problem more compact.
Finally, the noncausal solution for the control inputs and state trajectories can be obtained by solving the two-point BVP, e.g., with MATLAB's `bvp5c` solver. To make it more likely that a solution to the BVP is found, an appropriate initial guess is necessary. Using the solution from a trajectory optimization as initial guess is often an effective approach. Alternatively, output redefinition, introduced in Section 5.2.1.1, can be used to adapt the internal dynamics which then might be easier to solve. When a solution is found it can be implemented as initial guess for the original problem. The effectiveness of the stable inversion approach is validated experimentally for FLEXOR within Section 5.3. For further details on stable inversion it is pointed to the work of Chen and Paden [101], of Devasia et al. [102] and to the book of Seifried [68].

Chapter 4

Estimation

Chapter Contents

For state and output feedback control it is necessary to estimate the system state and the considered output in real-time. In this regard, a Kalman filter is discussed in this chapter to filter measurement signals and to obtain their time derivatives. In the case that the independent coordinates of a flexible link parallel robot, described with geometric constraints, can be measured the filtering gives the independent state. The complete state in chain coordinates is then obtained from the geometric constraints and the corresponding time derivatives. This procedure is denoted as linear state estimator since only Kalman filtering for a linear model is involved. Smoother signals can however be obtained with a nonlinear state estimator which makes use of the nonlinear dynamic flexible multibody model from Chapter 3. In this context, the popular unscented Kalman filter (UKF) is applied but with an adaption to the underlying system of DAEs.

For validation of the later considered end-effector trajectory tracking controllers, a direct measurement with a motion tracking camera is used. The necessary steps for the reconstruction from marker pixel data to the physical end-effector pose are discussed. Finally, it is demonstrated that flexible multibody models can be utilized to estimate unknown parameters such as the mass of a handling object.

4.1 State Estimation

First, signal filtering is used to reduce noise within measurements and to obtain their time derivatives. The filtered signals are then utilized within state estimators. On the one hand, within a simple linear state estimator which is based on a kinematic robot model. On the other hand, filtered signals are used within a UKF which incorporates a nonlinear dynamic flexible multibody model. The UKF further reduces noise and observation spillover. The obtained smoothed signals through filtering can significantly improve the performance of the later applied feedback controllers.

4.1.1 Signal Filtering

Often, measurements are noisy and their time derivatives are needed. To filter measurement signals, an effective and computationally efficient approach is to model each signal independently as a linear single degree of freedom system. The neglected dynamics is then represented by process noise. Based on Singer [105], the discrete equations of motion of each sensor signal can be written as

$$\begin{bmatrix} s_n \\ \dot{s}_n \\ \ddot{s}_n \end{bmatrix} = \begin{bmatrix} 1 & T & \left(\alpha T - 1 + \mathrm{e}^{-\alpha T}\right)/\alpha^2 \\ 0 & 1 & \left(1 - \mathrm{e}^{-\alpha T}\right)/\alpha \\ 0 & 0 & \mathrm{e}^{-\alpha T} \end{bmatrix} \begin{bmatrix} s_{n-1} \\ \dot{s}_{n-1} \\ \ddot{s}_{n-1} \end{bmatrix} + \boldsymbol{w}_{n-1}, \tag{4.1a}$$

$$y_n = s_n + v_n. \tag{4.1b}$$

Here, s_n is the current sensor signal of the model. The term y_n is the system output which is measured by the considered sensor. The vector $\boldsymbol{w}_{n-1}$ is a discrete time white noise sequence, which is a function of the correlation coefficient α, being empirically chosen for each sensor, and the sample time T. The extensive terms within $\boldsymbol{w}_{n-1}$ can be found in [105]. Also, v_n is white Gaussian noise. A Kalman filter for linear systems is here used for state estimation, see, e.g., the book of Thrun et al. [20]. This gives the filtered, i.e., estimated sensor signal $\hat{s}$, denoted by the hat symbol, as well as its time derivatives $\hat{\dot{s}}$ and $\hat{\ddot{s}}$.

The filter performance can be seen in Figure 4.1 for Flexor with $T = 1\,\mathrm{ms}$. Here, the encoder measurements of slider A as well as the strain gauge measurements on link 2, represented as curvature κ_2, are considered. The Kalman filtered signals are much smoother than the measurements. Especially the obtained time derivatives benefit from filtering. Without filtering the velocities have extreme jumps and cannot be meaningfully used within feedback control. These measurement time derivatives are calculated with the difference quotient

$$\dot{y}_n = \frac{y_n - y_{n-1}}{T}. \tag{4.2}$$

In contrast, the filtered velocities are directly provided by the Kalman filter. It should be noted that a major benefit of the presented method over simple

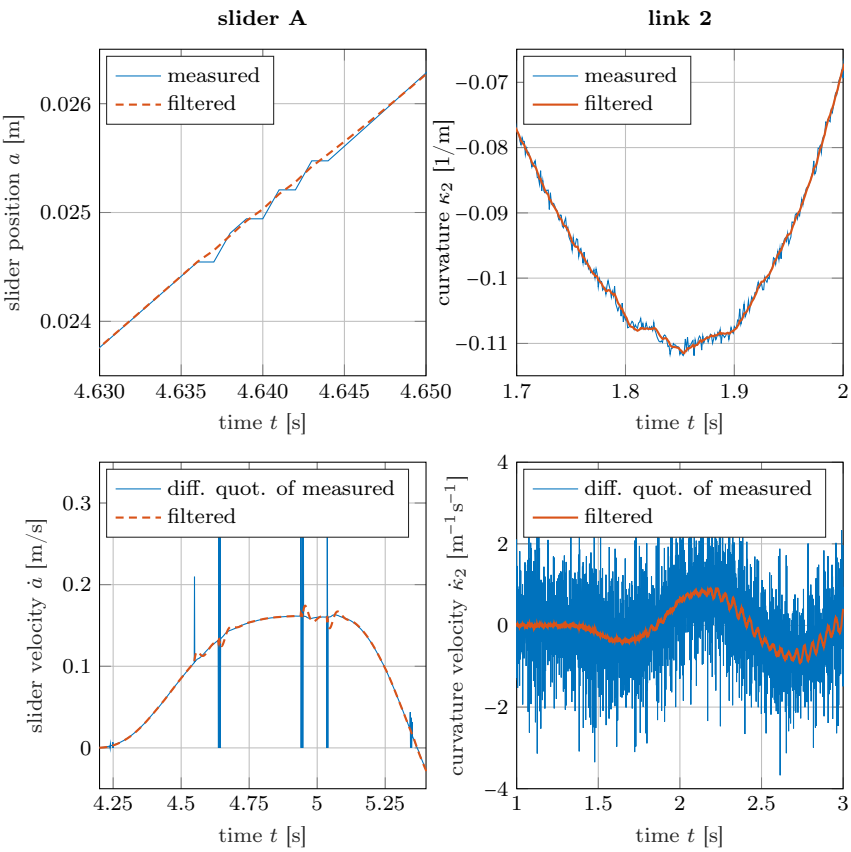


Figure 4.1: Measurements and corresponding time derivatives obtained via Equation (4.2) compared to Kalman filtered measurements.

low-pass filtering is that it typically introduces negligible time delay. Also, the filter performs effectively even for slider A although the measurements show steps instead of the modeled white Gaussian noise.
Throughout this thesis, all sensor related signals will also be denoted as measurements even if they are Kalman filtered. This serves to focus on fundamental differences such as if signals are only simulated.

4.1.2 Linear State Estimator

If enough sensors are available to measure the independent coordinates $\boldsymbol{q}_\mathrm{i}$ of a flexible link parallel robot, the geometric constraints and the signal filtering of

Section 4.1.1 allow to estimate the complete state $\boldsymbol{x}$ in chain coordinates. In the following, this approach is directly applied to FLEXOR for clarification purposes. Here, the estimated system output

$$\hat{\boldsymbol{y}}_\mathrm{f} = \begin{bmatrix} \hat{\kappa}_2 & \hat{\kappa}_3 & \hat{a} & \hat{b} & \hat{\gamma} & \dot{\hat{\kappa}}_2 & \dot{\hat{\kappa}}_3 & \dot{\hat{a}} & \dot{\hat{b}} & \dot{\hat{\gamma}} \end{bmatrix}^\mathrm{T}, \tag{4.3}$$

obtained via Kalman filtering as described in Section 4.1.1, together with the kinematics of FLEXOR shall be used to estimate the system state $\boldsymbol{x} = [\boldsymbol{q}^\mathrm{T}, \dot{\boldsymbol{q}}^\mathrm{T}]^\mathrm{T}$ with

$$\boldsymbol{q} = \begin{bmatrix} \boldsymbol{q}_\mathrm{e}^\mathrm{T} & \boldsymbol{q}_\mathrm{a}^\mathrm{T} & \boldsymbol{q}_\mathrm{u}^\mathrm{T} \end{bmatrix}^\mathrm{T} = \begin{bmatrix} q_{\mathrm{e},2} & q_{\mathrm{e},3} & a & b & \gamma & \alpha & \beta \end{bmatrix}^\mathrm{T}. \tag{4.4}$$

Therefore, observability is needed. A definition and sufficient condition of global observability of nonlinear systems can be found, e.g., in the book of Adamy [106, pp. 288/295]. It is typically difficult to show but for the considered system type and the available output $\hat{\boldsymbol{y}}_\mathrm{f}$ the complexity reduces significantly. From the output $\hat{\boldsymbol{y}}_\mathrm{f}$, the quantities $\hat{\boldsymbol{q}}_\mathrm{a}$ and $\dot{\hat{\boldsymbol{q}}}_\mathrm{a}$ are directly available. Based on the utilized beam model, the curvatures $\boldsymbol{\kappa}$ are linearly related to the elastic coordinates $\boldsymbol{q}_\mathrm{e}$ by

$$\boldsymbol{\kappa} = \underbrace{\begin{bmatrix} \partial \psi_z / \partial x \big|_{x_{\mathrm{s},1}} \\ \vdots \\ \partial \psi_z / \partial x \big|_{x_{\mathrm{s},i}} \\ \vdots \end{bmatrix}}_{\widetilde{\boldsymbol{C}}_\mathrm{e}} \boldsymbol{q}_\mathrm{e}, \tag{4.5}$$

see Equation (2.5). Here, $x_{\mathrm{s},i}$ denotes the i-th strain gauge measurement location on the flexible links. In the considered case of using as many strain gauge measurements as elastic coordinates, namely two for both, the sensors can be appropriately placed such that the constant output matrix $\widetilde{\boldsymbol{C}}_\mathrm{e}$ has full rank. This means that $\widetilde{\boldsymbol{C}}_\mathrm{e} \in \mathbb{R}^{2\times 2}$ can be inverted to calculate the elastic coordinates $\hat{q}_{\mathrm{e},2}$, $\hat{q}_{\mathrm{e},3}$ from the measured curvatures $\hat{\kappa}_2$, $\hat{\kappa}_3$. Since analogously

$$\dot{\boldsymbol{\kappa}} = \widetilde{\boldsymbol{C}}_\mathrm{e} \dot{\boldsymbol{q}}_\mathrm{e} \tag{4.6}$$

holds, $\hat{\boldsymbol{q}}_\mathrm{e}$ and $\dot{\hat{\boldsymbol{q}}}_\mathrm{e}$ are determined. Thus, only the additional unactuated coordinates $\hat{\boldsymbol{q}}_\mathrm{u}$, i.e., the angles $\hat{\alpha}$ and $\hat{\beta}$ as well as the corresponding time derivatives are missing to complete the state $\hat{\boldsymbol{x}}$. Due to the kinematic loop at the sliders, $\boldsymbol{q}_\mathrm{u}$ can be regarded as dependent coordinates $\boldsymbol{q}_\mathrm{d}$ while $\boldsymbol{q}_\mathrm{i} = [\boldsymbol{q}_\mathrm{e}^\mathrm{T}, \boldsymbol{q}_\mathrm{a}^\mathrm{T}]^\mathrm{T}$ are the independent coordinates. With available estimates of the independent coordinates, the dependent coordinates are obtained by solving the constraints $\boldsymbol{c}(\hat{\boldsymbol{q}}_\mathrm{i}, \hat{\boldsymbol{q}}_\mathrm{d}) = \boldsymbol{0}$ for $\hat{\boldsymbol{q}}_\mathrm{d}$. This may be done numerically by root finding via the Newton-Raphson method. Analogously to step 3) of Figure 3.19, with available $\hat{\boldsymbol{q}}$ and $\dot{\hat{\boldsymbol{q}}}_\mathrm{i}$ the velocities are then computed via

$$\dot{\hat{\boldsymbol{q}}}_\mathrm{u} = \dot{\hat{\boldsymbol{q}}}_\mathrm{d} = -\boldsymbol{C}_\mathrm{d}^{-1} (\boldsymbol{C}_\mathrm{i} \dot{\hat{\boldsymbol{q}}}_\mathrm{i} + \boldsymbol{c}'). \tag{4.7}$$

This delivers the complete state $\hat{\boldsymbol{x}}$ and consequently observability is provided in all feasible system configurations.

Within the presented method, all measured signals rely on Kalman filtering based on the linear model of Equation (4.1). Since the nonlinearity of the constraint equations $\boldsymbol{c} = \mathbf{0}$ is introduced after the actual estimation, the method is denoted as *linear state estimator* throughout this research.

4.1.3 Unscented Kalman Filter for DAEs

The linear state estimator from Section 4.1.2 is a simplified and computationally efficient method to estimate the state of flexible link parallel robots. Nevertheless, more effective filtering of noise and unmodeled higher-order modes can be necessary for feedback control. Also, a possibility for systematic sensor data fusion is desired, e.g., to include the rotary joint encoder measurements for Flexor. Therefore, the well-known UKF [20, 107] will now be applied for nonlinear state estimation. As opposed to the linear state estimator, it incorporates the nonlinear system dynamics. The UKF framework is chosen because of its ability to handle noisy measurements and it provides straightforward sensor data fusion. Also, it is well suited for real-time estimation for nonlinear systems since Kalman algorithms are not computationally expensive [108]. Instead of linearizing the nonlinear system dynamics as within the extended Kalman filter (EKF), the UKF uses the so-called *unscented transformation*. According to Wan and van der Merwe [109], this renders the UKF typically more accurate than the EKF while sharing a similar level of complexity.
To apply the UKF to parallel robots described by DAEs, the algorithm needs to be adapted. For instance, Teixeira et al. [22] discuss four algorithms for nonlinear equality-constrained state estimation based on the UKF. The presented research, however, uses a different approach via minimal coordinates. It is similar to [21] but a projection tangential to the constraint manifold is used based on a QR decomposition instead of coordinate partitioning. This prevents singularities within the calculation of the projection matrix. Subsequently, the standard UKF algorithm is applied to the resulting ODEs in minimal, i.e., independent coordinates. Here, the minimal form reduces the necessary sigma points. The dependent coordinates are obtained by satisfying the algebraic constraint equations with the Newton-Raphson method in each time step. The details are discussed in the following which are based on the publication [30] of the author.

First, an appropriate system description needs to be selected for the considered flexible link parallel robots. Since there are unknown and potentially significant friction effects within the actuators, kinematic inputs are used instead of motor current, i.e., force inputs. This means that the actuator motions are no degrees of freedom anymore, which reduces the system dimension of the model for the estimator. The problem of estimating the kinematic actuator quantities is thus shifted to the Kalman filter which uses the linear single degree of freedom system of Equation (4.1) for each actuator encoder measurement. This is meaningful since Kalman filtered encoder measurements are typically very smooth and accurate

such that no further filtering based on a dynamic robot model is needed. Only using the Kalman filter for the actuators is also computationally very efficient. In the case of FLEXOR, the remaining coordinates $\boldsymbol{q}_\mathrm{k} \in \mathbb{R}^{f_\mathrm{k}}$ consist of the two angles α and β as well as of the elastic coordinates $q_{\mathrm{e},2}$ and $q_{\mathrm{e},3}$ of link 2 and link 3, respectively. The main idea of the nonlinear state estimator is to transform the DAEs of the model of a flexible link parallel robot to ODEs for which the standard UKF algorithm can be applied. The transformation is performed via a projection based on a QR decomposition in each time step. Consequently, Equation (3.83) is used to describe the forward dynamics in minimal form which yields

$$\dot{\boldsymbol{\zeta}} = \boldsymbol{J}_{\mathrm{k,q}}^{\mathrm{T}} \left(\dot{\boldsymbol{q}}_\mathrm{k} + \boldsymbol{Q}_\mathrm{k} \boldsymbol{R}_\mathrm{k}^{-\mathrm{T}} \boldsymbol{c}_\mathrm{k}' \right), \tag{4.8a}$$

$$\ddot{\boldsymbol{\zeta}} = \left(\boldsymbol{J}_{\mathrm{k,q}}^{\mathrm{T}} \boldsymbol{M}_\mathrm{k} \boldsymbol{J}_{\mathrm{k,q}} \right)^{-1} \boldsymbol{J}_{\mathrm{k,q}}^{\mathrm{T}} \left(\boldsymbol{f}_\mathrm{k} + \boldsymbol{B}_\mathrm{k} \ddot{\boldsymbol{q}}_\mathrm{a} + \boldsymbol{M}_\mathrm{k} \boldsymbol{Q}_\mathrm{k} \boldsymbol{R}_\mathrm{k}^{-\mathrm{T}} \boldsymbol{c}_\mathrm{k}'' \right) \tag{4.8b}$$

for the new independent coordinates $\boldsymbol{\zeta} \in \mathbb{R}^{f_\mathrm{k} - n_\mathrm{c}}$. Here, n_c dependent coordinates $\boldsymbol{\nu}$ occur, with $n_\mathrm{c} = 2$ for FLEXOR due to the kinematic loop at the sliders. This means that in this research the UKF is used only for an unconstrained end-effector within the model. Nevertheless, for an end-effector with contact Equation (4.8) can still be used but with an additional force input at the end-effector to incorporate force measurements. This is especially meaningful if the environment is unknown and can therefore not be modeled. Following the local coordinate method of Hairer et al. [110, chapter 4] the coordinates $\boldsymbol{q}_\mathrm{k}$ and $\dot{\boldsymbol{q}}_\mathrm{k}$ can be decomposed as

$$\underbrace{\begin{bmatrix} \boldsymbol{q}_\mathrm{k} \\ \dot{\boldsymbol{q}}_\mathrm{k} \end{bmatrix}}_{\boldsymbol{x}_\mathrm{k}} = \underbrace{\begin{bmatrix} \boldsymbol{q}_{\mathrm{k},0} \\ \dot{\boldsymbol{q}}_{\mathrm{k},0} \end{bmatrix}}_{\boldsymbol{x}_{\mathrm{k},0}} + \underbrace{\begin{bmatrix} \boldsymbol{J}_{\mathrm{k,q}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{J}_{\mathrm{k,q}} \end{bmatrix}}_{\widetilde{\boldsymbol{J}}_{\mathrm{k,q}}} \underbrace{\begin{bmatrix} \boldsymbol{\zeta} \\ \dot{\boldsymbol{\zeta}} \end{bmatrix}}_{\boldsymbol{z}} + \underbrace{\begin{bmatrix} \boldsymbol{Q}_\mathrm{k} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{Q}_\mathrm{k} \end{bmatrix}}_{\widetilde{\boldsymbol{Q}}_\mathrm{k}} \underbrace{\begin{bmatrix} \boldsymbol{\nu} \\ \boldsymbol{\nu}' \end{bmatrix}}_{\boldsymbol{n}}, \tag{4.9}$$

with $\boldsymbol{z} \in \mathbb{R}^{2(f_\mathrm{k} - n_\mathrm{c})}$ and $\boldsymbol{n} \in \mathbb{R}^{2n_\mathrm{c}}$ being the local coordinates. In the estimator, the start value $\boldsymbol{x}_{\mathrm{k},0}$ represents $\hat{\boldsymbol{x}}_{\mathrm{k},n-1}^{+}$ with the plus denoting a posterior quantity and $n-1$ being the last time step before the current time step n. A minus would describe a prior quantity. With the error covariance matrix $\boldsymbol{P}_{zz,n-1}^{+}$ of the last time step the origin-centered sigma points $\boldsymbol{\mathcal{Z}}_{i,n-1}$ are generated in $\boldsymbol{z}$-coordinates, with bold calligraphic letters representing vectors. Based on Equation (4.9), the results are inserted into

$$\boldsymbol{\mathcal{X}}_{i,n-1} = \hat{\boldsymbol{x}}_{\mathrm{k},n-1}^{+} + \widetilde{\boldsymbol{J}}_{\mathrm{k,q}} \boldsymbol{\mathcal{Z}}_{i,n-1}, \tag{4.10}$$

which is used to evaluate Equation (4.8). The results are then integrated with a forward Euler step to obtain $\overline{\boldsymbol{\mathcal{Z}}}_{i,n}^{*}$ for each sigma point denoted by i. Now, in accordance with the usual UKF algorithm, see, e.g., [20], the prior mean $\hat{\boldsymbol{z}}_n^{-}$ is obtained and utilized to calculate the prior error covariance matrix $\boldsymbol{P}_{zz,n}^{-}$. Thereby, the new sigma points $\overline{\boldsymbol{\mathcal{Z}}}_{i,n}$ are generated. Analog to Equation (4.10)

$$\overline{\boldsymbol{\mathcal{X}}}_{i,n} = \hat{\boldsymbol{x}}_{\mathrm{k},n-1}^{+} + \widetilde{\boldsymbol{J}}_{\mathrm{k,q}} \overline{\boldsymbol{\mathcal{Z}}}_{i,n} \tag{4.11}$$

is then used within the output function $\boldsymbol{h}$ leading to

$$\overline{\boldsymbol{\mathcal{Y}}}_{i,n} = \boldsymbol{h}(\overline{\boldsymbol{\mathcal{X}}}_{i,n}). \tag{4.12}$$

The following necessary steps are the remaining standard UKF calculations but applied to the independent coordinates from which the posterior quantities $\hat{\boldsymbol{z}}_n^+$ and $\boldsymbol{P}_{zz,n}^+$ are obtained. Based on this, the starting value

$$\hat{\boldsymbol{x}}_{\mathrm{k},n,0}^+ = \hat{\boldsymbol{x}}_{\mathrm{k},n-1}^+ + \widetilde{\boldsymbol{J}}_{\mathrm{k,q}} \hat{\boldsymbol{z}}_n^+ \tag{4.13}$$

can be found. It is then used within the Newton-Raphson method in order to obtain $\boldsymbol{\nu}$ which fulfills the algebraic constraints on position level $\boldsymbol{c} = \boldsymbol{0}$, see Equation (3.78b). A discussion on the Newton-Raphson Method with optimal basis can be found, e.g., in the thesis of Burkhardt [104, p. 66]. Based on Equations (3.82) and (4.9) it follows

$$\frac{\partial \boldsymbol{c}}{\partial \boldsymbol{\nu}} = \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}_\mathrm{k}} \frac{\partial \boldsymbol{q}_\mathrm{k}}{\partial \boldsymbol{\nu}} = \boldsymbol{C}_\mathrm{k} \boldsymbol{Q}_\mathrm{k} = \boldsymbol{R}_\mathrm{k}^\mathrm{T} \underbrace{\boldsymbol{Q}_\mathrm{k}^\mathrm{T} \boldsymbol{Q}_\mathrm{k}}_{\mathbf{I}_{n_\mathrm{c}}} = \boldsymbol{R}_\mathrm{k}^\mathrm{T}. \tag{4.14}$$

The Newton-Raphson steps are written as

$$\boldsymbol{\nu}_j = -\boldsymbol{R}_\mathrm{k}^{-\mathrm{T}} \boldsymbol{c}(\hat{\boldsymbol{q}}_{\mathrm{k},n,j-1}^+, \hat{\boldsymbol{q}}_{\mathrm{a},n}), \tag{4.15}$$

with

$$\hat{\boldsymbol{q}}_{\mathrm{k},n,j}^+ = \hat{\boldsymbol{q}}_{\mathrm{k},n,j-1}^+ + \boldsymbol{Q}_\mathrm{k} \boldsymbol{\nu}_j. \tag{4.16}$$

Here, j denotes the iteration step of the j_max iterations. To fulfill the constraints on velocity level, i.e., $\dot{\boldsymbol{c}} = \boldsymbol{0}$, the result of the Newton-Raphson method can be used within Equation (3.79a) which is solved for $\boldsymbol{\nu}'$. This yields the velocities

$$\hat{\dot{\boldsymbol{q}}}_{\mathrm{k},n}^+ = \hat{\dot{\boldsymbol{q}}}_{\mathrm{k},n,j_\mathrm{max}}^+ = \hat{\dot{\boldsymbol{q}}}_{\mathrm{k},n,0}^+ + \boldsymbol{Q}_\mathrm{k} \boldsymbol{\nu}'. \tag{4.17}$$

Finally, the output of interest such as the estimated end-effector pose and velocity can be calculated from the estimated state $\hat{\boldsymbol{x}}_{\mathrm{k},n}^+$.

The discussed algorithm is summarized in Figure 4.2. Here, the filtered actuator positions $\hat{\boldsymbol{q}}_{\mathrm{a},n}$ at the current time step as well as the corresponding time derivatives $\hat{\dot{\boldsymbol{q}}}_{\mathrm{a},n}$ and $\hat{\ddot{\boldsymbol{q}}}_{\mathrm{a},n}$ are treated as kinematic inputs. Also, $\boldsymbol{Q}_n$ and $\boldsymbol{R}_n$ denote the process and the measurement noise covariance matrices which are sources of additive white Gaussian noise.

For Flexor, the system output $\boldsymbol{y}_n$ utilized within the measurement corrections includes the curvatures $\hat{\kappa}_2$ and $\hat{\kappa}_3$ obtained via Kalman filtered strain gauge measurements. The output also includes the corresponding curvature velocities as these ensure that small offsets of the zero-point of the strain gauges do not cause the estimated first time derivatives of the elastic coordinates to drift. These offsets typically result from static deflections of the spring steel links which are

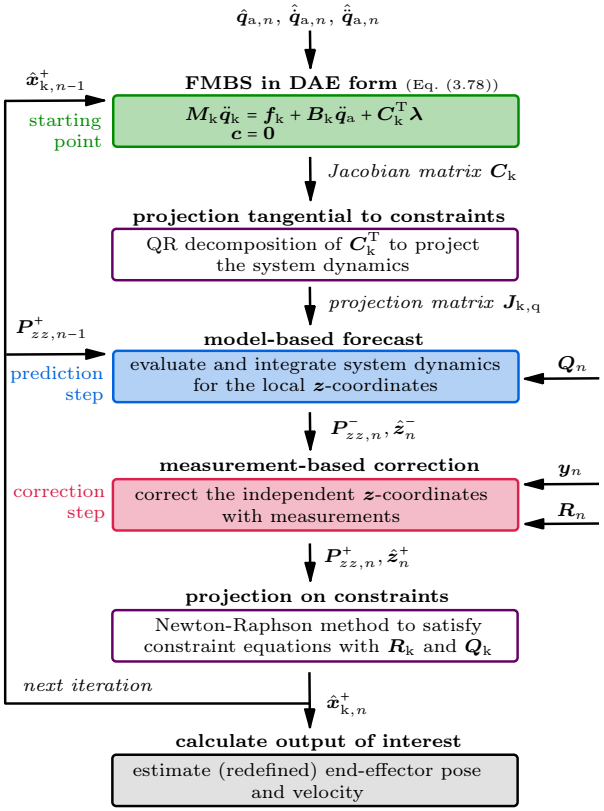


Figure 4.2: Algorithm of the UKF for flexible multibody systems in DAE form.

not perfectly straight. For the Newton-Raphson method four iteration steps are used which typically leads to negligible Euclidean norms for $\boldsymbol{c}$ and $\dot{\boldsymbol{c}}$.

To experimentally validate the UKF it is applied to Flexor, where the actuators are directly controlled via a gamepad. The obtained arbitrary motion is well suited to show the excitation of higher-order modes. The resulting curvature κ_2 of link 2 is illustrated in Figure 4.3. Here, the measured curvature corresponds to the Kalman filtered measurement signal used within the output $\boldsymbol{y}_n$ for the UKF. The curvature of the UKF is obtained by transforming the estimation of the elastic coordinate $q_{\mathrm{e},2}$ via Equation (4.5). Moreover, the simulated curvature is computed via forward time integration of Equation (3.84). The integration is done with the forward Euler method running at a step size of 1 ms, which is what the UKF also relies on.
In the magnification of Figure 4.3 at around 3 s a significant higher-order frequency

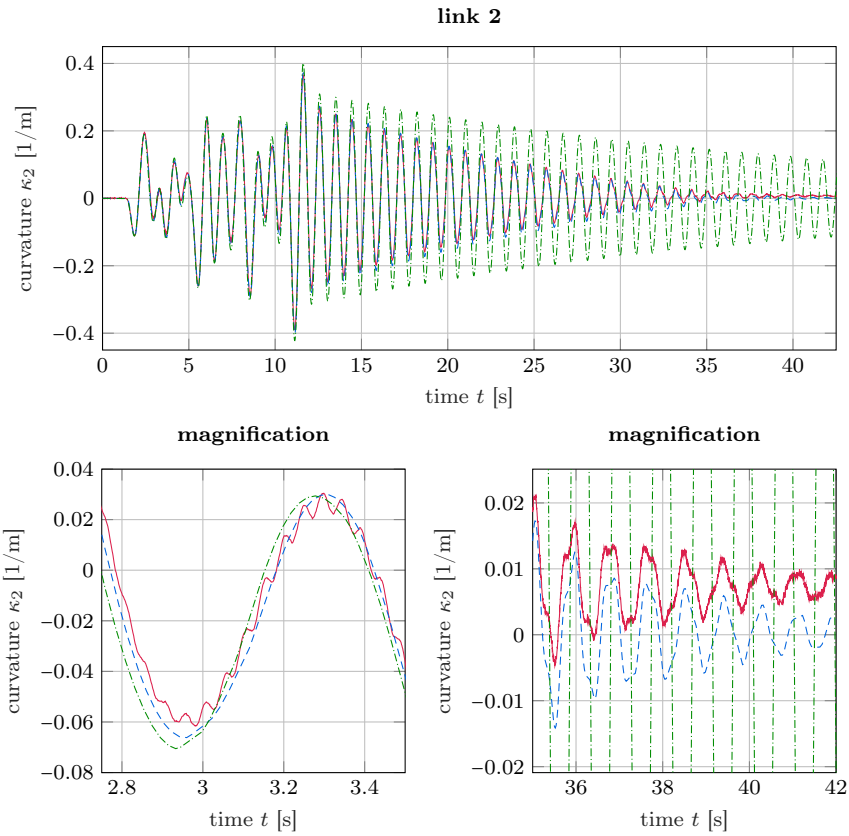


Figure 4.3: Measured (——), estimated by a UKF (– – –) and simulated (–·–·–) curvature κ_2 of link 2.

can be seen in the measurement signal, which is related to the second bending eigenmode of link 2. Since the model within the UKF incorporates only the first bending eigenfrequency of link 2 as well as of link 3, higher frequencies are filtered effectively without introducing significant delay. Filtering such unmodeled frequencies is very important considering the so-called *spillover effect*. The later utilized feedback controllers are designed for a finite number of controlled modes, being the first two bending eigenmodes in the case of Flexor. But they are applied to real flexible links which possess an infinite number of additional uncontrolled modes when assuming continua. This results in two types of spillover, the *control spillover* meaning the excitation of uncontrolled modes, and the *observation spillover* meaning that uncontrolled modes contaminate the sensor measurements [111, 112]. Spillover can deteriorate the control performance and

even cause instability [113]. According to Balas [112], filtering may substantially reduce observation spillover. Especially low-pass filtering is discussed by Liu and Hou [113] as a natural solution, which can however introduce significant delay. Also, the Kalman filtering approach from Section 4.1.1 is not appropriate to filter frequencies which are close to each other. Thus, the UKF is typically much better suited to cope with observation spillover in order to prevent feedback of uncontrolled modes.
In the magnification of Figure 4.3 at around 40 s it is also seen that the estimation further reduces noise and tends to zero while the measurement shows a small offset. This offset may reflect the reality better since the spring steel of the flexible links is not perfectly straight when no oscillations occur. Nevertheless, such offsets are typically unwanted within, e.g., state controllers as the system never comes to rest when a zero deformation is desired. The offset also differs with the slider positions such that a simple zeroing at the start is not sufficient. Thus, the estimation by the UKF ensures that controllers get a signal where such undesired offsets are significantly reduced. The estimated curvature of the UKF also matches closely the amplitude and phase of the measurements over the whole scenario time. This is not the case for the simulation which is expected since it is not updated with measurements as opposed to the UKF.
The estimation performance for the state variables other than $q_{\mathrm{e},2}$, which are not illustrated here, is similarly effective. In particular, the reduced noise leads to a much smoother $\dot{q}_{\mathrm{e},2}$ and $\dot{q}_{\mathrm{e},3}$ for the UKF compared to Kalman filtering of the single curvature signals.

In conclusion, the presented investigation shows that the proposed UKF algorithm can be used to obtain a smooth state estimate with highly reduced observation spillover for flexible multibody systems in DAE form.

4.2 Output Pose Estimation

In robotics the end-effector pose is typically the output of interest. However, measurements of the end-effector of a flexible link robot cannot be obtained only via motor encoders. Based on such encoders together with strain gauge measurements and a flexible robot model indirect estimations could be conducted via the UKF presented in Section 4.1.3. Nevertheless, for validation purposes of model-based controllers a direct measurement method is needed which is not based on such a robot model. Therefore, marker-based real-time end-effector reconstruction with the IR motion tracking camera from Section 2.2.3.4 is discussed in the following.
Before the reconstruction a two-step calibration is performed with the camera being mounted at a fixed pose with fixed focus. In the first step, the intrinsic camera parameters are identified. They are needed to convert marker data from pixels into physical coordinates with respect to a camera-fixed coordinate system.

In the second step, the extrinsic parameters are identified being the pose, i.e., the position and rotation of the camera relative to the inertial system of the robot. Afterwards, the utilized methods for output reconstruction in 2D and in 3D are discussed. The main calculations are based on the C++ version of the open source computer vision library OpenCV, which is well suited for real-time applications.

Starting point is the ideal pinhole camera model, see, e.g., [114, chapter 11], where the aperture of the camera is reduced to a point, see Figure 4.4. Here, the standard 2D case of Flexor with gripper and without contact is used for visualization purposes. The approaches of this section, however, can be utilized for general 2D and 3D pose estimation. In reality the image plane of Figure 4.4

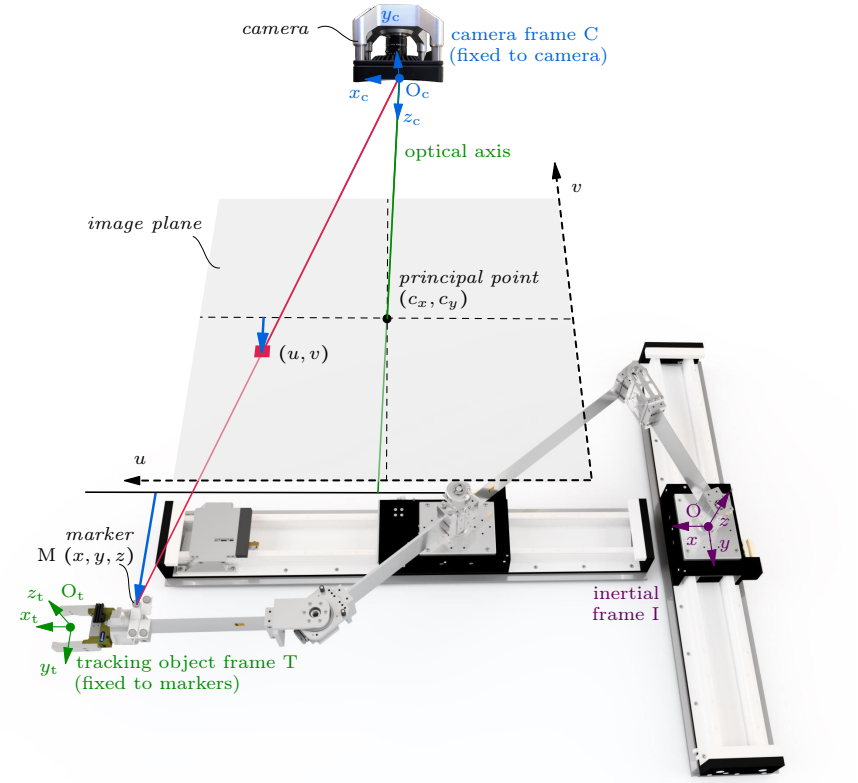


Figure 4.4: Pinhole camera model applied to Flexor.

occurs behind the pinhole denoted by O_c and is mirrored. For simplification this is prevented in the model by positioning the image plane between the camera and the marker M under consideration. The goal is now to find the marker position with respect to the origin O being represented in the inertial frame I.

4.2.1 Intrinsic Parameters

Firstly, the relation between image coordinates u and v in pixels and physical Cartesian coordinates with respect to the camera coordinate system C needs to be formulated. Based on [114, 115], this can be written as

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\text{camera matrix}} \frac{1}{z_c} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}. \tag{4.18}$$

Here, the u- and x_c-axis point in the same direction as well as the v- and y_c-axis. The parameters f_x and f_y are the focal lengths in pixels. The coordinates c_x and c_y describe the principal point being typically the image center expressed in pixels. These four intrinsic parameters characterize the camera via the so-called *camera matrix*.
Since real cameras are subject to lens distortion the ideal pinhole camera model needs to be extended to obtain more accurate localization results. Here, tangential and radial distortion is considered up to terms of 6th order according to [115, p. 382] by

$$\tilde{x} = \frac{x_c}{z_c}, \tag{4.19a}$$

$$\tilde{y} = \frac{y_c}{z_c}, \tag{4.19b}$$

$$r^2 = \tilde{x}^2 + \tilde{y}^2, \tag{4.19c}$$

$$\bar{x} = \tilde{x}\frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + 2p_1\tilde{x}\tilde{y} + p_2\left(r^2 + 2\tilde{x}^2\right), \tag{4.19d}$$

$$\bar{y} = \tilde{y}\frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + p_1\left(r^2 + 2\tilde{y}^2\right) + 2p_2\tilde{x}\tilde{y}. \tag{4.19e}$$

With these distortion terms Equation (4.18) is adapted to

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{d}\left(\frac{1}{z_c}\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}\right) = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \bar{x} \\ \bar{y} \\ 1 \end{bmatrix}. \tag{4.20}$$

Here, $\boldsymbol{d}(\,)$ is the distortion function which maps the camera coordinates x_c, y_c and z_c to distorted coordinates $\bar{x}$ and $\bar{y}$ as described by Equations (4.19a)–(4.19e). The coefficients k_1 to k_6 describe the radial distortion and p_1 as well as p_2 describe the tangential distortion. These 8 distortion coefficients also belong to the intrinsic parameters which are camera dependent.

4.2.2 Extrinsic Parameters

Secondly, the extrinsic camera parameters are needed which describe the rotation and translation between the camera system C and the inertial system I. The vector from the origin O_c of the camera coordinate system C to the center of the marker M represented in the camera system C can be expressed as

$$\underbrace{\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}}_{\boldsymbol{r}_{O_c M|C}} = \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}}_{\boldsymbol{S}_{CI}} \underbrace{\begin{bmatrix} x \\ y \\ z \end{bmatrix}}_{\boldsymbol{r}_{OM|I}} + \underbrace{\begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}}_{\boldsymbol{r}_{O_c O|C}} . \tag{4.21}$$

Here, $\boldsymbol{S}_{CI}$ is the matrix describing the rotation from the inertial to the camera system and $\boldsymbol{r}_{O_c O|C}$ is the translation from O_c to O expressed in the camera system C. They both represent the extrinsic camera parameters and characterize the camera pose relative to the inertial system. The vector $\boldsymbol{r}_{OM|I}$ is the marker position in the inertial frame which shall finally be calculated. Together with the intrinsic relation of Equation (4.20) the complete projection of a marker M from the inertial system to the image plane follows as

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{d}\left(\frac{1}{z_c} \left(\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \right) \right). \tag{4.22}$$

4.2.3 Calibration

The objective of the calibration is to identify the discussed intrinsic and extrinsic camera parameters. The utilized calibration workflow is depicted in the middle column of Figure 4.5. It also summarizes the later covered 2D and 3D reconstructions.

To obtain the intrinsic parameters, i.e., the camera matrix and the distortion coefficients, the calibration board shown in Figure 2.19 is used. Several shots of the board are taken within different poses. The camera extracts all 110 foil marker centers which are equally distributed over an area of 45 cm × 50 cm. The resulting 2D image points in pixels of each shot together with the physical positions of the markers in the tracking object frame, i.e., the 3D marker pattern are handed to the OpenCV function `calibrateCamera`. This function is based on Zhang [116] and uses the Levenberg-Marquardt optimization algorithm to minimize the reprojection error between image points and projected object points. The function output are the required intrinsic parameters. The effect of the identified lens distortion is shown in Figure 4.6. With larger distance from the center, a significant distortion can be seen. This confirms the need to take the lens distortion into account. The edges of the image are not included since they are not of interest for the considered measurements.

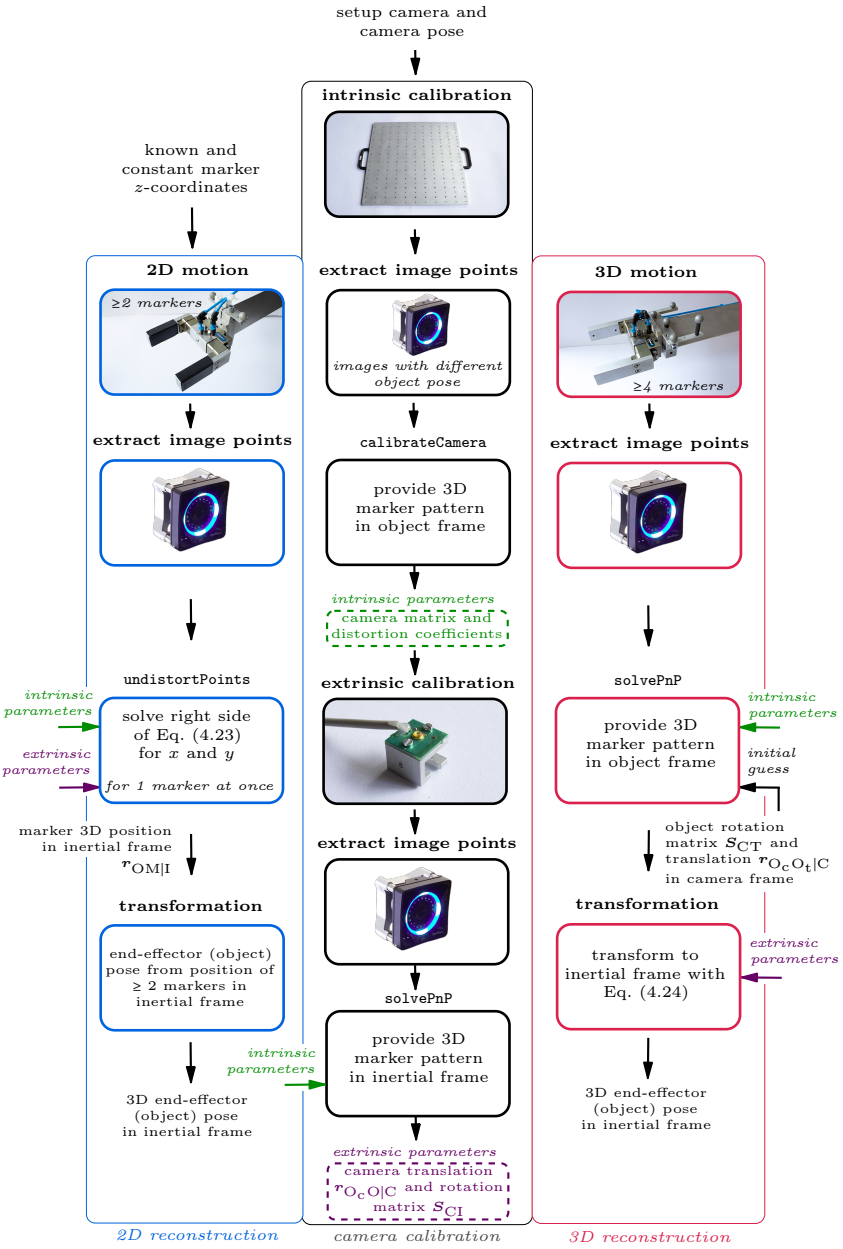


Figure 4.5: Calibration and reconstruction process.

