Multibody System Dynamics (2022) 56:1–28
https://doi.org/10.1007/s11044-022-09836-x

End-effector trajectory tracking of flexible link parallel robots using servo constraints

Merlin Morlock[1] · Markus Burkhardt[2] · Robert Seifried[1] · Peter Eberhard[2]

Received: 30 June 2021 / Accepted: 17 June 2022 / Published online: 14 July 2022

Abstract
We apply the concept of servo constraints to end-effector trajectory tracking control of parallel robots with structural link flexibilities. Such servo constraints deliver the inverse robot model where solution approaches via projections are proposed, which transform the resulting differential-algebraic equations to ordinary differential equations. The applicable solution process depends on the existence and stability of the internal dynamics. When using the exact end-effector of flexible link robots as output, this internal dynamics is usually unstable. Then a two-point boundary value problem is considered in the framework of stable inversion to obtain the noncausal solution offline. This solution is used as a feedforward control, which is initially combined only with actuator feedback control. To also account for errors within the link flexibility, the well-known linear–quadratic regulator is adapted to end-effector trajectory tracking based on differential-algebraic equations. Finally, we propose a systematic input–output feedback linearization approach, which uses servo constraints for flexible link parallel robots. Here a minimum phase system is obtained by tracking a redefined end-effector output, which is an approximation of the exact end-effector position. All control concepts are validated experimentally with a parallel robot having a highly flexible link. The results allow us to compare different control approaches and show the superior performance of controllers that rely on a flexible multibody model in contrast to classical rigid multibody modeling.

Keywords Feedback linearization · Linear–quadratic regulator · Model inversion · Trajectory tracking · Algebraic constraints · Flexible link

1 Introduction

In the last years the importance of energy and material efficiency has increased substantially. In the field of robotic manipulators, these challenges can be overcome by lightweight designs. Such designs also provide more safety for human–robot interaction, which is a

✉ R. Seifried
robert.seifried@tuhh.de

1 Institute of Mechanics and Ocean Engineering, Hamburg University of Technology, Hamburg, Germany

2 Institute of Engineering and Computational Mechanics, University of Stuttgart, Stuttgart, Germany

considerably growing application area. However, these advantages come at the expense of oscillations due to structural flexibilities. If these flexibilities are significant, then they need to be considered within the control design. This is often a complicated task as such systems are in general underactuated, i.e., they have less control inputs than degrees of freedom. Consequently, many control approaches from rigid-body robotics cannot be directly applied. Additionally, such flexible robots are often nonminimum phase systems when using the exact end-effector as output, which makes the control even more challenging. It should be noted that here the description of the end-effector as *exact* differentiates it from later introduced end-effector approximations.

In this paper, we consider such structural flexibilities in the form of flexible link robots, with focus on the end-effector trajectory tracking control. This topic has been extensively studied in the last decades [2, 11, 22]. Still, a significant part of the literature reports only theoretical and numerical results or considers only single-link flexible manipulators [29] or serial robots. Especially, the literature on flexible link parallel robots is rare [16, 35]. Thus this research aims at contributing to this field by presenting a complete process from modeling over end-effector trajectory tracking control design to the experimental validation with a flexible two-link parallel robot.

Usually, parallel robots are described by differential-algebraic equations (DAEs), and serial robots by ordinary differential equations (ODEs). Thus, mathematically, serial robots can be regarded as a simpler subgroup of parallel robots. Therefore the developed techniques can be also applied to purely serial robots with flexible links.

1.1 Modeling

The occurring link flexibilities are modeled with the linear finite element method (FEM), and a modal reduction [26] is used to reduce the number of elastic degrees of freedom. Based on the floating frame of reference approach [32], the motion of a flexible body is separated into a large nonlinearly described reference frame motion and linear deformations described with respect to this reference frame. This yields a computationally efficient model for state estimation and control.

Still, geometric nonlinearities such as the foreshortening effect [21] can have a large influence on the actual end-effector kinematics. In this research, we efficiently describe geometric nonlinear effects by solely adapting the kinematics of the considered output.