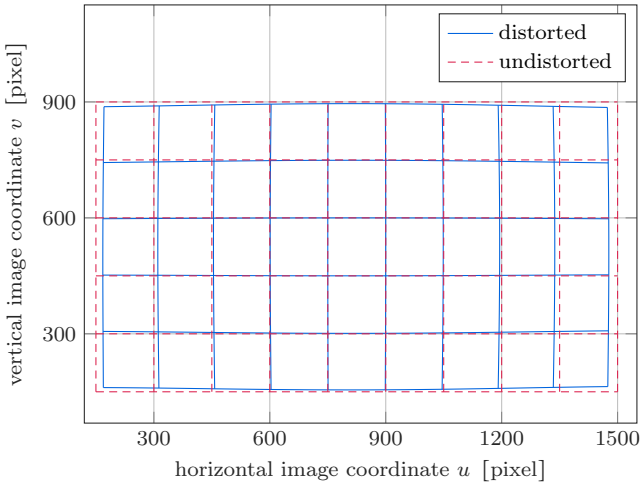


Figure 4.6: Experimentally identified lens distortion.

With the identified intrinsic parameters, the extrinsic parameters can be obtained via the OpenCV function `solvePnP`. It uses the 3D pattern of n markers and the corresponding 2D image points within the arising Perspective-n-Point (PnP) problem. To obtain the camera pose in relation to the inertial system, the 3D marker positions need to be provided in the inertial frame I. Instead of a board, 10 active IR LEDs are used as illustrated in Figure 2.21, which are attached to the granite table of the robot as shown in Figure 2.22. Again, the reprojection error is minimized with the Levenberg-Marquardt optimization algorithm. For the utilized setup, the calibration turns out to be sufficiently accurate with a maximum absolute error in the xy-plane of around 1.3 mm between actual LED position and reconstructed LED position. If the intrinsic distortion parameters are not used, i.e., when they are all set to zero in the intrinsic calibration and a subsequent extrinsic calibration is performed, typically much higher errors occur. For the LEDs a test resulted in a roughly three times larger absolute error in the xy-plane on average. This confirms again the need to consider the lens distortion. Finally, the output of `solvePnP` yields the extrinsic parameters, i.e., the rotation vector which is transformed via Rodrigues' rotation formula [115] to the rotation matrix $\boldsymbol{S}_{\mathrm{CI}}$ and the translation vector $\boldsymbol{r}_{\mathrm{O_cO|C}}$.

4.2.4 2D Reconstruction

If it is known that the motion to be tracked is only in 2D, such as for the standard setup of Flexor, a simplified reconstruction process is applicable. With a given third coordinate it is possible to directly obtain the 2D position of single markers.

Without loss of generality, the considered 2D motion is in the xy-plane with known marker z-coordinate. Tests confirm that reconstructing a motion in the z_c-direction, i.e., in the direction of the optical axis, typically involves larger errors than in the x_c- and y_c-direction. Therefore, the camera is mounted vertically over the 2D motion which results in a z_c-axis and a z-axis being very close to parallel. Since the z-coordinate is known an increased reconstruction accuracy of the xy-coordinates is ensured. The reconstruction process to obtain a marker position in the inertial system can be regarded as the inversion of the projection of these marker coordinates from the inertial system to the image plane as described by Equation (4.22). Here, the OpenCV function `undistortPoints` is applied to compensate the identified lens distortion. It needs the intrinsic camera matrix and the distortion coefficients as well as the observed, i.e., distorted image coordinates u and v of the considered marker. The `undistortPoints` function iteratively calculates the left side of the rearranged Equation (4.22)

$$\underbrace{\boldsymbol{d}^{-1}\left(\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}\right)}_{\texttt{undistortPoints}} = \frac{1}{z_c}\left(\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}\right). \tag{4.23}$$

Here, $\boldsymbol{d}^{-1}(\,)$ denotes the inverse distortion function. With the left side calculated and with identified extrinsic parameters on the right side as well as with a given marker z-coordinate, Equation (4.23) can be solved analytically for the three unknowns. These are the camera z_c-coordinate and the required x- and y-position of the marker in the inertial system. Repeating these computations for at least a second marker allows to calculate the complete tracking object, i.e., end-effector pose.

For validation of the proposed 2D reconstruction a marker is attached to slider B of Flexor. The position of the marker on the slider is manually measured and added to the linear encoder measurements. The result is then compared to marker camera measurements, see Figure 4.7. These camera measurements come directly from the 2D reconstruction with the identified intrinsic and extrinsic parameters. The processed marker data arrives with a latency in the control loop of the real-time target, which can be seen when comparing the encoder and the camera measurements. Due to the typically smooth motions and the small camera latency, a simple linear extrapolation of 3 ms based on 9 past values of the camera measurement signal is used since it effectively reduces the errors coming from the latency with negligible computational load. Alternative and more complex possibilities are discussed, e.g., by van der Merwe et al. [117] in order to compensate sensor latency within Kalman filters for linear or linearized systems and for nonlinear systems.
In Figure 4.7a) the x-coordinate, i.e., the direction of motion of the slider coordinate b, is visualized. For a movement of over 20 cm the maximum Euclidean error norm between encoder and camera measurements is 2 mm without extrapolation

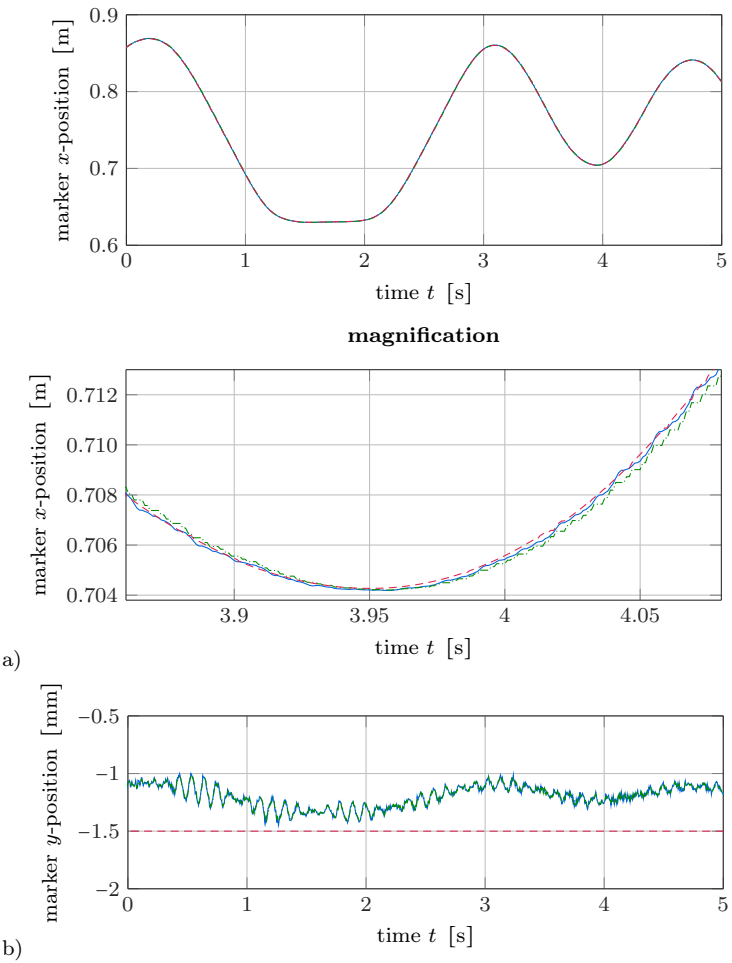


Figure 4.7: Marker attached to slider B for 2D reconstruction: camera measurements with extrapolation (——) and without (-·-·-·) compared to linear encoder measurements (- - - -).

and 1.3 mm with extrapolation. As a result of these small errors one can hardly see any difference in the upper plot. A magnification in the middle plot shows that the camera measurements without extrapolation have significant steps. The reason is that the camera sample time is 2.8 ms while the curves are sampled with 1 ms since they come from the real-time target.
Figure 4.7b) confirms that no significant motion occurs in the y-direction since slider B only moves in the x-direction. Here, the constant offset of the encoder measurements comes from the marker being attached slightly off the slider center.

In conclusion, the 2D marker reconstruction is successfully validated. The demonstrated accuracy is relatively high but can worsen in different areas of the camera image. Nonetheless, it is well suited to assess the end-effector trajectory tracking performance of the later considered control concepts, which lead to errors of several millimeters for FLEXOR.

4.2.5 3D Reconstruction

To reconstruct a 3D motion with the tracking camera at least four markers are necessary, that form a rigid body with known relative positions, to ensure a unique solution. Again, the OPENCV function `solvePnP` is used which needs the intrinsic parameters being the camera matrix and the distortion coefficients as well as the 2D image coordinates in pixels. By utilizing the last reconstructed pose, after a successful initialization, as initial guess for the next pose the robustness of the algorithm is increased. In contrast to the extrinsic calibration the marker positions are defined in a local object fixed coordinate system and not in the inertial system. Consequently, `solvePnP` computes the rotation matrix $\boldsymbol{S}_{\mathrm{CT}}$ and the translation vector $\boldsymbol{r}_{\mathrm{O_cO_t|C}}$ between the tracking object frame T and the camera frame C. A simple coordinate transformation with the extrinsic parameters gives the required translation and rotation

$$\boldsymbol{r}_{\mathrm{OO_t|I}} = \boldsymbol{S}_{\mathrm{CI}}^{\mathrm{T}} \left(\boldsymbol{r}_{\mathrm{O_cO_t|C}} - \boldsymbol{r}_{\mathrm{O_cO|C}} \right), \tag{4.24a}$$

$$\boldsymbol{S}_{\mathrm{IT}} = \boldsymbol{S}_{\mathrm{CI}}^{\mathrm{T}} \boldsymbol{S}_{\mathrm{CT}}, \tag{4.24b}$$

between the tracking object frame and the inertial frame.

The presented 3D reconstruction is now experimentally validated for the five marker setup of FLEXOR as shown on the right of Figure 2.18. The sliders are fixed to their zero positions and link 3 is replaced by a rigid link and is rotated by 90° analogously to Figure 2.5 on the right. Then, link 3 is raised slowly to ensure that no oscillations are excited in the flexible link 2. The camera-based reconstruction of the resulting end-effector motion is compared to a reconstruction based on the rotary motor encoder together with the utilized link kinematics. The reconstructed end-effector positions are shown in Figure 4.8. The maximum absolute errors are 4 mm in x- and y-direction as well as 8 mm in z-direction. The occurring higher errors compared to the much simpler 2D reconstruction from Section 4.2.4 are expected. Also, with the z-axis being almost exactly parallel to the optical axis of the camera the reconstruction of the z-coordinate has the smallest accuracy. The accuracy additionally varies for different end-effector poses. Still, the camera- and the encoder-based approach perform quite similar, which confirms the effectiveness of the presented 3D camera-based reconstruction. Finally, it should be noted that the marker pattern needs to be known very precisely as small deviations can lead to significant differences in the reconstruction results. Also, for a large camera distance compared to the depth of the object along the optical axis two solutions could be obtained [118] while one is correct

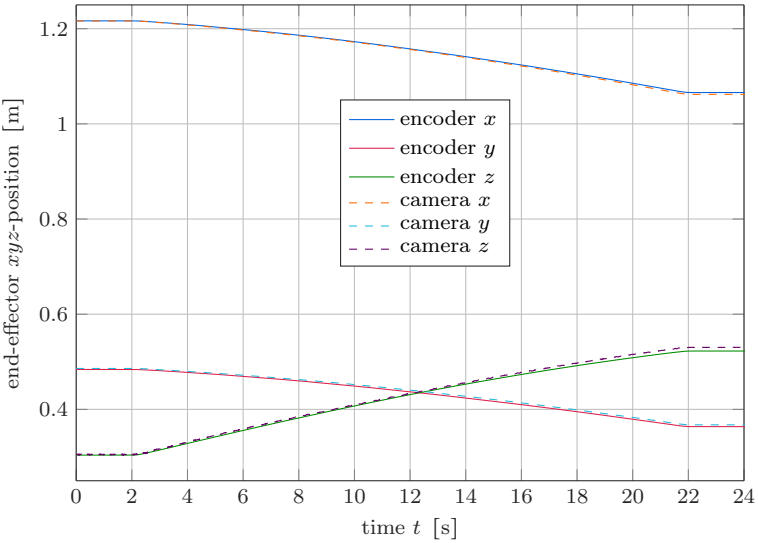


Figure 4.8: Encoder- versus camera-based 3D reconstruction for a rigid link 3.

and one is wrong. Therefore, the utilized marker pattern is not coplanar to further increase the robustness of the reconstruction.

4.3 Parameter Estimation

Section 3.2.1 shows that flexible link models being essentially purely based on data from CAD and FEM tools can describe the flexibilities quite accurately without any experimental system identification. Only for the damping such identifications are needed. However, there might be cases with further uncertain parameters such as an unknown mass of a handling object. If this mass cannot be identified it could introduce severe errors and instability in the later applied model-based control concepts.
In the literature, adaptive controllers have been applied for such cases. Feliu et al. [119] use an adaptive controller for a flexible single-link manipulator with changes in payload by estimating the variations in the eigenfrequency of the link. Pradhan and Subudhi [120] discuss a real-time adaptive controller for end-effector trajectory tracking of a robot with two flexible links and variable payloads. In contrast, in the presented approach the identification is not done online to prevent an initially large model mismatch and the connected potential instabilities when using elastic feedback signals. Instead, the system response

from a short excitation is used in a post-processing step to identify the new payload, i.e., handling object mass via a nonlinear flexible multibody model.

In the investigated scenario FLEXOR in a 2D configuration grasps a bottle. It is assumed that the position of its center of gravity relative to the gripper is known but its mass is assumed to be unknown. With all actuators at their zero positions, a fast motion of slider A of -15 cm is performed to excite significant oscillations. It should be noted that while the damping of the first eigenmode for the single flexible links could be measured quite accurately, it turns out that the overall system damping can nevertheless vary noticeably. This might be caused amongst others by variations in wiring and tubing. Therefore, the mass of the bottle being modeled as point mass and the linear damping of link 2 are determined via optimization. Here, the damping parameter of the first eigenmode of link 2 identified in Section 3.2.1 serves as initial guess to reduce the optimization time and to ensure convergence. For the optimization the nonlinear least squares method is used to minimize the sum of the squared error between the response of the experiment and the simulations. Here, the measurements of the rotary encoder in the middle of link 2 are used as response which are compared to the simulated angle β. For the simulations the measured actuator motions are used as kinematic inputs within a forward time integration of Equation (3.84).
The results for the optimized parameters for a case with and a case without the handling object mass are shown in Figure 4.9. This means that the case without actual handling object is also optimized to ensure an accurate reference mass. The difference between this reference mass and the optimized mass for the case with the handling object is then used as estimated bottle mass. It results in 578 g instead of the actual 630 g, being a difference of 52 g and a relative error of 8.3%.

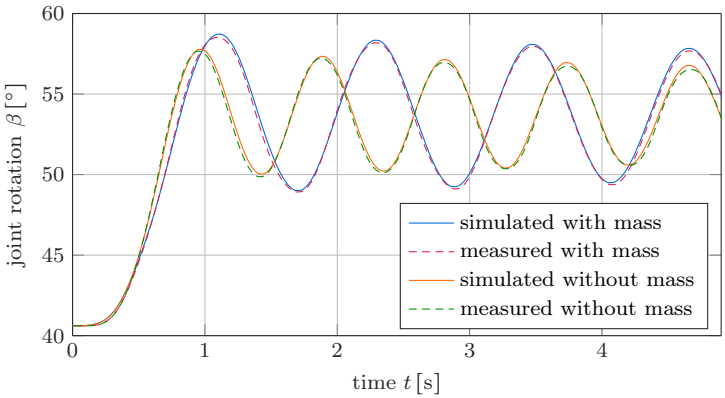


Figure 4.9: Estimation of handling object mass.

It can be concluded that the close match of the estimated and the real bottle mass as well as of the measurements and the simulations in Figure 4.9 compactly shows the applicability of flexible multibody models also in scenarios with initially unknown parameters. It is demonstrated that such parameters can change the system dynamics significantly, which renders an accurate identification essential for an effective and safe model-based control performance.

Chapter 5

End-Effector Trajectory Tracking Control

Chapter Contents

In the previous chapters, the flexible link parallel robot FLEXOR has been introduced. Also, modeling, inversion and solution techniques as well as state and end-effector output pose estimation have been discussed for such system types. In this chapter, these different parts are combined in order to track the end-effector trajectory of flexible link parallel robots, being the centerpiece of this research. Here, FLEXOR serves as main application example to experimentally validate the proposed approaches.

For the most part, the end-effector trajectory tracking control approaches are based on model inversion. An overview of the presented approaches is shown in Figure 5.1. The internal dynamics of the utilized models decides on how the controllers are implemented and for which applications they are suited. Therefore, the approaches are classified according to the existence and the stability of the internal dynamics, analogously to Figure 3.3. Throughout this research, the standard 2D setup of FLEXOR with free end-effector is the main application focus.

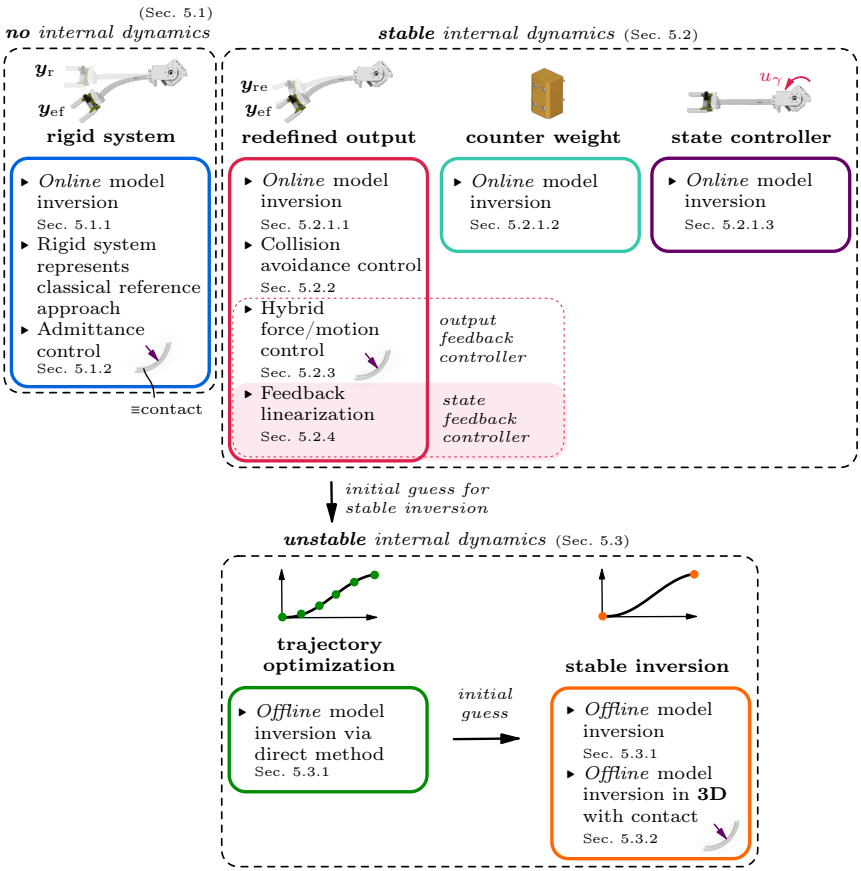


Figure 5.1: Overview of the presented end-effector trajectory tracking control approaches.

Nevertheless, Figure 5.1 shows that an admittance control, a hybrid force/motion control and a 3D case with contact are also considered.
In this chapter, firstly control approaches are discussed which rely on an equivalent rigid model without internal dynamics. Here, the rigid end-effector $\boldsymbol{y}_r$ is tracked instead of the exact end-effector $\boldsymbol{y}_{ef}$. Secondly, to enable classical inversion of flexible multibody models, via the solution of an IVP, three concepts are proposed to obtain a stable internal dynamics. Ensuring stability for a nonlinear time-variant system, such as the internal dynamics, is often difficult. Therefore, in this research, the stability of the internal dynamics is investigated locally for a constant output, i.e., by examining the linearized zero dynamics. For the considered robotic systems and trajectory tracking scenarios the stability of the

linearized zero dynamics even results in a stable internal dynamics. For FLEXOR, the control approaches without internal dynamics and with stable linearized zero dynamics, i.e., with stable internal dynamics, are not only computed online but are even real-time capable. Based on the concept where a redefined output $\boldsymbol{y}_{\mathrm{re}}$ close to the exact end-effector $\boldsymbol{y}_{\mathrm{ef}}$ is tracked, two output feedback controllers are applied. Here, feedback linearization additionally uses state feedback, which tends to be difficult to realize robustly within experiments for flexible link robots. Therefore, the focus of this chapter clearly lies on model inversion control. Thirdly, two concepts are discussed for an unstable internal dynamics, which is typically the case when tracking the exact end-effector of flexible link robots. Here, the model inversion needs to be pre-calculated offline. Such concepts are mainly interesting for repetitive tasks, since the solution is quite involved.
The section references depicted in Figure 5.1 show where the different approaches are introduced. Nevertheless, the rigid model inversion without internal dynamics is used throughout this chapter to compare the developed controllers, which use a flexible multibody model, to the classical rigid body approach. In most cases, this will emphasize the importance of using a flexible multibody model. Likewise, the redefined output concept is also compared to the controllers with unstable internal dynamics in order to show similarities and differences. Within experiments, this will demonstrate that concepts which introduce small modeling errors to obtain a stable internal dynamics often perform similarly effective as concepts without such errors.

5.1 No Internal Dynamics

To cope with an unstable internal dynamics caused by flexible links, which often occurs within end-effector trajectory tracking, a trivial approach is to neglect the flexibility. Thus, the end-effector of an equivalent rigid model can be tracked which has no internal dynamics. Instead of eliminating it, it can also be rendered stable by redefining the end-effector output of a flexible model to being the equivalent rigid end-effector output $\boldsymbol{y}_{\mathrm{r}}$. In this case the system is usually minimum phase with a stable internal dynamics, which could then also be counted to Section 5.2. This is related to the fact that for such an equivalent rigid end-effector the zero dynamics typically does not contain any rigid body motion but consists only of the stable dynamics of the structural oscillations, compare Seifried [68, p. 189]. For FLEXOR, both approaches lead to the same actuator motions within a model inversion. Only different forces and a different torque occur for the actuators which however cannot be passed to the utilized SIEMENS velocity controller. Therefore, both approaches can be used interchangeably.
In this section, an equivalent rigid model is inverted by solving the algebraic loop closing constraints and servo constraints as well as the corresponding first time derivatives via root finding. This corresponds to inverse kinematics of the fully

actuated equivalent rigid system, see Section 3.6.1. The purpose of this section is to show that it is typically inappropriate to use such a rigid model inversion for a free end-effector trajectory tracking scenario for flexible link robots. However, for HRI, such as manual guidance where the end-effector is in contact with a human hand, a rigid model inversion can be very effective and safe.

5.1.1 Trajectory Tracking

First, it shall be demonstrated that it is usually inadequate to use a rigid model inversion for end-effector trajectory tracking of a flexible link robot. The utilized control structure is shown in Figure 5.2. Here, a user input velocity is generated on the host computer via a game pad. It is sent to the real-time target for limitation as well as smoothing via a low-pass filter with a cutoff frequency at 1.4 Hz, giving the desired end-effector velocity trajectory $\dot{\boldsymbol{y}}_\mathrm{d}$ as well as its time derivative $\ddot{\boldsymbol{y}}_\mathrm{d}$. If the rigid model inversion is purely based on kinematics then $\ddot{\boldsymbol{y}}_\mathrm{d}$ is not needed. The filter is crucial as the input velocities applied by the game pad can be extremely steep with edges being not feasible. The forward Euler method is then used to integrate the velocity signals giving $\boldsymbol{y}_\mathrm{d}$. For the standard 2D case of FLEXOR it contains the desired xy-position and the desired rotation angle around the z-axis according to $\boldsymbol{y}_\mathrm{d} = [x_\mathrm{d},\, y_\mathrm{d},\, \vartheta_\mathrm{d}]^\mathrm{T}$. For the rigid model inversion the rigid end-effector is used as tracking output, i.e., $\boldsymbol{y}_\mathrm{o} = \boldsymbol{y}_\mathrm{r}$. Analogously to $\boldsymbol{y}_\mathrm{d}$, $\boldsymbol{y}_\mathrm{r} = [x_\mathrm{r},\, y_\mathrm{r},\, \vartheta_\mathrm{r}]^\mathrm{T}$ here contains the corresponding rigid end-effector 2D position and rotation, represented in the inertial frame with respect to its origin O. The rigid inversion thus uses $n_\mathrm{s} = 3$ servo constraints and transforms the desired end-effector trajectory to actuator motions, which need to be realized by the robot. Here, disturbances such as friction in the actuators can significantly reduce the control performance. Therefore, the desired actuator trajectories $\boldsymbol{q}_\mathrm{a,d}$ on position and $\dot{\boldsymbol{q}}_\mathrm{a,d}$ on velocity level are sent to the actuator cascade control for each actuator, which accurately realizes the desired motions. Here, the voltages $\boldsymbol{u}_\mathrm{v}$ and the currents $\boldsymbol{i}$ incorporate the corresponding u_v and i element of each actuator. The xy-position x_ef and y_ef as well as the rotation ϑ_ef around the z-axis of the exact end-effector $\boldsymbol{y}_\mathrm{ef} = [x_\mathrm{ef},\, y_\mathrm{ef},\, \vartheta_\mathrm{ef}]^\mathrm{T}$, which are represented in the inertial frame with respect to its origin O, are reconstructed from two passive markers, see Section 4.2.4. This yields $\hat{\boldsymbol{y}}_\mathrm{ef}$ being used to judge the control performance.

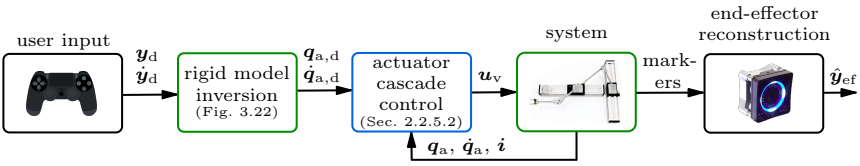


Figure 5.2: Control structure of the rigid model inversion.

The control structure is applied to FLEXOR within an experiment, which is visualized in Figure 5.3. The rigid model inversion is straightforward and real-time capable but large undesired oscillations are clearly visible. The corresponding camera-based measurements are shown in Figure 5.4. Here, an end-effector motion solely in y-direction is commanded via the game pad. Still significant oscillations of several centimeters occur in the x-direction. The maximum absolute rotation error of the end-effector amounts to 13°.

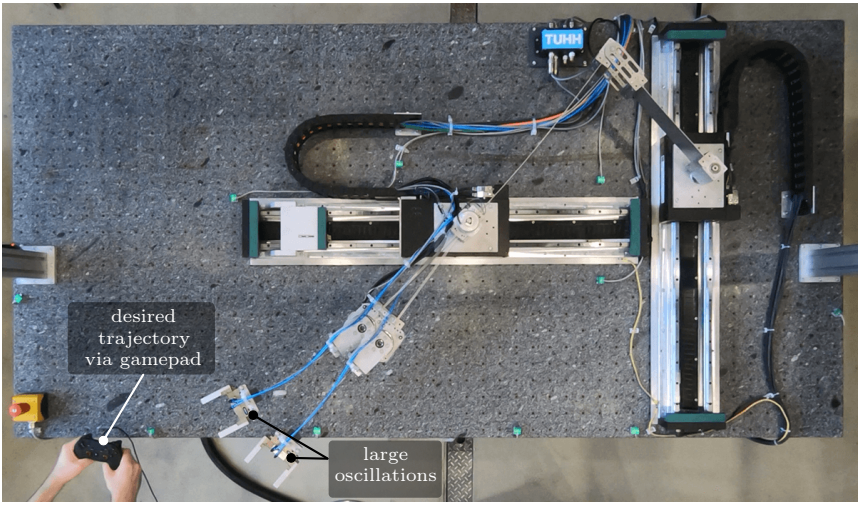


Figure 5.3: Rigid model inversion leads to significant oscillations for a free end-effector.

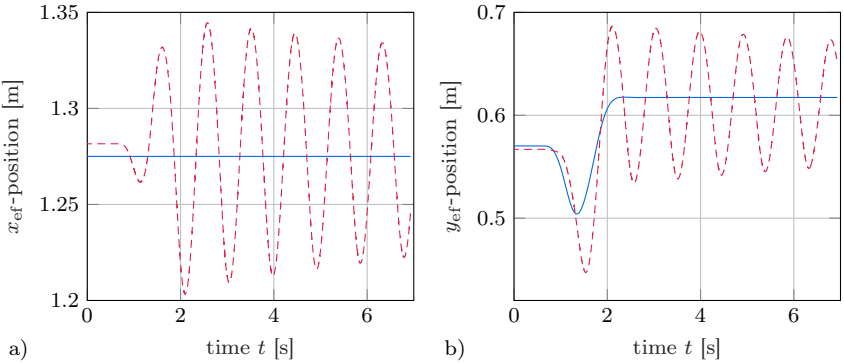


Figure 5.4: End-effector trajectory tracking experiments based on the rigid model inversion: desired (——) and measured (- - - -) end-effector position.

It can be concluded that the rigid model inversion tends to perform very poorly in trajectory tracking scenarios for a free end-effector of flexible link robots. This motivates the application of a flexible multibody model in Sections 5.2 and 5.3.

5.1.2 Manual Guidance

In Section 5.1.1, the poor performance of a rigid model inversion for Flexor has been presented for trajectory tracking of a free end-effector. However, the robust and straightforward behavior can be advantageous for contact scenarios involving HRI, such as within manual guidance, i.e., when a human pushes or drags the robot onto a trajectory by hand. Using a rigid model inversion for manual guidance of a flexible link robot is reasonable since:

- it is safe in contrast to a flexible model inversion which is unstable or could become unstable when guiding the robot, putting the human worker at risk;
- the human prevents or damps most oscillations automatically by keeping contact. This leads to oscillations of significantly higher frequencies and typically much smaller amplitudes;
- the human corrects occurring errors in the position and rotation tracking automatically which makes a flexible model inversion unnecessary;
- oscillations after the guidance when the contact is released can be handled by switching to an oscillation damping controller of Chapter 6.

Manual guidance can be used amongst others to teach trajectories for repetitive tasks or to use a robot as helping hand for object manipulation. Robots with flexible links are especially advantageous for such HRI scenarios, as they have a smaller weight and a reduced stiffness which leads to an increased safety.

Robots with flexible links are inherently compliant but the steady-state end-effector pose can usually not be reasonably changed by manual contact. Therefore, the idea is to impose a compliant behavior onto the end-effector pose to enable manual guidance. In this regard, an indirect force control law shall be used, i.e., which does not explicitly control the contact force and torque to desired values. Popular and very suitable methods are impedance and admittance control which are very similar. Here, the end-effector error is related to the contact force by a mass-spring-damper system. Details can be found in the comprehensive chapter by Villani and De Schutter [121].
As opposed to rigid robots, impedance control for flexible link robots has been rarely covered in the literature [122–124]. These three references also only include simulations. Likewise, very few publications exist on HRI for flexible link robots especially for manual guidance. A rare result is documented by Malzahn and Bertram [125], who use an admittance controller for manual guidance of a flexible link serial robot within experiments. It is realized on joint basis using strain measurements at the links. In contrast, the presented research considers a flexible

link parallel robot within experiments and an admittance control is applied to the end-effector, which is manually guided based on a force/torque sensor.

For FLEXOR in the considered 2D case, the admittance controller has the form

$$\begin{bmatrix} m_\mathrm{p} & 0 & 0 \\ 0 & m_\mathrm{p} & 0 \\ 0 & 0 & m_\mathrm{r} \end{bmatrix} (\ddot{\boldsymbol{y}}_\mathrm{c} - \ddot{\boldsymbol{y}}_\mathrm{d}) + \begin{bmatrix} d_\mathrm{p} & 0 & 0 \\ 0 & d_\mathrm{p} & 0 \\ 0 & 0 & d_\mathrm{r} \end{bmatrix} (\dot{\boldsymbol{y}}_\mathrm{c} - \dot{\boldsymbol{y}}_\mathrm{d}) + \begin{bmatrix} c_\mathrm{p} & 0 & 0 \\ 0 & c_\mathrm{p} & 0 \\ 0 & 0 & c_\mathrm{r} \end{bmatrix} (\boldsymbol{y}_\mathrm{c} - \boldsymbol{y}_\mathrm{d}) = \begin{bmatrix} F_{\mathrm{s},x} \\ F_{\mathrm{s},y} \\ T_{\mathrm{s},z} \end{bmatrix}. \tag{5.1}$$

This is a system of three decoupled mass-spring-dampers, one for the end-effector position in x-direction, one for the y-direction as well as one for the rotation ϑ around the z-axis. Equation (5.1) is treated as IVP and integrated with the forward Euler method to obtain the reference trajectories $\boldsymbol{y}_\mathrm{c}$ and $\dot{\boldsymbol{y}}_\mathrm{c}$. These references are then tracked, with the rigid end-effector used as tracking output, i.e., $\boldsymbol{y}_\mathrm{o} = \boldsymbol{y}_\mathrm{r}$, instead of the desired trajectory $\boldsymbol{y}_\mathrm{d}$ and its time derivatives. Thus, the desired trajectory, which needs to be provided by the user, is smoothly adapted based on measurements of the force/torque sensor from Section 2.2.3.3. Here, $F_{\mathrm{s},x}$, $F_{\mathrm{s},y}$ and $T_{\mathrm{s},z}$ denote the measured forces and the torque at the sensor in the corresponding directions of the inertial system I.
The complete control structure is shown in Figure 5.5. The algorithm is denoted as admittance control since no end-effector feedback is used within the mass-spring-damper system. Still, in contrast to the classical admittance control discussed by Villani and De Schutter [121], the usage of a subsequent end-effector feedback is neglected. The reason is that feedback of the exact end-effector of FLEXOR destabilizes the system due to the non-minimum phase system property. Therefore, this feedback is replaced by a rigid model inversion and the actuator cascade controller. This ensures that the equivalent rigid end-effector tracks the trajectories which come from the mass-spring-damper system. The link elasticity is only needed to obtain the end-effector, i.e., force/torque sensor rotation $\hat{\vartheta}_\mathrm{ef}$.

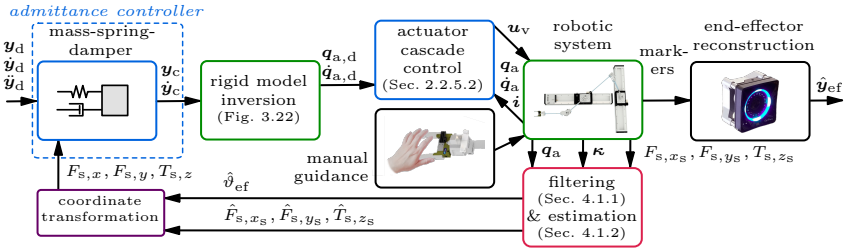


Figure 5.5: Control structure with admittance controller applied to the end-effector of a flexible link parallel robot.

This is then used to transform the filtered forces $\hat{F}_{\mathrm{s},x_\mathrm{s}}$ and $\hat{F}_{\mathrm{s},y_\mathrm{s}}$ from coordinates of the sensor system to the inertial system. The direction of the torque $\hat{T}_{\mathrm{s},z_\mathrm{s}}$ does not change with the 2D force/torque sensor rotation.

The conducted experiment is illustrated in Figure 5.6. First, a motion is performed in y-direction, then in x-direction. Afterwards, the gripper is rotated via a torque around the z-axis and a torque around the force/torque sensor y-axis results in opening the gripper to place the handling object. Via the sensor, the end of the manual guidance can be detected in order to switch, e.g., to an oscillation damping controller. The selected parameters within the experiment are shown in Table 5.1. Since the spring constants c_p and c_r are set to zero the value $\boldsymbol{y}_\mathrm{d}$ is not used and the robot will not return to its starting pose, when assuming that $\boldsymbol{y}_\mathrm{d}$ is constant. The desired velocities and accelerations are permanently set to zero, i.e., $\dot{\boldsymbol{y}}_\mathrm{d} = \ddot{\boldsymbol{y}}_\mathrm{d} = \boldsymbol{0}$, which is also needed for a typical guidance scenario. Furthermore, the control parameters for the rotation need to be scaled down as the measured torque values are much smaller than the occurring forces. This depends on the lever arm between sensor and guidance contact point and needs to be chosen appropriately for every robot. In the considered experiment, maximum absolute curvatures of around $0.15\,\mathrm{m}^{-1}$ for κ_2 at link 2 and of around $0.4\,\mathrm{m}^{-1}$ for κ_3 at link 3 occur. Especially for link 2, this is a very small value which is directly connected to the admittance control gains. They are tuned such that the compliant movement of the robot is not too large but significant enough to keep the deformations small. This is confirmed by Figure 5.6, where hardly any deformations of the links can be seen.
The corresponding measurements are shown in Figure 5.7. In the path plot of Figure 5.7a) the desired motion applied by the human hand is compared to the actually measured end-effector position. The difference at the start point is related to measurement and modeling errors. The tracking differences when moving are however related to the fact that a rigid model inversion is used, that does not take the deformations of the links into account which are caused by the human hand. This difference should not be considered as error since the objective here is not to exactly track the desired motion, but to ensure that the directions and the shape are similar. As this is the case, the human user can simply adjust the position and rotation as needed to compensate for such differences.
In the end-effector trajectories of Figure 5.7b) it can be observed that oscillations only occur after the guidance when the contact is released at around 14 s. This can be detected with the force/torque sensor to switch to an oscillation damping controller. Additional dead zones are applied to the force/torque sensor measure-

Table 5.1: Admittance control parameters for manual guidance.

	mass	**damper**	**spring**
x, y-position	$m_\mathrm{p} = 5\,\mathrm{kg}$	$d_\mathrm{p} = 20\,\mathrm{kg/s}$	$c_\mathrm{p} = 0\,\mathrm{kg/s^2}$
rotation ϑ	$m_\mathrm{r} = m_\mathrm{p}/(20\,\mathrm{rad/m^2})$	$d_\mathrm{r} = d_\mathrm{p}/(20\,\mathrm{rad/m^2})$	$c_\mathrm{r} = c_\mathrm{p}/(20\,\mathrm{rad/m^2})$

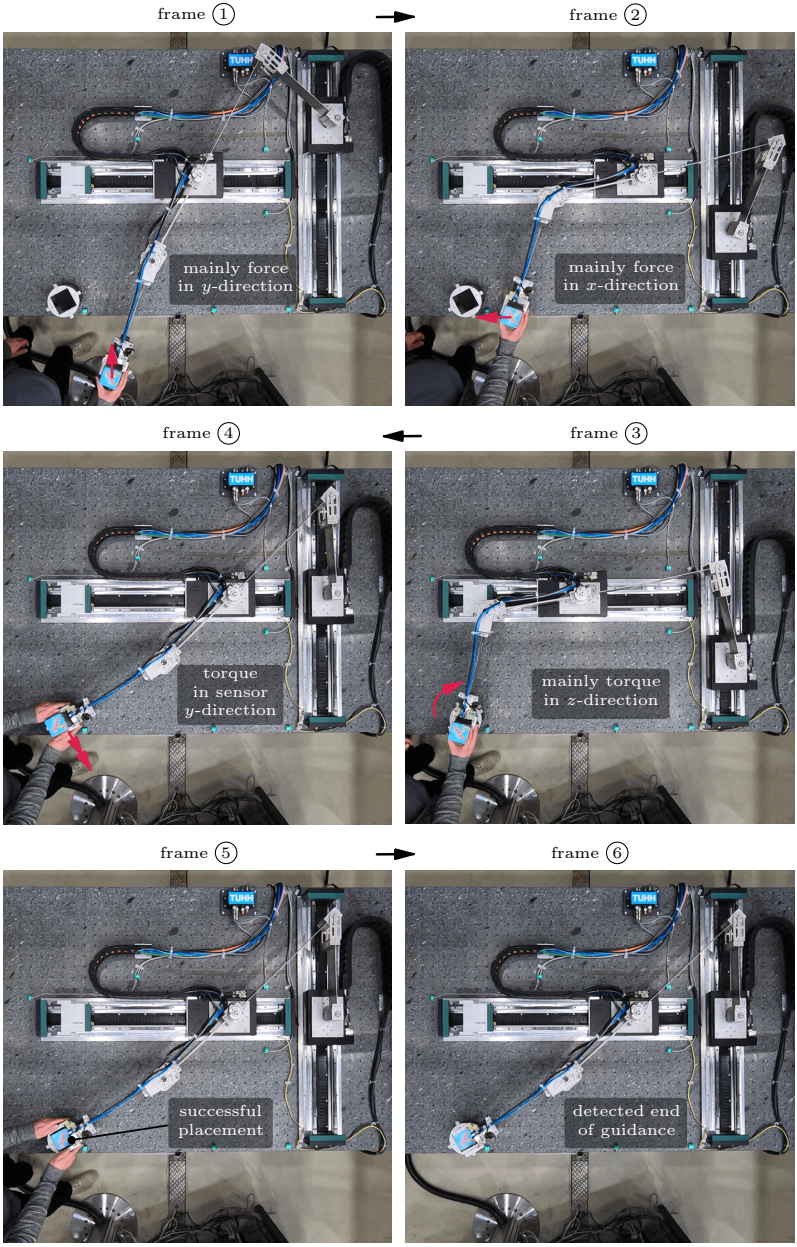


Figure 5.6: Manual guidance within an object placement scenario.