1.2 End-effector trajectory tracking

The discussed modeling approaches are the basis for the main focus of the paper. This is the end-effector trajectory tracking control for flexible link parallel robots based on an inverse model using servo constraints. These servo constraints [3] are additional algebraic equations, which restrict the output to a desired trajectory. To simplify the solution for the considered model inversion-based controllers, we transform the obtained DAEs to ODEs based on projections. As mentioned before, when taking the exact end-effector of flexible link robots as output, the zero dynamics is usually unstable, i.e., the system is nonminimum phase; see, e.g., [30]. To still obtain bounded results when tracking the exact end-effector, we use the framework of stable inversion [9], where a two-point boundary value problem (BVP) needs to be solved offline. Alternatively to stable inversion, we use the concept of output redefinition to obtain a stable internal dynamics. This allows us to get bounded results by solving an initial value problem (IVP) via forward integration instead of a usually much more complicated BVP. The IVP may even allow a model inversion in real-time. Here we realize the redefinition by directly weighting the elastic deformation in the nonlinear end-effector output function based on [24].

1.2.1 Linear–quadratic regulator

Within initial experiments the feedforward control approaches based on model inversion are not combined with feedback of the flexibilities but only with feedback control of actuator-related quantities.

Still, the flexible model inversion computes the complete system state, including elastic coordinates, which will be used within a linear–quadratic regulator (LQR) applied to end-effector trajectory tracking. In [1, 8, 27], LQRs are applied experimentally to end-effector trajectory tracking of serial robots with two flexible links. The desired state trajectories come here from a rigid-body inverse kinematics. In contrast to this, the presented approach calculates the desired state trajectories with a dynamic flexible model inversion. Also, instead of serial robots, parallel robots are considered, which complicates the control design. In this regard, the DAEs are transformed to ODEs via projection, which allows us to use the standard LQR algorithm after a linearization of the minimal form.

1.2.2 Feedback linearization

Since the LQR does not directly reduce the end-effector trajectory tracking errors, the concept of feedback linearization [14] is considered.

Feedback linearization has been applied to flexible link robots before. Nevertheless, due to the typically nonminimum 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 [2], such as in [20], where end-effector trajectory tracking of a serial robot with flexible and rigid links is considered. Hereby, a collocated output, being the joint angles, is used for the input–output feedback linearization. Also, in [15], end-effector trajectory tracking of a serial robot with two flexible links is performed via computed torque, being a special form of full state feedback linearization, but based only on a rigid-body model. In [4] the theory of a computed torque method for serial flexible link robots is discussed, which only controls the rigid coordinates onto a desired trajectory. An additional stabilization of the elastic oscillations is then superimposed. Still, in [11] a point along a one-link flexible arm is tracked via input–output feedback linearization within simulations. The idea of output redefinition is also employed in [23], where an output close to the exact end-effector is used for trajectory tracking of a serial two-link robot with one flexible link. There the tracking is realized via input–output linearization 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 position. However, in contrast to the cited literature, we consider flexible link robots that also have a parallel part. Here we propose a systematic approach to realize the input–output feedback linearization via servo constraints.

1.3 Actuator cascade control

All control concepts are designed to convert the user-defined desired end-effector trajectories to actuator position, velocity and current, i.e., force or torque, trajectories. These actuator trajectories are then sent to actuator cascade controllers, which feed back the corresponding actuator measurements. This ensures that disturbances such as friction within the actuators can be effectively compensated.