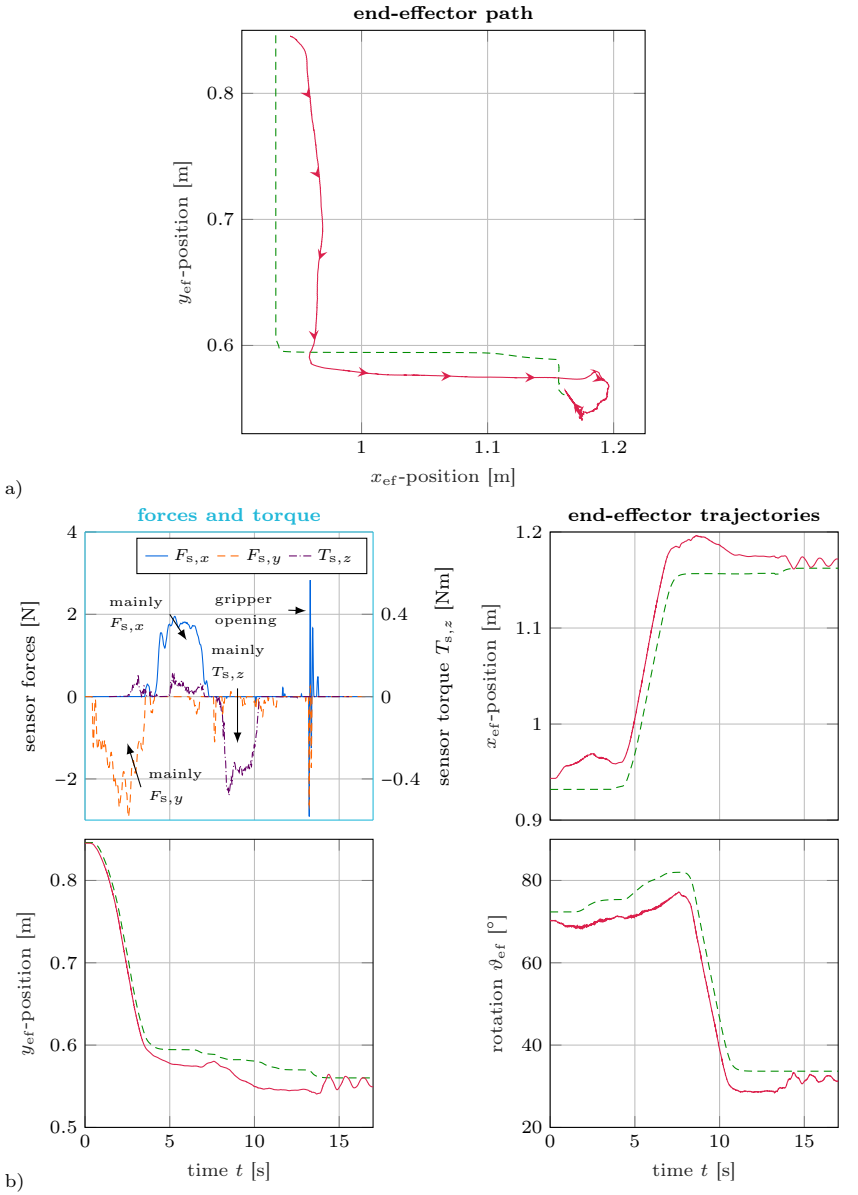


Figure 5.7: Manual guidance experiment with a) the end-effector path and b) corresponding trajectories, for the desired end-effector (- - - -) and the measured end-effector (➤——, ——).

ments at 1 N and 0.15 Nm. This stops the input to the admittance controller and ensures that no small forces and torque or noise excite the system. These dead zones are the reason for the extremely straight desired motion in Figure 5.7a). They are also applied in the forces and torque plot in Figure 5.7b), which are the signals used within the admittance control. When comparing the end-effector trajectories with the force and torque trajectories it can be seen that they are directly related as intended. First, mainly a motion in y-direction is introduced through the force $F_{\mathrm{s},y}$, then in x-direction via $F_{\mathrm{s},x}$ and afterwards mainly a rotation around the z-axis is commanded via the torque $T_{\mathrm{s},z}$.

In conclusion, a rigid model inversion together with an admittance controller can be an effective approach for manual guidance also for robots with highly flexible links. In the following sections it is shown how trajectories, which could be generated via such a manual guidance scenario, can be tracked closely also for a free end-effector of a flexible link robot.

5.2 Stable Internal Dynamics

End-effector trajectory tracking for robots with flexible links has been extensively investigated in the last decades [11, 126, 127]. However, a significant part of the literature reports just theoretical and numerical results or experiments for robots with only a single flexible link. Amongst others, Kwon and Book [128] use an offline computed inverse dynamics control for a flexible single-link manipulator being limited to linear systems. End-effector trajectory tracking for a two-link parallel robot with one flexible link is conducted by Burkhardt et al. [46] based on the inversion of a dynamic flexible multibody model. The calculations are also performed offline, based on stable inversion [101]. Still, in [23–25, 129] serial robots with two flexible links are controlled experimentally but based on rigid inverse kinematics. Also, Zhang et al. [13] use input shaping and inverse kinematics for a parallel robot with three flexible links within experiments.
The focus of this section is to extend the related literature by using a dynamic flexible multibody model inversion in real-time for end-effector trajectory tracking of a parallel robot with flexible links in experiments. However, flexible link robots are typically non-minimum phase for the exact end-effector as output, i.e., the zero dynamics is unstable as well as the internal dynamics. Classical inversion then leads to unbounded results. Stable inversion of such systems usually results in a noncausal solution which is bounded but it has to be computed offline as a two-point BVP [68, 101]. Therefore, in Section 5.2.1 three approaches are proposed to render initially non-minimum phase flexible link robots minimum phase. They result even in a stable internal dynamics for the considered scenarios for Flexor and enable real-time model inversion based on classical inversion via forward time integration. Experiments then show that using such a stable flexible model inversion, instead of rigid inverse kinematics often applied in the literature

as well as in Section 5.1.1, can highly improve the end-effector trajectory tracking control performance.
Flexible model inversion in real-time paves the way for further scenarios where:

- the desired end-effector trajectories are provided in real-time by a user;
- the desired end-effector trajectories need to be adapted in real-time;
- errors in the end-effector pose shall be controlled in a feedback loop.

From the three approaches to obtain a minimum phase system the most universal, being output redefinition, is implemented within such scenarios. This includes collision avoidance in Section 5.2.2 as well as hybrid force/motion control in Section 5.2.3. Feedback linearization using a redefined output is also considered in Section 5.2.4, where however algebraic model equations are utilized since the state comes from a UKF, see Section 4.1.3.

5.2.1 System Optimization

In contrast to previous works, which are typically based on offline pre-calculations or rigid body inverse kinematics for online control of flexible link robots, a flexible multibody model shall be inverted online for end-effector trajectory tracking. Due to the flexible links, the inverse model is a dynamic system itself. The dynamics of this inverse model corresponds to the internal dynamics. To perform the inversion of the flexible multibody model online the unstable internal dynamics, being typically the case when using the exact end-effector $\boldsymbol{y}_{\mathrm{ef}}$ as tracking output $\boldsymbol{y}_{\mathrm{o}}$, needs to be rendered stable. Together with the modal reduction, the floating frame of reference approach and the transformation from DAEs to ODEs from Chapter 3 even real-time capability can then be provided for Flexor.
Butterworth et al. [130] describe systematic ways for linear non-minimum phase systems to obtain stable approximate inverse models. However, for nonlinear systems no general design methodology exists. Therefore, in this section alternative ways are investigated to obtain a stable internal dynamics.

As mentioned on page 47, it is often difficult to ensure stability for a nonlinear time-variant system such as the internal dynamics, which is driven by the desired trajectory. As a result, stability analysis is often performed only for the zero dynamics [131]. For the investigated systems it turns out that the stability property of the zero dynamics is independent of the considered robot poses. Therefore, only a single working point is considered where the stability analysis is done locally by linearizing the zero dynamics. The linearization can be executed, e.g., with finite differences applied to Equation (3.63) for a constant output. For Flexor in 2D, the actuator middle positions $a = b = \gamma = 0$ and zero deformations $\boldsymbol{q}_{\mathrm{e}} = \boldsymbol{0}$ with the system being at rest are used as linearization point in all following investigations. Since the algebraic constraints $\boldsymbol{c} = \boldsymbol{0}$ and $\boldsymbol{s} = \boldsymbol{0}$ lead to dependent coordinates within $\boldsymbol{q}$ one obtains two zero eigenvalues for each constraint equation. As they do not contribute to the zero dynamics and its stability, they are removed from the

results. It should be noted that for designs with stable linearized zero dynamics large undesired oscillations can still occur within the actuator trajectories. Thus, the smoothness and robustness of the inversion should be verified within time simulations for the optimized designs. Since a dynamic real-time model inversion shall be utilized, which does not use state feedback, one can quite safely rely on time simulations. In contrast, concepts such as feedback linearization, where an estimated state is fed back often using disturbed elastic measurements, tend to demand a more conservative design for experiments.
Three different system optimization approaches are now discussed to obtain a stable linearized zero dynamics being summarized in Figure 5.8 and applied to Flexor. Firstly, one can redefine either the input or the output to gain a minimum phase system, i.e., a stable zero dynamics. The former is rarely done as it typically involves hardware changes. For instance, in [132, 133] the torque application point is relocated for a single flexible link with an experimental transmission mechanism. The more straightforward output redefinition is considered in the following by directly weighting the elastic deformations and rotations of the links. Secondly, a change of the system dynamics via a counter weight is discussed, which is attached at an advantageous location on the robot so that it has only a minor influence on the significant bending eigenmodes. Lastly, the rotational degree of freedom of the end-effector is used to stabilize the zero dynamics with a small motion of the rotary motor. The following content of this

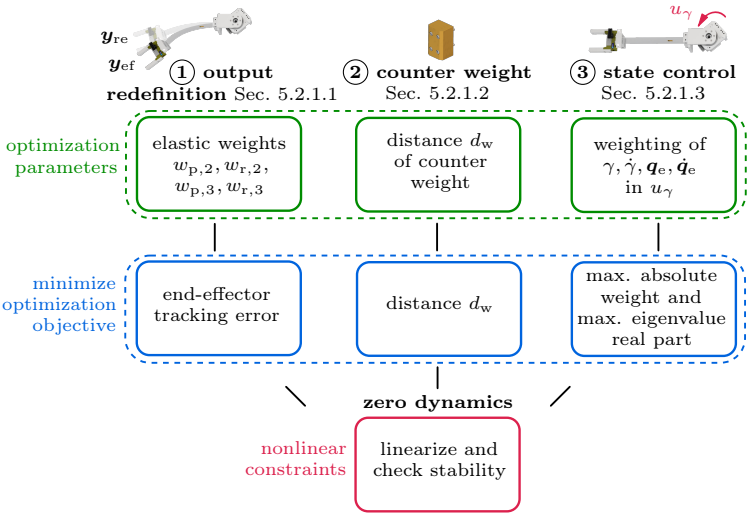


Figure 5.8: Overview of the system optimization approaches to render the linearized zero dynamics stable.

section is highly based on the author's publication [32] where more details can be found.

5.2.1.1 Output Redefinition

Output redefinition is a well-known approach to obtain a minimum phase system and only involves a different output selection. It has been applied to flexible link robots before. For example, De Luca and Lanari [134] vary the output along a single flexible link to achieve the minimum phase property. Another application to a flexible single-link robot is provided by Chodavarapu and Spong [135], who define a virtual angle of rotation including a weighted value of the tip slope in order to obtain a minimum phase system. In [15–17] serial robots with two flexible links are considered within trajectory tracking simulations using a linearly combined output. As opposed to this, the presented research does not use a linear output approximation but directly weights the elasticity in the nonlinear output function. Furthermore, this approach is experimentally applied to FLEXOR.

Instead of choosing the exact end-effector $\boldsymbol{y}_{\mathrm{ef}}$ as output $\boldsymbol{y}_{\mathrm{o}}$ a so-called *redefined end-effector* $\boldsymbol{y}_{\mathrm{re}}$ shall be found which yields a minimum phase system. A straightforward approach is to choose an equivalent rigid end-effector $\boldsymbol{y}_{\mathrm{r}}$, which typically results in such a minimum phase system as described on page 119. This however comes at the cost of a poor tracking performance. Thus, the objective is to find an output $\boldsymbol{y}_{\mathrm{re}}$ which is close to the exact end-effector $\boldsymbol{y}_{\mathrm{ef}}$ and additionally renders the system minimum phase. Since for flexible link robots the elasticity usually introduces an unstable internal dynamics for the exact end-effector as output, a promising approach is to weight this link elasticity in the output function. This also results in zero steady-state redefinition error for scenarios without static deflections, e.g., without gravitational effects and without contact to the environment. It should be noted that this weighting is similar to the single scalar weighting factor μ within the so-called *μ-tip rate*, which results for the flexible systems in [136–138] even in passivity.
A weighting of the elasticity has also been done by the author in [28, 31] for a single flexible link. The considered robot FLEXOR, however, has two flexible links in series for the standard 2D setup and the end-effector rotation is also part of the system output. Instead of weighting each of the two elastic coordinates with one parameter it turns out that using separate weights $w_{\mathrm{p},2}$, $w_{\mathrm{p},3}$ for the deflections and $w_{\mathrm{r},2}$, $w_{\mathrm{r},3}$ for the elastic rotations can significantly improve the tracking performance. The position $\boldsymbol{r}_{\mathrm{re}}$ and rotation ϑ_{re} of the redefined end-effector

output $\boldsymbol{y}_{\mathrm{re}} = [\boldsymbol{r}_{\mathrm{re}}^{\mathrm{T}}, \vartheta_{\mathrm{re}}]^{\mathrm{T}}$ follow as

$$\begin{aligned}\boldsymbol{r}_{\mathrm{re}} = &\begin{bmatrix} o+b \\ 0 \end{bmatrix} + \begin{bmatrix} \cos(\beta_{\mathrm{R}}) & -\sin(\beta_{\mathrm{R}}) \\ \sin(\beta_{\mathrm{R}}) & \cos(\beta_{\mathrm{R}}) \end{bmatrix} \begin{bmatrix} \ell_{2_2} \\ w_{\mathrm{p},2}\phi_2 q_{\mathrm{e},2} \end{bmatrix} \\ &+ \begin{bmatrix} \cos(\gamma_{\mathrm{R}}) & -\sin(\gamma_{\mathrm{R}}) \\ \sin(\gamma_{\mathrm{R}}) & \cos(\gamma_{\mathrm{R}}) \end{bmatrix} \begin{bmatrix} \ell_3 \\ w_{\mathrm{p},3}\phi_3 q_{\mathrm{e},3} \end{bmatrix}, \end{aligned} \tag{5.2a}$$

$$\vartheta_{\mathrm{re}} = \gamma_{\mathrm{R}} + w_{\mathrm{r},3}\psi_3 q_{\mathrm{e},3}. \tag{5.2b}$$

When all elastic weighting parameters $w_{\mathrm{p},2}$, $w_{\mathrm{r},2}$, $w_{\mathrm{p},3}$ and $w_{\mathrm{r},3}$ are set to 1, then $\boldsymbol{y}_{\mathrm{re}}$ corresponds to the exact end-effector output $\boldsymbol{y}_{\mathrm{ef}}$. This means that the term $\boldsymbol{r}_{\mathrm{re}}$ yields the exact end-effector position $\boldsymbol{r}_{\mathrm{ef|I}}$ in 2D, represented in the inertial frame I with respect to its origin O. Also, in this case ϑ_{re} describes the rotation ϑ_{ef} of the exact end-effector with respect to the inertial frame. When all four elastic weighting parameters are set to 0, one obtains an equivalent rigid end-effector $\boldsymbol{y}_{\mathrm{r}}$. In Equation (5.2a), the parameter o denotes the offset in $x-$direction between the inertial frame and the zero position of the coordinate b, see Figure 3.11. The parameters ℓ_3 and ℓ_{2_2} are the undeformed lengths of link 3 and of the left part of link 2. The angles β_{R} and γ_{R} describe the rotation of the second and the weighted rotation of the third floating frame of reference with respect to the inertial frame. Here, β_{R} corresponds to the angle β of an equivalent rigid system. The angle γ_{R} consists of β_{R}, the elastic rotation of link 2 at the rotary motor weighted by $w_{\mathrm{r},2}$ and of the actuator angle γ, i.e.,

$$\gamma_{\mathrm{R}} = \beta_{\mathrm{R}} + w_{\mathrm{r},2}\psi_2 q_{\mathrm{e},2} + \gamma. \tag{5.3}$$

The shape functions for the deflections in the local $y-$directions and for the elastic rotations around the z-axis, evaluated at the tips of link 2 and 3, are denoted by ϕ and ψ with corresponding indices.

The objective is now to find a parameter setting for the four elastic weights such that the linearized zero dynamics becomes stable at the considered working point and such that a minimal end-effector tracking error is introduced. Here, the error between the exact and redefined end-effector is calculated for a static deformation scenario caused by a force of 5 N applied in negative y-direction at the end-effector. To obtain a scalar error measure the translational and rotational errors are combined, whereby the rotational error is scaled by the factor 2 mm/1°. Using a brute-force search in the weighting parameter interval $[0, 2]$ results in $w_{\mathrm{p},2} = 0.78$, $w_{\mathrm{r},2} = 0.92$, $w_{\mathrm{p},3} = 0.86$ and $w_{\mathrm{r},3} = 0.50$, which also leads to smooth and robust time simulations of the inverse model.

In order to provide an impression of the highly nonlinear problem the three weighting parameters $w_{\mathrm{p},2}$, $w_{\mathrm{p},3}$ and $w_{\mathrm{r},2}$, which influence the end-effector position, are varied. At the same time the weight $w_{\mathrm{r},3}$ is changed in the interval $[0, 2]$ to check if the linearized zero dynamics can become stable for the different combinations of the other three weights. The obtained boundary of the stable region is shown in Figure 5.9. This stable region is large for the three weights being between 0 and 1. Only for values close to 1 the linearized zero dynamics

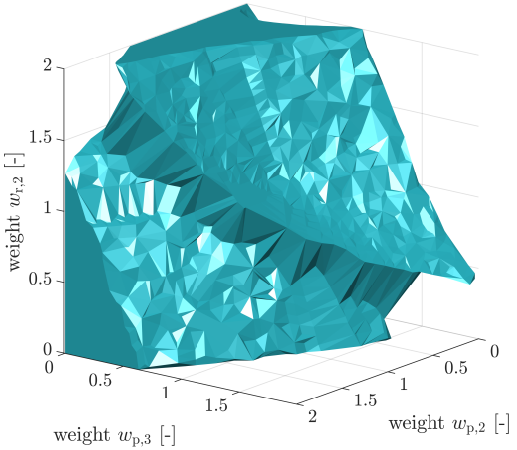


Figure 5.9: Region with stable linearized zero dynamics for the redefined output.

becomes unstable. For weights above 1 no general statement can be made as the shape is very irregular.

5.2.1.2 Counter Weight

A second concept is to change the system itself, such as by modifying the mass distribution of the robot, to render it minimum phase. Seifried [69] uses small counter weights in passive joint rigid manipulators to ensure the minimum phase property. In the work of Bastos et al. [18], a rather large mass close to the end-effector turns a passive joint rigid manipulator minimum phase. The flexible single-link robot considered by De Luca [19] becomes minimum phase when increasing the tip inertia. The idea of changing the mass distribution is now applied to Flexor. Instead of adding mass and inertia close to the end-effector similar to [18, 19], the presented solution adds a small counter weight near the actuators on the robot base. Its effectiveness is later confirmed within experiments which has not been done within the cited references.

The design and placement of such a counter weight is nontrivial and typically needs to be supported by a brute-force search or optimization. Addition of mass has been tested for the connections of the flexible links, i.e., for the joints on the sliders, the joint at O_2 connecting link 1 and 2, the rotary motor and the end-effector. It turns out that for reasonable parameters only a placement of mass either at the joint at O_2 or at the rotary motor can make the system minimum phase, whereas mass at O_2 has a much larger impact. Moreover, attaching mass at O_2 in negative x_2-direction is much more efficient than in positive or negative y_2-direction. A placement in positive x_2-direction does not affect the stability of the zero dynamics. The utilized counter weight consists of 1 cm thick plates made

of brass. Its high density makes it very compact and its color helps to differentiate it from the original robot. For the selected material and the shape matching the joint, the required distance d_w between the center of gravity of the counter weight and O_2 can be calculated. Four such plates turn the system minimum phase for the exact end-effector as output with a robust behavior within time simulations of the inverse model. Thus, no additional output redefinition is necessary. The resulting counter weight is shown in Figure 5.10, having the mass $m_w = 1.334\,\text{kg}$, the inertia $I_{w,zz} = 450.3\,\text{kg}\,\text{mm}^2$ and the distance $d_w = 43.3\,\text{mm}$. Due to the position of the counter weight and its small mass, only a minor influence occurs on the significant bending eigenmodes of the robot.
Finally, it should be noted that one does not need to actually attach the counter weight to the real system. By only adding the weight in the model inversion control a small modeling error is introduced, similar to output redefinition, without physically changing the system, see Section 5.2.1.4. This can be helpful if changes to the real system are difficult to perform or undesired. In this research, this modeling error does not introduce any stability issues since no feedback of the robot state or output is used within the model inversion for the counter weight case.

5.2.1.3 State Control with Kinematic Redundancy

As third approach a controller shall be used to stabilize the internal dynamics. This is done by Das et al. [139] for a quadrotor using a robust control term. For flexible link robots almost no control approaches exist to make such systems

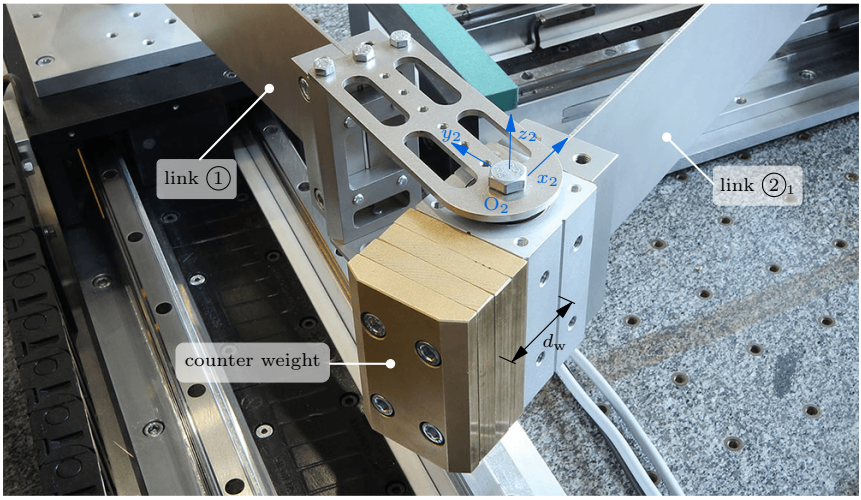


Figure 5.10: Counter weight.

minimum phase. A rare result is documented by Wang and Chen [140], who stabilize the unstable part of the zero dynamics with a controller within simulations for a flexible single-link manipulator. The control approach shows macroscopic tracking errors. In contrast, an approach is introduced in the following which can be used for exact end-effector tracking with only numerical errors for flexible multi-link robots with kinematic redundancy.

For FLEXOR kinematic redundancy exists when not all three servo constraints need to be fulfilled. This could be the case for rotationally symmetric handling objects where the end-effector rotation might not be of interest. The obtained redundancy can then be used to make the system minimum phase.
Thus, one constraint is removed from Equation (3.41c) reducing the number of servo constraints to $n_\mathrm{s} = 2$. With these two servo constraints only the end-effector position but not its rotation shall be tracked. For FLEXOR a stable linearized zero dynamics typically leads to a stable internal dynamics. Therefore, the idea is now to find a linear controller for the rotary motor input of the form

$$u_\gamma = \left(\sum_{i=1}^{2f} w_{\gamma,i} x_i \right) \Big/ K_\mathrm{R}, \tag{5.4}$$

which stabilizes the linearized zero dynamics. Here, K_R is the motor constant, x_i are the elements of the state vector $\boldsymbol{x} = [\boldsymbol{q}^\mathrm{T}, \dot{\boldsymbol{q}}^\mathrm{T}]^\mathrm{T}$ and $w_{\gamma,i}$ are the weights, i.e., control gains to be optimized. Equation (5.4) internally feeds back the state within the model inversion. This means that the controller feedback is only present within the inverse model and no measurements from the physical system are needed. To implement this control approach one needs to separate the input matrix $\boldsymbol{B}$ and the control inputs $\boldsymbol{u} = [u_\mathrm{a}, u_\mathrm{b}, u_\gamma]^\mathrm{T}$ according to

$$\boldsymbol{B}\boldsymbol{u} = \bar{\boldsymbol{B}}_\ell \underbrace{\begin{bmatrix} u_\mathrm{a} \\ u_\mathrm{b} \end{bmatrix}}_{\bar{\boldsymbol{u}}_\ell} + \bar{\boldsymbol{b}}_\gamma u_\gamma(\boldsymbol{q}, \dot{\boldsymbol{q}}). \tag{5.5}$$

For solving the inverse model it is again transformed from DAEs to ODEs. But now the reduced input matrix $\bar{\boldsymbol{B}}_\ell$ instead of $\boldsymbol{B}$ needs to be used to obtain the projection matrix and the term $\bar{\boldsymbol{b}}_\gamma u_\gamma$ is not canceled by this projection. Details can be found in Section 3.5.2, where $\bar{\boldsymbol{B}}_\mathrm{r} = \bar{\boldsymbol{b}}_\gamma$ and $\bar{\boldsymbol{u}}_\mathrm{r} = u_\gamma$ hold for the considered application.
Now, the weights $w_{\gamma,i}$ are varied via MATLAB's genetic algorithm command `ga`. Since feeding back a, b, α and β prevents the robot to come to rest at the end point and as their time derivatives typically did not improve the results, only the angle γ and the elastic coordinates $q_{\mathrm{e},2}$, $q_{\mathrm{e},3}$ as well as their time derivatives are used within the optimization. Here, the nonlinear constraint is imposed that the linearized zero dynamics has to be stable, i.e., all eigenvalues λ_j need to be in the left half-plane. The objective function to be minimized is chosen as

$$J = \max |w_{\gamma,i}| + \max \left(\mathrm{Re}(\lambda_j) \right). \tag{5.6}$$

It balances small control gains to reduce the control effort and fast stable eigenvalues. The latter shall ensure that the eigenvalues are not too close to the stability limit. For the exact end-effector position as tracking output an optimized state controller is

$$u_\gamma = (46.9\gamma + 2q_{\mathrm{e},2} + 28.5q_{\mathrm{e},3} + 18.4\dot{\gamma} - 2.2\dot{q}_{\mathrm{e},2} + 66.7\dot{q}_{\mathrm{e},3})/K_{\mathrm{R}}. \tag{5.7}$$

It should be noted that the elastic coordinates $q_{\mathrm{e},2}$ and $q_{\mathrm{e},3}$ are positive for positive deformations at the link tips in the y-direction of the corresponding floating frame of reference. Typically, they take values between -1 and 1 with 0 being the undeformed state. The SI units of the control gains have been neglected in Equation (5.7) for readability. Although Equation (5.7) renders the system minimum phase it leads to significant oscillations within the actuator motions and the elastic coordinates. This is not desired and is challenging for the integrator of the inverse model. It makes a reduction of the step size necessary or the use of a variable-step solver which is not suitable for real-time control. This issue can be avoided by combining the state control approach with the output redefinition from Section 5.2.1.1. Thus, the redefined end-effector position is used as tracking output, i.e., $\boldsymbol{y}_{\mathrm{o}} = \boldsymbol{r}_{\mathrm{re}}$. As $w_{\mathrm{r},3}$ has no influence since the end-effector rotation is not being tracked only the other elastic weights need to be selected. By choosing $w_{\mathrm{p},2} = w_{\mathrm{p},3} = w_{\mathrm{r},2} = 0.97$ only a small approximation is introduced, which is significantly closer to the exact end-effector compared to the elastic weights of Section 5.2.1.1. An optimized control law is

$$u_\gamma = (-6.6\gamma - 6.3q_{\mathrm{e},2} + 6.6q_{\mathrm{e},3} - 5.9\dot{\gamma} - 2.3\dot{q}_{\mathrm{e},2} + 6.6\dot{q}_{\mathrm{e},3})/K_{\mathrm{R}}, \tag{5.8}$$

which yields much smoother results than Equation (5.7) without output redefinition.

It is worth noting that since the rotation is not tracked one could also conveniently set γ to zero by fixing the motor instead of using a control law for u_γ such as Equation (5.8). As in this case much smaller elastic weights are necessary for the output redefinition the tracking performance would deteriorate substantially. Therefore, a control action in u_γ is needed.

5.2.1.4 Experiments

In this section, experiments are conducted on Flexor to validate the real-time applicability and the end-effector trajectory tracking performance of the model inversion control via the three approaches proposed in the previous sections. The utilized control structure is depicted in Figure 5.11. Analogously to Section 5.1.1, the user input velocity is generated via a game pad and is smoothed with a low-pass filter with a cutoff frequency at 1 Hz. This is especially important when using a flexible model, since badly conditioned signals can be difficult for the time integration within a model inversion and might lead to an internal dynamics with significant oscillation amplitudes. The filter gives the desired end-effector output velocity trajectory $\dot{\boldsymbol{y}}_{\mathrm{d}}$ as well as its time derivative $\ddot{\boldsymbol{y}}_{\mathrm{d}}$. The need for this desired

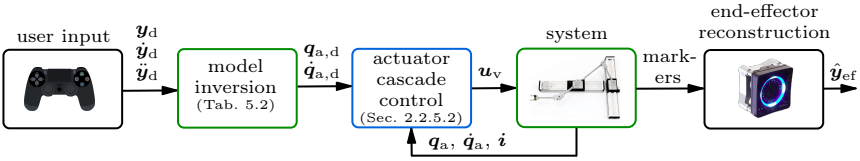


Figure 5.11: Control structure of the real-time model inversion.

acceleration is the only difference to the control structure of the rigid model inversion of Figure 5.2. The simple forward Euler method is used to integrate the velocity signals which yields $\boldsymbol{y}_\mathrm{d}$. It contains the desired xy-position and in the case the rotation is tracked also the desired angle ϑ around the z-axis. Thus, $n_\mathrm{s} = 2$ or $n_\mathrm{s} = 3$ servo constraints are used, depending on the model inversion. The desired output trajectory with its time derivatives are then passed to the inverse model. It is realized without measurement feedback, i.e., as feedforward control. Thus, typically no stability issues arise being especially important when humans interact with the robot such as via a game pad. Using no feedback to the model inversion requires an accurate description of the flexibilities. Even if feedback would be utilized, an accurate model is needed to ensure a stable operation. Moreover, despite no feedback of the end-effector tracking error is used, the long-term drift through modeling errors between a flexible model and the real robot will be small for typical scenarios which include short stops where both come to rest.
When flexibilities are considered within the inverse model, the desired actuator trajectories, $\boldsymbol{q}_\mathrm{a,d}$ on position and $\dot{\boldsymbol{q}}_\mathrm{a,d}$ on velocity level, are obtained via a time integration in real-time. These trajectories are then sent to the actuator cascade control which feeds back the corresponding actuator measurements.

Now, the three developed approaches which render the dynamic inverse of the flexible multibody model stable are compared to classical inverse kinematics for an equivalent rigid multibody model. The utilized settings are summarized in Table 5.2. For the three cases which rely on a flexible model, the formulation via a QR decomposition in chain coordinates is used. This prevents singularities, which can occur for a classical coordinate partitioning, being especially advantageous for the considered real-time model inversion. To solve dynamic inverse models on the real-time target the explicit fourth-order Runge–Kutta method is used for time integration with a step size h being fixed at 1 ms. The definition of explicit Runge-Kutta methods can be found, e.g., in the book of Hairer et al. [110, pp. 28-30]. This is slightly modified by explicitly adding the input $\boldsymbol{u}$.

Definition
For the general system of non-autonomous first-order ODEs

$$\dot{\boldsymbol{x}} = \boldsymbol{f}_\mathrm{g}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{5.9}$$

Table 5.2: Real-time model inversion cases.

case	elastic DOFs	tracking output	equations of motion formulation	solution method	parameters
rigid (Sec. 5.1.1)	none	*rigid* end-effector position and rotation	only algebraic constraint equations (Fig. 3.22)	root finding	equivalent rigid model
redefined output (Sec. 5.2.1.1)	first two bending eigenmodes	*redefined* end-effector position and rotation (Eq. (5.2))	form in chain coordinates via QR decomposition (Fig. 3.15)	classical inversion via forward integration of an IVP	$w_{\mathrm{p},2} = 0.78,\ w_{\mathrm{p},3} = 0.86,$ $w_{\mathrm{r},2} = 0.92,\ w_{\mathrm{r},3} = 0.50$
counter weight (Sec. 5.2.1.2)	first two bending eigenmodes	*exact* end-effector position and rotation	form in chain coordinates via QR decomposition (Fig. 3.15)	classical inversion via forward integration of an IVP with *input extrapolation*	$d_{\mathrm{w}} = 43.3\,\mathrm{mm},$ $m_{\mathrm{w}} = 1.334\,\mathrm{kg},$ $I_{\mathrm{w},zz} = 450.3\,\mathrm{kg\,mm}^2$
state control (Sec. 5.2.1.3)	first two bending eigenmodes	*redefined* end-effector position, *no rotation* (Eq. (5.2a))	form in chain coordinates via QR decomposition with free u_γ (Fig. 3.17)	classical inversion via forward integration of an IVP	$w_{\mathrm{p},2} = w_{\mathrm{p},3} = w_{\mathrm{r},2} = 0.97,$ $u_\gamma = (-6.6\gamma - 6.3q_{\mathrm{e},2} + 6.6q_{\mathrm{e},3}$ $-5.9\dot{\gamma} - 2.3\dot{q}_{\mathrm{e},2} + 6.6\dot{q}_{\mathrm{e},3})/K_{\mathrm{R}}$

an s-stage explicit Runge-Kutta method is given by

$$\boldsymbol{x}_{t_1} = \boldsymbol{x}(t_0) + h \sum_{i=1}^{s} b_i \boldsymbol{k}_i, \tag{5.10a}$$

$$\text{with} \quad \boldsymbol{k}_i = \boldsymbol{f}_{\mathrm{g}}\Big(t_0 + c_i h, \boldsymbol{x}(t_0) + h \sum_{j=1}^{s} a_{ij} \boldsymbol{k}_j, \boldsymbol{u}(t_0 + c_i h)\Big), \tag{5.10b}$$

$$c_i = \sum_{j=1}^{s} a_{ij}. \tag{5.10c}$$

The value $\boldsymbol{x}_{t_1}$ denotes the numerical approximation of $\boldsymbol{x}(t_1) = \boldsymbol{x}(t_0 + h)$ and the coefficients b_i as well as a_{ij} are real numbers with $a_{ij} = 0$ for $i \leq j$.

The coefficients of the utilized explicit fourth-order Runge-Kutta method displayed according to Butcher [141] are

$$\begin{array}{c|ccc} c_1 & a_{11} & \cdots & a_{1s} \\ \vdots & \vdots & & \vdots \\ c_s & a_{s1} & \cdots & a_{ss} \\ \hline & b_1 & \cdots & b_s \end{array} \quad \Longrightarrow \quad \begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{2}{6} & \frac{2}{6} & \frac{1}{6} \end{array} . \tag{5.11}$$

Within end-effector trajectory tracking experiments, the input $\boldsymbol{u}$ to the considered model inversion actually corresponds to the desired end-effector output trajectory on position, velocity and acceleration level. In real-time experiments where the desired trajectory is generated online, this input $\boldsymbol{u}$ is only known at previous and the current time step t_n. Thus, the future inputs $\boldsymbol{u}(t_n + c_i h)$ need to be approximated. For the output redefinition approach and for the state control it is sufficient to use the current value as approximation, i.e.,

$$\boldsymbol{u}(t_n + c_i h) \approx \boldsymbol{u}(t_n). \tag{5.12}$$

For the counter weight, however, it turns out that the numerical integration errors noticeably influence the experimental performance. Therefore, a linear extrapolation

$$\boldsymbol{u}(t_n + c_i h) \approx \boldsymbol{u}(t_n) + \dot{\boldsymbol{u}}(t_n) c_i h \tag{5.13}$$

is performed, where $\dot{\boldsymbol{u}}(t_n)$ is either given or approximated by the difference quotient

$$\dot{\boldsymbol{u}}(t_n) \approx \frac{\boldsymbol{u}(t_n) - \boldsymbol{u}(t_{n-1})}{h}. \tag{5.14}$$

This straightforward extrapolation significantly reduces the occurring integration errors, when compared to the standard of Equation (5.12) which keeps the input constant within one integration step. Since the desired end-effector trajectory is given on position, velocity and acceleration level, the difference quotient only needs to be applied for extrapolating the accelerations. When considering, e.g., a

velocity signal the related element of the time derivative $\dot{\boldsymbol{u}}(t_n)$ corresponds to an already given acceleration signal. In the following the extrapolation method is exclusively used for the counter weight case.
It is worth noting that instead of extrapolation, the inputs could be delayed by one time step. Then the delayed but actual user inputs can be used and interpolated. Besides extrapolation, reducing the step size also helps but even with $h = 0.5\,\text{ms}$ the integration errors are significant and the computational load for real-time control exceeds the performance of a single core. Increasing the counter weight mass reduces the integration errors as well. Since the influence on the system dynamics rises, this method is also not pursued further.

To ensure repeatable experiments, the raw user input signals coming from a game pad are recorded once and applied in real-time to signal filtering and the model inversion. The game pad based desired end-effector trajectory $\boldsymbol{y}_\text{d}$ can be seen in Figure 5.12. This smooth line trajectory lets the end-effector move roughly 0.5 m in 2 s with constant rotation. The other curves within Figure 5.12

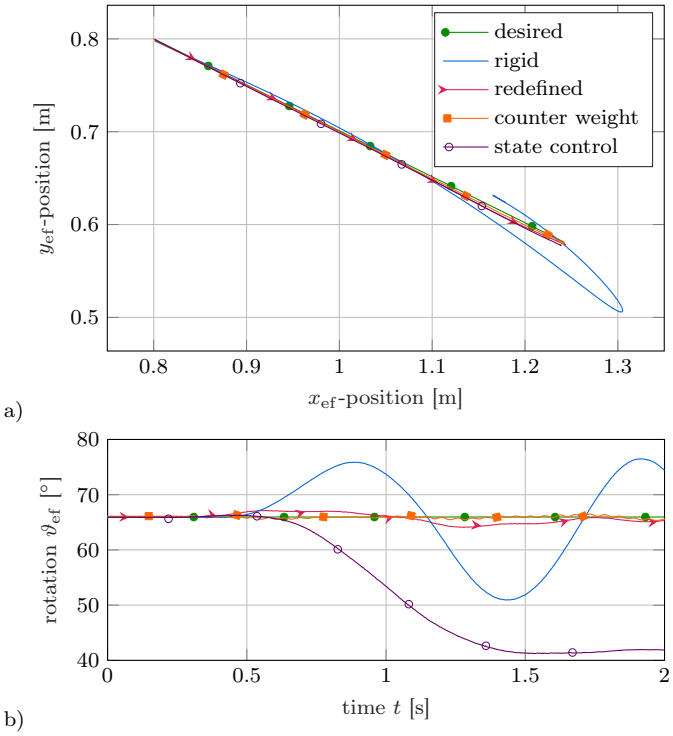


Figure 5.12: Desired and measured end-effector a) positions and b) rotations for the line trajectory.

come from camera measurements showing the end-effector trajectory tracking performance. Here, classical rigid multibody model inversion demonstrates a poor tracking behavior since significant bending is induced for the considered trajectory. The three flexible minimum phase cases perform very effectively and quite similar within the path plot of Figure 5.12a). In Figure 5.12b), the huge difference within the end-effector rotation of the state control compared to the redefined and counter weight case can be seen. Since the state control does not track the end-effector rotation it leads to a completely different final end-effector angle. Also, ripples can be seen within the counter weight case, where the counter weight is used in the model inversion and on the actual robot. They come most likely from the underlying steeper actuator trajectories as the exact end-effector is tracked.
Figure 5.13 shows the robot at the end of the considered trajectory tracking scenario, using the rigid case as well as the flexible case with redefined output. Here, the significant difference within the end-effector rotation angle and the larger bending for the rigid case is visible.
The corresponding tracking errors are quantified in Figure 5.14 for the experiments and simulations. Here, the maximum Euclidean norm of the position errors and the maximum absolute rotational error are given. Each bar of the experiments is the mean of three measurements which are very repeatable. In the simulations the errors come solely from the inverse model, i.e., no forward model is utilized. Thus, the errors introduced through output redefinition and by not tracking the end-effector rotation in the case of the state controller can be identified. In the experiments other error sources such as parameter deviations, friction, measurement errors and unmodeled dynamics occur.
In Figure 5.14, the huge errors for the rigid model inversion and the large rotational deviation of the state control can be seen again. For such significant tracking errors and for moderate errors within simulations, such as for the redefined output, these simulations match the experiments quite closely. However, for the counter weight, being attached to the inverse robot model, one only has numerical errors in the simulation as the exact end-effector is tracked. Here, both counter

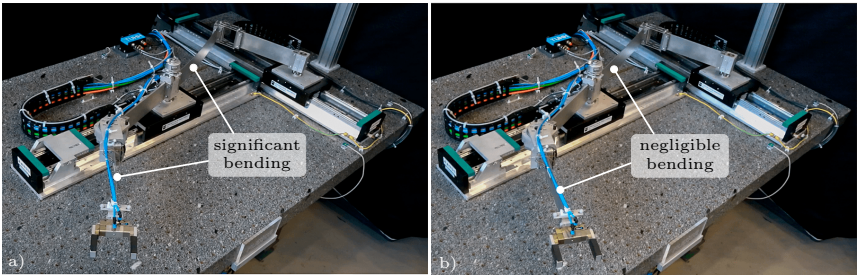


Figure 5.13: Snapshots of the test rig at the end of the desired line trajectory for a) the rigid case and b) the flexible redefined output case.

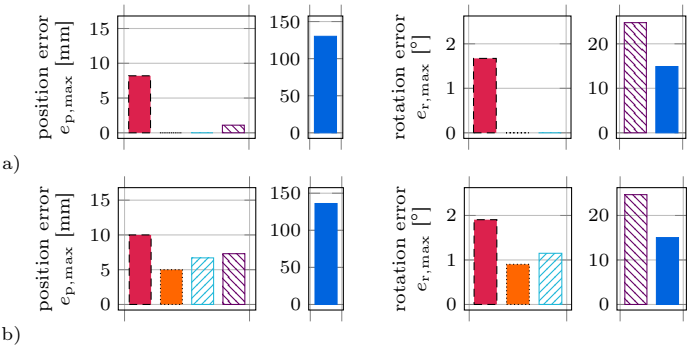


Figure 5.14: Simulated a) and measured b) end-effector trajectory tracking errors for the redefined output ( ), the counter weight ( ), the counter weight only within the model inversion ( ), the state controller ( ) and the rigid model ( ) for the line trajectory.

weight simulations perform identically since they rely on the same inverse model. Applying this model inversion to the experimental setup leads to much higher errors. The reason are the mentioned error sources such as measurement and modeling errors as well as actuator tracking errors, which contribute to a base error within all experiments. Nevertheless, with tracking the exact end-effector the experimental counter weight errors are the smallest. Even with the counter weight only in the model inversion but not on the real robot, the experimental errors are only slightly larger than without this modeling error. This confirms that the effect of the counter weight on the elastic oscillations is very small.
It can be concluded that all flexible cases perform effectively within simulations as well as within experiments. Also, it might be worthwhile to accept tracking errors in the control design, such as for the output redefinition or by not using the counter weight on the real robot, since experimental errors can prevail.

The corresponding actuator motions of the end-effector trajectory tracking experiments are visualized in Figure 5.15. Here, the redefined case shows still some internal dynamics after the desired end-effector motion slowly fades out, which is however stable. This can be seen in the magnification of the slider position a in Figure 5.15a) which oscillates even at the depicted 2.7 s and beyond. Nevertheless, the amplitudes are small enough to ensure a safe operation and to have a negligible influence on the end-effector. These oscillations do not occur for the counter weight but a rather sharp startup motion is present at around 0.25 s, which can again be seen in the magnification of the slider position a in Figure 5.15a). With tracking the exact end-effector this approach tends to produce actuator trajectories which are more difficult to realize than the other cases. This leads here to a slightly increased error within the actuator cascade control.
It should be noted that the initial conditions of the robot let the motor angle γ

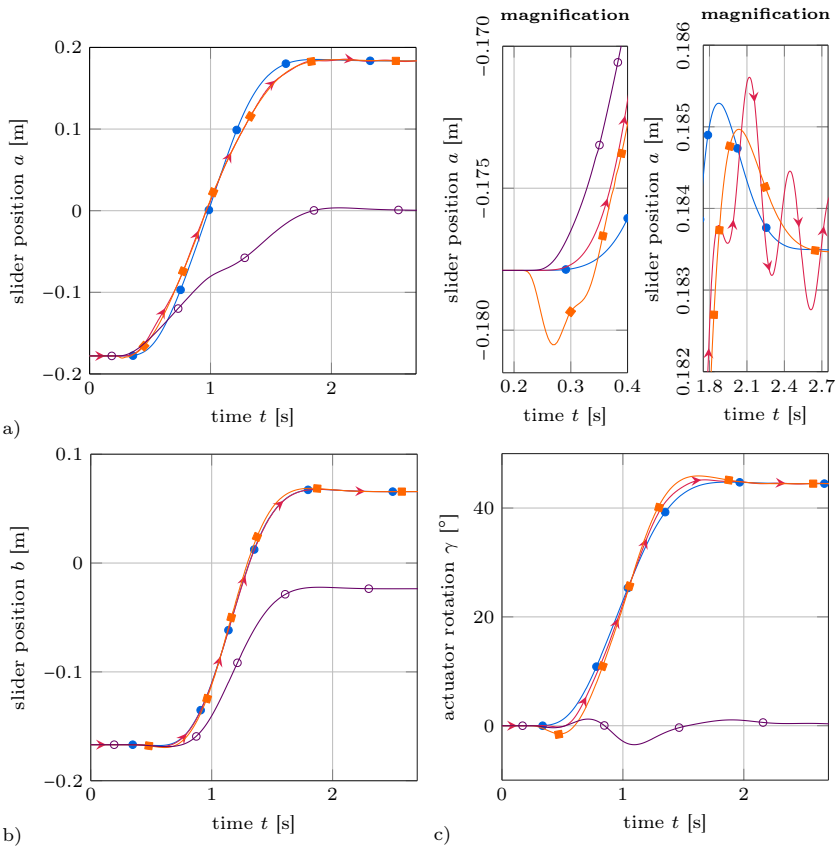


Figure 5.15: Measured actuator positions for the redefined output (——), the counter weight (——), the state controller (——) and the rigid model (——) for the line trajectory.

start at zero being useful for the state control to prevent control inputs at startup. It turns out that for the state control the motion of the rotary motor is extremely small with a change in γ below 3.5° from the starting point compared to the other cases with around 45°. Also, the sliders perform a much smaller motion with a reduction of around 50 % in position a. In specific scenarios, the state control could consequently be helpful to reduce energy consumption if the end-effector rotation is not of interest.

In summary, three approaches are proposed to render the internal dynamics of FLEXOR stable, based on the linearized zero dynamics. They show an effective end-effector trajectory tracking performance using a dynamic model inversion in real-time experiments. The necessity of taking the flexibilities into account within

the control design is also confirmed. It is demonstrated that control concepts with small errors in the design, such as by using a redefined output, can indeed result in a very comparable experimental performance as concepts without such design errors. The proposed approaches are also valuable for other controllers such as feedback linearization and when the end-effector trajectory needs to be adapted or controlled in real-time, being discussed in the subsequent sections.

5.2.2 Collision Avoidance Control

Flexible link robots are well suited for HRI since they offer increased safety. Although such robots are compliant, for the interaction with humans it is still required to implement collision avoidance control in real-time. This is necessary to prevent injuries and damage as the occurring impacts and deformations in the case of a collision can still be devastating.
In this section, collision avoidance is combined with end-effector trajectory tracking. For the collision avoidance two control concepts are considered, one preventing collisions with the actuator limits and the other one avoiding robot collisions with external dynamic obstacles. Since the objective is not to provide optimal collision avoidance algorithms but to show the applicability to flexible link robots, computationally efficient concepts are used. This ensures real-time execution, as the computational complexity of flexible models typically does not allow to implement costly approaches such as optimization. Common real-time collision avoidance algorithms issue commands on actuator level which can cause large oscillations within flexible link robots. Thus, in this research the needed evasive motions at the points which are about to collide are used to adapt the desired end-effector trajectory, even for actuator limits and points other than the end-effector. In contrast to existing real-time collision avoidance concepts, which are essentially completely based on rigid models, a dynamic real-time flexible model inversion is then used for trajectory tracking. Thereby, output redefinition leads to a minimum phase system and a classical rigid model inversion is utilized for comparison. This means that the rigid and the redefined output cases from Table 5.2 will be considered throughout this section, only that the weighting parameters of Table 5.3 are used. These weights, being iterated via time simulations, tend to provide a more robust behavior compared to the output redefinition weights of Table 5.2. Especially as feedback to the model inversion is needed to avoid collisions and as mainly HRI scenarios are of interest, an increased robustness is advantageous. Since the redefined output is the only case which uses a flexible model, it is also referred to as flexible model inversion in this section.
The main contribution of this section is the extension of the related rigid body literature on collision avoidance to flexible link robots and combining it with end-effector trajectory tracking. The following results are highly based on the author's publication [33], where more details can be found for further reading.

Table 5.3: Output redefinition weights for actuator limit and obstacle collision avoidance.

link	position	rotation
2	$w_{\mathrm{p},2} = 0.8$	$w_{\mathrm{r},2} = 1.07$
3	$w_{\mathrm{p},3} = 0.8$	$w_{\mathrm{r},3} = 0.5$

5.2.2.1 Actuator Limit Avoidance

A significant part of the literature related to actuator limit avoidance focuses on rigid redundant robots, where the desired end-effector trajectory can often still be realized. The pseudoinversion with null space projection is a popular method used for such scenarios [142]. This is applied by Flacco et al. [143], who discuss the iterative online inversion of the differential task kinematics for redundant robots in the framework of hard joint limits. In contrast, the presented research uses a flexible underactuated robot where the desired end-effector trajectory needs to be adapted to prevent collisions. Therefore, concepts relying only on pseudoinversion of the differential task kinematics are inapplicable. As mentioned before, existing real-time collision avoidance concepts typically directly adapt the desired actuator motion of rigid robots, which can excite significant oscillations in flexible link robots. Therefore, in the proposed approach the actuator motions are adapted only indirectly by changing the desired end-effector trajectory. A flexible model inversion is then used to ensure an effective trajectory tracking performance. Since here a dynamic model has to be integrated in real-time, optimization such as reported by Bosscher and Hedman [144] and other iterative approaches are usually infeasible.

Within this section an actuator limit avoidance control is introduced which also avoids self-collisions for FLEXOR, in the case of reasonable link deformations. The considered scenario is that a user provides desired end-effector trajectories, e.g., with a game pad, which leave the workspace of the robot. To prevent a collision an automatic braking command is issued to this desired end-effector motion when an actuator limit position is approached. Thus, the user input is adapted such that the actuators do not hit their limits. This results in end-effector trajectories that stay in the robot's workspace. While checking each individual actuator is straightforward it is also computationally more efficient than surveilling if the end-effector is close to the workspace boundaries. To provide smooth braking a standard sigmoid function or logistic curve with an absolute value is used of the form

$$f(q_{\mathrm{a}}) = \frac{1}{1 + \mathrm{e}^{(|q_{\mathrm{a}}| - h_{\mathrm{a}})r}}. \tag{5.15}$$

This yields braking factors between 1 and 0 for each actuator when q_{a}, denoting an element from the actuated coordinates $\boldsymbol{q}_{\mathrm{a}}$, approaches the corresponding actuator limit. Within Equation (5.15), $h_{\mathrm{a}} > 0$ specifies the value of $|q_{\mathrm{a}}|$ where $f(q_{\mathrm{a}})$ gives a half and a large $r > 0$ results in a steep transition rate between minimum and maximum values. Individual parameters h_{a} and r are chosen

for each actuator so that braking is initiated at a reasonable distance from the limits without being too sharp. As mentioned before, such smooth signals are particularly necessary for flexible link robots as badly conditioned signals can be challenging for the time integrator within a flexible model inversion and might lead to an internal dynamics with substantial oscillation amplitudes.
In Figure 5.16 the braking factors for FLEXOR are plotted. It can be seen that the sigmoid function from Equation (5.15) is symmetric to the zero, i.e., middle position of each actuator. At these points cubic splines are used to obtain a smooth transition. Slider B has a smaller range of motion due to the slider limit stop, see Figure 2.7. Note that the factors are only depending on the current position and also slow down a motion away from the actuator limits when being close to one. This seemingly conservative choice helps to avoid jumps and edges in the braking factor for direction changes. This is very important since the actuators can oscillate as part of the internal dynamics of the later used flexible model inversion.
The resulting end-effector trajectory tracking control structure with actuator limit avoidance is shown in Figure 5.17. Here, the user inputs the velocity trajectories online for the end-effector translation and rotation in 2D. If these inputs come from a game pad they are low-pass filtered with a cutoff frequency at 1.4 Hz to remove steps. This ensures a physically meaningful translational and rotational end-effector input velocity $\dot{\boldsymbol{y}}_\mathrm{i}$. The product of all three braking factors provides the conservative but smooth braking factor b_Π between 0 and 1 being multiplied to all three components of $\dot{\boldsymbol{y}}_\mathrm{i}$. Using the same factor for each component prevents a direction distortion of the user input. Afterwards, a low-pass filter is applied with a cutoff frequency at 30 Hz. This ensures that no high frequencies are fed back, being important for a numerically stable model inversion, while providing a small delay for braking maneuvers. With the filter $\dot{\boldsymbol{y}}_\mathrm{d}$ as well as its time derivative $\ddot{\boldsymbol{y}}_\mathrm{d}$ are obtained and integration of $\dot{\boldsymbol{y}}_\mathrm{d}$ via the forward Euler method gives $\boldsymbol{y}_\mathrm{d}$. Then, $\ddot{\boldsymbol{y}}_\mathrm{d}$, $\dot{\boldsymbol{y}}_\mathrm{d}$ and $\boldsymbol{y}_\mathrm{d}$ are passed to the model inversion, which is in ODE form if flexibilities are considered. Again, the model inversion does not use state feedback which ensures a more robust behavior. This is particularly important for

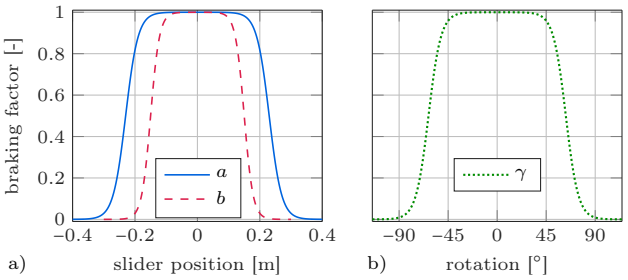


Figure 5.16: Braking factors realized by the sigmoid function.

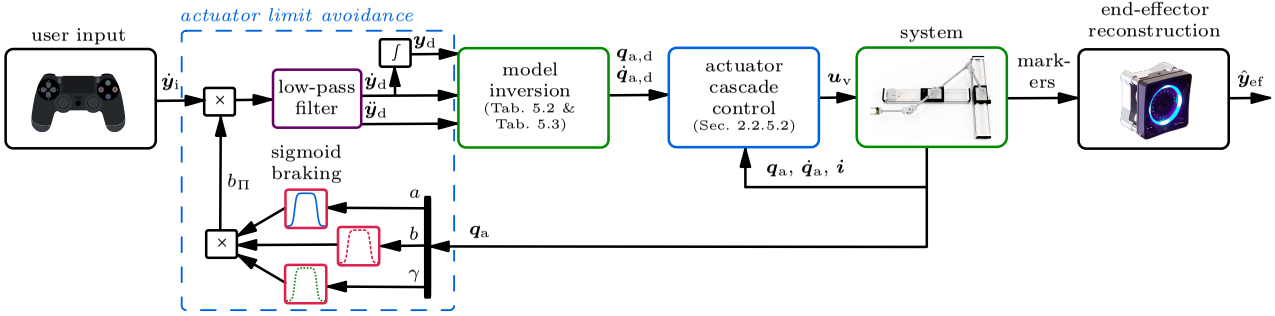


Figure 5.17: Control structure of the actuator limit avoidance with model inversion.

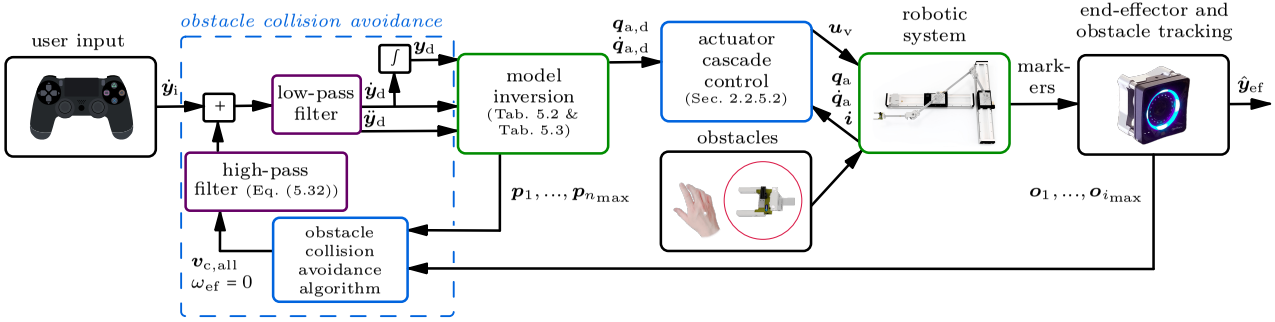


Figure 5.18: Control structure of the obstacle collision avoidance with model inversion.

the involved HRI in the considered collision avoidance scenarios. As previously discussed, the calculated desired actuator positions $\boldsymbol{q}_{\mathrm{a,d}}$ and velocities $\dot{\boldsymbol{q}}_{\mathrm{a,d}}$ are then accurately realized by the cascade control. Also, the end-effector pose is reconstructed from camera measurements.
For a convenient comparison, the control structure of the obstacle collision avoidance is also illustrated here, see Figure 5.18. Its differences and details will be clarified in the subsequent section.

Experiments are now conducted to validate the necessity and effectiveness of the actuator limit avoidance for user defined end-effector trajectories, coming here from a game pad. To provide repeatable and comparable results the user input is recorded once and then applied online. The actuator motions of the considered scenario are shown in Figure 5.19. Here, limited actuator motion is compared to unlimited actuator motion. Not using a limit avoidance, i.e., $b_{\Pi} = 1$, leads to a desired motion hitting the physical actuator limits such as for the slider position a at around 1.7 s. This is independent of using a flexible or rigid model

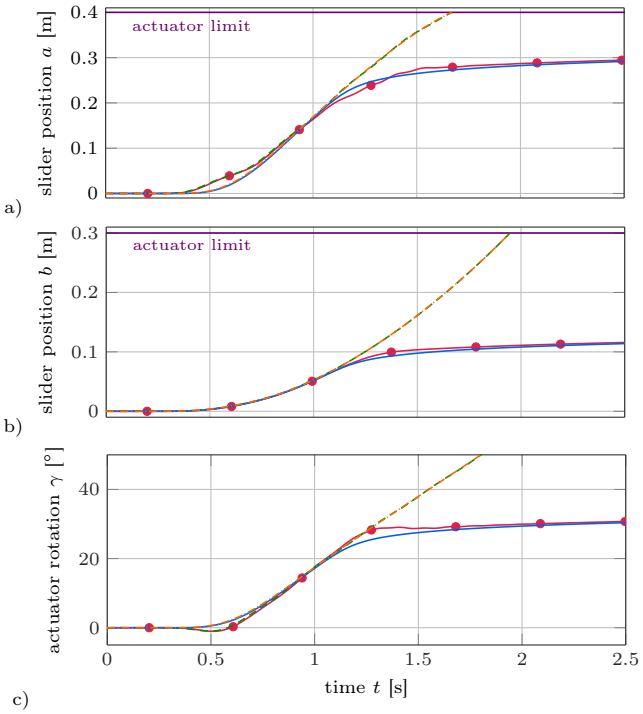


Figure 5.19: Experimentally measured actuator positions with limit avoidance and flexible (●——) as well as rigid (——) inversion, and simulated actuator positions without limit avoidance and flexible (- - - -) as well as rigid (-·-·-) inversion.

inversion. Since this would result in a crash or emergency stop motion with large and dangerous oscillations, unlimited cases are only applied within simulations. With active actuator limit avoidance the motion diverges from the unlimited cases when approaching the limits. This starts at around 1 s where b_{Π} begins to drop from a value of 1. For both cases, flexible and rigid, there is still a substantial safety distance which the user can only reduce with extremely small velocity as the underlying sigmoid functions quickly approach zero close to the limits. For the flexible model inversion braking at the end-effector does not directly lead to a stop motion in the actuators. This is due to the internal dynamics which will come to rest shortly afterwards. Therefore, the actuator curves for the flexible case are more uneven. Further significant differences between the rigid and flexible cases can be seen within the position a and the rotation γ, which are required to ensure an effective end-effector trajectory tracking performance.
The superior trajectory tracking performance of the flexible model inversion can be seen in the end-effector measurements of Figure 5.20. Here, the flexible model inversion tracks the position and rotation, adapted by the actuator limit avoidance, very effectively leading to a fast and safe braking maneuver. The poor results of the rigid inversion, even before the actuator limit avoidance becomes significant, are clearly visible for the considered trajectory as it excites substantial bending. In the path plot of Figure 5.20a), one can also see that the limit avoidance does not distort the direction of the user input since the desired paths from the flexible and rigid case lie exactly over the infeasible case without limit avoidance, which eventually leaves the robot's workspace. A difference between the desired motion of the flexible and rigid case can only be observed in Figure 5.20b), where the change over time is taken into account. Finally, since the desired rotation is constant, Figure 5.20c) shows three horizontal lines which are closely tracked in the flexible case.

The presented experimental results clearly show the effectiveness of the actuator limit avoidance and the significant advantage of the flexible model inversion, compared to the classical rigid body approach, in a realistic scenario. It is concluded that the proposed control approach enables a safe HRI, e.g., by using inputs from a game pad, for flexible link robots.

5.2.2.2 Obstacle Collision Avoidance

Many real-time concepts for dynamic obstacle collision avoidance are based on the artificial potential field method [145, 146]. Also, a large part of the utilized concepts directly adapts the actuator motions which is, as explained before, not appropriate for flexible link robots. The elastic strip method of Brock and Khatib [147] is better suited as it adapts the desired trajectory in the task space instead of the configuration, i.e., joint space. However, its computational complexity is higher than potential field methods [148]. Again, optimization approaches, discussed, e.g., in [149, 150], might be used in real-time for rigid robots. This is however usually not possible for flexible link robots which rely on a dynamic

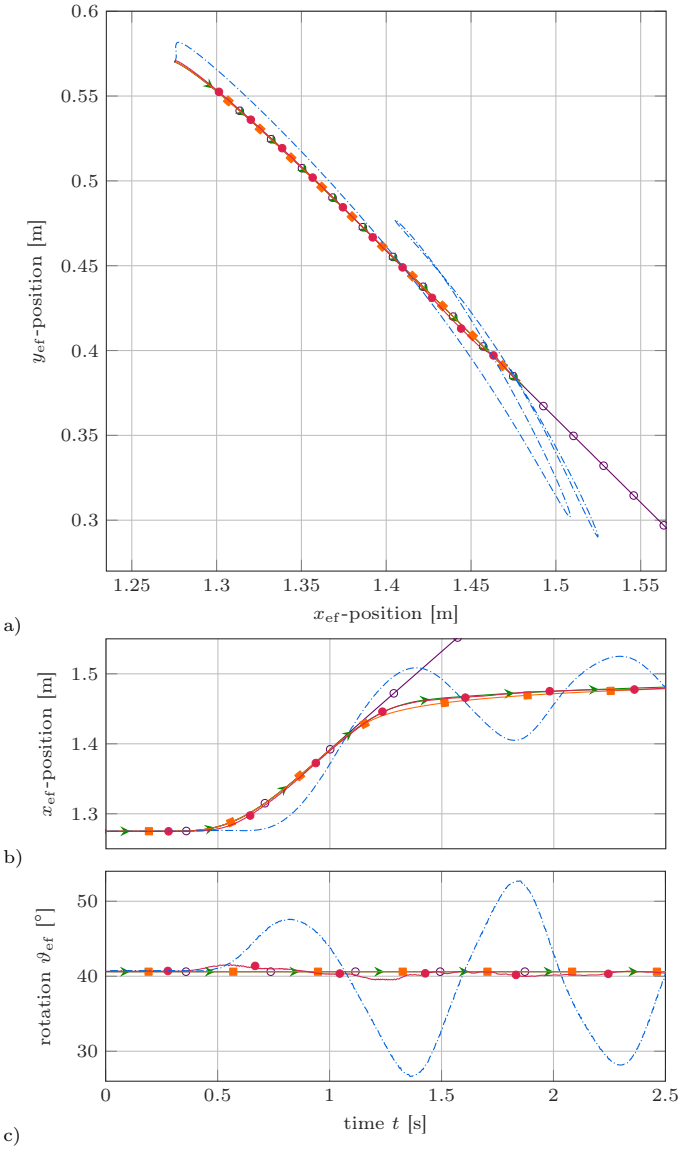


Figure 5.20: End-effector trajectory tracking experiments with actuator limit avoidance for flexible (desired ➤——, measured ●——) and rigid (desired ■——, measured ·-·-·) model inversion, as well as a case without actuator limit avoidance (desired ○——).

model that needs to be solved via time integration.
Collision avoidance for flexible link robots is very rarely found in the literature and, as opposed to the here presented research, it is not implemented within real-time experiments and only considers static obstacles as well as only serial robots. Ata and Myo [151] implement obstacle avoidance as a planning task within simulations, which are based on optimization, for a robot with a rigid and a flexible link. Also, Karray et al. [152] and Mclean and Cameron [153] show path planning with obstacle avoidance for kinematically redundant flexible manipulators being however not implemented in experiments.

In this research, an algorithm is used which applies repulsive velocities to avoid external dynamic obstacles. It has been proposed by Flacco et al. [154, 155] for rigid robots. The algorithm can be regarded as efficient and simple form of the classical artificial potential field method, which guarantees fast calculations being necessary for a safe HRI in real-time. To enable the application to flexible link robots, the algorithm is adapted. A major difference of the presented research in contrast to [154] is that the distances between obstacles and the control points on the robot body, other than the end-effector, are not translated into joint velocity constraints. As this is not feasible for flexible link robots repulsive actions are used instead, both for the end-effector and the other control points on the robot. These actions are combined into a single adapted end-effector trajectory. Then, as opposed to [154], the actuator motions are not kinematically obtained by pseudoinversion but by inversion of a dynamic flexible link robot model. The adapted algorithm is summarized in Figure 5.21 and its details are discussed in the following.

The utilized obstacle collision avoidance algorithm aims to avoid a collision between $n_{\max}$ control points, where the n-th control point is located at $\boldsymbol{p}_n \in \mathbb{R}^3$ on the robot, and $i_{\max}$ obstacles, where the i-th obstacle is located at $\boldsymbol{o}_i \in \mathbb{R}^3$. Initially, for a single control point described by the position vector $\boldsymbol{p}$ and one obstacle described by the position vector $\boldsymbol{o}$ a repulsive translational velocity is introduced as

$$\boldsymbol{v}_{\mathrm{r}}(\boldsymbol{p}, \boldsymbol{o}) = v(\boldsymbol{p}, \boldsymbol{o})\boldsymbol{d}(\boldsymbol{p}, \boldsymbol{o}) \tag{5.16}$$

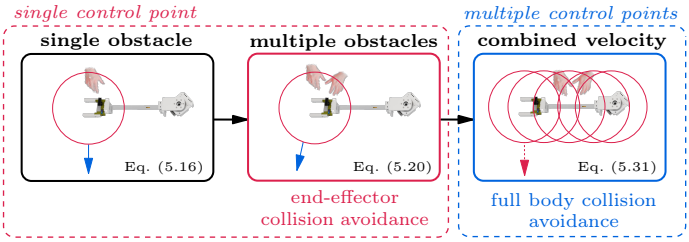


Figure 5.21: Overview of the obstacle collision avoidance algorithm.

to prevent a collision, with

$$\boldsymbol{d}(\boldsymbol{p},\boldsymbol{o}) = \frac{\boldsymbol{p}-\boldsymbol{o}}{\|\boldsymbol{p}-\boldsymbol{o}\|}. \tag{5.17}$$

Here, $\boldsymbol{d}(\boldsymbol{p},\boldsymbol{o})$ is the normalized position vector between $\boldsymbol{p}$ and $\boldsymbol{o}$ with its direction pointing away from the obstacle and $\|\|$ denotes the Euclidean norm. The term

$$v(\boldsymbol{p},\boldsymbol{o}) = \frac{v_{\max}}{1+\mathrm{e}^{\left(\|\boldsymbol{p}-\boldsymbol{o}\|\left(\frac{2}{\rho}\right)-1\right)\sigma}} \tag{5.18}$$

is the magnitude of the repulsive velocity. It consists of the maximum speed $v_{\max}$ at which the control point $\boldsymbol{p}$ can move and is shaped by a sigmoid function. Thus, the shape is similar to Figure 5.16, but is defined only for a positive distance between obstacle and control point. Equation (5.18) depends on the parameter $\rho > 0$ which defines at which distance between control point $\boldsymbol{p}$ and obstacle $\boldsymbol{o}$ the repulsive velocity becomes significant. It can be regarded as the radius of a collision avoidance zone around the control point $\boldsymbol{p}$. The parameter $\sigma > 0$ is used to shape the slope of the sigmoid function.
So far the repulsive velocity $\boldsymbol{v}_{\mathrm{r}}$ provides a straightforward way of collision avoidance for a single obstacle. However, for multiple obstacles the repulsive velocity $\boldsymbol{v}_{\mathrm{r}}$ needs to be adjusted, such that the control point can avoid all obstacles. By combination of the repulsive velocities $\boldsymbol{v}_{\mathrm{r}}(\boldsymbol{p},\boldsymbol{o}_i)$ the new direction of the repulsive velocity for all considered obstacles is obtained as

$$\boldsymbol{d}_{\mathrm{all}}(\boldsymbol{p},\boldsymbol{o}_1,\ldots,\boldsymbol{o}_{i_{\max}}) = \frac{\sum_{i=1}^{i_{\max}} \boldsymbol{v}_{\mathrm{r}}(\boldsymbol{p},\boldsymbol{o}_i)}{\left\|\sum_{i=1}^{i_{\max}} \boldsymbol{v}_{\mathrm{r}}(\boldsymbol{p},\boldsymbol{o}_i)\right\|}. \tag{5.19}$$

To ensure a collision avoidance only the closest obstacle $\boldsymbol{o}_{\min}$ is taken into account for the magnitude, which yields the repulsive velocity for all considered obstacles

$$\boldsymbol{v}_{\mathrm{r,all}}(\boldsymbol{p},\boldsymbol{o}_1,\ldots,\boldsymbol{o}_{i_{\max}}) = v(\boldsymbol{p},\boldsymbol{o}_{\min})\boldsymbol{d}_{\mathrm{all}}(\boldsymbol{p},\boldsymbol{o}_1,\ldots,\boldsymbol{o}_{i_{\max}}). \tag{5.20}$$

Until now, the collision avoidance algorithm is only introduced for a single control point, e.g., only for the end-effector. For a safe collision avoidance between a human and a robot more control points need to be added to realize a full body collision avoidance. Since only the end-effector shall be tracked the repulsive velocities for the other control points need to be transformed into an end-effector velocity.
Starting point is the time derivative of a general position vector

$$\boldsymbol{r}_{\mathrm{g}}(\boldsymbol{q}) = \boldsymbol{r}_{\mathrm{g}}(\boldsymbol{q}_{\mathrm{i}},\boldsymbol{q}_{\mathrm{d}}(\boldsymbol{q}_{\mathrm{i}})), \tag{5.21}$$

which gives the corresponding translational velocity

$$\boldsymbol{v}_{\mathrm{g}} = \frac{\mathrm{d}\boldsymbol{r}_{\mathrm{g}}(\boldsymbol{q}_{\mathrm{i}},\boldsymbol{q}_{\mathrm{d}}(\boldsymbol{q}_{\mathrm{i}}))}{\mathrm{d}t} = \frac{\partial \boldsymbol{r}_{\mathrm{g}}}{\partial \boldsymbol{q}_{\mathrm{i}}}\dot{\boldsymbol{q}}_{\mathrm{i}} + \frac{\partial \boldsymbol{r}_{\mathrm{g}}}{\partial \boldsymbol{q}_{\mathrm{d}}}\frac{\partial \boldsymbol{q}_{\mathrm{d}}}{\partial \boldsymbol{q}_{\mathrm{i}}}\dot{\boldsymbol{q}}_{\mathrm{i}}. \tag{5.22}$$

As discussed in Section 3.3.1, the coordinates $\boldsymbol{q} = [\boldsymbol{q}_\mathrm{e}^\mathrm{T}, \boldsymbol{q}_\mathrm{a}^\mathrm{T}, \boldsymbol{q}_\mathrm{u}^\mathrm{T}]^\mathrm{T}$ contain the elastic coordinates $\boldsymbol{q}_\mathrm{e}$, the actuated coordinates $\boldsymbol{q}_\mathrm{a}$ and further unactuated coordinates $\boldsymbol{q}_\mathrm{u}$. For the forward model of FLEXOR, the independent coordinates are chosen as $\boldsymbol{q}_\mathrm{i} = [\boldsymbol{q}_\mathrm{e}^\mathrm{T}, \boldsymbol{q}_\mathrm{a}^\mathrm{T}]^\mathrm{T}$ and the dependent coordinates as $\boldsymbol{q}_\mathrm{d} = \boldsymbol{q}_\mathrm{u}$. Furthermore, the variation of the geometric constraints for fixed time yields

$$\delta \boldsymbol{c}(\boldsymbol{q}_\mathrm{i}, \boldsymbol{q}_\mathrm{d}(\boldsymbol{q}_\mathrm{i})) = \underbrace{\frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}_\mathrm{i}}}_{\boldsymbol{C}_\mathrm{i}} \delta \boldsymbol{q}_\mathrm{i} + \underbrace{\frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}_\mathrm{d}}}_{\boldsymbol{C}_\mathrm{d}} \frac{\partial \boldsymbol{q}_\mathrm{d}}{\partial \boldsymbol{q}_\mathrm{i}} \delta \boldsymbol{q}_\mathrm{i} = \boldsymbol{0}, \tag{5.23}$$

which needs to hold for all independent variations $\delta \boldsymbol{q}_\mathrm{i}$. From Equation (5.23), the partial derivative

$$\frac{\partial \boldsymbol{q}_\mathrm{d}}{\partial \boldsymbol{q}_\mathrm{i}} = -\boldsymbol{C}_\mathrm{d}^{-1} \boldsymbol{C}_\mathrm{i} \tag{5.24}$$

required within Equation (5.22) is obtained. In regard of the considered planar setup of FLEXOR, a 2D case is now discussed. Thus, Equation (5.22) can be used to express the planar translational end-effector velocity $\boldsymbol{v}_\mathrm{ef} \in \mathbb{R}^2$ and any other control point velocity $\boldsymbol{v}_{\mathrm{p}n} \in \mathbb{R}^2$, with the time derivative of the independent coordinates $\dot{\boldsymbol{q}}_\mathrm{i}$, according to

$$\boldsymbol{v}_\mathrm{ef} = \boldsymbol{J}_\mathrm{ef}(\boldsymbol{q}) \underbrace{\begin{bmatrix} \dot{\boldsymbol{q}}_\mathrm{e} \\ \dot{\boldsymbol{q}}_\mathrm{a} \end{bmatrix}}_{\dot{\boldsymbol{q}}_\mathrm{i}}, \tag{5.25a}$$

$$\boldsymbol{v}_{\mathrm{p}n} = \boldsymbol{J}_{\mathrm{p}n}(\boldsymbol{q}) \begin{bmatrix} \dot{\boldsymbol{q}}_\mathrm{e} \\ \dot{\boldsymbol{q}}_\mathrm{a} \end{bmatrix}. \tag{5.25b}$$

Here, $\boldsymbol{J}_\mathrm{ef}$ and $\boldsymbol{J}_{\mathrm{p}n}$ are Jacobian matrices. The end-effector also has a rotational degree of freedom which is not directly relevant for obstacle avoidance and thus needs to be selected by the user. Therefore, the collision avoidance motion of the control points of Equation (5.25b) is extended by the rotational velocity ω_ef of the end-effector as

$$\begin{bmatrix} \boldsymbol{v}_{\mathrm{p}n} \\ \omega_\mathrm{ef} \end{bmatrix} = \boldsymbol{J}_{\mathrm{mix},\mathrm{p}n}(\boldsymbol{q}) \begin{bmatrix} \dot{\boldsymbol{q}}_\mathrm{e} \\ \dot{\boldsymbol{q}}_\mathrm{a} \end{bmatrix}, \tag{5.26}$$

with the mixed Jacobian $\boldsymbol{J}_{\mathrm{mix},\mathrm{p}n} \in \mathbb{R}^{3\times(f_\mathrm{a}+f_\mathrm{e})}$. While the control point velocity $\boldsymbol{v}_{\mathrm{p}n}$ is determined by the collision avoidance algorithm, setting $\omega_\mathrm{ef} = 0$ shall avoid a change of the end-effector rotation through a collision avoidance motion. By rearranging Equations (5.25a) and (5.26), the control point velocity is transformed into an end-effector velocity

$$\boldsymbol{v}_{\mathrm{ef},\mathrm{p}n} = \boldsymbol{J}_\mathrm{ef}(\boldsymbol{q}) \boldsymbol{J}_{\mathrm{mix},\mathrm{p}n}^{+}(\boldsymbol{q}) \begin{bmatrix} \boldsymbol{v}_{\mathrm{p}n} \\ 0 \end{bmatrix}. \tag{5.27}$$

Since the considered system is underactuated, with the elastic coordinates $\boldsymbol{q}_\mathrm{e}$ being not directly actuated, $\boldsymbol{J}_{\mathrm{mix},\mathrm{p}n}$ is non-square with more columns than rows. Using

a pseudoinverse, denoted by $^+$, for Equation (5.26) will thus lead to necessary elastic deformation velocities $\dot{\boldsymbol{q}}_\mathrm{e}$ that cannot be directly controlled. Consequently, only taking the kinematics of a dynamic inverse system into account could result in an undesired behavior. Using alternatively a dynamic inverse for each control point is typically not feasible since this often involves a non-minimum phase system behavior. Even if this is not the case, the computational cost of a dynamic model inversion would significantly limit the number of control points. Therefore, the idea is to use the Jacobians of the equivalent rigid system instead, turning Equation (5.27) into

$$\boldsymbol{v}_{\mathrm{ef},\mathrm{p}n} = \boldsymbol{J}_{\mathrm{ef,r}}(\boldsymbol{q}_\mathrm{a}, \boldsymbol{q}_\mathrm{u}) \boldsymbol{J}^{-1}_{\mathrm{mix},\mathrm{p}n,\mathrm{r}}(\boldsymbol{q}_\mathrm{a}, \boldsymbol{q}_\mathrm{u}) \begin{bmatrix} \boldsymbol{v}_{\mathrm{p}n} \\ 0 \end{bmatrix}. \tag{5.28}$$

As the columns of the Jacobians with respect to the elastic velocities $\dot{\boldsymbol{q}}_\mathrm{e}$ vanish, the rigid Jacobian $\boldsymbol{J}_{\mathrm{mix},\mathrm{p}n,\mathrm{r}} \in \mathbb{R}^{3\times f_\mathrm{a}}$ is square with $f_\mathrm{a} = 3$. The rigid Jacobians of Equation (5.28) adapt the end-effector velocity $\boldsymbol{v}_{\mathrm{ef},\mathrm{p}n}$ again via kinematic relations. Using this as desired end-effector velocity later within a dynamic flexible model inversion will also not result in the original desired $\boldsymbol{v}_{\mathrm{p}n}$. Nevertheless, this approach aims at preventing collisions for the underlying rigid body motion, which is usually an effective way for obstacle avoidance since the flexible system basically oscillates around this motion. Occurring errors due to simplifications are treated by the algorithm through running in a feedback loop. Additionally, this straightforward adaption of the end-effector velocity is computationally more efficient than calculating the pseudoinverse of the Jacobians of the flexible system. This is advantageous for the required real-time capability.
Calculating all repulsive velocities $\boldsymbol{v}_{\mathrm{r,all},\mathrm{p}n}$ at each control point via Equation (5.20) and transforming them via Equation (5.28) into $\boldsymbol{v}_{\mathrm{r,all,ef},\mathrm{p}n}$ then yields the combined repulsive velocity

$$\boldsymbol{v}_\mathrm{c} = \boldsymbol{v}_{\mathrm{r,all,ef}} + \sum_{n=2}^{n_\mathrm{max}} \boldsymbol{v}_{\mathrm{r,all,ef},\mathrm{p}n}. \tag{5.29}$$

Here, $\boldsymbol{v}_{\mathrm{r,all,ef}}$ is the repulsive velocity at the end-effector control point which does not need to be transformed. Since $\boldsymbol{v}_\mathrm{c}$ might exceed the maximum allowed velocity only its direction is used by normalizing analogously to Equation (5.19) as

$$\boldsymbol{d}_{\mathrm{c,all}} = \frac{\boldsymbol{v}_\mathrm{c}}{\|\boldsymbol{v}_\mathrm{c}\|}. \tag{5.30}$$

The magnitude is chosen similar to Equation (5.20), which gives the final combined repulsive velocity

$$\boldsymbol{v}_{\mathrm{c,all}} = v(\boldsymbol{p}_\mathrm{min}, \boldsymbol{o}_\mathrm{min}) \boldsymbol{d}_{\mathrm{c,all}}. \tag{5.31}$$

Here, $\boldsymbol{p}_\mathrm{min}$ and $\boldsymbol{o}_\mathrm{min}$ denote the closest control point and obstacle pair. Since this pair can change throughout operation, the resulting minimum or critical distance is low-pass filtered with a cutoff frequency at 12 Hz to remove edges.

Applying $\boldsymbol{v}_{\mathrm{c,all}}$ to avoid a collision moves the end-effector away from its original trajectory. Therefore, a high-pass filter is used for each component of $\boldsymbol{v}_{\mathrm{c,all}}$, such that the robot smoothly returns to the original trajectory when the obstacles are not in range anymore. The filter is realized via the state-space model

$$y_{\mathrm{f}} = \dot{x}_{\mathrm{f}} = a_{\mathrm{f}} x_{\mathrm{f}} + u_{\mathrm{f}}. \tag{5.32}$$

In Equation (5.32), the input u_{f} is a component of $\boldsymbol{v}_{\mathrm{c,all}}$. To ensure an effective collision avoidance, a scalar $a_{\mathrm{f}} = -1.2\,\mathrm{s}^{-1}$ is used to obtain a slow filtering. Finally, the output y_{f}, being the filtered combined repulsive velocity component, is added to the end-effector input velocity $\dot{\boldsymbol{y}}_{\mathrm{i}}$. The resulting control structure with model inversion can be seen in Figure 5.18. Here, the same low-pass filter as in Section 5.2.2.1 with a cutoff frequency at 30 Hz is applied, which provides the time derivative $\ddot{\boldsymbol{y}}_{\mathrm{d}}$ for the model inversion.

The obstacle collision avoidance control is now applied to FLEXOR within two different experimental scenarios. For the considered 2D setup the obstacles are assumed to occur in the same plane as the robot. Therefore, a single marker is sufficient to track the center of each dynamic obstacle, which are here the hands of a human worker. For a full body collision avoidance the amount of control points, their positions and the corresponding radius ρ need to be selected such that the area of interest is completely covered. Still, the covered area needs to be small enough to avoid unnecessary collision avoidance movements. In this research, five control points including the end-effector are utilized for link 3 of FLEXOR, see Figure 5.22. For control purposes, the current positions of the control points on the robot are fed back from the corresponding model inversion, compare Figure 5.18. For the flexible case this prevents difficult marker-based tracking on the flexible link or the need for a state estimator, based, e.g., on strain gauge measurements to reconstruct the control point positions. For the rigid case this ensures a stable operation, since feeding back large oscillations from a flexible link robot into a rigid model inversion can cause instability.
To provide repeatable and comparable results the user input and the obstacle motions are recorded once and are then applied online. Nevertheless, the delay of the camera system with data processing is with around 3 ms small enough

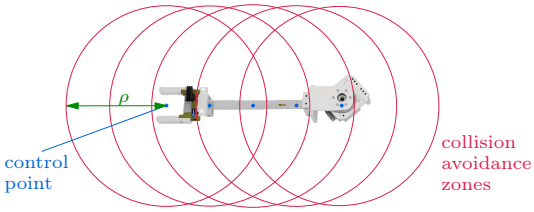


Figure 5.22: Control points used for obstacle collision avoidance on link 3 with $\rho \approx 0.29\,\mathrm{m}$.

to successfully avoid obstacles also without pre-recording marker data. This is demonstrated in the accompanying video of the author's article [33].