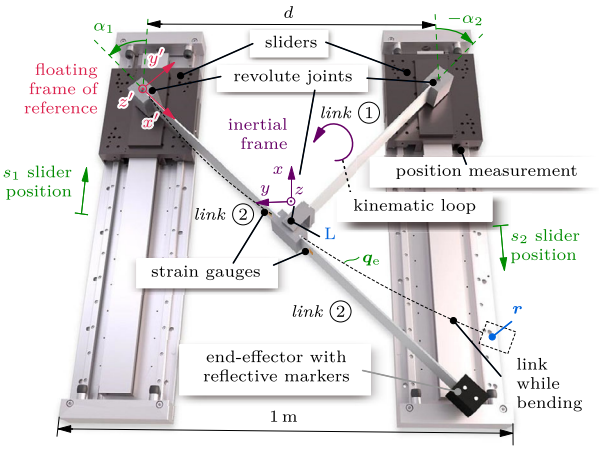

Fig. 1 Overview of the considered flexible link parallel robot

1.4 Main contribution

The main contribution of this research is the proposal of an input–output feedback linearization formulation for flexible link parallel robots, i.e., flexible multibody systems with kinematic loops. Here geometric and servo constraints are combined resulting in a compact control law. Furthermore, we present an approach that uses a dynamic flexible model inversion together with an LQR. Here we propose a design that enables the application of the standard LQR algorithm to flexible multibody systems with algebraic constraint equations. For these model-based controllers, we examine different approaches that transform the equations of motion from DAEs to ODEs. We experimentally validate all concepts within end-effector trajectory tracking scenarios with a flexible link parallel robot.

1.5 Structure

In Sect. 2, we introduce the system that is used as an application example and for validation purposes throughout the paper. In Sect. 3, we discuss the utilized modeling, inversion, and solution approaches. In Sect. 4, we experimentally apply model-based end-effector trajectory tracking controllers.

2 Considered system

To clarify and validate the discussed modeling and control techniques, throughout this paper, we use a real flexible link parallel robot as an application example. Its components and notation can be seen in Fig. 1. Here the motor currents are the control inputs $\boldsymbol{u}$ of two ironless linear motors from KML, which actuate the system. Three 2 mm thin spring steel sheets are used to create a short link 1 and a long highly elastic link 2. They are connected by revolute joints and mounted on the motor sliders, which results in a parallel robot. This parallel part is often also denoted as a kinematic loop. The utilized sensors comprise optical encoders to measure the $f_\mathrm{a} = 2$ actuator positions $\boldsymbol{s} = [s_1,\, s_2]^\mathrm{T}$, i.e., the slider positions, as well as $f_\kappa = 2$ strain gauge measurement points on the long link 2 to obtain information on the curvature $\boldsymbol{\kappa} \in \mathbb{R}^{f_\kappa}$. For validation purposes of the end-effector trajectory tracking performance,

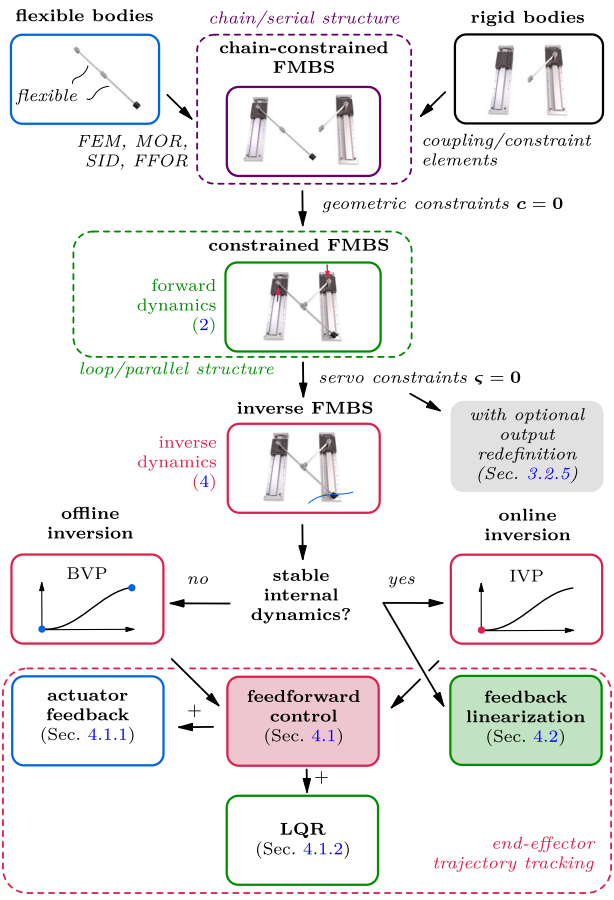

Fig. 2 Considered modeling and control approaches

a camera is mounted over the robot. Two reflective markers are used to reconstruct the xy-position of the exact end-effector $\boldsymbol{r}$. The advantage of this camera setup is that it enables a direct measurement without relying on a robot model, but the disadvantage is that the image processing needs to be done offline in a post-processing step. As a result, all later presented end-effector measurements are instead obtained with an estimator running in real-time. The sample time of the real-time target is 0.5 ms, which is also the sample time of the estimators and controllers within experiments.