In the first experimental scenario, two static obstacles, represented by the hands of a human worker, collide with a desired end-effector trajectory. This trajectory describes a line in xy-coordinates. The obstacle collision avoidance control is used to prevent such a collision. Snapshots from the scenario with a rigid and a flexible model inversion are shown in Figure 5.23. Here, the markers on the hands represent the obstacle centers. It can be seen that the obstacle collision avoidance algorithm adapts the desired line trajectory between start and endpoint. This results in a curvy motion, which successfully avoids a collision for the rigid and the flexible model inversion. Nevertheless, the rigid model inversion leads to heavy oscillations at the endpoint which can be dangerous if the obstacles and the robot approach each other before they are damped out.
The corresponding measurements are shown in Figure 5.24. The utilized parameters for all control points are listed in Table 5.4, where the choice depends on the considered scenario. In Figure 5.24a), the path plot visualizes the successful collision avoidance and the return close to the trajectory endpoint. Here, the flexible model inversion tracks the desired trajectory very closely, while the significant oscillations of the rigid case can be seen again. In contrast to the end-effector position, which needs to be adapted to avoid a collision, the desired end-effector rotation still follows the original constant user input, see Figure 5.24b). As expected, the flexible case also tracks the rotation much better than the rigid case.

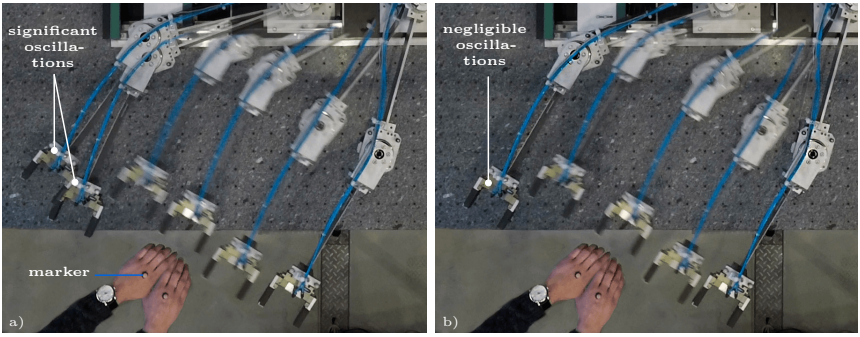


Figure 5.23: Snapshots of the test rig for a) the rigid and b) the flexible model inversion combined with static obstacle collision avoidance for a motion from right to left.

Table 5.4: Obstacle collision avoidance parameters for static obstacles.

parameter	value
ρ	0.29 m
σ	5
$v_{\max}$	0.6 m/s

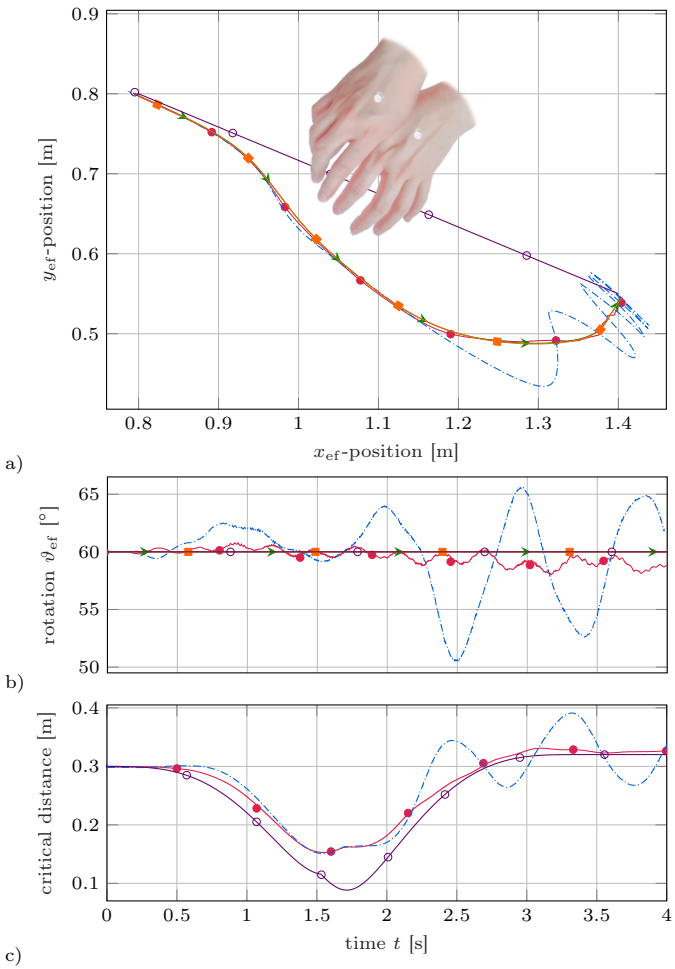


Figure 5.24: Static obstacles: end-effector trajectory tracking experiments with obstacle collision avoidance for the flexible (desired, measured) and rigid (desired, measured) model inversion, as well as the desired motion without obstacle collision avoidance ().

Finally, in Figure 5.24c) the critical distance is shown. This denotes the distance between the closest obstacle and control point pair, which for this scenario is always with respect to the end-effector. Therefore, the critical distance is here based on camera measurements of the end-effector. The minimal critical distance for both the flexible and the rigid model inversion is quite similar. But more importantly, it is also significantly larger than without the obstacle collision

avoidance, which would lead to a collision between the robot and the fingers of the human worker.

In the second experimental scenario, a dynamic obstacle approaches multiple control points on the robot. The obstacle is again a human hand, which now approaches link 3 of FLEXOR at rest. Thus, the user input trajectory is reduced to a fixed point to show how the robot behaves just by avoiding a collision. The hand first moves closely parallel to link 3, approaches it and then moves away from it. Here, for a full body collision avoidance the utilized control points as shown in Figure 5.22 become crucial. In Figure 5.25 the corresponding measurements are shown. The parameters from Table 5.4 are used again but now with $\rho = 0.35\,\mathrm{m}$ to obtain larger avoidance zones. Clearly, Figure 5.25a) and b) confirm that the flexible model inversion is required to keep the end-effector oscillations small. Figure 5.25a) also demonstrates that the end-effector successfully returns to its start position when the obstacle is far enough away. In Figure 5.25c) the critical distance can be seen which depends on the control point being closest to the hand. As the control points on link 3, except the end-effector, are not measured by the camera the illustrated critical distance depends on data from the flexible multibody model. The results indicate that both cases, rigid and flexible, avoid the hand with almost the same minimal critical distance of approximately $0.12\,\mathrm{m}$. This represents a substantial improvement compared to the scenario without obstacle collision avoidance. Nevertheless, further approaching the robot when using the rigid model inversion can be dangerous due to the significant oscillations.

In summary, the discussed experiments validate the effectiveness of the obstacle collision avoidance algorithm in combination with a flexible model inversion. As a result, the proposed control approach enables a collision-free interaction between human workers and flexible link robots.

5.2.3 Hybrid Force/Motion Control

Up to now it was not necessary to control the end-effector trajectory tracking error due to the high accuracy of the utilized flexible multibody model. However, especially in contact scenarios having a good model can be difficult since often the environment is only roughly known. In such scenarios, moderate modeling errors can lead to completely different contact forces. Also, flexible link robots are especially promising for scenarios with contact to a basically rigid environment. The inherent compliance of the links reduces contact forces and the connected risk of damage. Therefore, feedback of the tracking output error shall be used within the framework of hybrid force/motion control. In this research, the contact force and the end-effector motion shall be controlled on position level, which is commonly denoted as *hybrid force/position control*.

Hybrid force/motion control is a well-known technique on which detailed background information is provided, e.g., by Villani and De Schutter [121]. It is

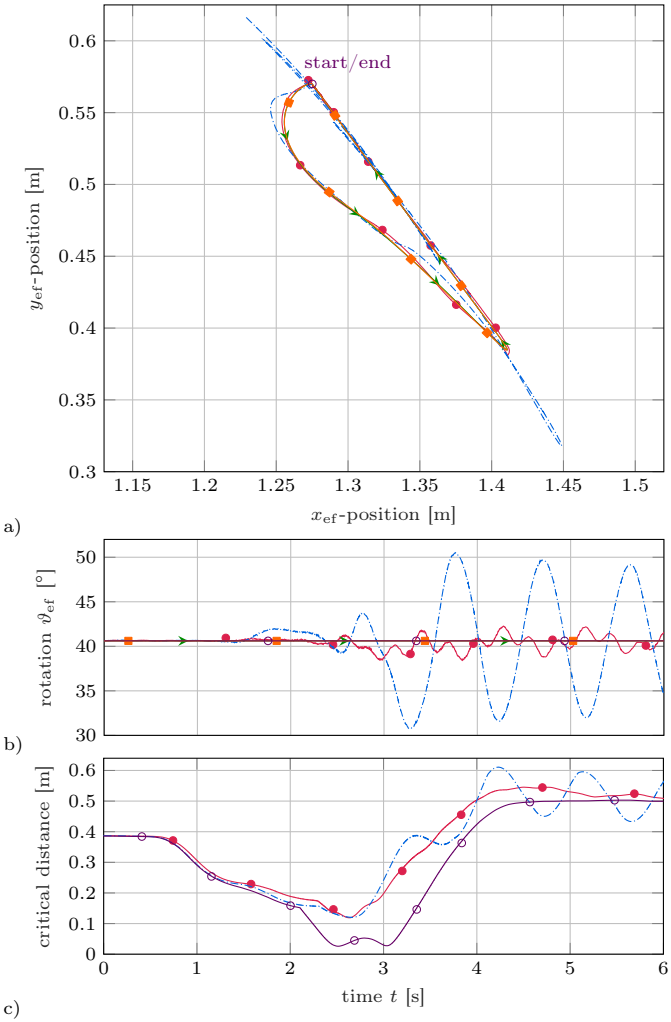


Figure 5.25: Dynamic obstacle: end-effector trajectory tracking experiments with obstacle collision avoidance for the flexible (desired ➤——, measured ●——) and rigid (desired ■——, measured -·-·-) model inversion, as well as the desired motion without obstacle collision avoidance (○——).

a direct force control technique, which explicitly controls the contact force to desired values via a force feedback loop. The main idea is to control the force along the constrained task directions and the motion along the unconstrained task directions. Thus, the end-effector motion and the normal contact force

are controlled simultaneously within two independent subspaces. Therefore, in contrast to indirect force control methods, such as the admittance control of Section 5.1.2, the environment needs to be known relatively accurately.
For flexible link robots, hybrid force/position control has been investigated for several decades. In the work of Lew and Book [156] from 1993, hybrid force/position control for flexible link manipulators with multiple contacts is considered. However, they use a quasi-static assumption and only look at regulation instead of trajectory tracking. Matsuno et al. [157] also utilize a quasi-static assumption, where static relations between the contact forces and the elastic deformations are used. Based on this, a hybrid force/position control is derived for flexible link manipulators. Matsuno et al. [158] also use the singular perturbation method for reducing the dynamic model of flexible manipulators to a slow and a fast subsystem. This serves to design a hybrid force/position controller. In the article of Madani and Moallem [43] from 2011, hybrid force/position control of a flexible link parallel manipulator is investigated. However, the controller only uses kinematic relations. In contrast, the presented research includes the complete system dynamics via model inversion based on servo constraints, to control the normal contact force error and the end-effector error on position level. Thus, a dynamic flexible model inversion in real-time analog to Section 5.2.2, which has proven to be quite robust, is now combined with output feedback control within a contact scenario.

The utilized control structure is illustrated in Figure 5.26. Here, the desired end-effector trajectory $\boldsymbol{y}_\mathrm{d}$ and its time derivatives are provided similar to the previous sections. For the considered 2D scenario in the xy-plane of the inertial system, this includes trajectories for the desired end-effector x, y-positions as well as for its rotation ϑ. However, now also a trajectory of the desired normal contact force $F_\mathrm{c,d}$ is required. To simplify the results the flat contact wall of Figure 2.9 is used which is aligned parallel to the y-axis. The utilized linear output controllers of the hybrid force/position control read

$$F_\mathrm{v} = F_\mathrm{c,d} + I_\mathrm{c} \int_0^t (F_\mathrm{c,d} - F_\mathrm{c}) \mathrm{d}\tau, \tag{5.33a}$$

$$\ddot{\boldsymbol{y}}_\mathrm{v} = \ddot{\boldsymbol{y}}_\mathrm{d} + \begin{bmatrix} D_x & 0 & 0 \\ 0 & D_y & 0 \\ 0 & 0 & D_\vartheta \end{bmatrix} (\dot{\boldsymbol{y}}_\mathrm{d} - \dot{\boldsymbol{y}}_\mathrm{o}) + \begin{bmatrix} P_x & 0 & 0 \\ 0 & P_y & 0 \\ 0 & 0 & P_\vartheta \end{bmatrix} (\boldsymbol{y}_\mathrm{d} - \boldsymbol{y}_\mathrm{o}) \,. \tag{5.33b}$$

The forward Euler method is used for the integration of the integral force control within Equation (5.33a) and for $\ddot{\boldsymbol{y}}_\mathrm{v}$ of the PD position control. The resulting term $\boldsymbol{y}_\mathrm{v}$ and its time derivatives are then utilized to replace the corresponding $\boldsymbol{y}_\mathrm{d}$ terms within the model inversion. Also, F_v replaces $F_\mathrm{c,d}$ within the model inversion.
In the framework of hybrid force/position control it needs to be ensured that only the unconstrained direction is controlled via the utilized PD position control to not interfere with the integral force control. The decoupling is realized by

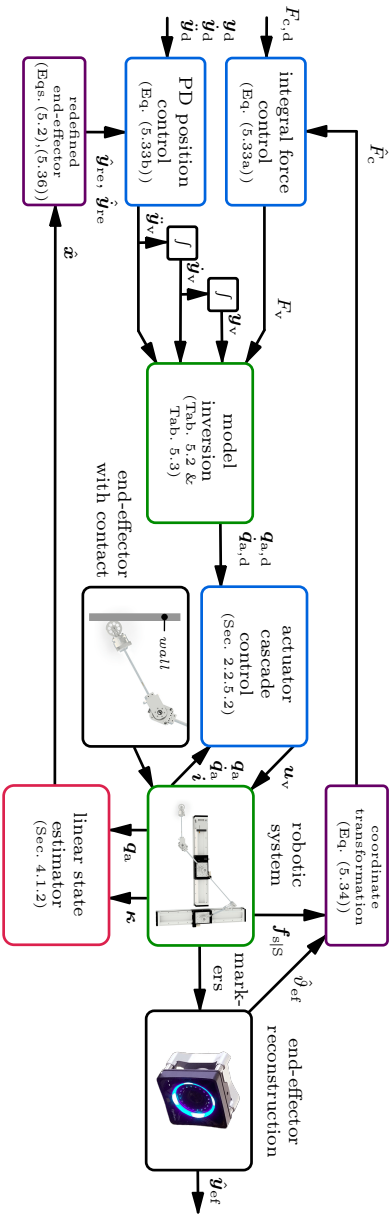


Figure 5.26: Structure of the hybrid force/position control.

using PD gains only for the direction tangential to the wall and for the rotation. Thus, within Equation (5.33) positive control gains I_c, P_y, P_ϑ, D_y, $D_\vartheta > 0$ are applied whereas $P_x = D_x = 0$, since the force control acts in x-direction. Hence, the end-effector y-position and its rotation ϑ are feedback controlled but not the x-position. Then, for an accurate model inversion Equation (5.33) leads to a stable error dynamics.
The normal contact force F_c corresponds here to the sensor force $F_{s,x}$ in the x-direction of the inertial frame. Therefore, the sensor force measurements $\boldsymbol{f}_{s|S} = [F_{s,x_s}, F_{s,y_s}, F_{s,z_s}]^T$ in the sensor frame S need to be transformed to the inertial frame I. This is done via the camera-based estimate of the exact end-effector rotation $\hat{\vartheta}_{ef}$, which is inserted into the force/torque sensor rotation matrix

$$\boldsymbol{S}_{IS} = \begin{bmatrix} \cos(\vartheta_{ef}) & -\sin(\vartheta_{ef}) & 0 \\ \sin(\vartheta_{ef}) & \cos(\vartheta_{ef}) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{5.34}$$

For a flexible model inversion exclusively the redefined output case is used in this section, with the weights of Table 5.3. Thus, the servo constraints

$$\boldsymbol{s}(t, \boldsymbol{q}) = \boldsymbol{y}_o(\boldsymbol{q}) - \boldsymbol{y}_d(t) = \boldsymbol{0} \tag{5.35}$$

then use the tracking output $\boldsymbol{y}_o = \boldsymbol{y}_{re}$ to ensure a stable internal dynamics. The influence of the contact force on the stability of the linearized zero dynamics turns out to be small. Within tests instability occurred only for relatively large forces, which will not be considered within experiments. The time derivative of $\boldsymbol{y}_o(\boldsymbol{q}) = \boldsymbol{y}_{re}(\boldsymbol{q})$ within Equation (5.33) can be obtained via

$$\dot{\boldsymbol{y}}_{re} = \frac{\partial \boldsymbol{y}_{re}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}}. \tag{5.36}$$

This can be evaluated with the state $\boldsymbol{x} = [\boldsymbol{q}^T, \dot{\boldsymbol{q}}^T]^T$, being estimated based on the measured actuator coordinates $\boldsymbol{q}_a$ and the link curvatures $\boldsymbol{\kappa}$.
It is worth noting that the flexible model inversion constrains the redefined end-effector to the wall, while on the real robot the exact end-effector is at the wall. This can be regarded as small modeling error which the control approach needs to handle. Also, similar to the previous cases without output feedback, a typical scenario with short stops, where the model and the real robot come to rest, will prevent long-term drifts through modeling errors. This ensures that the state of the flexible model inversion stays close to the state of the real robot.
As reference, again an equivalent rigid model inversion is considered. Here, the hybrid force/position control is not utilized, i.e., the corresponding PID gains are all set to zero, but only rigid inverse kinematics is applied. The reason is that using feedback of the flexibilities of the robot together with a rigid model inversion can lead to instability. Additionally, the normal contact force F_c does not occur in the algebraic constraint equations, which completely define the actuator motions of the rigid model.

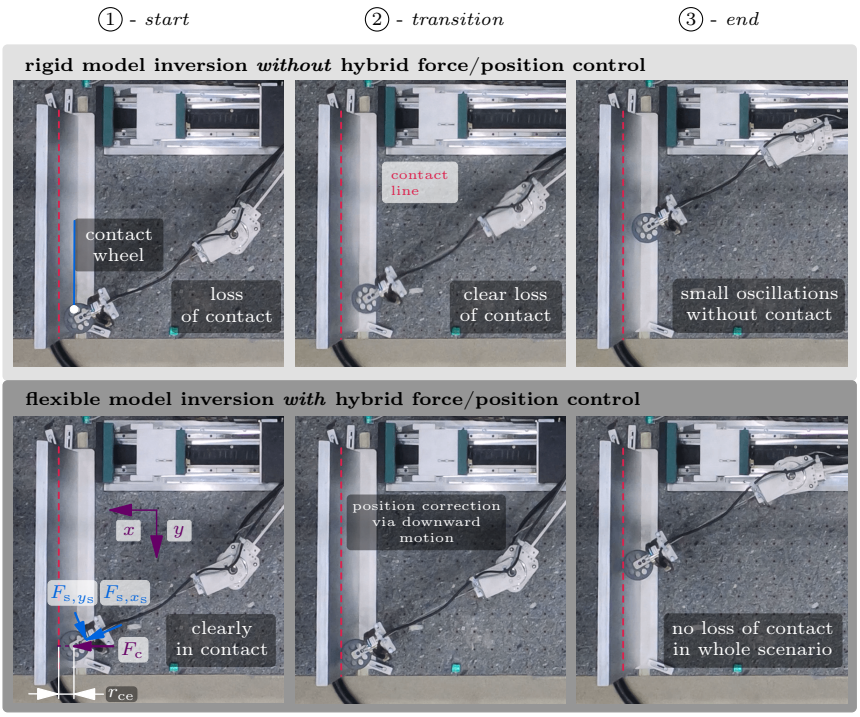


Figure 5.27: Snapshots of the hybrid force/position control scenario.

The considered scenario is illustrated in Figure 5.27. The end-effector shall follow a line, i.e., a flat wall of 0.25 m in 2 s with constant rotation and a normal contact force which smoothly changes from -7 N to -3 N. As the normal contact force F_c is always aligned with the x-axis, the negative sign implies that the end-effector shall be pushed into the negative x-direction.
Here, the rigid model inversion without hybrid force/position control is compared to the flexible model inversion with hybrid force/position control using the gains of Table 5.5. The advantages of the hybrid force/position control become clear especially in scenarios with moderate modeling errors. Therefore, the wall is shifted by 2.5 cm in positive x-direction, i.e., to the left. This shall introduce a realistic error, since the environment is often not exactly known. This error causes the rigid case to lose contact throughout the whole scenario, see Figure 5.27, despite permanent contact is desired. Also, the loss of contact leads to visible oscillations at the trajectory end. In contrast, the flexible case leads to link deformations right at the start. Therefore, the spring-like behavior of the flexible links ensures that the contact is not lost despite the wall position error. In

Table 5.5: Hybrid force/position control gains.

	force control	*position control*	
	I gain	**P gain**	**D gain**
x-position	-	$P_x = 0\,\mathrm{s}^{-2}$	$D_x = 0\,\mathrm{s}^{-1}$
y-position	-	$P_y = 1\,\mathrm{s}^{-2}$	$D_y = 1\,\mathrm{s}^{-1}$
rotation ϑ	-	$P_\vartheta = 1\,\mathrm{s}^{-2}$	$D_\vartheta = 1\,\mathrm{s}^{-1}$
normal contact force F_c	$I_\mathrm{c} = 0.8\,\mathrm{s}^{-1}$	-	-

combination with the hybrid force/position control, the flexible case is able to closely track the desired trajectory with permanent contact. The permanent contact also keeps the occurring oscillations highly reduced. The active output control can be seen in the second frame of the flexible model inversion. Here, the end-effector has moved slightly downwards, even before the desired trajectory starts to change, in order to correct initial output errors.
As mentioned in Section 3.3.3, the contact wheel center shall follow a desired trajectory, which has an offset from the wall surface of the wheel radius r_ce when having contact. The circular wheel shape ensures that the normal contact force always points to the wheel center. This simplifies the modeling since with the wheel center a fixed force application point can be chosen. Moreover, using a rolling contact highly reduces friction effects at the end-effector such that they can be neglected in the model. As a result, it is assumed that only the normal force F_c occurs at the contact.

The corresponding measurements are shown in Figure 5.28. It can be seen that both the rigid and the flexible case without hybrid force/position control perform very unsatisfyingly. While the rigid case has no contact to the wall throughout the whole scenario, Figure 5.28c) shows that the flexible case loses contact near the trajectory end due to the unmodeled shifting of the wall by 2.5 cm. This loss of contact causes the clearly visible oscillations in Figure 5.28a) and b). When the end-effector is in contact, oscillations of significantly higher frequencies occur with highly reduced amplitudes.
For the flexible case with hybrid force/position control, the errors for the feedback controlled redefined end-effector all converge to zero. However, since the exact end-effector is plotted in Figure 5.28a) and b), the desired steady-state values are not exactly reached. This is amongst others related to the fact that the exact and the redefined end-effector also differ within steady state due to the static deformations in contact scenarios. Nevertheless, the steady-state errors for the exact end-effector are very small when using the flexible case with hybrid force/position control. Since the visualized normal contact force F_c is also feedback controlled, it converges to the desired value in Figure 5.28c).
As mentioned before, the link elasticity is especially advantageous for contact scenarios, where it highly reduces the contact forces and the related risk of damage. The inherent compliance also helps to reduce the effect of modeling

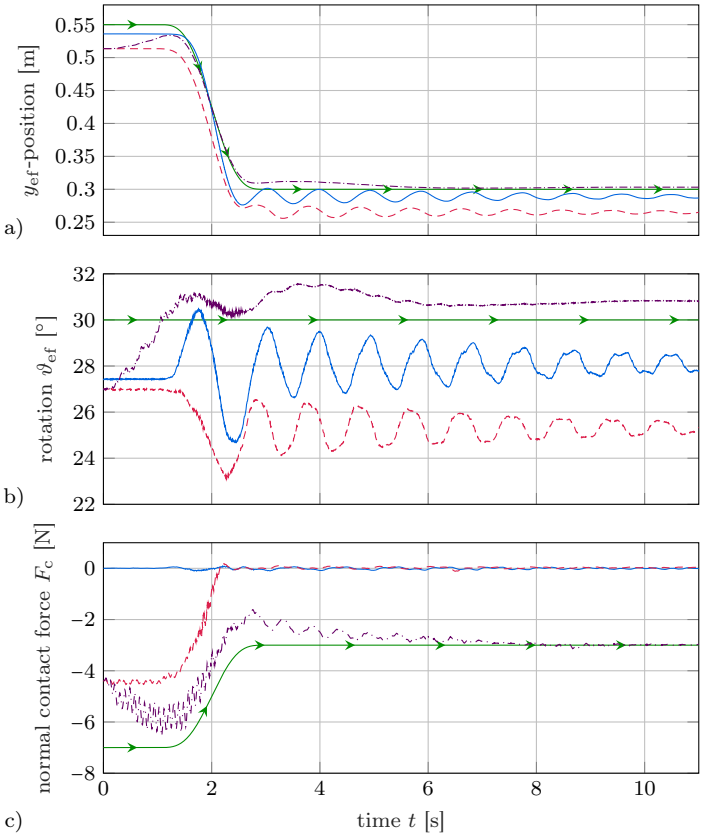


Figure 5.28: End-effector trajectory tracking experiments with contact for the rigid model inversion (——), the flexible model inversion without (– – – –) and with hybrid force/position control (–·–·–) as well as the desired trajectory (—➔—).

errors on the contact force. This can be seen in Figure 5.28c) where the flexible cases only lead to an initial force error of around 2.5 N. If in contrast a rigid link robot would slightly push into a wall due to modeling errors, the occurring contact forces can be devastating.

In summary, it is shown that the proposed real-time inversion of a dynamic flexible multibody model can also be successfully used within the framework of hybrid force/position control. Despite the complexity of contact scenarios, which involve amongst others uneven force measurements and moderate modeling errors of the environment, simultaneous force and position control is effectively realized in an experiment. Moreover, the utilized control structure can be applied

analogously to free end-effector scenarios, whereby no force feedback would be used.

5.2.4 Feedback Linearization

In the previous Section 5.2.3, it is shown that the proposed real-time model inversion approach can also be used together with output feedback control to account for tracking errors. Still, no state feedback is used, which tends to be more robust and consequently renders the experimental implementation often much easier. However, scenarios are possible where the internal state of such a model inversion differs significantly from the real flexible link robot. Therefore, the well-known and highly systematic concept of feedback linearization, which uses output and state feedback, is now adapted to flexible link parallel robots. Due to the existence of an internal dynamics when using a flexible model, full state feedback linearization is not possible. Since no rigid reference approach is considered, this section consequently focuses on input-output feedback linearization using a flexible multibody model.

The concept of feedback linearization [64] has been applied to flexible link robots before. Nevertheless, due to the typically non-minimum phase system property when tracking the exact end-effector, input-output feedback linearization has been mainly used to track joint trajectories for flexible link robots [11]. Such as by Li [159], who considers end-effector trajectory tracking of a serial robot with a flexible and a rigid link. Hereby, a collocated output, being the joint angles, is used for the input-output feedback linearization. Also, Jiang [129] performs end-effector trajectory tracking of a serial robot with two flexible links via computed torque, being a special form of full state feedback linearization, but based only on a rigid body model. Boyer and Khalil [160] discuss the theory of a computed torque method for serial flexible link robots, which only controls the rigid coordinates onto a desired trajectory. An additional stabilization of the elastic oscillations is then superimposed. Still, De Luca and Siciliano [126] track a point along a one-link flexible arm via input-output feedback linearization within simulations. The idea of output redefinition is also employed by Moallem et al. [161], who use an output close to the exact end-effector for trajectory tracking of a serial two-link robot with one flexible link. The tracking is realized via input-output linearization being validated within experiments.
The presented research also experimentally applies the concept of input-output feedback linearization for trajectory tracking of a redefined output close to the exact end-effector. However, in contrast to the cited literature, the considered system type are flexible link robots which also have a parallel part. Here, a systematic approach to realize the input-output feedback linearization is proposed via servo constraints.

5.2.4.1 The Lambda-Kinematics

The Lambda-Kinematics depicted in Figure 2.2j) has been used as test rig before and during the major development phase of Flexor, which finished in 2019. Therefore, some concepts have only been tested for the Lambda-Kinematics, which is located at the University of Stuttgart at the Institute of Engineering and Computational Mechanics. One of these concepts is feedback linearization. Due to its systematic approach, universality and high relevance for many applications it is discussed in this thesis but with an experimental validation using the Lambda-Kinematics. As both, the Lambda-Kinematics and Flexor, are flexible link parallel robots the modeling and control design are analogue. Therefore, no details are discussed here on the flexible multibody model of the Lambda-Kinematics. Instead, the application of feedback linearization to flexible link parallel robots is the focus.
The different components of the Lambda-Kinematics are shown in Figure 5.29. In regard of the similarities to Flexor, the same notation is used for the sake of comprehensibility. Here, the motor currents $\boldsymbol{u} = [u_\mathrm{a}, u_\mathrm{b}]^\mathrm{T}$ are the control inputs of two ironless linear motors from KML which actuate the system. Three 2 mm thin spring steel sheets are used for the short link 1 and the long link 2. The connection of both links at the point L forms the parallel part, i.e., the kinematic loop. The robot owes its name to the resulting shape, which resembles the letter λ. Similar to Flexor, link 1 is approximated as rigid body since it exhibits only negligible oscillations for normal operation. Link 2, however, can show substantial deformation amplitudes being exemplarily visualized in Figure 5.29 via a dashed

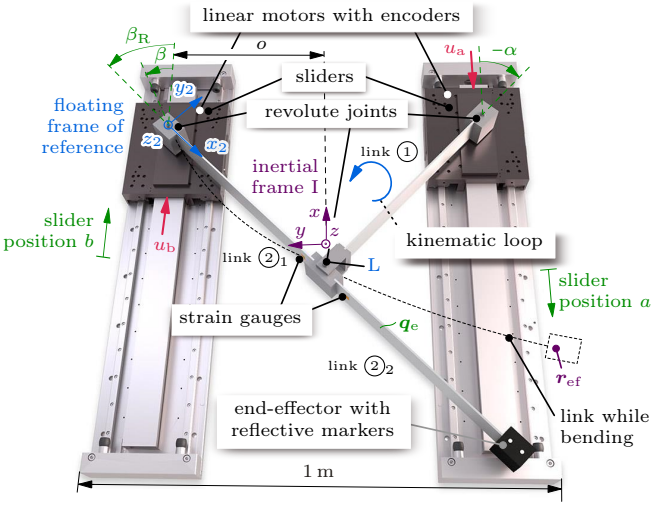


Figure 5.29: Overview of the Lambda-Kinematics.

curve. Here, only the first bending eigenmode is significant. Thus, the elastic coordinates $\boldsymbol{q}_\mathrm{e}$ only consist of the corresponding scalar $q_{\mathrm{e},2}$. The utilized sensors comprise strain gauges in Wheatstone bridge circuits, attached to link 2, in order to obtain information on the curvature $\boldsymbol{\kappa}$ at different positions. Furthermore, incremental optical linear encoders measure the actuated positions $\boldsymbol{q}_\mathrm{a} = [a,\, b]^\mathrm{T}$ of the sliders, each having a range of motion of ±30 cm. The main differences compared to Flexor are that the Lambda-Kinematics does not have:

- a third serial link with a rotary motor;
- end-effector variations such as a gripper or a force/torque sensor with a contact element;
- a real-time capable motion tracking camera system.

Due to the first two bullet points, the Lambda-Kinematics only allows the investigation of rather academic scenarios. Regarding the third point, a camera is mounted vertically over the experimental rig similar to Flexor. However, it can only be used with offline image processing to identify the end-effector motion, based, e.g., on two retroreflective foil markers. Since the robot moves in the xy-plane, the 2D-position of this end-effector $\boldsymbol{r}_{\mathrm{ef|I}}$ in the inertial frame I forms the output of interest. Besides, the utilized image processing is not automatically synchronized to the time stamps of the real-time system. Also, for feedback linearization a stable zero dynamics is realized by output redefinition. Since processing the camera measurements is cumbersome as well as not real-time capable and as the virtual redefined output shall also be evaluated, all outputs are exclusively obtained via a UKF, analogously to Section 4.1.3. The camera measurements have only been used to validate this UKF, but are not further used in this research for the Lambda-Kinematics. The sample time of the real-time target is 0.5 ms, which is also the sample time of the estimators and the model-based controllers within experiments.
The Lambda-Kinematics was mainly developed by Markus Burkhardt. For further details it is therefore referred to his doctoral thesis [104] and to the article of Burkhardt et al. [46].

5.2.4.2 Feedback Linearization with Servo Constraints

For input-output feedback linearization a minimum phase system is again obtained by output redefinition, analogously to Section 5.2.1.1. For the Lambda-Kinematics the redefined output $\boldsymbol{y}_\mathrm{re}$ to be tracked only contains the 2D end-effector position but no rotation, i.e., $\boldsymbol{y}_\mathrm{re} = \boldsymbol{r}_\mathrm{re}$. The redefinition is done by weighting the only retained elastic coordinate $q_{\mathrm{e},2}$, related to the first bending eigenmode of link 2, by the factor $w_{\mathrm{p},2}$ for the elastic deformation. If this weight is chosen as $w_{\mathrm{p},2} = 1$ the exact end-effector is obtained and $w_{\mathrm{p},2} = 0$ gives an equivalent rigid end-effector.
The fulfilled sufficient conditions for the proposed input-output feedback linearization based on geometric and servo constraints, via a static state feedback law, can be summarized as:

1) The system is minimum phase, due to the tracking of an appropriate redefined output $\boldsymbol{y}_{\mathrm{re}}$.

2) The number of inputs and outputs, introduced via servo constraints, is equal.

3) The vector relative degree related to the geometric and servo constraints, being $\boldsymbol{r} = \{2, ..., 2\}$, is well-defined with a regular extended decoupling matrix $\boldsymbol{\Delta}$ of Equation (3.45).

In regard of condition 2), for the Lambda-Kinematics two inputs and two outputs are available. It is worth noting that when α is -90° or β is 90°, or when being close to such an angle depending on the elastic deformation, the actuators can move the end-effector only in the x-direction and $\boldsymbol{\Delta}$ becomes singular. This behavior is similar to Flexor, for which the singular constellations are discussed and visualized in Section 3.4.2. Since such singular constellations are only reached close to the actuator limits, for both the Lambda-Kinematics and Flexor, they do not prevent the fulfillment of condition 3) for a typical operation.

The implemented control structure, without contact at the end-effector, is illustrated in Figure 5.30 being explained in the following. Feedback linearization is usually applied to systems without geometric constraint equations. Still, for the considered system type geometric constraints due to the parallel part exist but also servo constraints are utilized. To apply the concept of input-output feedback linearization nonetheless, these constraints are incorporated via their second time derivatives. Together with the system dynamics equation (3.45) has been obtained, see Section 3.4.1. This equation already represents the static state feedback law of the input-output feedback linearization to obtain the required control inputs $\boldsymbol{u}$, as well as the Lagrange multipliers $\boldsymbol{\lambda}$. To evaluate Equation (3.45) the state $\boldsymbol{x} = [\boldsymbol{q}^{\mathrm{T}}, \dot{\boldsymbol{q}}^{\mathrm{T}}]^{\mathrm{T}}$ is needed, which is estimated via a UKF based on the measured actuator coordinates $\boldsymbol{q}_{\mathrm{a}}$ and the link curvatures $\boldsymbol{\kappa}$.
It should be noted that no transformation to the (Byrnes-Isidori) input–output normal form [162] is needed to obtain the input-output linearizing controller. Similarly, in Section 3.5.1 the internal dynamics is directly obtained via projections instead of a transformation to such an input–output normal form. The internal dynamics is also not needed to calculate the input-output linearizing controller, but it is required for stability analysis. Here, the internal dynamics is used to find a redefined output that renders this internal dynamics stable, similar to Section 5.2.1.1.
By applying the control $\boldsymbol{u}$ obtained via Equation (3.45) all nonlinear terms are theoretically canceled, which yields a linear differential input-output relation between the new control input $\ddot{\boldsymbol{y}}_{\mathrm{v}}$ and the tracking output $\boldsymbol{y}_{\mathrm{re}}$. This is ensured by adapting the servo constraints according to

$$\boldsymbol{s}(t, \boldsymbol{q}) = \boldsymbol{y}_{\mathrm{o}}(\boldsymbol{q}) - \boldsymbol{y}_{\mathrm{d}}(t) = \boldsymbol{0} \quad \rightarrow \quad \boldsymbol{s}(t, \boldsymbol{q}) = \boldsymbol{y}_{\mathrm{re}}(\boldsymbol{q}) - \boldsymbol{y}_{\mathrm{v}}(t, \boldsymbol{q}) = \boldsymbol{0}. \tag{5.37}$$

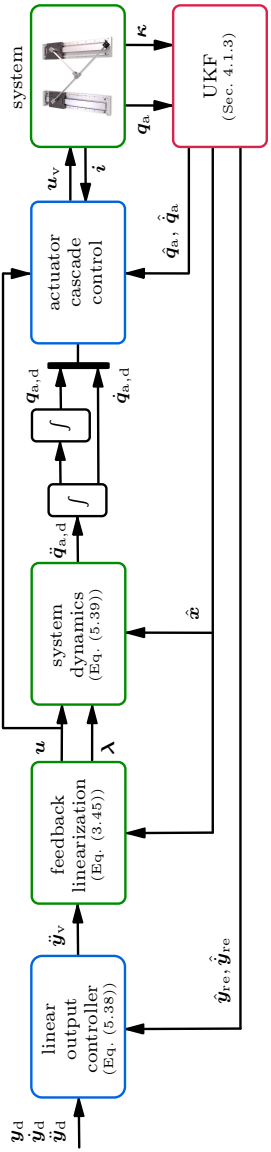


Figure 5.30: Control structure of the input-output feedback linearization based on geometric and servo constraints.

This means that the redefined end-effector is implemented as tracking output, i.e., $\boldsymbol{y}_\mathrm{o} = \boldsymbol{y}_\mathrm{re}$ and the desired trajectory $\boldsymbol{y}_\mathrm{d}$ is replaced by $\boldsymbol{y}_\mathrm{v}$. In contrast to FLEXOR, for the LAMBDA-KINEMATICS the velocity control is not done within the motion controller but within SIMULINK. Therefore, in Figure 5.30 the calculated model-based current control inputs $\boldsymbol{u}$ are as well provided to the actuator cascade control. Nevertheless, the overall control approach can be applied to both systems and to other flexible link robots.
Selecting the new control input as the second time derivative of the desired output trajectory, i.e., $\ddot{\boldsymbol{y}}_\mathrm{v} = \ddot{\boldsymbol{y}}_\mathrm{d}$, gives $\ddot{\boldsymbol{y}}_\mathrm{re} = \ddot{\boldsymbol{y}}_\mathrm{d}$ in an ideal scenario. Obviously, the input-output feedback linearization is an exact linearization of the nonlinear system leading to such a linear relation, which is much easier to control. Therefore, to compensate occurring errors in the control output $\boldsymbol{y}_\mathrm{re}$ the simple linear controller

$$\ddot{\boldsymbol{y}}_\mathrm{v} = \ddot{\boldsymbol{y}}_\mathrm{d} + \begin{bmatrix} D & 0 \\ 0 & D \end{bmatrix} (\dot{\boldsymbol{y}}_\mathrm{d} - \dot{\boldsymbol{y}}_\mathrm{re}) + \begin{bmatrix} P & 0 \\ 0 & P \end{bmatrix} (\boldsymbol{y}_\mathrm{d} - \boldsymbol{y}_\mathrm{re}) \tag{5.38}$$

is used for the LAMBDA-KINEMATICS, which yields a stable output error dynamics for $P, D > 0$. Regarding the Lagrange multipliers $\boldsymbol{\lambda}$ as additional control inputs one can analogously stabilize the geometric constraints $\boldsymbol{c}(\boldsymbol{q}) = \boldsymbol{0}$ [163]. This is however not necessary when using a real system where these constraints cannot drift.
As a next step, the motor currents $\boldsymbol{u}$ are transformed to actuator motions to guarantee amongst others an effective friction compensation via the utilized actuator cascade control. This transformation is simply done by plugging the available control inputs $\boldsymbol{u}$ and the Lagrange multipliers $\boldsymbol{\lambda}$ as well as the state $\boldsymbol{x}$ into the system dynamics (3.22a). This gives

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}^{-1} \left(\boldsymbol{f} + \boldsymbol{B}\boldsymbol{u} + \boldsymbol{C}^\mathrm{T} \boldsymbol{\lambda} \right), \tag{5.39}$$

where $\ddot{\boldsymbol{q}}$ contains the needed actuator accelerations $\ddot{\boldsymbol{q}}_\mathrm{a}$. Evaluating Equation (5.39) is computationally very efficient since the inverse of $\boldsymbol{M}$ is already known from using Equation (3.45) and the forward Euler method is accurate enough for the subsequent time integration. The resulting desired actuator motions $\boldsymbol{q}_\mathrm{a,d}$ and $\dot{\boldsymbol{q}}_\mathrm{a,d}$ are then closely reproduced via the actuator cascade control.
Alternatively, if the currents $\boldsymbol{u}$ are not needed, e.g., to apply the control only on position and velocity level, the desired actuator accelerations $\ddot{\boldsymbol{q}}_\mathrm{a,d}$ can be obtained directly via Equation (3.63b). Here, again a replacement of $\ddot{\boldsymbol{y}}_\mathrm{d}$ by $\ddot{\boldsymbol{y}}_\mathrm{v}$ is needed within $\boldsymbol{\theta}''_\mathrm{q}$.
It is worth noting that a major difference to the previously discussed dynamic model inversion controllers, which use an internal time integration in real-time, is that the feedback linearization only uses an algebraic, i.e., static control law which is computationally much more efficient, compare Table 2.4. Nevertheless, the feedback linearization needs a state estimator which can be computationally expensive. This is especially the case for the utilized UKF which internally integrates the system dynamics.

Now, the feedback linearization is applied experimentally to the LAMBDA-KINEMATICS for a desired line trajectory with a length of $0.2\sqrt{2}\,\mathrm{m}$ to be tracked within 0.5 s, see Figure 5.31. Within these plots it can be observed that a weight $w_{\mathrm{p},2}$ closer to a value of 1, i.e., choosing a redefined output $\boldsymbol{y}_{\mathrm{re}}$ closer to the exact end-effector output $\boldsymbol{y}_{\mathrm{ef}}$, yields a more effective trajectory tracking behavior of $\boldsymbol{y}_{\mathrm{ef}}$. Also, the redefined output, which is actually tracked by the controller, is kept very close to the desired end-effector trajectory $\boldsymbol{y}_{\mathrm{d}}$.
It should be noted that all results in this section, whether of the exact or of the redefined end-effector, are based on estimations of the UKF which does not use camera measurements. Thus, as opposed to FLEXOR the results are not directly measured and consequently they do not show all errors coming from the robot model. Also, the utilized SI units for the PD gains are dropped for the sake of readability.

The corresponding maximum tracking errors for the line trajectory are shown in Figure 5.32a) for different weights $w_{\mathrm{p},2}$ and PD control gains. The bars represent mean values of three different experiments. Here, the errors for the redefined end-effector $\boldsymbol{y}_{\mathrm{re}}$ are quite small, such that only a minor improvement is visible when using non-zero PD gains. Nevertheless, the PD gains help to reduce the errors in all cases and also have a positive effect on the exact end-effector $\boldsymbol{y}_{\mathrm{ef}}$. The main error within the exact end-effector comes obviously from output redefinition being needed to obtain a minimum phase system. This redefinition error can only be safely decreased if, e.g., disturbances are reduced to allow a higher value

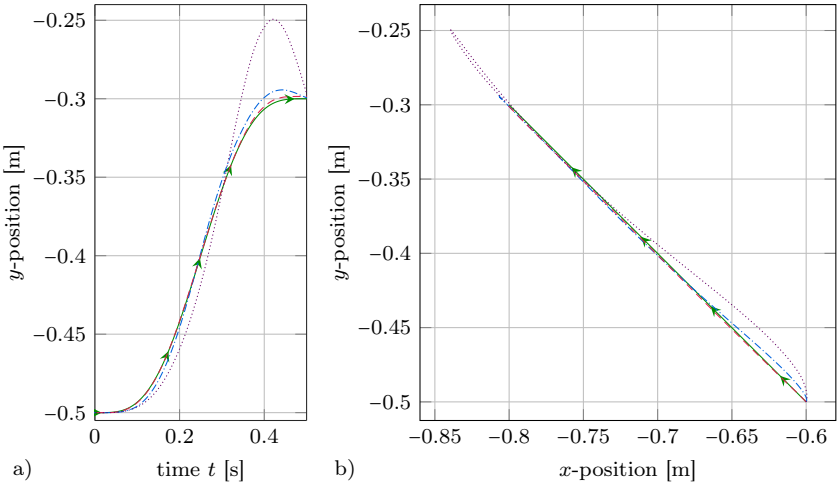


Figure 5.31: End-effector trajectory tracking experiments for input-output feedback linearization with $P = D = 10$ showing a) the end-effector y-position and b) the end-effector path for $\boldsymbol{y}_{\mathrm{ef}}$ with $w_{\mathrm{p},2} = 0.75$ (-·-·-), for $\boldsymbol{y}_{\mathrm{re}}$ with $w_{\mathrm{p},2} = 0.75$ (- - - -) and for $\boldsymbol{y}_{\mathrm{ef}}$ with $w_{\mathrm{p},2} = 0.3$ (·····), as well as for the desired trajectory (——→).

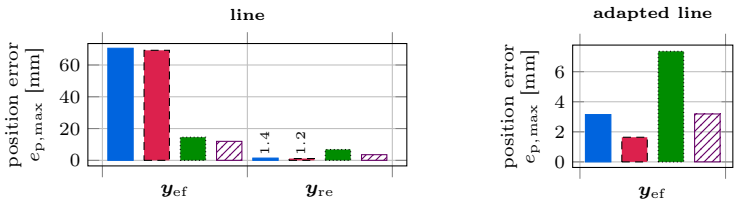


Figure 5.32: End-effector trajectory tracking errors for input-output feedback linearization experiments with $w_{p,2} = 0.3$, $P = D = 0$ ( ), $w_{p,2} = 0.3$, $P = D = 10$ ( ), $w_{p,2} = 0.75$, $P = D = 0$ ( ) and $w_{p,2} = 0.75$, $P = D = 10$ ( ).

than the robust value of $w_{p,2} = 0.75$. The experimental setup under consideration turned out to become unstable roughly at $w_{p,2} = 0.85$. This is still somewhat away from the numerically robust weight of $w_{p,2} = 0.9625$, which can be safely used within a model inversion to obtain a minimum phase system. Thus, the redefinition error for the feedback linearization is significantly higher than for the model inversion. As a result, the feedback linearization based on output redefinition is mainly useful for scenarios where using a state feedback is so advantageous that it can outweigh this error discrepancy.
Nevertheless, for a case in which the desired end-effector trajectory is known in advance, such as for repetitive tasks, the error introduced through output redefinition can be pre-compensated. Initially, the trajectory tracking problem, formulated, e.g., as a BVP, is solved offline for the exact end-effector as output. The calculated states from this model inversion can then be used to evaluate the corresponding redefined end-effector motion. Applying this motion, denoted as adapted "line" trajectory, as new desired trajectory will ensure that a close tracking of $\boldsymbol{y}_{re}$ of this adapted line results in a close tracking of $\boldsymbol{y}_{ef}$ of the original line. The highly reduced tracking errors of the original line for the exact end-effector via this trajectory adaption method can be seen in Figure 5.32b). Furthermore, the PD gains now improve the exact end-effector trajectory tracking error by a significant percentage. Interestingly, the $w_{p,2} = 0.3$ case performs better than the $w_{p,2} = 0.75$ case for such an adapted trajectory. However, as this could not be confirmed within simulations it is most likely an experimental effect occurring through the different weighting of the elastic coordinate.

In summary, the concept of input-output feedback linearization, based on servo constraints, is successfully applied to a parallel robot with a flexible link. The effective end-effector trajectory tracking is validated experimentally. The tracking performance is even further improved by adapting the desired trajectory offline based on tracking of the exact end-effector. Due to the resulting unstable internal dynamics a BVP is solved. Consequently, this approach involves an element which actually belongs to the next Section 5.3 and thus offers a first outlook on its advantages. Section 5.3 will compare the redefined output case and the BVP case in closer detail.

5.3 Unstable Internal Dynamics

In the previous Section 5.2, approaches are proposed to obtain a minimum phase system description for flexible link robots. These approaches even lead to a stable internal dynamics within different real-time end-effector trajectory tracking scenarios and result in an effective control performance. Nevertheless, when using the exact end-effector as output flexible link robots usually exhibit a non-minimum phase system behavior. In some instances it might be undesired or difficult to adapt this system property resulting in an unstable internal dynamics. As calculating a bounded solution for such an unstable inverse system is much more involved and needs to be done offline, it is mainly useful for repetitive tasks. Throughout this section it is shown that non-minimum phase system descriptions lead to a similar experimental end-effector trajectory tracking performance as the output redefinition approach based on Section 5.2.1.1, which results in a minimum phase system.

The structure utilized for all controllers in this section is shown in Figure 5.33. Here, the user provides a complete desired end-effector trajectory $\boldsymbol{y}_\mathrm{d}$ with corresponding time derivatives in advance. In a scenario with end-effector contact, a nonzero desired normal contact force $F_\mathrm{c,d}$ needs to be provided as well. For all cases the model inversion is then pre-calculated offline for consistency. Nevertheless, equivalent rigid and minimum phase systems can be inverted online if needed. The obtained desired actuator motions $\boldsymbol{q}_\mathrm{a,d}$ on position and $\dot{\boldsymbol{q}}_\mathrm{a,d}$ on velocity level are sent to the actuator cascade control. Thus, like in Section 5.2.1 only feedback of the actuator measurements is used, but neither measurements of the elastic deformations nor measurements of the contact forces are fed back. This ensures a stable operation since the model-based part is implemented as feedforward control. Similar to Section 5.2.1, the control performance is therefore highly dependent on the model of the flexibilities. Even if feedback is used within model-based controllers, a relatively accurate model is necessary to ensure a stable behavior.

It should be noted that it is not possible to use output feedback if the system is

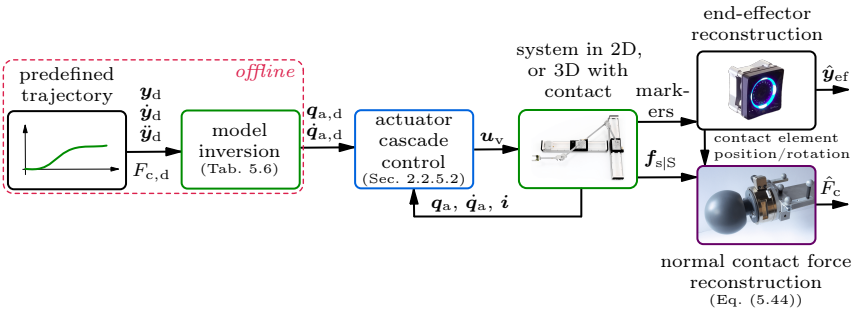


Figure 5.33: Control structure of the offline model inversion.

non-minimum phase. Nevertheless, one could use feedback of the elastic deformations via the LQR or the modal damping controller which will be discussed in Chapter 6. They can be implemented between the model inversion and the actuator cascade control. As the model inversion also calculates the desired elastic state $\boldsymbol{q}_{\mathrm{e,d}}$, $\dot{\boldsymbol{q}}_{\mathrm{e,d}}$ these controllers allow to reduce tracking errors within the link deformations being measured, e.g., by strain gauges. Since the resulting structure would be analog to Chapter 6 and as both controllers are mainly useful for scenarios with larger modeling errors, which is not the case here, they are however not further considered in this section. For more information on using these controllers for end-effector trajectory tracking of flexible link robots it is pointed to the author's publications [29, 36].
Finally, as visualized in Figure 5.33 the control performance is evaluated with the camera-based end-effector reconstruction $\hat{\boldsymbol{y}}_{\mathrm{ef}}$ and with the reconstruction of the magnitude of the normal contact force $\hat{F}_{\mathrm{c}}$ based on the force measurements $\boldsymbol{f}_{\mathrm{s|S}}$ and the camera measurements.

For the Lambda-Kinematics described in Section 5.2.4.1, feedforward control based on an offline performed stable inversion of the full dynamic model has been applied before within experiments, which also only rely on feedback of actuator measurements [46]. These results are now extended to Flexor which has an additional flexible link and besides a 2D scenario also a 3D scenario with contact to a wall is considered. Here, stable inversion from Section 3.6.4 is compared to model inversion via an equivalent rigid kinematic model from Section 3.6.1, to model inversion via output redefinition from Section 5.2.1.1 and to the trajectory optimization approaches from Sections 3.6.3.1 and 3.6.3.2. For the implementation details it is referred to the referenced sections. In the following the focus lies on the experimental comparison in order to show the different properties of these alternative methods to solve the end-effector trajectory tracking problem for flexible link robots.

5.3.1 Two-Dimensional System

Initially, the standard 2D setup of Flexor with a free end-effector is considered. For the investigated end-effector trajectory tracking problem the five different model inversion cases outlined in Table 5.6 are compared. The output which is tracked within the model and the resulting internal dynamics determine which solution methods are applicable. For the rigid case no internal dynamics exists and the problem reduces to inverse kinematics which can be solved by root finding. In this case, the rigid end-effector is tracked which corresponds to the redefined output of Equation (5.2) with zero weighting parameters, i.e., $w_{\mathrm{p},2} = w_{\mathrm{p},3} = w_{\mathrm{r},2} = w_{\mathrm{r},3} = 0$. For such a redefined output but, e.g., with the weighting parameters of the collision avoidance scenario from Section 5.2.2, the flexible system is minimum phase and a simple IVP can be solved for model inversion. In contrast, for the exact end-effector, which corresponds to

Table 5.6: Model inversion cases.

case	elastic DOFs	tracking output	equations of motion formulation	solution method	parameters	internal dynamics
rigid (Sec. 3.6.1)	none	*rigid* end-effector position and rotation	only algebraic constraint equations (Fig. 3.22)	root finding	equivalent rigid model	none
redefined output (Sec. 5.2.1.1)	first two bending eigenmodes	*redefined* end-effector position and rotation (Eq. (5.2))	form in chain coordinates via QR decomposition (Fig. 3.15)	classical inversion via forward integration of an IVP	$w_{\mathrm{p},2} = 0.8$, $w_{\mathrm{p},3} = 0.8$, $w_{\mathrm{r},2} = 1.07$, $w_{\mathrm{r},3} = 0.5$	stable
BVP (Sec. 3.6.4)	first two bending eigenmodes	*exact* end-effector position and rotation	minimal form via mixed coordinate partitioning with $\boldsymbol{q}_\mathrm{e}$ as independent coordinates (Fig. 3.16)	stable inversion for a two-point BVP		unstable
optimization with **three servo constraints** (Sec. 3.6.3.1)	first two bending eigenmodes	*exact* end-effector position and rotation	minimal form via mixed coordinate partitioning with $\boldsymbol{q}_\mathrm{e}$ as independent coordinates (Fig. 3.16)	nonlinear programming solver	▸ no bounds on design variables ▸ 2.5 s included before and after the desired end-effector motion	unstable
optimization with **two servo constraints** (Sec. 3.6.3.2)	first two bending eigenmodes	*redefined* end-effector position, *no rotation* (Eq. (5.2a))	minimal form via mixed coordinate partitioning with $\boldsymbol{q}_\mathrm{e}$, γ as independent coordinates and free u_γ (Fig. 3.18)	nonlinear programming solver	▸ $w_{\mathrm{p},2} = w_{\mathrm{p},3} = w_{\mathrm{r},2} = 0.97$ ▸ steady-state boundary conditions ▸ without pre-/post-actuation	unstable

$w_{p,2} = w_{p,3} = w_{r,2} = w_{r,3} = 1$, the system is non-minimum phase and either a BVP or a trajectory optimization problem, converted to a nonlinear programming problem, has to be solved. For the considered BVP and optimization cases the equations of motion are used in minimal form which makes the problem formulation more compact.
It is worth noting that in the optimization case with two servo constraints the obtained solution is causal, i.e., no pre-actuation phase occurs. Also, a post-actuation is prevented. As a result, slider A might need to perform very steep start and end motions with large accelerations in order to excite the required bending deformation for trajectory tracking of the exact end-effector. This is numerically and experimentally challenging. Therefore, a slight output redefinition is used as in Section 5.2.1.3, which has a negligible effect on the experimental end-effector tracking performance but significantly smooths the actuator motions.

Experimental snapshots of selected cases are shown in Figure 5.34 for a desired line trajectory of 0.5 m with constant rotation within 1.5 s. As the motion is mainly transverse to the longitudinal axis of the elastic links, the rigid case leads to extreme oscillations at the trajectory end. In contrast, the other two cases which are based on a flexible model result in very small oscillations. For the BVP case the line with constant rotation is closely tracked, whereas the optimization with two servo constraints only tracks the line but not the rotation, which is not constant anymore. Consequently, the overall motion as well as the start and end pose is completely different since the obtained kinematic redundancy is amongst others used to reduce the link deformations, see Section 3.6.3.2.
The corresponding measurements are plotted in Figure 5.35. In the path plot of Figure 5.35a) the inaccurate position tracking performance of the rigid model inversion is again visible while all other approaches track the line trajectory closely. Figure 5.35b) confirms the completely different end-effector rotation for the optimization case with two servo constraints. The maximum Euclidean norm of the position errors and the maximum absolute rotational error in Figure 5.35c) quantify these observations. Here, each bar represents the mean of three experiments which are very repeatable. It can be clearly seen that the large effort for solving a BVP or an optimization problem, when tracking the exact end-effector, does not lead to notable improvements compared to the computationally efficient and real-time capable redefined output case. This is caused by modeling and measurement errors. Tracking the exact end-effector would therefore be mainly advantageous for systems where no redefined output close to the exact end-effector can be found to render the system minimum phase. Also, within purely simulative studies the tracking of the exact end-effector leads to negligible errors which clearly outperforms the tracking of a redefined output, see Figure 5.14a).
It should be noted that both optimization cases have been solved based on an initial guess of zero for all design variables, which includes zero link deformations. In contrast, the BVP could not be solved for the exact end-effector with an initial

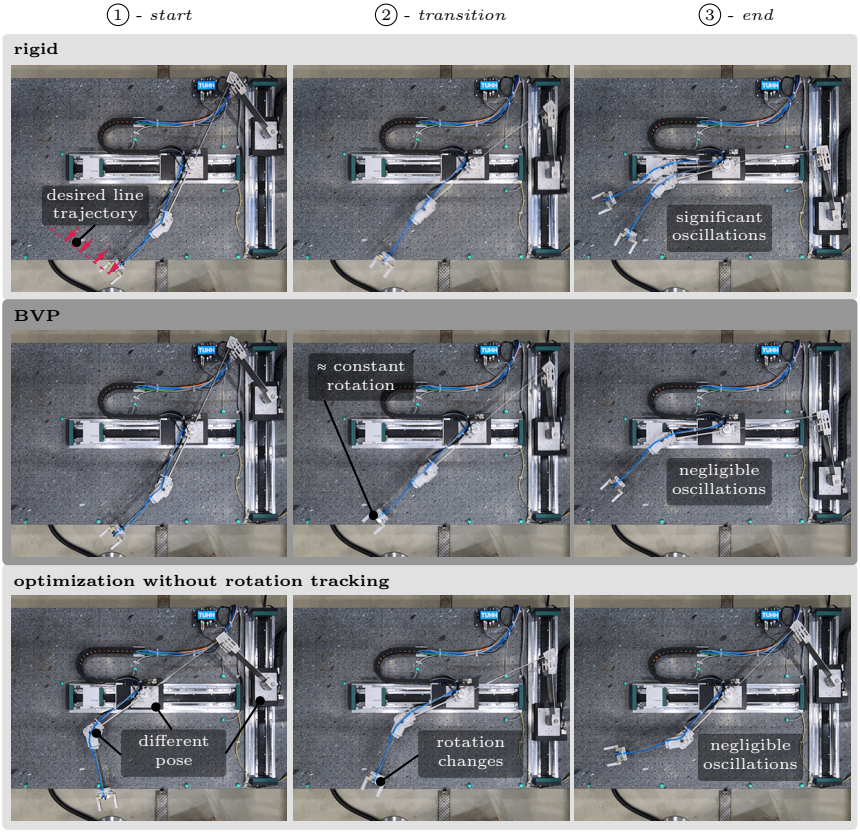


Figure 5.34: End-effector line trajectory tracking with constant rotation based on three different model inversion cases.

guess of zero deformation, i.e., a rigid initial guess. An often promising approach to obtain an appropriate initial guess is output redefinition. Already a minor output redefinition, which still results in a non-minimum phase system, can simplify the solution of the BVP. Here, the weights $w_{\mathrm{p},2} = w_{\mathrm{p},3} = w_{\mathrm{r},2} = w_{\mathrm{r},3} = 0.99$ allow to solve the BVP even with an initial guess of zero deformation. The result can then be used as initial guess to solve the BVP for tracking of the exact end-effector. Alternatively, the result of the redefined minimum phase case or of the trajectory optimization with three servo constraints can be used as initial guess for the BVP. The key difference is that a more accurate initial guess makes it more likely to find a solution for the BVP in a shorter amount of time.

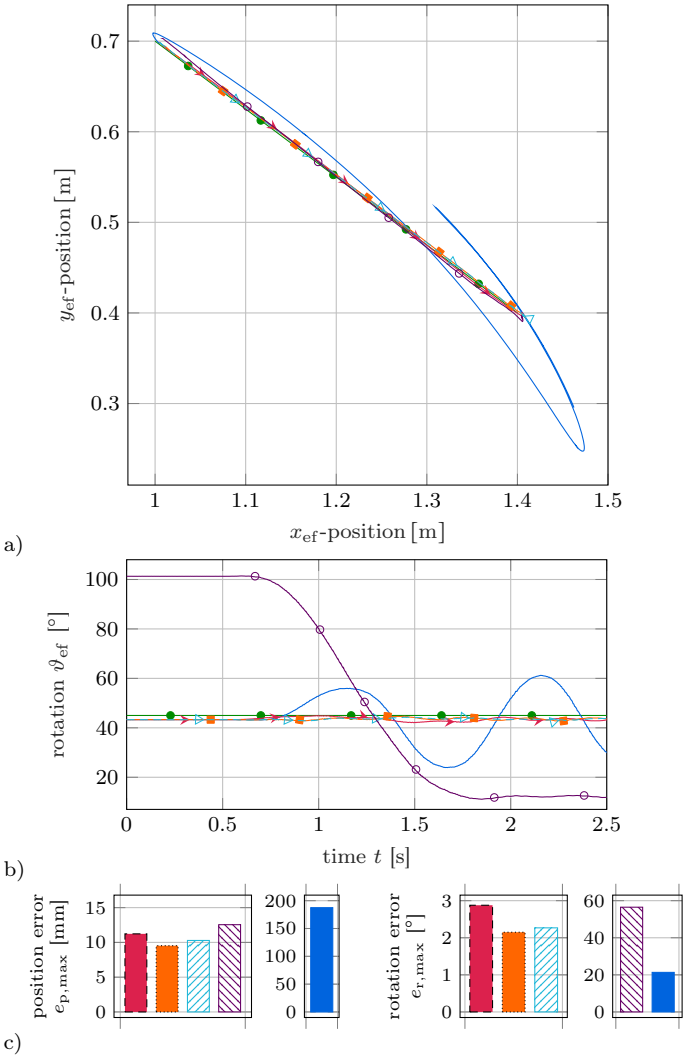


Figure 5.35: End-effector trajectory tracking experiments: measured end-effector pose and errors for the rigid case (——, ■), the redefined output case (➤——, ■), the BVP case (■——, ■), the optimization case with three servo constraints (▷- - -, ▨) as well as with two servo constraints (○——, ▨) and the desired trajectory (●——).

Figure 5.36 shows the corresponding actuator motions. In the magnification plot of the slider position a the pre-actuation phase becomes apparent for the BVP

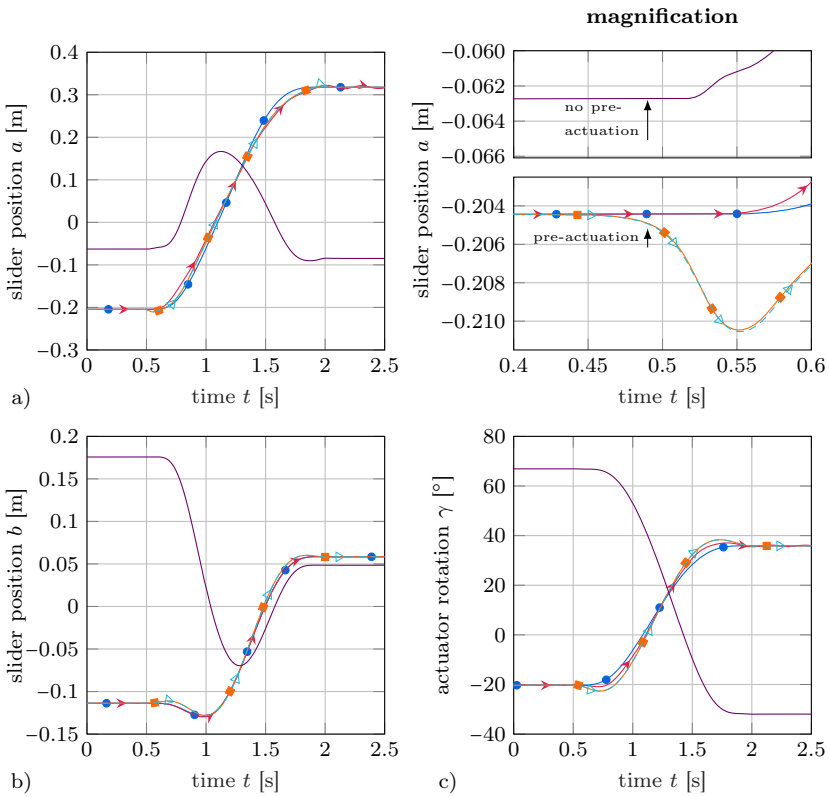


Figure 5.36: End-effector trajectory tracking experiments: measured actuator positions for the rigid case (●——), the redefined output case (➤——), the BVP case (■——), the optimization case with three servo constraints (▷- - -) and with two servo constraints (——).

case and the optimization case with three servo constraints. This pre-actuation starts before the desired end-effector motion which begins at 0.5 s. It is needed to introduce a link deformation before the output moves in order to track the desired trajectory with the exact end-effector. As mentioned before, the optimization case with two servo constraints yields a causal solution, i.e., no pre-actuation is needed. Also, its completely different actuator motions can be seen.
These different motions are needed to minimize the link deformations, i.e., curvatures, see Figure 5.37. Especially the curvature for link 2 is significantly reduced. Although it is very small, the measured and simulated signals match qualitatively quite well. These results confirm that the presented modeling and control approach allows to make accurate simulation-based predictions and optimizations which can be transferred almost directly to an experiment. For the other three

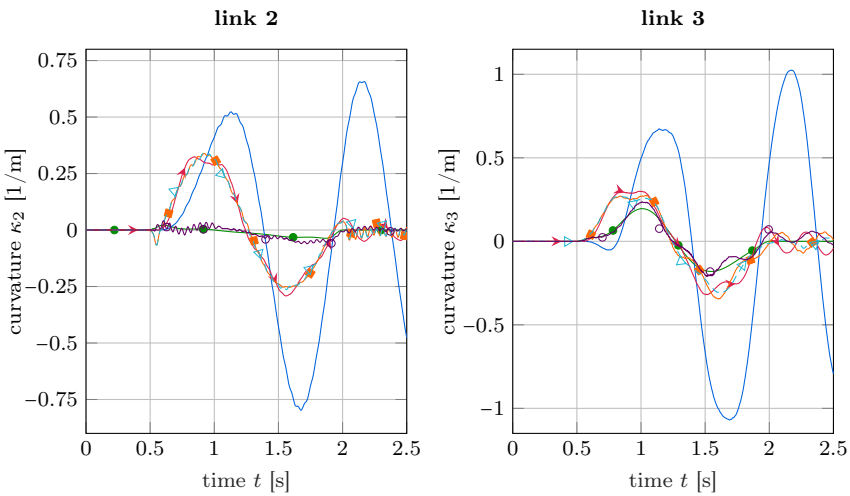


Figure 5.37: End-effector trajectory tracking experiments: measured curvatures for the rigid case (——), the redefined output case (——), the BVP case (——) and the optimization case with three servo constraints (- - -). The optimization case with two servo constraints is measured (——) and simulated (——).

flexible model inversion cases the elastic deformation is much larger, which is necessary to also follow the desired end-effector rotation. They perform very similar, while for the BVP and the optimization case with three servo constraints the pre-actuation phase leads to a small minimum just after 0.5 s within link 2. This excites a minor bending in the opposite direction to ensure an accurate tracking of the exact end-effector.

In summary, the need for a flexible model to obtain close trajectory tracking of the end-effector is again confirmed. The four presented flexible cases are almost equally effective in tracking the considered line trajectory. Thus, the most appropriate approach depends mainly on the application and the user preferences. For real-time trajectory adaptions a minimum phase system, e.g., via output redefinition, is needed. If such a redefined minimum phase output close to the exact end-effector cannot be found for a repetitive task, the BVP or trajectory optimization is recommended. In such an instance the trajectory optimization is well suited to find an accurate initial guess for the BVP to simplify its numerical solution. In the case of kinematic redundancy, the trajectory optimization enables a straightforward realization of additional objectives, such as the minimization of the link deformations, besides end-effector trajectory tracking. This also allows to obtain a causal solution without pre-actuation.

5.3.2 Three-Dimensional System with Contact

To further validate the effectiveness of the proposed offline calculated model inversion approach of Figure 5.33, a 3D trajectory tracking scenario is considered for FLEXOR. Through flipping the rotary motor by 90° as shown in Figure 2.5 on the right, link 3 now rotates upwards. Since the additional gravitational effects together with a motion can lead to extreme deformations of link 3, its thickness is increased from 1 mm to 1.5 mm. As contact scenarios are experimentally more challenging, this section focuses on the contact between the spherical 3D contact element introduced in Figure 2.6 and a higher version of the quadrant-shaped wood wall of Figure 2.9.

From the model inversion cases introduced in Table 5.6 a reduced set is considered to cover one case of each system type. It comprises the rigid case without internal dynamics, the redefined case with stable internal dynamics and the BVP case with unstable internal dynamics. From a modeling stand point no structural changes occur in the equations of motion, including the constraint equations, compared to the free end-effector case in 2D. As explained in Section 3.4, the desired normal contact force $F_{\mathrm{c,d}}$ is simply nonzero now and gravity needs to be taken into account.
Firstly, the rigid model inversion has been investigated. Large undesired forces, which occurred already at the initial conditions, confirm that a flexible model is required for the considered 3D contact scenario to a stiff wall. To prevent damaging the robot no trajectory tracking experiments are conducted based on the rigid model inversion. Secondly, the utilized redefined output $\boldsymbol{y}_{\mathrm{re}}$ in 3D corresponds to the position vector $\boldsymbol{r}_{\mathrm{re}}$

$$\begin{aligned}\boldsymbol{y}_{\mathrm{re}} = \boldsymbol{r}_{\mathrm{re}} = &\begin{bmatrix} o+b \\ 0 \\ z_0 \end{bmatrix} + \begin{bmatrix} \cos(\beta_{\mathrm{R}}) & -\sin(\beta_{\mathrm{R}}) & 0 \\ \sin(\beta_{\mathrm{R}}) & \cos(\beta_{\mathrm{R}}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \ell_{2_2} \\ w_{\mathrm{p},2}\phi_2 q_{\mathrm{e},2} \\ 0 \end{bmatrix} \\ &+ \begin{bmatrix} \cos(\beta_{\mathrm{R}} + w_{\mathrm{r},2}\psi_2 q_{\mathrm{e},2}) & -\sin(\beta_{\mathrm{R}} + w_{\mathrm{r},2}\psi_2 q_{\mathrm{e},2}) & 0 \\ \sin(\beta_{\mathrm{R}} + w_{\mathrm{r},2}\psi_2 q_{\mathrm{e},2}) & \cos(\beta_{\mathrm{R}} + w_{\mathrm{r},2}\psi_2 q_{\mathrm{e},2}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &\begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ 0 & 0 & -1 \\ \sin(\gamma) & \cos(\gamma) & 0 \end{bmatrix} \begin{bmatrix} \ell_3 \\ w_{\mathrm{p},3}\phi_3 q_{\mathrm{e},3} \\ 0 \end{bmatrix}. \end{aligned} \tag{5.40}$$

In Equation (5.40), the motor angle γ is zero when the undeformed link 3 is horizontal and an increasing γ raises link 3. For such a horizontal link 3, its center is at the constant height z_0 from the table surface of the robot. The other variables and parameters are analog to Equation (5.2). Similar to the 2D case, if all three elastic weighting design parameters are set to 1, i.e., $w_{\mathrm{p},2} = w_{\mathrm{p},3} = w_{\mathrm{r},2} = 1$,

then $\boldsymbol{r}_{\mathrm{re}}$ equals the exact end-effector 3D position

$$\boldsymbol{r}_{\mathrm{ef|I}} = \begin{bmatrix} x_{\mathrm{ef}} \\ y_{\mathrm{ef}} \\ z_{\mathrm{ef}} \end{bmatrix}. \tag{5.41}$$

It is located in the center of the contact element sphere and given in inertial coordinates with respect to the origin O of the inertial frame. It should be noted that for 3D and contact scenarios link deformations also occur in steady state. As a result, the redefined output differs from the exact end-effector also at the trajectory start and end, in contrast to the 2D scenario with free end-effector of the previous Section 5.3.1. For model inversion of the redefined output case the servo constraints

$$\boldsymbol{s}(t, \boldsymbol{q}) = \boldsymbol{y}_{\mathrm{o}}(\boldsymbol{q}) - \boldsymbol{y}_{\mathrm{d}}(t) = \boldsymbol{0} \tag{5.42}$$

use the tracking output $\boldsymbol{y}_{\mathrm{o}} = \boldsymbol{y}_{\mathrm{re}} = \boldsymbol{r}_{\mathrm{re}}$ with $\boldsymbol{y}_{\mathrm{d}} = [x_{\mathrm{d}}, y_{\mathrm{d}}, z_{\mathrm{d}}]^{\mathrm{T}}$. To obtain a minimum phase system and even a stable internal dynamics, the setting of Table 5.6 is used with the same weighting parameters. The only difference is that the parameter $w_{\mathrm{r},3}$ is not needed since the end-effector rotation is not included in the 3D redefined output of Equation (5.40).
Thirdly, for the BVP case also the settings from Table 5.6 are applied where the exact end-effector position is utilized as tracking output, i.e., $\boldsymbol{y}_{\mathrm{o}} = \boldsymbol{y}_{\mathrm{ef}} = \boldsymbol{r}_{\mathrm{ef|I}}$. Here, the solution of the redefined output case is used as initial guess.

The considered scenario and kinematic details are shown in Figure 5.38. For trajectory tracking, the 2D position of the contact element on the wall as well

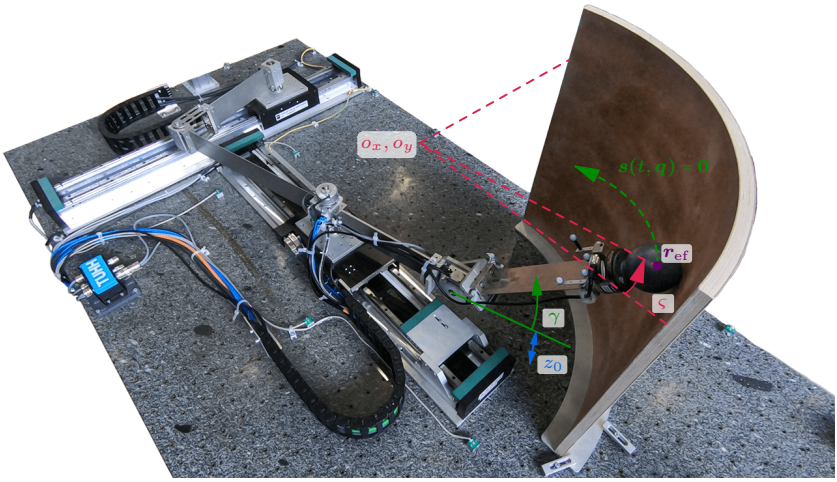


Figure 5.38: Contact scenario in 3D.

as the magnitude of the normal contact force are of interest. This means that the tracking output $\boldsymbol{y}_\mathrm{o}$ of the model, being a 3D position vector, differs from the output of interest for control and evaluation purposes. The 2D position on the wall is described by the height z_ef and the angle ς, which corresponds to cylindrical coordinates with constant radius. The angle ς is defined in the first quadrant according to

$$\varsigma = \arctan\Big((y_\mathrm{ef} - o_y)/(x_\mathrm{ef} - o_x)\Big). \tag{5.43}$$

Here, the origin of the circle described by the wall is at $x = o_x$ and $y = o_y$. The normal contact force $\boldsymbol{f}_{\mathrm{c|I}} = [F_{\mathrm{c},x}, F_{\mathrm{c},y}, F_{\mathrm{c},z}]^\mathrm{T}$ in the inertial frame needs to be reconstructed from the force measurements $\boldsymbol{f}_{\mathrm{s|S}} = [F_{\mathrm{s},x_\mathrm{s}}, F_{\mathrm{s},y_\mathrm{s}}, F_{\mathrm{s},z_\mathrm{s}}]^\mathrm{T}$ and the torque measurements $\boldsymbol{t}_{\mathrm{s|S}} = [T_{\mathrm{s},x_\mathrm{s}}, T_{\mathrm{s},y_\mathrm{s}}, T_{\mathrm{s},z_\mathrm{s}}]^\mathrm{T}$ at the sensor origin $\mathrm{O_s}$ given in the sensor frame S. To clarify the relation, a free body diagram of a general contact scenario is illustrated in Figure 5.39. It can be seen that besides the different orientation of the inertial and the sensor frame the unknown friction force $\boldsymbol{f}_{\mathrm{f|I}} = [F_{\mathrm{f},x}, F_{\mathrm{f},y}, F_{\mathrm{f},z}]^\mathrm{T}$ and the mass m_ce of the spherical contact element, which includes the mass of the sensor adapter plate, need to be taken into account. Since it is difficult to accurately model friction effects and a detailed description is out of scope of this research, they shall be substantially reduced. This could be done with a rolling contact similar to Section 5.2.3. Since available solutions such as ball casters typically only allow a small range of contact angles, as over the half of the contact ball needs to be inside the housing to be form locked, a rolling contact is not applied here. Instead, the utilized contact sphere is rigidly

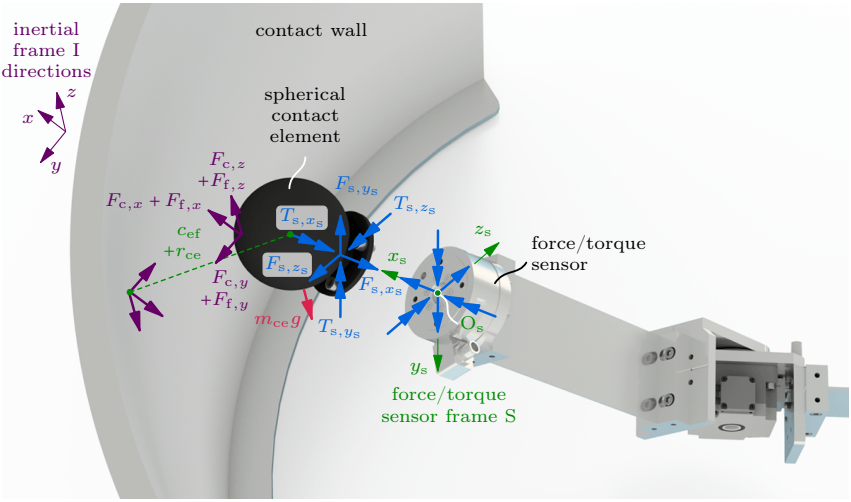


Figure 5.39: Free body diagram of the contact element in 3D.

attached to the robot and allows a wide range of contact angles while sliding on the wall. The occurring friction between these two contact bodies is significantly reduced by a dry lubrication with Teflon spray. The remaining small friction force $\boldsymbol{f}_\mathrm{f}$ is neglected in the following also since an accurate quantification is difficult. Therefore, the principle of linear momentum is sufficient to obtain the unknown normal contact force as

$$\boldsymbol{f}_{\mathrm{c|I}} = \boldsymbol{S}_{\mathrm{IS}} \boldsymbol{f}_{\mathrm{s|S}} + m_{\mathrm{ce}} \left(\ddot{\boldsymbol{r}}_{\mathrm{ef|I}} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \right) \tag{5.44}$$

and the measured torques $\boldsymbol{t}_{\mathrm{s|S}}$ are not needed. Equation (5.44) incorporates the minor approximation that the center of gravity of the contact element lies in the sphere center at the end-effector point. The force/torque sensor rotation matrix $\boldsymbol{S}_{\mathrm{IS}}$ equals the rotation matrix $\boldsymbol{S}_{\mathrm{IT}}$, which comes from the marker-based camera 3D reconstruction described in Section 4.2.5. This reconstruction also delivers the 3D end-effector position $\boldsymbol{r}_{\mathrm{ef|I}}$, which is differentiated twice via Kalman filtering as described in Section 4.1.1. The gravitational constant is denoted by g. For the considered contact scenario, the wall is vertical so that no normal contact force in z-direction occurs. With the remaining xy-components of Equation (5.44) the magnitude of the normal contact force is obtained via

$$F_\mathrm{c} = \sqrt{F_{\mathrm{c},x}^2 + F_{\mathrm{c},y}^2}, \tag{5.45}$$

which completes the output of interest.

The corresponding measurements of the three components of this output are shown in Figure 5.40 for the output redefinition and the BVP case. The trajectory tracking performance of both cases is again very similar. The overall larger errors compared to the 2D case with free end-effector are expected. On the one hand, the accuracy of the camera-based end-effector reconstruction in 3D is smaller than in 2D. And on the other hand, errors in the wall placement and in the wall shape as well as neglected friction in the contact zone disturb the trajectory tracking. The influence on the contact force tends to be more severe than on signals on position level. For instance, only in the force measurements of Figure 5.40c) a steep change occurs just after the first second at the trajectory start. Still, both the end-effector pose on the wall as well as the normal contact force are tracked relatively closely. The close force tracking is also enabled by the considered motion which makes use of the link elasticity. This inherent compliance is a major advantage of flexible robots in contact scenarios. In contrast, small modeling errors for a rigid robot can lead to very large undesired contact forces and damage.

In summary, the proposed modeling and control concepts are able to yield an effective end-effector trajectory tracking performance also within 3D contact scenarios. The minimum phase case with redefined output performs again very similar to the non-minimum phase case which tracks the exact end-effector. Since

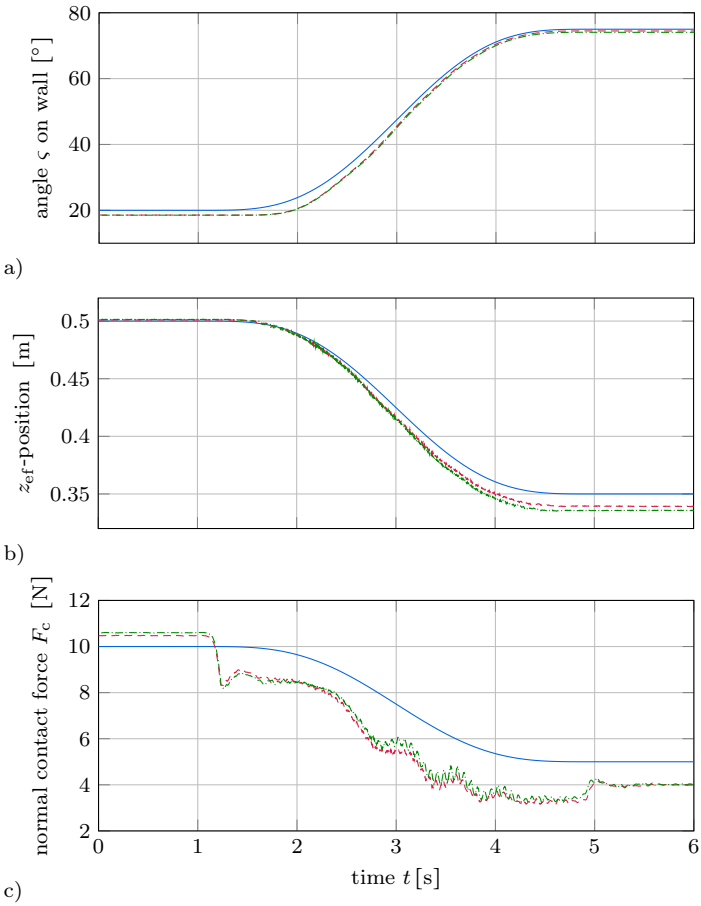


Figure 5.40: End-effector 3D trajectory tracking experiments with contact for the redefined output case (-·-·-) and the BVP case (- - - -) as well as the desired trajectory (——).

accurate tracking of the normal contact force is difficult especially in a 3D scenario with sliding contact, the real-time capable redefined output case could be used in future investigations with additional output feedback control analogously to Section 5.2.3.

Chapter 6

Oscillation Damping Control

Chapter Contents

In this chapter active oscillation damping of flexible link parallel robots is discussed. The two proposed concepts can be used amongst others:

- for working point changes after which the system shall quickly come to rest;
- to damp oscillations caused by external disturbances;
- after a trajectory tracking scenario to damp the remaining oscillations.

The damping of occurring oscillations is ensured by enforcing desired elastic deformations and corresponding time derivatives of zero within the controllers. In experiments, this chapter exclusively considers the planar case for FLEXOR, since the damping of the coupled oscillations between link 2 and link 3 is of major interest. Also, only the free end-effector case is investigated as within permanent contact scenarios the oscillations are of much higher frequencies and typically of significantly smaller amplitudes.

A large variety of oscillation damping approaches exists in the literature. An interesting method is based on internal resonance which has been applied in [164] to a cantilever beam simulation. Here, an internally resonant pair is generated via a controller, which is used to damp the plant. Since the torque requirements of the presented internal resonance controller are unidirectional it suits applications which are actuated by, e.g., thrusters or tendons, but it is in this form not applicable to classically actuated robots such as FLEXOR. Springer et al. [165] use model predictive control for oscillation damping of a flexible link robot. However, as model predictive control tends to be computationally expensive it is not further considered in this research. Input or command shaping [9, 166] can also be used to reduce vibrations of the flexible links. But being a feedforward approach renders it not meaningful for the desired application scenarios which include unmodeled disturbances.
Another oscillation damping approach is wave-based control proposed by O'-Connor and Lang [167]. The idea is that an actuator motion launches waves and

simultaneously absorbs the echoed, i.e., returning waves in the considered flexible structure. The approach has been applied experimentally to a single flexible link in [168]. While for such a single link and for a serial flexible link robot in [169] wave-based control proved to be effective, still several open questions regarding the theory of wave-based control need to be solved according to Malzahn and Bertram [169]. Nevertheless, these authors showed the equivalence of the resulting control structure of wave-based control and a modal damping approach based on direct strain feedback. Therefore, such a well-understood modal approach is pursued in Section 6.3 which applies the basic idea of direct strain feedback control, being the feedback of the time derivative of the link curvatures in order to directly increase the system damping [170]. A modal approach very similar to [169] has been discussed earlier by Brüls [171, p. 201], which is based on inertial damping schemes. Here, the modal damping is increased via the rigid variables which are used to feed back the velocity of the modal coordinates. These approaches of direct strain feedback and inertial damping control lead to the same control structure and are very similar to the presented approach in Section 6.3. However, on the one hand within direct strain feedback the strain is often finally used instead of the time derivative and it does not automatically imply a modal decomposition. On the other hand, the inertial damping schemes are developed for macro/micro-manipulators. Therefore, the approach discussed in this research is denoted as *modal damping control* to differentiate it from the techniques in [169, 171]. It is compared to the LQR from Section 6.2 being applied to oscillation damping, since both concepts are systematic, well-understood and effective.
In [23–25], LQRs are applied experimentally to serial robots with two flexible links. There, the desired state trajectories come from rigid body inverse kinematics. In opposition to this, here parallel instead of serial robots are considered, which complicates the control design. In this regard, the DAEs are transformed to ODEs via projection, which allows to use the standard LQR algorithm after a linearization of the minimal form.
The main contribution of this chapter is to discuss and relate these two active oscillation damping approaches, being the modal damping control and the LQR, to design them for parallel flexible link robots and to show their promising experimental performance on Flexor.

6.1 Overview

The main steps to arrive at the proposed oscillation damping controllers for flexible link parallel robots are summarized in Figure 6.1, which are discussed throughout this chapter. After modeling of the flexible links, two different formulations of the equations of motion of the forward model are utilized. For the LQR the standard formulation is used with actuator forces or corresponding

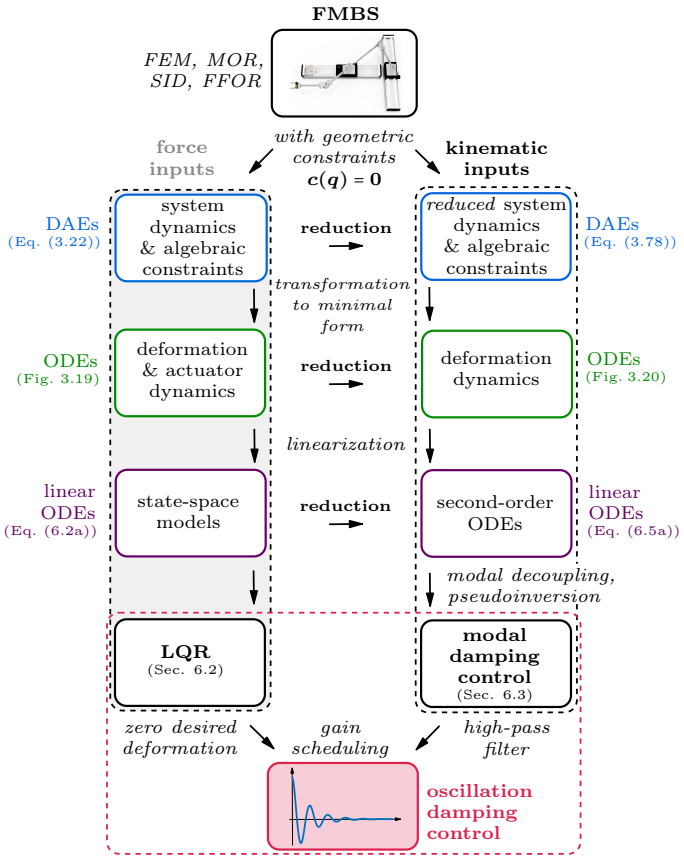


Figure 6.1: Overview of the LQR and the modal damping controller for oscillation damping of flexible link parallel robots.

currents as inputs. This ensures that the complete state can be controlled. For the modal damping control only the damping of the structural oscillations shall be increased. Here, a formulation with kinematic inputs for the actuators is advantageous.

Initially, a transformation from DAEs to ODEs in minimal form is performed via coordinate partitioning. Such a formulation in minimal, i.e., independent coordinates $\boldsymbol{q}_\mathrm{i}$ ensures that the damping control designs work since the equations of motion related to the dependent coordinates and the corresponding zero eigenvalues are canceled. Then, linearizations are conducted at different steady-state working points for the considered planar case, i.e., with zero deformations and inputs $\boldsymbol{u} = \boldsymbol{0}$. This is reasonable as typically the nonlinearity mainly comes

from the underlying rigid body poses and the flexible links oscillate around these points. All these steps eventually enable that powerful linear techniques such as the considered LQR and a modal decoupling to increase the damping can be applied to parallel robots like FLEXOR. The obtained controllers are interpolated, i.e., gain scheduled between the linearization points to account for nonlinearities within the kinematics. In future investigations, linear parameter-varying models might be used for a more systematic analysis and design of such gain-scheduled controllers, compare Hoffmann and Werner [172].
Besides, a reduction of the formulation from force inputs to kinematic inputs can be done within DAE form as shown in Section 3.5.4. Nevertheless, this reduction can be also performed at a later step of the procedure such as after the linearization.

6.2 Linear-Quadratic Regulator

Using an LQR for oscillation damping of flexible link robots can be advantageous as it not only damps the oscillations but controls the complete state, which also ensures a correct robot positioning. The LQR formalism, see, e.g., [173], is designed for linear forward models in state-space representation. Therefore, in the first step the DAEs (3.22), which describe a general forward model of flexible link parallel robots, are transformed to state space via projections. Here, a representation in independent coordinates, i.e., in minimal form, is necessary as the dependent coordinates introduce zero eigenvalues which render FLEXOR to be not stabilizable. Consequently, the forward model of Figure 3.19, i.e., the function $\dot{\boldsymbol{x}}_\mathrm{i} = \boldsymbol{f}_\mathrm{i}^\mathrm{fw}(\boldsymbol{x}_\mathrm{i}, \boldsymbol{u})$, is utilized for the LQR.
In the second step, this function needs to be linearized to enable the application of the LQR formalism. The linearization is done with respect to $\boldsymbol{x}_\mathrm{i}$ and $\boldsymbol{u}$ in order to obtain a linear system with independent state variables. The linearization is performed at steady-state points, i.e., the state $\boldsymbol{x}_\mathrm{i} = [\boldsymbol{q}_\mathrm{i}^\mathrm{T}, \dot{\boldsymbol{q}}_\mathrm{i}^\mathrm{T}]^\mathrm{T}$ is chosen as $\boldsymbol{x}_{\mathrm{i,s}} = [\boldsymbol{q}_{\mathrm{i,s}}^\mathrm{T}, \boldsymbol{0}^\mathrm{T}]^\mathrm{T}$ and the input $\boldsymbol{u}$ is chosen as $\boldsymbol{u}_\mathrm{s} = \boldsymbol{0}$. For FLEXOR, $\boldsymbol{q}_\mathrm{i}$ is now of dimension

$$f_\mathrm{i} = f - n_\mathrm{c} - n_\mathrm{s} = 7 - 2 - 0 = 5. \tag{6.1}$$

Here, the fixed set $\boldsymbol{q}_\mathrm{i} = [\boldsymbol{q}_\mathrm{e}^\mathrm{T}, \boldsymbol{q}_\mathrm{a}^\mathrm{T}]^\mathrm{T}$ is used where $\boldsymbol{q}_\mathrm{e}$ consists again only of the first two bending eigenmodes of link 2 and link 3, respectively. This is different, e.g., from the model inversion formulated as a BVP in Section 5.3, where only the elastic coordinates $\boldsymbol{q}_\mathrm{e}$ are selected as independent. For $\boldsymbol{q}_{\mathrm{i,s}}$ different actuator positions are chosen at each linearization point with zero link deformation.
Analogously to $\boldsymbol{f}_\mathrm{i}^\mathrm{fw}$ within Figure 3.19, the output $\boldsymbol{y}_\mathrm{o}(\boldsymbol{q}) \in \mathbb{R}^{f_\mathrm{o}}$ can also be written as a function $\boldsymbol{y}_\mathrm{o}(\boldsymbol{q}_\mathrm{i})$ with internal root finding for the dependent coordinates $\boldsymbol{q}_\mathrm{d} = \boldsymbol{q}_\mathrm{d}(\boldsymbol{q}_\mathrm{i})$. Thus, this output function can also be linearized with respect to the independent state $\boldsymbol{x}_\mathrm{i}$, where the derivatives with respect to $\dot{\boldsymbol{q}}_\mathrm{i}$ are zero. The

linear state-space representation in independent coordinates then follows as

$$\dot{\tilde{\boldsymbol{x}}}_\mathrm{i} = \left.\frac{\partial \boldsymbol{f}_\mathrm{i}^\mathrm{fw}}{\partial \boldsymbol{x}_\mathrm{i}}\right|_{\substack{\boldsymbol{x}_\mathrm{i,s}\\ \boldsymbol{u}_\mathrm{s}}} \tilde{\boldsymbol{x}}_\mathrm{i} + \left.\frac{\partial \boldsymbol{f}_\mathrm{i}^\mathrm{fw}}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}_\mathrm{i,s}\\ \boldsymbol{u}_\mathrm{s}}} \boldsymbol{u} = \widetilde{\boldsymbol{A}}_\mathrm{i}\tilde{\boldsymbol{x}}_\mathrm{i} + \widetilde{\boldsymbol{B}}_\mathrm{i}\boldsymbol{u}, \tag{6.2a}$$

$$\tilde{\boldsymbol{y}}_\mathrm{o} = \left.\frac{\partial \boldsymbol{y}_\mathrm{o}}{\partial \boldsymbol{x}_\mathrm{i}}\right|_{\boldsymbol{x}_\mathrm{i,s}} \tilde{\boldsymbol{x}}_\mathrm{i} = \widetilde{\boldsymbol{C}}_\mathrm{i}\tilde{\boldsymbol{x}}_\mathrm{i}, \tag{6.2b}$$

with $\tilde{\boldsymbol{x}}_\mathrm{i} = \boldsymbol{x}_\mathrm{i} - \boldsymbol{x}_\mathrm{i,s}$ representing variations around the considered steady-state working point $\boldsymbol{x}_\mathrm{i,s}$. Here, $\widetilde{\boldsymbol{A}}_\mathrm{i} \in \mathbb{R}^{2f_\mathrm{i}\times 2f_\mathrm{i}}$ is the constant system matrix, $\widetilde{\boldsymbol{B}}_\mathrm{i} \in \mathbb{R}^{2f_\mathrm{i}\times f_\mathrm{a}}$ is the constant input matrix and $\widetilde{\boldsymbol{C}}_\mathrm{i} \in \mathbb{R}^{f_\mathrm{o}\times 2f_\mathrm{i}}$ is the constant output matrix. The linearization to obtain numerical values for these matrices can be done, e.g., via finite differences.
For the LQR the cost function being minimized for infinite final time reads

$$J = \int_0^\infty \tilde{\boldsymbol{x}}_\mathrm{i}^\mathrm{T}\widetilde{\boldsymbol{Q}}_\mathrm{i}\tilde{\boldsymbol{x}}_\mathrm{i} + \boldsymbol{u}^\mathrm{T}\widetilde{\boldsymbol{R}}_\mathrm{i}\boldsymbol{u}\,\mathrm{d}t. \tag{6.3}$$

Selecting $\widetilde{\boldsymbol{Q}}_\mathrm{i} = \widetilde{\boldsymbol{C}}_\mathrm{i}^\mathrm{T}\widetilde{\boldsymbol{C}}_\mathrm{i} \in \mathbb{R}^{2f_\mathrm{i}\times 2f_\mathrm{i}}$ minimizes $\tilde{\boldsymbol{y}}_\mathrm{o}^\mathrm{T}\tilde{\boldsymbol{y}}_\mathrm{o} \geq 0$, which guarantees that $\widetilde{\boldsymbol{Q}}_\mathrm{i}$ is positive semi-definite. Since the output $\boldsymbol{y}_\mathrm{o}$ does not depend on the velocities $\dot{\boldsymbol{q}}_\mathrm{i}$, the $\widetilde{\boldsymbol{Q}}_\mathrm{i}$-matrix contains nonzero elements only in the upper left block. The matrix $\widetilde{\boldsymbol{R}}_\mathrm{i} \in \mathbb{R}^{f_\mathrm{a}\times f_\mathrm{a}}$ is chosen as diagonal with positive elements to obtain a positive definite matrix. The control gains $\boldsymbol{K}_\mathrm{lqr} \in \mathbb{R}^{f_\mathrm{a}\times 2f_\mathrm{i}}$ for the state feedback control law

$$\boldsymbol{u} = -\boldsymbol{K}_\mathrm{lqr}\tilde{\boldsymbol{x}}_\mathrm{i} \tag{6.4}$$

can then be calculated, e.g., via MATLAB's `lqr` command, which solves the arising algebraic Riccati equation. For each linearization point an individual gain matrix $\boldsymbol{K}_\mathrm{lqr}$ is computed. To account for nonlinearities nonetheless, these gain matrices are interpolated, i.e., a gain scheduling is performed for $\boldsymbol{K}_\mathrm{lqr}$. This is possible since the same fixed set of independent coordinates $\boldsymbol{q}_\mathrm{i}$ is used for all linearization points.
The proposed control structure is shown in Figure 6.2. Here, it can be seen that the standard LQR control law from Equation (6.4) is slightly adapted by

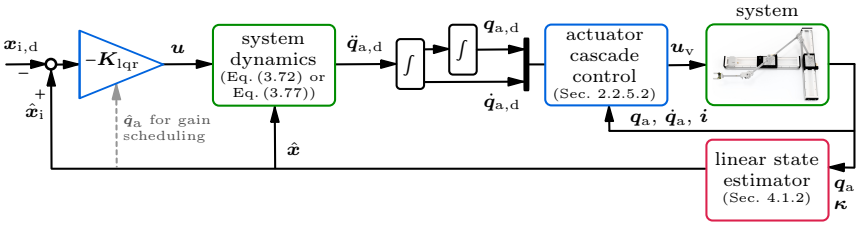


Figure 6.2: Control structure of the LQR applied to oscillation damping of flexible link parallel robots.

feeding back a trajectory tracking error. This is the error between the independent state $\hat{\boldsymbol{x}}_\mathrm{i}$, obtained by a linear state estimator, and the desired independent state $\boldsymbol{x}_\mathrm{i,d}$. This ensures that the state trajectories $\boldsymbol{x}_\mathrm{i,d}$ are closely tracked. To provide oscillation damping, the desired elastic coordinates and the corresponding first time derivatives within $\boldsymbol{x}_\mathrm{i,d}$ are permanently set to zero.
Directly applying the inputs $\boldsymbol{u}$ on current, i.e., force level is usually not meaningful because of friction effects within the actuators. Therefore, the inputs are converted to actuator motions via the projected system dynamics and the estimated state in order to realize these motions via the actuator cascade control, compare Figure 6.2. Here, Equation (3.72) can be used, which is based on the coordinate partitioning approach. However, since the incorporated projection needs to be applied in real-time it is advisable to use Equation (3.77) where the projection matrix $\boldsymbol{J}_\mathrm{c,q}$ is computed based on a QR decomposition. This prevents possible singularities within $\boldsymbol{C}_\mathrm{d}$ which can occur through a manual selection of independent coordinates. By evaluating the right side of Equation (3.77) $\ddot{\boldsymbol{q}}$ is obtained, from which the actuator accelerations $\ddot{\boldsymbol{q}}_\mathrm{a}$ are selected. These accelerations are then integrated via the simple forward Euler method. Finally, the resulting desired actuator motions $\boldsymbol{q}_\mathrm{a,d}$ and $\dot{\boldsymbol{q}}_\mathrm{a,d}$ can be accurately reproduced via the actuator cascade control.
As pointed out in Section 3.3.4, for the LQR an appropriate modeling of available gearings within the actuators is crucial. If they are neglected, the transformation from input currents or forces to accelerations can lead to very large actuator accelerations even if inputs $\boldsymbol{u}$ of zero come from the LQR. This renders a meaningful control design and tuning almost impossible especially within experiments where several other unmodeled effects and disturbances complicate the application.

6.3 Modal Damping Control

The idea of modal damping control is to actively increase the damping of the eigenmodes of a flexible multibody system which are related to structural oscillations. It has been applied to the Lambda-Kinematics, being discussed in the papers [26, 27, 29] of the author. This section extends these results by applying the modal damping control to Flexor, which has an additional flexible link leading to coupled oscillations for the considered planar case. This renders especially the experimental realization more difficult.
Starting point of the modal damping control is a reduction of the equations of motion by using kinematic inputs, see Equation (3.78). Utilizing the actuator kinematics as inputs is reasonable since it can be controlled quite accurately by the actuator cascade controller outlined in Section 2.2.5.2. With the transformation discussed in Section 3.5.4.1 ODEs are obtained in minimal form. Here, the elastic coordinates are chosen as independent, i.e., $\boldsymbol{q}_\mathrm{i} = \boldsymbol{q}_\mathrm{e}$, which is usually

possible for flexible link robots. This allows to focus only on the deformation dynamics. Then, a linearization, e.g., with finite differences, of the forward model within Figure 3.20 needs to be performed. This eventually gives the linear equations of motion

$$\widetilde{\boldsymbol{M}}_\mathrm{e}\ddot{\boldsymbol{q}}_\mathrm{e} + \widetilde{\boldsymbol{D}}_\mathrm{e}\dot{\boldsymbol{q}}_\mathrm{e} + \widetilde{\boldsymbol{K}}_\mathrm{e}\boldsymbol{q}_\mathrm{e} = \widetilde{\boldsymbol{B}}_\mathrm{e}\ddot{\boldsymbol{q}}_\mathrm{a}, \tag{6.5a}$$

$$\dot{\boldsymbol{\kappa}} = \widetilde{\boldsymbol{C}}_\mathrm{e}\dot{\boldsymbol{q}}_\mathrm{e}, \tag{6.5b}$$

at each steady-state working point. It is worth noting that for the considered system type the linearizations with respect to $\boldsymbol{q}_\mathrm{a}$ and $\dot{\boldsymbol{q}}_\mathrm{a}$ vanish in steady state. Here, the $f_\mathrm{e} \times f_\mathrm{e}$ matrices $\widetilde{\boldsymbol{M}}_\mathrm{e}$, $\widetilde{\boldsymbol{D}}_\mathrm{e}$ and $\widetilde{\boldsymbol{K}}_\mathrm{e}$ are the constant mass, damping and stiffness matrices of the linearized dynamics of the elastic coordinates. For Flexor, $\widetilde{\boldsymbol{D}}_\mathrm{e}$ and $\widetilde{\boldsymbol{K}}_\mathrm{e}$ correspond to the constant matrices coming from the linear finite element models of the single links, but $\widetilde{\boldsymbol{M}}_\mathrm{e}$ changes at the different linearization points when varying the actuator positions. Thus, $\widetilde{\boldsymbol{M}}_\mathrm{e}$ accounts for the different mass distribution which significantly changes the eigenmodes over the workspace. The new input matrix is $\widetilde{\boldsymbol{B}}_\mathrm{e}$. For the outputs it is proposed that the time derivatives of the link curvatures $\dot{\boldsymbol{\kappa}}$ are used, which are linearly related to the elastic coordinates $\dot{\boldsymbol{q}}_\mathrm{e}$ by

$$\widetilde{\boldsymbol{C}}_\mathrm{e} = \begin{bmatrix} \partial\psi_z/\partial x\big|_{x_{\mathrm{s},1}} \\ \vdots \\ \partial\psi_z/\partial x\big|_{x_{\mathrm{s},i}} \\ \vdots \end{bmatrix}, \tag{6.6}$$

see Equation (2.5). Here, $x_{\mathrm{s},i}$ denotes the i-th measurement location on the flexible links where, e.g., strain gauges are used. For Flexor, the encoder on slider B could also be used to quantify the elastic coordinate of link 2. Then, one would need to extract the elastic rotation from the overall encoder rotation.
As the elastic coordinates are coupled via the mass matrix $\widetilde{\boldsymbol{M}}_\mathrm{e}$ and the objective is to increase the modal damping, it is necessary to perform a modal transformation to decouple the eigenmodes. A modal matrix $\boldsymbol{V}$ can be calculated for an undamped version of Equation (6.5a). Then, Equation (6.5) is transformed to

$$\mathbf{I}\ddot{\boldsymbol{q}}_\mathrm{m} + \underbrace{\boldsymbol{V}^\mathrm{T}\widetilde{\boldsymbol{D}}_\mathrm{e}\boldsymbol{V}}_{\boldsymbol{D}_\mathrm{m}}\dot{\boldsymbol{q}}_\mathrm{m} + \underbrace{\boldsymbol{V}^\mathrm{T}\widetilde{\boldsymbol{K}}_\mathrm{e}\boldsymbol{V}}_{\boldsymbol{K}_\mathrm{m}}\boldsymbol{q}_\mathrm{m} = \underbrace{\boldsymbol{V}^\mathrm{T}\widetilde{\boldsymbol{B}}_\mathrm{e}}_{\boldsymbol{B}_\mathrm{m}}\ddot{\boldsymbol{q}}_\mathrm{a}, \tag{6.7a}$$

$$\dot{\boldsymbol{\kappa}} = \underbrace{\widetilde{\boldsymbol{C}}_\mathrm{e}\boldsymbol{V}}_{\boldsymbol{C}_\mathrm{m}}\dot{\boldsymbol{q}}_\mathrm{m}. \tag{6.7b}$$

The mass matrix is transformed to the identity matrix $\mathbf{I}$, since $\boldsymbol{V}$ here consists of massorthonormal eigenvectors. The vector of the modal coordinates $\boldsymbol{q}_\mathrm{m}$ follows from

$$\boldsymbol{q}_\mathrm{e} = \boldsymbol{V}\boldsymbol{q}_\mathrm{m}. \tag{6.8}$$

The diagonal matrix $\boldsymbol{K}_\mathrm{m}$ consists of the squared undamped eigenfrequencies. In general, $\boldsymbol{D}_\mathrm{m}$ is not a diagonal matrix but is often diagonally dominant such as for the measured damping of FLEXOR. Thus, according to the decoupling approximation [174] the off-diagonal elements are neglected.
For the modal damping control it is assumed that an equal or larger number of inputs is available than significant eigenmodes. This condition is true for FLEXOR to which the following control design steps are explicitly applied to simplify the explanations. Nevertheless, these steps are applicable in an analog way to other flexible link robots where the discussed conditions are also fulfilled. As still only the first two link bending eigenmodes are significant for FLEXOR one can either apply a modal reduction to the single flexible links before assembling the flexible multibody system or now by reducing the matrix $\boldsymbol{V}$ to the corresponding eigenvectors. Thus, Equation (6.7) results in the decoupled equations of motion of the first two bending eigenmodes of the robot

$$\begin{bmatrix} \ddot{q}_{\mathrm{m},1} \\ \ddot{q}_{\mathrm{m},2} \end{bmatrix} + \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix} \begin{bmatrix} \dot{q}_{\mathrm{m},1} \\ \dot{q}_{\mathrm{m},2} \end{bmatrix} + \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix} \begin{bmatrix} q_{\mathrm{m},1} \\ q_{\mathrm{m},2} \end{bmatrix} = \bar{\boldsymbol{B}}_\mathrm{m} \ddot{\boldsymbol{q}}_\mathrm{a}, \tag{6.9a}$$

$$\dot{\boldsymbol{\kappa}} = \begin{bmatrix} \dot{\kappa}_2 \\ \dot{\kappa}_3 \end{bmatrix} = \bar{\boldsymbol{C}}_\mathrm{m} \begin{bmatrix} \dot{q}_{\mathrm{m},1} \\ \dot{q}_{\mathrm{m},2} \end{bmatrix} = \bar{\boldsymbol{C}}_\mathrm{m} \dot{\boldsymbol{q}}_\mathrm{m}. \tag{6.9b}$$

Here, one strain measurement point for link 2 and one for link 3 is used resulting in a square $\bar{\boldsymbol{C}}_\mathrm{m} \in \mathbb{R}^{2\times 2}$ matrix. Also, $\bar{\boldsymbol{B}}_\mathrm{m} \in \mathbb{R}^{2\times 3}$ is non-square as three accelerations within $\ddot{\boldsymbol{q}}_\mathrm{a} \in \mathbb{R}^3$ actuate two eigenmodes. To obtain direct control of the modal damping, the control law then follows as

$$\ddot{\boldsymbol{q}}_\mathrm{a} = -\bar{\boldsymbol{B}}_\mathrm{m}^{+} \boldsymbol{P}_\mathrm{m} \bar{\boldsymbol{C}}_\mathrm{m}^{-1} \dot{\boldsymbol{\kappa}}. \tag{6.10}$$

With $^+$ denoting the Moore-Penrose pseudoinverse and

$$\boldsymbol{P}_\mathrm{m} = \begin{bmatrix} P_1 & 0 \\ 0 & P_2 \end{bmatrix} \tag{6.11}$$

being the control gains which need to be tuned. Inserting Equations (6.10) and (6.11) into Equation (6.9) gives

$$\begin{bmatrix} \ddot{q}_{\mathrm{m},1} \\ \ddot{q}_{\mathrm{m},2} \end{bmatrix} + \begin{bmatrix} d_1 + P_1 & 0 \\ 0 & d_2 + P_2 \end{bmatrix} \begin{bmatrix} \dot{q}_{\mathrm{m},1} \\ \dot{q}_{\mathrm{m},2} \end{bmatrix} + \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix} \begin{bmatrix} q_{\mathrm{m},1} \\ q_{\mathrm{m},2} \end{bmatrix} = \boldsymbol{0}. \tag{6.12}$$

With positive control gains, i.e., $P_1, P_2 > 0$ the damping is increased and both decoupled equations are asymptotically stable due to $d_1, d_2, k_1, k_2 > 0$. Since for FLEXOR three actuators damp out two eigenmodes, they are redundantly actuated rendering the use of a pseudoinversion in Equation (6.10) meaningful. As the complete robot is still underactuated, the modal damping control, which only considers the structural oscillations, will cause a drift of the actuator positions. This is typically undesired especially as the actuator limits could

be reached. It is prevented by a filter for the actuator accelerations $\ddot{\boldsymbol{q}}_\mathrm{a}$ which structurally corresponds to a mass-spring-damper system of the form

$$\dot{\boldsymbol{x}}_\mathrm{hp} = \begin{pmatrix} 0 & 1 \\ -4 & -2 \end{pmatrix} \boldsymbol{x}_\mathrm{hp} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \ddot{q}_\mathrm{a}, \tag{6.13a}$$

$$\boldsymbol{y}_\mathrm{hp} = \boldsymbol{x}_\mathrm{hp}, \tag{6.13b}$$

for each actuator. The complex eigenvalues of this filter are well below the significant bending eigenmodes to keep the influence on the oscillation damping small. Here, the input $\ddot{q}_\mathrm{a}$ is the acceleration of a single actuator and $\boldsymbol{y}_\mathrm{hp} = \boldsymbol{x}_\mathrm{hp}$ contains the position and velocity that belong to the filtered version of $\ddot{q}_\mathrm{a}$. When damping the oscillations the input $\ddot{q}_\mathrm{a}$ finally becomes zero which leads to $\boldsymbol{x}_\mathrm{hp} \to \boldsymbol{0}$, i.e., the desired zero drift in the filtered position and velocity is obtained. Therefore, the filter is just denoted as high-pass filter in the following.

The resulting control structure is shown in Figure 6.3. Here, the overall modal damping controller motion denoted by $\boldsymbol{q}_\mathrm{a,mdc}$ and $\dot{\boldsymbol{q}}_\mathrm{a,mdc}$ is sent to the actuator cascade controller to ensure an accurate realization. Within an application, the system is disturbed, e.g., by an actuator motion denoted as $\boldsymbol{q}_\mathrm{a,d}$, $\dot{\boldsymbol{q}}_\mathrm{a,d}$ or by an external contact. The oscillations due to these excitations of the system are eventually damped out by the modal damping controller. To ensure an effective performance over the whole workspace, the controller uses different actuator positions for the linearization points. The matrices $\bar{\boldsymbol{C}}_\mathrm{m}^{-1}$ and $\bar{\boldsymbol{B}}_\mathrm{m}^{+}$ are then scheduled accordingly. It should be noted that it is crucial to sort the eigenmodes and use the same sign for the eigenvectors within $\boldsymbol{V}$ at the different linearization points to provide a meaningful gain scheduling.
In Figure 6.4 the element of $\bar{\boldsymbol{B}}_\mathrm{m}^{+}$ related to the lower eigenmode and to $\ddot{a}$ is plotted at $\gamma = 0°$ over the slider positions at steady state. In the lower right, the circle describing $\alpha = 0°$, see Equation (3.47), lies over the zero line of the gains, i.e., where a direction change happens. This corresponds to the result shown in Figure 3.14 as slider A cannot actuate the system at $\alpha = 0°$. Here, the change of sign is unproblematic since two modes are redundantly damped by

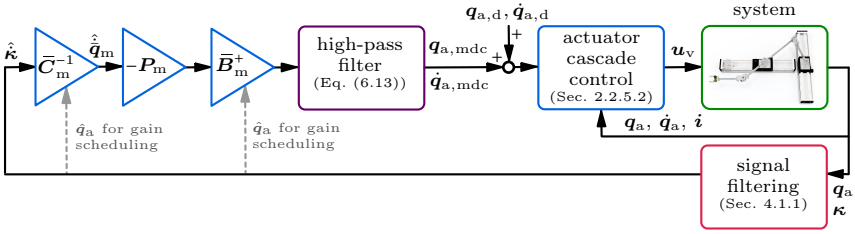


Figure 6.3: Structure of the modal damping controller for oscillation damping of flexible link parallel robots.

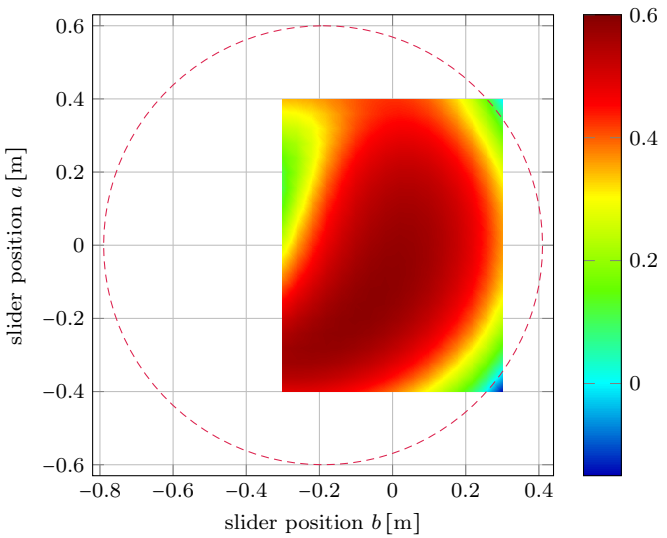


Figure 6.4: Element of $\bar{\boldsymbol{B}}^{+}_{\mathrm{m}}$ ( ) related to the lower eigenmode and $\ddot{a}$ for the modal damping control at $\gamma = 0°$ as well as the circle for $\alpha = 0°$ (- - - -) of Equation (3.47).

three actuators. Nevertheless, it again confirms the highly nonlinear actuation of FLEXOR which needs to be taken into account within the control design.

6.4 Experiments

For the experiments with FLEXOR, both oscillation damping controllers are computed in a pre-processing step at linearization points where the slider positions are varied in steps of 4 cm and steps of 20° are used for the rotary motor angle γ. The pre-processing allows to catch possible singularities through a manual coordinate partitioning before applying the controllers and it significantly reduces the computational load within experiments. Here, computationally efficient linear interpolations are used to schedule the controllers at the current actuator positions. Now, a typical placement scenario is investigated, where an unmodeled but very light box of 83 g needs to be dropped into the corresponding appliance, see Figure 6.5. Here, the first oscillation near the end-point is shown as well as the placement of the handling object with and without oscillation damping. The significant improvement with the LQR is clearly visible, which reduces the undesired oscillations quickly to ensure a safe object placement.
The corresponding curvature measurements are presented in Figure 6.6, showing the weak passive damping of the flexible links. The coupling of the link oscillations

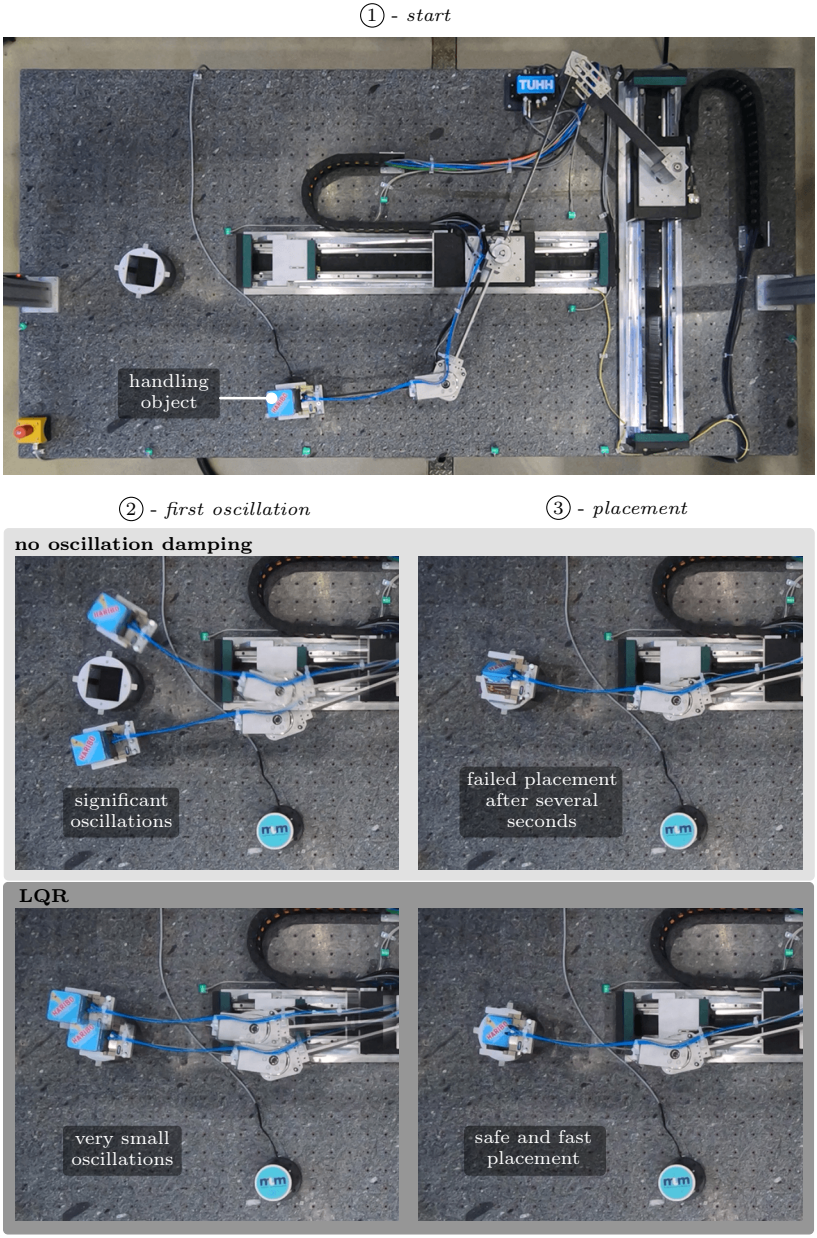


Figure 6.5: Placement scenario without and with oscillation damping via an LQR.

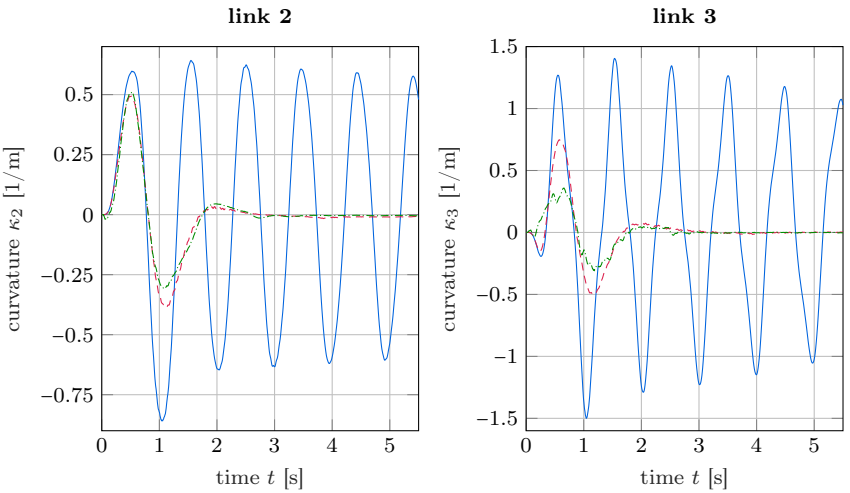


Figure 6.6: Working point change experiments without oscillation damping (——), with modal damping control (- - - -) and with an LQR (-·-·-·-).

can be seen especially in the plot of link 3 as the shape of the oscillation without damping controller differs significantly from a damped sine wave. Both proposed oscillation damping controllers introduce substantial damping and the maximum curvatures are highly reduced, which can provide a much safer operation. The detailed performance depends on the chosen gains and the filter for the modal damping control. The utilized control parameters for the LQR are

$$\widetilde{\boldsymbol{R}}_{\mathrm{i}} = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.2 \end{bmatrix} \mathrm{A}^{-2}. \tag{6.14}$$

Additionally using the exact end-effector $\boldsymbol{y}_{\mathrm{ef}}$ as output, i.e., $\boldsymbol{y}_{\mathrm{o}} = \boldsymbol{y}_{\mathrm{ef}}$, ensures that its error is kept small via $\widetilde{\boldsymbol{Q}}_{\mathrm{i}} = \widetilde{\boldsymbol{C}}_{\mathrm{i}}^{\mathrm{T}} \widetilde{\boldsymbol{C}}_{\mathrm{i}}$. This is crucial for the considered object placement scenario. It should be noted that since the LQR is still no output controller, no problems occur when using the exact end-effector for the control design. For the modal damping controller a constant gain matrix

$$\boldsymbol{P}_{\mathrm{m}} = \begin{bmatrix} 4 & 0 \\ 0 & 8 \end{bmatrix} \mathrm{s}^{-1} \tag{6.15}$$

is used. A higher gain is utilized for the higher mode to provide a somewhat similar damping behavior. As shown in Figure 6.6 such a straightforward choice can perform very effectively. Nevertheless, selecting the gains is up to the needed damping behavior. Thus, P_1 and P_2 could also be scheduled, e.g., such that the

damping matches a fixed portion of the corresponding critical damping. With these control parameters and the filter from Equation (6.13) the modal approach results in larger maximum curvatures for the considered scenario, but tends to damp the significant oscillations within a similar time as the LQR.
The corresponding actuator motions can be seen in Figure 6.7. Here, the actuator differences a_δ, b_δ and γ_δ are the differences of the case without damping control compared to the cases with modal damping control and with the LQR, respectively. The LQR performs a more significant corrective motion. While for a_δ the shape is quite similar for both controllers, the difference motion of the other actuators varies substantially in phase and amplitude. Nevertheless, both controllers damp the oscillations effectively and closely reach the end point in few seconds. This considerably shortens the needed time to safely place a handling object when compared to the case without oscillation damping.

For both controllers an estimator is used which does not rely on a dynamic robot model. They are therefore directly applicable to typical scenarios with external disturbances, e.g., where the robot is manually pushed at different points, see Figure 6.8. Here, both controllers are robust enough to quickly damp the occurring oscillations even with the utilized unmodeled bottle weight of 630 g which lowers the smallest eigenfrequency by 23%. Using, e.g., the UKF from Section 4.1.3 for such scenarios would require trusting the measurements to a large extent which significantly reduces the advantage of using a dynamic robot model within the estimation. Nevertheless, it should be noted that as no such robot model is utilized for the estimation, additional low-pass filters are used to further smoothen the signals. This reduces observation spillover as described on page 103. High-pass filters additionally ensure that the strain gauge measurements go to zero when the robot is at rest.

The LQR leads to a load of 21% on a single core of the real-time target, while the load of the modal damping control stays around 1% or less. The reason is that the inverse matrices $\bar{\boldsymbol{C}}_\mathrm{m}^{-1}$ and $\bar{\boldsymbol{B}}_\mathrm{m}^{+}$ of the modal approach are all pre-calculated and linearly interpolated within lookup tables to realize the actuator position dependent gain scheduling. The computational load of the part of the LQR to obtain the control currents $\boldsymbol{u}$ is also below 1% as here the pre-calculated values for $\boldsymbol{K}_\mathrm{lqr}$ are again only linearly interpolated within lookup tables. But transforming the control currents $\boldsymbol{u}$ to accelerations $\ddot{\boldsymbol{q}}_\mathrm{a,d}$ via the system dynamics significantly increases the load. This renders the modal approach more suitable in cases with minor computational power. Furthermore, as the modal damping control only uses the time derivative of the curvature it has no problems with offsets in the curvature, i.e., the strain gauge measurements. These typically result from static deflections of the spring steel links which are not perfectly straight. The modal approach is also easier to tune since only two gains need to be chosen, whereas for the LQR the $\boldsymbol{K}_\mathrm{lqr}$-matrix has to be recalculated at all linearization points. Also, the system dynamics needs to be implemented on the real-time target for the LQR, which increases the implementation effort. Nevertheless, the LQR controls

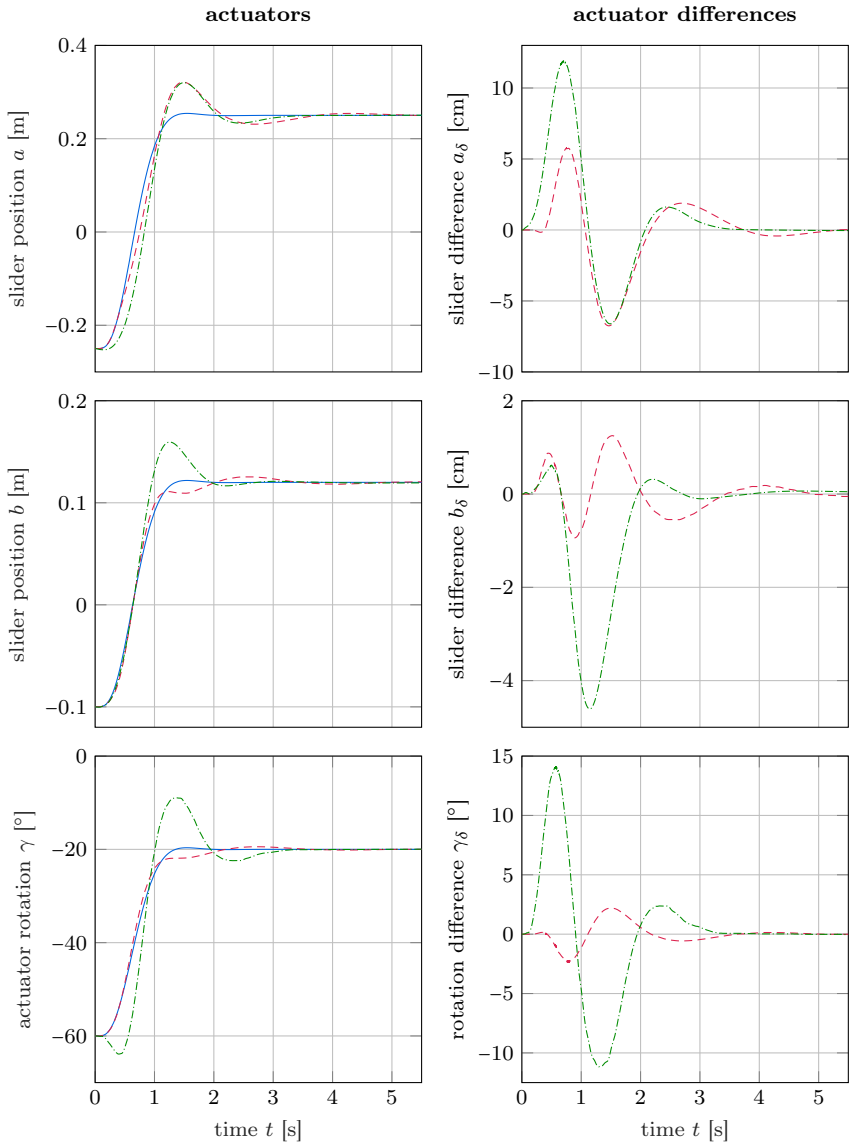


Figure 6.7: Working point change experiments without oscillation damping (——), with modal damping control (– – –) and with an LQR (–·–·–).

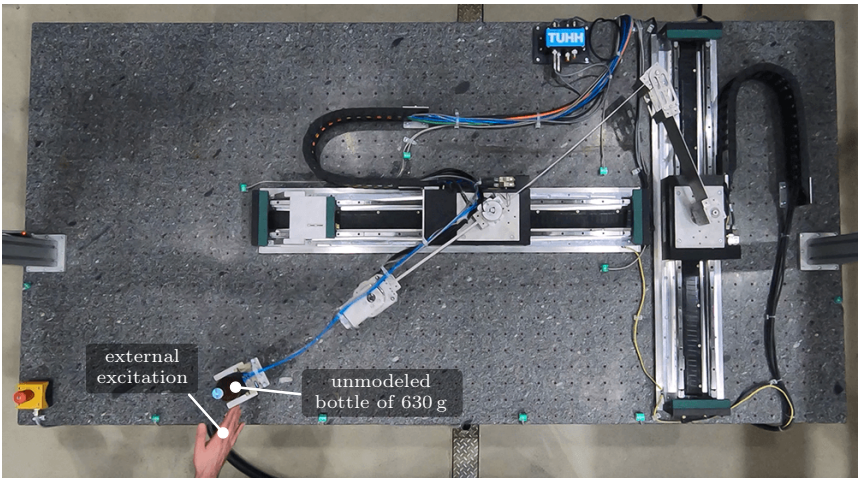


Figure 6.8: Oscillation damping scenario with external disturbance and modeling error.

the complete state and thus does not need an additional filter to ensure a correct actuator positioning.

As mentioned on page 174, both the LQR and the modal damping control can also be applied to end-effector trajectory tracking scenarios. Here, a flexible model inversion should provide the desired state trajectory $\boldsymbol{x}_{\mathrm{i,d}}$ and control inputs for the LQR as well as the trajectories of the desired curvature time derivative $\dot{\boldsymbol{\kappa}}_{\mathrm{d}}$, of $\boldsymbol{q}_{\mathrm{a,d}}$ and of $\dot{\boldsymbol{q}}_{\mathrm{a,d}}$ for the modal damping controller. Using these trajectories and not a desired deformation of zero eventually enables end-effector trajectory tracking. For both controllers this has been applied to the Lambda-Kinematics, for which results can be found in the publications [29, 36] of the author. Here, both control approaches tend to be especially useful for scenarios with large modeling errors within the flexible model inversion to limit the deformation and to quickly damp remaining oscillations.

Chapter 7

Conclusions

With modern lightweight designs and the need for inherent compliance within human-robot interaction, an increasing amount of flexibility is introduced in robots nowadays. Through a reduction of the robot mass energy consumption can be decreased and interactions are safer. Flexible link robots are a natural way to approach these topics. Nevertheless, the obtained compliance within flexible links introduces several challenges. These include amongst others underactuation, undesired oscillations and often an unstable internal dynamics when controlling the end-effector. Therefore, it is typically not possible to directly apply established methods from rigid body robotics but more sophisticated approaches are necessary. The presented research deals with a variety of the occurring difficulties for flexible link robots with serial and parallel parts, such that it is possible to eventually benefit from the advantages which they offer. In this regard, the complete process is discussed from the development of a new flexible link parallel robot, over the compact as well as accurate modeling and model inversion, over state and output estimation, to end-effector trajectory tracking control and active oscillation damping. As this research focuses on concepts which are systematic and experimentally promising, several well-known control techniques are adapted to flexible link parallel robots with a focus on computational efficiency.
Initially, to validate the developed concepts and to outline the implementation details a new parallel flexible link robot called Flexor is built. This robot and a structurally similar robot called the Lambda-Kinematics are used as application examples throughout this research to confirm the experimental relevance of the presented concepts. Hardware and software details are given to provide ideas on how the occurring challenges can be handled when designing a flexible link robot. The flexible multibody modeling approach based on the floating frame of reference formulation is applied to accurately and efficiently describe the robot dynamics. The underlying linear finite element models allow to use linear model order reduction such as the straightforward modal truncation. This leads to compact equations of motion to ensure computational efficiency. Model inversion is performed with the concept of servo constraints and transformations to ordinary differential equations via projections simplify the numerical solution process. These modeling steps even enable classical model inversion in real-time for Flexor if the internal dynamics is stable. If the internal dynamics is unstable, which is usually the case when tracking the exact end-effector of flexible link robots, classical inversion leads to unbounded results. For such cases trajectory

optimization and the concept of stable inversion are considered to obtain bounded solutions offline.
To enable state feedback controllers, state estimation is required. In this regard, it is presented how the standard unscented Kalman filter can be applied to flexible link parallel robots. For such robots output estimation based on a direct measurement technique is essential to accurately evaluate end-effector trajectory tracking controllers. Therefore, a vision-based approach by tracking markers on the robot is implemented. With a motion tracking camera for the markers and by applying algorithms from the OpenCV library the end-effector reconstruction for Flexor runs in real-time with very small delay.
The centerpiece of this research is end-effector trajectory tracking control. First, it is shown that for manual guidance scenarios often a rigid inverse model together with the elastic end-effector rotation is sufficient. However, for a free end-effector the complete consideration of the link flexibility is crucial. Connected to this a real-time capable dynamic inverse model is needed for online trajectory adaptions. This leads to the development of three different methods to obtain a stable internal dynamics. These are output redefinition by weighting the elastic deformations and rotations, adding a small counter weight at an advantageous location or using kinematic redundancy for stabilization. All three approaches perform effectively within real-time inversion experiments based on a full dynamic flexible multibody model. Since output redefinition is the most general of these approaches and straightforward to apply, it is considered within a variety of further scenarios. This includes the avoidance of collisions with the actuator limits and with dynamic obstacles, such as human hands, in real-time. It is also successfully utilized within the dynamic model inversion for a hybrid force/position controller. Moreover, using an input-output feedback linearization controller based on output redefinition and servo constraints confirms that state feedback control based on the presented unscented Kalman filter is viable for end-effector trajectory tracking.
Then, trajectory tracking results are shown for the exact end-effector, which results in a non-minimum phase system, i.e., an unstable internal dynamics. Firstly, trajectory optimization is applied to model inversion which is also used to simultaneously minimize the link deformations. Secondly, the concepts which render the internal dynamics stable or the trajectory optimization results provide an initial guess for the boundary value problem within the stable inversion approach. In experiments, the end-effector trajectory tracking performances with redefined minimum phase output and with exact end-effector output prove to be very similar. In applications, it might therefore be advantageous to use a redefined output which enables model inversion in real-time for sufficient computational power.
Finally, the end-effector trajectory is tracked within 3D experiments. These also include contact at the end-effector where a desired motion and a desired contact force trajectory need to be fulfilled. The resulting desired actuator motions originate from a model where the contact is realized by algebraic constraint

equations, i.e., no contact force based models are used.
After an actuator motion or after external disturbances residual oscillations occur within flexible link robots. This often requires active oscillation damping controllers. In this regard, a linear-quadratic regulator and a modal damping controller are applied to flexible link parallel robots. The nonlinearity of the experimentally considered parallel robot is included by gain scheduling between linearization points. The large potential of both controllers for oscillation damping is then shown within a placement scenario.

In summary, this research shows that it is indeed possible to cope with a variety of inherent difficulties within flexible link parallel robots. All control concepts are experimentally validated with two flexible multi-link parallel robots. The promising performance and the practical relevance of the discussed controllers are confirmed in realistic experimental scenarios. In the course of the experiments with flexible link robots, the necessity for modern controllers based on flexible models becomes apparent as they clearly outperform controllers based on classical rigid multibody models in most scenarios. This thesis can thus serve as a basis for future research within model-based control designs or for the extension to different scenarios such as state or output feedback control within 3D. A transfer to industrial lightweight applications is also possible.

Compliance in industrial robotics is amongst others driven by the demand for collaborative robots. In many cases such robots use flexible joints whereas flexible links are still very rare. A main reason is that the involved control challenges are significant. However, since it is possible to deal with many of these problems as outlined in this research and since the need for lightweight designs or completely compliant robots is increasing, a growing interest in robots which can exhibit structural oscillations is very likely.

References

[1] B. Siciliano and L. Villani. *Robot Force Control*. Springer, 1999.

[2] W. J. Book, O. Maizza-Neto, and D. E. Whitney. Feedback control of two beam, two joint systems with distributed flexibility. *Journal of Dynamic Systems, Measurement and Control*, 1975.

[3] C. T. Kiang, A. Spowage, and C. K. Yoong. Review of control and sensor system of flexible manipulator. *Journal of Intelligent and Robotic Systems*, 77(1):187–213, 2015.

[4] M. Sayahkarajy, Z. Mohamed, and A. A. Mohd Faudzi. Review of modelling and control of flexible-link manipulators. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 230(8):861–873, 2016.

[5] S. K. Dwivedy and P. Eberhard. Dynamic analysis of flexible manipulators, a literature review. *Mechanism and Machine Theory*, 41(7):749–777, 2006.

[6] R. H. Cannon, Jr. and E. Schmitz. Initial experiments on the end-point control of a flexible one-link robot. *The International Journal of Robotics Research*, 3(3):62–75, 1984.

[7] J. W. Lee, J. D. Huggins, and W. J. Book. Experimental verification of a large flexible manipulator. In *American Control Conference*, pages 1021–1028, 1988.

[8] G. Naganathan and A. Soni. An analytical and experimental investigation of flexible manipulator performance. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 767–773, 1987.

[9] A. De Luca and W. Book. Robots with flexible elements. In *Handbook of Robotics*, pages 287–319. Springer, 2008.

[10] A. A. Shabana. Flexible multibody dynamics: Review of past and recent developments. *Multibody System Dynamics*, 1(2):189–222, 1997.

[11] M. Benosman and G. Le Vey. Control of flexible manipulators: A survey. *Robotica*, 22(5):533–545, 2004.

[12] K. Lochan, B. Roy, and B. Subudhi. A review on two-link flexible manipulators. *Annual Reviews in Control*, 42:346–367, 2016.

[13] Q. Zhang, J. K. Mills, W. L. Cleghorn, J. Jin, and Z. Sun. Dynamic model and input shaping control of a flexible link parallel manipulator considering the exact boundary conditions. *Robotica*, 33(6):1201–1230, 2015.

[14] S. Karande, P. Nataraj, P. Gandhi, and M. M. Deshpande. Control of parallel flexible five bar manipulator using QFT. In *IEEE International Conference on Industrial Technology*, pages 1–6, 2009.

[15] R. Seifried, M. Burkhardt, and A. Held. Trajectory control of serial and parallel flexible manipulators using model inversion. In *Multibody Dynamics*, volume 28, pages 53–75. Springer, 2013.

[16] M. Moallem, R. Patel, and K. Khorasani. *Flexible-Link Robot Manipulators: Control Techniques and Structural Design*. Springer, 2000.

[17] M. Moallem, R. V. Patel, and K. Khorasani. An inverse dynamics control strategy for tip position tracking of flexible multi-link manipulators. *Journal of Robotic Systems*, 14(9):649–658, 1997.

[18] G. Bastos, Jr., R. Seifried, and O. Brüls. Analysis of stable model inversion methods for constrained underactuated mechanical systems. *Mechanism and Machine Theory*, 111:99–117, 2017.

[19] A. De Luca. Zero dynamics in robotic systems. In *Nonlinear Synthesis*, pages 68–87. Springer, 1991.

[20] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, 2005.

[21] R. Pastorino, D. Richiedei, J. Cuadrado, and A. Trevisaniolzorlock. State estimation using multibody models and non-linear Kalman filters. *International Journal of Non-Linear Mechanics*, 53:83–90, 2013.

[22] B. O. Teixeira, J. Chandrasekar, L. A. Tôrres, L. A. Aguirre, and D. S. Bernstein. State estimation for linear and non-linear equality-constrained systems. *International Journal of Control*, 82(5):918–936, 2009.

[23] C. M. Oakley and R. H. Cannon, Jr. Anatomy of an experimental two-link flexible manipulator under end-point control. In *IEEE Conference on Decision and Control*, pages 507–513, 1990.

[24] M. Bai, D. H. Zhou, and H. Schwarz. Adaptive augmented state feedback control for an experimental planar two-link flexible manipulator. *IEEE Transactions on Robotics and Automation*, 14(6):940–950, 1998.

[25] J. Carusone, K. S. Buchan, and G. M. T. D'Eleuterio. Experiments in end-effector tracking control for structurally flexible space manipulators. *IEEE Transactions on Robotics and Automation*, 9(5):553–560, 1993.

[26] M. Morlock, M. Burkhardt, and R. Seifried. Friction compensation, gain scheduling and curvature control for a flexible parallel kinematics robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2354–2359, 2015.

[27] M. Burkhardt, M. Morlock, R. Seifried, and P. Eberhard. Active damping control for an underactuated multibody system. *PAMM*, 15(1):49–50, 2015.

[28] M. Morlock, M. Burkhardt, and R. Seifried. Control concepts for a parallel manipulator with flexible links. *PAMM*, 16(1):819–820, 2016.

[29] M. Morlock, M. Burkhardt, and R. Seifried. Control of vibrations for a parallel manipulator with flexible links — Concepts and experimental results. *Journal of Physics: Conference Series*, 744(012069), 2016.

[30] M. Morlock, C. Schröck, M. Burkhardt, and R. Seifried. Nonlinear state estimation for trajectory tracking of a flexible parallel manipulator. *IFAC-PapersOnLine*, 50(1):3449–3454, 2017.

[31] M. Morlock, N. Meyer, M.-A. Pick, and R. Seifried. Modeling and trajectory tracking control of a new parallel flexible link robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 6484–6489, 2018.

[32] M. Morlock, N. Meyer, M.-A. Pick, and R. Seifried. Real-time trajectory tracking control of a parallel robot with flexible links. *Mechanism and Machine Theory*, 158:104220, 2021.

[33] M. Morlock, V. Bajrami, and R. Seifried. Trajectory tracking with collision avoidance for a parallel robot with flexible links. *Control Engineering Practice*, 111:104788, 2021.

[34] J. Timke, M. Morlock, D.-A. Dücker, and R. Seifried. Learning of a basketball free throw with a flexible link robot. In *ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, DETC2021-71660, V009T09A033, 2021.

[35] N. Meyer, C. Schwartz, M. Morlock, and R. Seifried. Systematic design of particle dampers for horizontal vibrations with application to a lightweight manipulator. *Journal of Sound and Vibration*, 510:116319, 2021.

[36] M. Morlock, M. Burkhardt, R. Seifried, and P. Eberhard. End-effector trajectory tracking of flexible link parallel robots using servo constraints. *Multibody System Dynamics*, 56:1–28, 2022.

[37] H. Cruijssen, M. Ellenbroe, M. Henderson, H. Petersen, P. Verzijden, and M. Visser. The European Robotic Arm: A high-performance mechanism finally on its way to space. In *Aerospace Mechanisms Symposium*, pages 319–334, 2014.

[38] A. Pertsch and O. Sawodny. Modeling of coupled bending and torsional oscillations of an inclined aerial ladder. In *American Control Conference*, pages 4098–4103, 2013.

[39] P. Schlott, F. Rauscher, and O. Sawodny. Modelling the structural dynamics of a tower crane. In *IEEE International Conference on Advanced Intelligent Mechatronics*, pages 763–768, 2016.

[40] L. Han, M. Chen, Q. Wu, and X. Li. Sliding mode control using disturbance observer for a flexible link robot. In *IEEE International Workshop on Variable Structure Systems*, pages 448–453, 2016.

[41] H. Gattringer. *Starr-elastische Robotersysteme: Theorie und Anwendungen*. Springer, 2011.

[42] Y.-Q. Yu, Z.-C. Du, J.-X. Yang, and Y. Li. An experimental study on the dynamics of a 3-RRR flexible parallel robot. *IEEE Transactions on Robotics*, 27(5):992–997, 2011.

[43] M. Madani and M. Moallem. Hybrid position/force control of a flexible parallel manipulator. *Journal of the Franklin Institute*, 348(6):999–1012, 2011.

[44] Z. Chu and J. Cui. Experiment on vibration control of a two-link flexible manipulator using an input shaper and adaptive positive position feedback. *Advances in Mechanical Engineering*, 7(10):1–13, 2015.

[45] S. Briot and W. Khalil. Recursive and symbolic calculation of the elasto-dynamic model of flexible parallel robots. *The International Journal of Robotics Research*, 33(3):469–483, 2014.

[46] M. Burkhardt, R. Seifried, and P. Eberhard. Experimental studies of control concepts for a parallel manipulator with flexible links. *Journal of Mechanical Science and Technology*, 29(7):2685–2691, 2015.

[47] J.-P. Merlet and C. Gosselin. Parallel mechanisms and robots. In *Handbook of Robotics*, pages 269–285. Springer, 2008.

[48] H. Wittel, D. Muhs, D. Jannasch, and J. Voßiek. *Roloff/Matek Maschinenelemente: Normung, Berechnung, Gestaltung*. Springer, 21st edition, 2013.

[49] R. Isermann. *Mechatronic Systems — Fundamentals*. Springer, 2005.

[50] R. Franke, J. Malzahn, T. Nierobisch, F. Hoffmann, and T. Bertram. Vibration control of a multi-link flexible robot arm with Fiber-Bragg-Grating sensors. In *IEEE International Conference on Robotics and Automation*, pages 3365–3370, 2009.

[51] K. Hoffmann. An introduction to stress analysis using strain gauges. Accessed on 11th April 2020 on www.hbm.com.

[52] M. Bao. *Analysis and Design Principles of MEMS Devices*. Elsevier, 2005.

[53] J. X. Zhang and K. Hoshino. *Molecular Sensors and Nanodevices*, chapter 6. Academic Press, Elsevier, 2nd edition, 2019.

[54] R. R. Craig and A. J. Kurdila. *Fundamentals of Structural Dynamics*, chapter 12. John Wiley & Sons, 2nd edition, 2006.

[55] Ansys, Inc. *Ansys Mechanical APDL 15.0 - Documentation*.

[56] G. Barbato and A. Bray. Sensors for measuring force. In *Sensors: Mechanical Sensors*, volume 7, pages 437–482. Wiley Online Library, 1993.

[57] D. Stefănescu. *Handbook of Force Transducers*. Springer, 2011.

[58] G. P. Russo. *Aerodynamic Measurements*, chapter 1. Woodhead Publishing, 2011.

[59] L. Mainetti, L. Patrono, and I. Sergi. A survey on indoor positioning systems. In *IEEE International Conference on Software, Telecommunications and Computer Networks*, pages 111–120, 2014.

[60] P. X. Liu, M. Meng, X. Ye, and J. Gu. An UDP-based protocol for internet robots. In *IEEE World Congress on Intelligent Control and Automation*, volume 1, pages 59–65, 2002.

[61] M. Spong, S. Hutchinson, and M. Vidyasagar. *Robot Modeling and Control*. John Wiley & Sons, 2005.

[62] O. Wallrapp. Standardization of flexible body modeling in multibody system codes, part I: Definition of standard input data. *Journal of Structural Mechanics*, 22(3):283–304, 1994.

[63] A. A. Shabana. *Dynamics of Multibody Systems*. Cambridge University Press, 3rd edition, 2005.

[64] A. Isidori. *Nonlinear Control Systems*. Springer, 3rd edition, 1995.

[65] J.-J. Slotine and W. Li. *Applied Nonlinear Control*, chapter 6. Prentice Hall, 1991.

[66] M. Fliess, J. Lévine, P. Martin, and P. Rouchon. On differentially flat nonlinear systems. *IFAC Proceedings Volumes*, 25(13):159–163, 1992.

[67] W. J. Terrell. *Stability and Stabilization: An Introduction*. Princeton University Press, 2009.

[68] R. Seifried. *Dynamics of Underactuated Multibody Systems*. Springer, 2014.

[69] R. Seifried. Integrated mechanical and control design of underactuated multibody systems. *Nonlinear Dynamics*, 67(2):1539–1557, 2012.

[70] D.-S. Kwon and W. J. Book. A time-domain inverse dynamic tracking control of a single-link flexible manipulator. *Journal of Dynamic Systems, Measurement, and Control*, 116(2):193–200, 1994.

[71] S. Sastry. *Nonlinear Systems: Analysis, Stability, and Control*. Springer, 1999.

[72] J. L. Junkins and Y. Kim. *Introduction to Dynamics and Control of Flexible Structures*, chapter 4.3.2. American Institute of Aeronautics and Astronautics, 1993.

[73] K.-J. Bathe. *Finite Element Procedures*. 2nd edition, 2014.

[74] R. Schwertassek and O. Wallrapp. *Dynamik flexibler Mehrkörpersysteme: Methoden der Mechanik zum rechnergestützten Entwurf und zur Analyse mechatronischer Systeme*. Springer, 1999.

[75] R. Schwertassek, O. Wallrapp, and A. A. Shabana. Flexible multibody simulation and choice of shape functions. *Nonlinear Dynamics*, 20(4): 361–380, 1999.

[76] A. K. Chopra. *Dynamics of Structures: Theory and Applications to Earthquake Engineering*. Prentice Hall, 4th edition, 2012.

[77] S. Adhikari. Damping modelling using generalized proportional damping. *Journal of Sound and Vibration*, 293(1):156–170, 2006.

[78] C. Nowakowski, J. Fehr, M. Fischer, and P. Eberhard. Model order reduction in elastic multibody systems using the floating frame of reference formulation. *IFAC Proceedings Volumes*, 45(2):40–48, 2012.

[79] A. C. Antoulas. *Approximation of Large-Scale Dynamical Systems*. SIAM, 2005.

[80] S. L. Campbell. High-index differential algebraic equations. *Journal of Structural Mechanics*, 23(2):199–222, 1995.

[81] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*. Springer, 2nd edition, 1996.

[82] S. Tschigg. Effiziente Kontaktberechnung in flexiblen Mehrkörpersystemen. In *MuM Notes in Mechanics and Dynamics*, volume 4. 2020.

[83] R. Seifried. Numerische und experimentelle Stoßanalyse für Mehrkörpersysteme. In *Schriften aus dem Institut für Technische und Numerische Mechanik der Universität Stuttgart*, volume 2. Shaker, 2005.

[84] P. Flores and H. M. Lankarani. *Contact Force Models for Multibody Dynamics*. Springer, 2016.

[85] C. Woernle. *Mehrkörpersysteme: Eine Einführung in die Kinematik und Dynamik von Systemen starrer Körper*. Springer, 2016.

[86] J. Mazurkiewicz. The basics of motion control—part 1. *Precision Motion Control, Power Transmission Design*, pages 43–45, 1995.

[87] R. Seifried, A. Held, and F. Dietmann. Analysis of feed-forward control design approaches for flexible multibody systems. *Journal of System Design and Dynamics*, 5(3):429–440, 2011.

[88] W. Blajer and K. Kołodziejczyk. A geometric approach to solving problems of control constraints: Theory and a DAE framework. *Multibody System Dynamics*, 11(4):343–64, 2004.

[89] C. Gosselin and J. Angeles. Singularity analysis of closed-loop kinematic chains. *IEEE Transactions on Robotics and Automation*, 6(3):281–290, 1990.

[90] P. Kunkel and V. Mehrmann. Index reduction for differential-algebraic equations by minimal extension. *ZAMM*, 84(9):579–597, 2004.

[91] R. Altmann, P. Betsch, and Y. Yang. Index reduction by minimal extension for the inverse dynamics simulation of cranes. *Multibody System Dynamics*, 36(3):295–321, 2016.

[92] S. S. Kim and M. J. Vanderploeg. QR decomposition for state space representation of constrained mechanical dynamic systems. *Journal of Mechanisms, Transmissions, and Automation in Design*, 108(2):183–188, 1986.

[93] R. Hirschorn. Invertibility of multivariable nonlinear control systems. *IEEE Transactions on Automatic Control*, 24(6):855–865, 1979.

[94] R. M. Hirschorn. Invertibility of nonlinear control systems. *SIAM Journal on Control and Optimization*, 17(2):289–297, 1979.

[95] M. Kelly. An introduction to trajectory optimization: How to do your own direct collocation. *SIAM Review*, 59(4):849–904, 2017.

[96] K. Springer, H. Gattringer, and P. Staufer. On time-optimal trajectory planning for a flexible link robot. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 227 (10):752–763, 2013.

[97] R. Seifried, G. Bastos, Jr., and O. Brüls. Computation of bounded feedforward control for underactuated multibody systems using nonlinear optimization. *PAMM*, 11(1):69–70, 2011.

[98] A. Lismonde, O. Brüls, and V. Sonneville. Solving the inverse dynamics of a flexible 3D robot for a trajectory tracking task. In *IEEE International Conference on Methods and Models in Automation and Robotics*, pages 194–199, 2016.

[99] J. T. Betts. *Practical Methods for Optimal Control and Estimation using Nonlinear Programming*, chapter 4. SIAM, 2nd edition, 2010.

[100] K. Graichen, V. Hagenmeyer, and M. Zeitz. A new approach to inversion-based feedforward control design for nonlinear systems. *Automatica*, 41 (12):2033–2041, 2005.

[101] D. Chen and B. Paden. Stable inversion of nonlinear non-minimum phase systems. *International Journal of Control*, 64(1):81–97, 1996.

[102] S. Devasia, D. Chen, and B. Paden. Nonlinear inversion-based output tracking. *IEEE Transactions on Automatic Control*, 41(7):930–942, 1996.

[103] H. Zhao and D. Chen. Tip trajectory tracking for multilink flexible manipulators using stable inversion. *Journal of Guidance, Control, and Dynamics*, 21(2):314–320, 1998.

[104] M. Burkhardt. Model-based feed-forward control for mechatronic systems with structural elasticity. In *Schriften aus dem Institut für Technische und Numerische Mechanik der Universität Stuttgart*, volume 59. Shaker, 2019.

[105] R. A. Singer. Estimating optimal tracking filter performance for manned maneuvering targets. *IEEE Transactions on Aerospace and Electronic Systems*, AES-6(4):473–483, 1970.

[106] J. Adamy. *Nichtlineare Regelungen*. Springer, 2009.

[107] E. A. Wan and R. van der Merwe. The unscented Kalman filter. In *Kalman Filtering and Neural Networks*, pages 221–280. John Wiley & Sons, 2001.

[108] G. Marafioti, S. Olaru, and M. Hovd. State estimation in nonlinear model predictive control, unscented Kalman filter advantages. In *Nonlinear Model Predictive Control*. Springer, 2009.

[109] E. A. Wan and R. van der Merwe. The unscented Kalman filter for nonlinear estimation. In *IEEE Adaptive Systems for Signal Processing, Communications and Control Symposium*, pages 153–158, 2000.

[110] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer, 2nd edition, 2006.

[111] M. Balas. Feedback control of flexible systems. *IEEE Transactions on Automatic Control*, 23(4):673–679, 1978.

[112] M. J. Balas. Modal control of certain flexible dynamic systems. *SIAM Journal on Control and Optimization*, 16(3):450–462, 1978.

[113] W. Liu and Z. Hou. A new approach to suppress spillover instability in structural vibration control. *Structural Control and Health Monitoring*, 11 (1):37–53, 2004.

[114] G. Bradski and A. Kaehler. *Learning OpenCV*. O'Reilly Media, Inc., 2008.

[115] OpenCV. *The OpenCV Reference Manual: Release 2.4.13.7*. `https://docs.opencv.org/2.4.13.7.zip`, 12th July 2018.

[116] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[117] R. van der Merwe, E. Wan, and S. Julier. Sigma-point Kalman filters for nonlinear estimation and sensor-fusion: Applications to integrated navigation. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 1–30, 2004.

[118] D. Oberkampf, D. F. DeMenthon, and L. S. Davis. Iterative pose estimation using coplanar feature points. *Computer Vision and Image Understanding*, 63(3):495–511, 1996.

[119] V. Feliu, K. S. Rattan, and H. B. Brown. Adaptive control of a single-link flexible manipulator. *IEEE Control Systems Magazine*, 10(2):29–33, 1990.

[120] S. K. Pradhan and B. Subudhi. Real-time adaptive control of a flexible manipulator using reinforcement learning. *IEEE Transactions on Automation Science and Engineering*, 9(2):237–249, 2012.

[121] L. Villani and J. De Schutter. Force control. In *Handbook of Robotics*, pages 161–185. Springer, 2008.

[122] A. Fayazi, N. Pariz, A. Karimpour, V. Feliu-Batlle, and S. H. HosseinNia. Adaptive sliding mode impedance control of single-link flexible manipulators interacting with the environment at an unknown intermediate point. *Robotica*, 38(9):1642–1664, 2020.

[123] G. Vossoughi and A. Karimzadeh. Impedance control of a two degree-of-freedom planar flexible link manipulator using singular perturbation theory. *Robotica*, 24(2):221–228, 2006.

[124] Z.-H. Jiang. Impedance control of flexible robot arms with parametric uncertainties. *Journal of Intelligent and Robotic Systems*, 42(2):113–133, 2005.

[125] J. Malzahn and T. Bertram. Collision detection and reaction for a multi-elastic-link robot arm. *IFAC Proceedings Volumes*, 47(3):320–325, 2014.

[126] A. De Luca and B. Siciliano. Trajectory control of a non-linear one-link flexible arm. *International Journal of Control*, 50(5):1699–1715, 1989.

[127] H. A. Talebi, K. Khorasani, and R. V. Patel. Experimental results on tracking control of a flexible-link manipulator: A new output re-definition approach. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 1090–1095, 1999.

[128] D.-S. Kwon and W. J. Book. An inverse dynamic method yielding flexible manipulator state trajectories. In *American Control Conference*, pages 186–194, 1990.

[129] Z.-H. Jiang. Workspace trajectory control of flexible robot manipulators using neural network and visual sensor feedback. In *IEEE Canadian Conference on Electrical and Computer Engineering*, pages 1502–1507, 2015.

[130] J. A. Butterworth, L. Y. Pao, and D. Y. Abramovitch. The effect of nonminimum-phase zero locations on the performance of feedforward model-inverse control techniques in discrete-time systems. In *American Control Conference*, pages 2696–2702, 2008.

[131] A. Isidori. The zero dynamics of a nonlinear system: From the origin to the latest progresses of a long successful story. *European Journal of Control*, 19(5):369–378, 2013.

[132] J.-H. Park and H. Asada. Design and analysis of flexible arms for minimum-phase endpoint control. In *American Control Conference*, pages 1220–1225, 1990.

[133] J.-H. Park and H. Asada. Design and control of minimum-phase flexible arms with torque transmission mechanisms. In *IEEE International Conference on Robotics and Automation*, pages 1790–1795, 1990.

[134] A. De Luca and L. Lanari. Achieving minimum phase behavior in a one-link flexible arm. In *International Symposium on Intelligent Robots*, pages 224–235, 1991.

[135] P. A. Chodavarapu and M. W. Spong. On noncollocated control of a single flexible link. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 1101–1106, 1996.

[136] C. J. Damaren. On the dynamics and control of flexible multibody systems with closed loops. *The International Journal of Robotics Research*, 19(3): 238–253, 2000.

[137] R. J. Caverly, J. R. Forbes, and D. Mohammadshahi. Dynamic modeling and passivity-based control of a single degree of freedom cable-actuated system. *IEEE Transactions on Control Systems Technology*, 23(3):898–909, 2015.

[138] A. Walsh and J. R. Forbes. Modeling and control of flexible telescoping manipulators. *IEEE Transactions on Robotics*, 31(4):936–947, 2015.

[139] A. Das, K. Subbarao, and F. Lewis. Dynamic inversion with zero-dynamics stabilisation for quadrotor control. *IET Control Theory Applications*, 3(3): 303–314, 2009.

[140] X. Wang and D. Chen. Output tracking control of a one-link flexible manipulator via causal inversion. *IEEE Transactions on Control Systems Technology*, 14(1):141–148, 2006.

[141] J. C. Butcher. Implicit Runge-Kutta processes. *Mathematics of Computation*, 18(85):50–64, 1964.

[142] C. A. Klein and C.-H. Huang. Review of pseudoinverse control for use with kinematically redundant manipulators. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-13(2):245–250, 1983.

[143] F. Flacco, A. De Luca, and O. Khatib. Control of redundant robots under hard joint constraints: Saturation in the null space. *IEEE Transactions on Robotics*, 31(3):637–654, 2015.

[144] P. Bosscher and D. Hedman. Real-time collision avoidance algorithm for robotic manipulators. In *IEEE International Conference on Technologies for Practical Robot Applications*, pages 113–122, 2009.

[145] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 500–505, 1985.

[146] C. W. Warren. Global path planning using artificial potential fields. In *IEEE International Conference on Robotics and Automation*, pages 316–321, 1989.

[147] O. Brock and O. Khatib. Elastic strips: Real-time path modification for mobile manipulation. In *Robotics Research*, pages 5–13. Springer, 1998.

[148] S. Haddadin, H. Urbanek, S. Parusel, D. Burschka, J. Roßmann, A. Albu-Schäffer, and G. Hirzinger. Real-time reactive motion generation based on variable attractor dynamics and shaped velocities. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3109–3116, 2010.

[149] H. Kaneko, T. Arai, K. Inoue, and Y. Mae. Real-time obstacle avoidance for robot arm using collision Jacobian. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 617–622, 1999.

[150] L. Balan and G. M. Bone. Real-time 3D collision avoidance method for safe human and robot coexistence. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 276–282, 2006.

[151] A. Ata and T. Myo. Optimal trajectory planning and obstacle avoidance for flexible manipulators using generalized pattern search. *World Journal of Modelling and Simulation*, 4(3):163–171, 2008.

[152] F. Karray, V. Modi, and J. Chan. Path planning with obstacle avoidance as applied to a class of space based flexible manipulators. *Acta Astronautica*, 37:69–86, 1995.

[153] A. Mclean and S. Cameron. Path planning and collision avoidance for redundant manipulators in 3D. In *Intelligent Autonomous Systems*, pages 381–388, 1995.

[154] F. Flacco, T. Kröger, A. De Luca, and O. Khatib. A depth space approach to human-robot collision avoidance. In *IEEE International Conference on Robotics and Automation*, pages 338–345, 2012.

[155] F. Flacco, T. Kroeger, A. De Luca, and O. Khatib. A depth space approach for evaluating distance to objects. *Journal of Intelligent and Robotic Systems*, 80(1):7–22, 2015.

[156] J. Y. Lew and W. J. Book. Hybrid control of flexible manipulators with multiple contact. In *IEEE International Conference on Robotics and Automation*, pages 242–247, 1993.

[157] F. Matsuno, T. Asano, and Y. Sakawa. Modeling and quasi-static hybrid position/force control of constrained planar two-link flexible manipulators. *IEEE Transactions on Robotics and Automation*, 10(3):287–297, 1994.

[158] F. Matsuno, O. Asano, and Y. Sakawa. Dynamic hybrid position/force control of n-link flexible manipulators. In *IEEE Conference on Decision and Control*, volume 2, pages 1803–1808, 1994.

[159] D. Li. Nonlinear control design for tip position tracking of a flexible manipulator arm. *International Journal of Control*, 60(6):1065–1082, 1994.

[160] F. Boyer and W. Khalil. An efficient calculation of flexible manipulator inverse dynamics. *The International Journal of Robotics Research*, 17(3): 282–293, 1998.

[161] M. Moallem, R. V. Patel, and K. Khorasani. Nonlinear tip-position tracking control of a flexible-link manipulator: Theory and experiments. *Automatica*, 37(11):1825–1834, 2001.

[162] C. I. Byrnes and A. Isidori. Local stabilization of minimum-phase nonlinear systems. *Systems & Control Letters*, 11(1):9–17, 1988.

[163] J. Chiou and S.-D. Wu. Constraint violation stabilization using input-output feedback linearization in multibody dynamic analysis. *Journal of Guidance, Control, and Dynamics*, 21(2):222–228, 1998.

[164] K. Tuer, M. Golnaraghi, and D. Wang. Development of a generalised active vibration suppression strategy for a cantilever beam using internal resonance. *Nonlinear Dynamics*, 5(2):131–151, 1994.

[165] K. Springer, F. J. Kilian, and H. Gattringer. Active vibration control of a flexible link robot with MPC. *IFAC Proceedings Volumes*, 45(17):163–168, 2012.

[166] L. Y. Pao. Strategies for shaping commands in the control of flexible structures. In *Japan/USA/Vietnam Workshop on Research and Education in Systems, Computation, and Control Engineering*, pages 309–318, 2000.

[167] W. O'Connor and D. Lang. Position control of flexible robot arms using mechanical waves. *Journal of Dynamic Systems, Measurement, and Control*, 120(3):334–339, 1998.

[168] W. J. O'Connor, F. Ramos de la Flor, D. J. McKeown, and V. Feliu. Wave-based control of non-linear flexible mechanical systems. *Nonlinear Dynamics*, 57(1):113–123, 2009.

[169] J. Malzahn and T. Bertram. On the equivalence of direct strain feedback and lumped parameter wave echo control for oscillation damping of elastic-link arms. *IEEE Robotics and Automation Letters*, 1(1):447–454, 2016.

[170] Z.-H. Luo. Direct strain feedback control of flexible robot arms: New theoretical and experimental results. *IEEE Transactions on Automatic Control*, 38(11):1610–1622, 1993.

[171] O. Brüls. *Integrated simulation and reduced-order modeling of controlled flexible multibody systems*. PhD thesis, Université de Liège, 2005.

[172] C. Hoffmann and H. Werner. A survey of linear parameter-varying control applications validated by experiments or high-fidelity simulations. *IEEE Transactions on Control Systems Technology*, 23(2):416–433, 2015.

[173] H. Kwakernaak and R. Sivan. *Linear Optimal Control Systems*. Wiley-Interscience, 1972.

[174] M. Morzfeld, F. Ma, and N. Ajavakom. On the decoupling approximation in damped linear systems. *Journal of Vibration and Control*, 14(12):1869–1884, 2008.

Acronyms

AC	alternating current
API	application programming interface
BVP	boundary value problem
CAD	computer-aided design
CPU	central processing unit
DAE	differential-algebraic equation
DAQ	data acquisition
DC	direct current
DLL	dynamic-link library
DOF	degree of freedom
EKF	extended Kalman filter
FBG	fiber Bragg grating
FEM	finite element method
FFOR	floating frame of reference
FFT	fast Fourier transform
FMBS	flexible multibody system
FPGA	field-programmable gate array
FPS	frames per second
GUI	graphical user interface
HDF	Hierarchical Data Format
HRI	human-robot interaction
I/O	input/output
IR	infrared
IRT	Isochronous Real-Time
IVP	initial value problem
LED	light-emitting diode

LQR	linear-quadratic regulator
MOR	model order reduction
NI	National Instruments
ODE	ordinary differential equation
PC	personal computer
PWM	pulse width modulation
RFID	radio-frequency identification
SDK	software development kit
SID	standard input data
TCP	Transmission Control Protocol
TTL	transistor-transistor logic
UDP	User Datagram Protocol
UKF	unscented Kalman filter
USB	Universal Serial Bus
WLAN	wireless local area network

With modern lightweight designs, driven by the need for material and energy efficiency, and the demand for inherent compliance within human-robot interaction an increasing amount of flexibility is introduced in robots nowadays. Through a reduction of the robot mass energy consumption is decreased and interactions are safer. Flexible link robots are a natural way to approach these topics. Nevertheless, the obtained compliance within flexible links introduces several challenges. These include amongst others underactuation, undesired structural oscillations and often an unstable internal dynamics when controlling the end-effector. Therefore, it is typically not possible to directly apply established methods from rigid body robotics but more sophisticated approaches are necessary.

The presented research deals with the challenges occurring for flexible link robots with serial and parallel parts, such that it is possible to eventually benefit from the variety of advantages which they offer. In this regard, the complete process is discussed from the development of a new modular flexible link parallel robot, over the compact as well as accurate modeling and model inversion, over state and output estimation, to end-effector trajectory tracking control and active oscillation damping.