3 Modeling

Since the application focus of this paper is model-based control for flexible link parallel robots, an accurate and computationally efficient model is required. The main steps used to obtain a flexible multibody model including kinematic loops are outlined in Fig. 2.

This illustration also shows how the subsequent model inversion via servo constraints is related to the later applied end-effector trajectory tracking control approaches. The details of this overview are discussed throughout this paper.

3.1 Forward model

In this research, the elements which show significant structural oscillations are modeled as flexible bodies via the linear FEM. This enables a straightforward description of arbitrary geometries, and with a large number of finite elements (FEs), a high model accuracy is ensured. Subsequently, a modal model order reduction (MOR) [26] is applied to reduce the number of elastic degrees of freedom by retaining only the significant eigenmodes. After a conversion to the standard input data (SID) format [34], the flexible parts are combined with the rigid bodies within the framework of the floating frame of reference (FFOR) [32]. Overall, this modeling process yields a computationally efficient flexible multibody system (FMBS), which is especially important for real-time estimation and control purposes.

Nevertheless, within this model occurring kinematic loops are virtually cut open, which leads to a flexible multibody system in serial, i.e., chain structure; see Fig. 2. These loops are closed again via algebraic geometric constraints $\boldsymbol{c} = \boldsymbol{0}$, which, together with the system dynamics, describe the forward model in its original loop, i.e., parallel structure.

These steps are applied to the considered exemplary system of Fig. 1. Here the long flexible link 2 consists of three rigid parts, which are connected by two spring steel sheets. Each sheet is described with 100 Timoshenko BEAM188 elements via ANSYS. The floating frame of reference for link 2 is chosen as a chord frame, which is fixed to the joint on the first slider, whereas the x'-axis always points to the revolute joint at the point L. The short link 1 is modeled as a rigid body since it exhibits only negligible oscillations. The arising flexible multibody system in chain structure, i.e., the two serial subsystems, is described with the chain coordinates $\boldsymbol{q} \in \mathbb{R}^f$. The introduced notion of chain coordinates represents a minimal set of coordinates needed to describe a system in chain structure. They consist of the joint angles α_1 and α_2, the slider positions s_1 and s_2, and the elastic coordinates $\boldsymbol{q}_\mathrm{e} \in \mathbb{R}^{f_\mathrm{e}}$ of link 2. For the considered system, it turns out that only the first bending eigenmode is significant in normal operation. Therefore the vector of the elastic coordinates reduces to a scalar $\boldsymbol{q}_\mathrm{e} = q_{\mathrm{e},1}$, i.e., $f_\mathrm{e} = 1$, which is used to describe the elastic deformation within all application scenarios. This results in $f = 5$ chain coordinates

$$\boldsymbol{q} = \begin{bmatrix} q_{\mathrm{e},1} & s_1 & s_2 & \alpha_1 & \alpha_2 \end{bmatrix}^\mathrm{T}. \tag{1}$$

The constrained flexible multibody system follows by introducing loop-closing constraints, which are formulated as n_c geometric constraint equations $\boldsymbol{c}(\boldsymbol{q}) = \boldsymbol{0}$. Similar to [31], the general formulation of the forward dynamics is then given by

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} = \boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{B}(\boldsymbol{q})\boldsymbol{u} + \boldsymbol{C}^\mathrm{T}(\boldsymbol{q})\boldsymbol{\lambda}, \tag{2a}$$

$$\boldsymbol{c}(\boldsymbol{q}) = \boldsymbol{0}, \tag{2b}$$

which is a set of DAEs with differentiation index 3. Here $\boldsymbol{M} \in \mathbb{R}^{f \times f}$ 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 $\boldsymbol{u} \in \mathbb{R}^{f_\mathrm{a}}$. The vector $\boldsymbol{f} \in \mathbb{R}^f$ contains the contributions of the Coriolis, centrifugal, gyroscopic, and elastic inner forces, as well as of the applied forces that are not related to the control inputs $\boldsymbol{u}$. Moreover, $\boldsymbol{C} = \partial \boldsymbol{c} / \partial \boldsymbol{q} \in \mathbb{R}^{n_\mathrm{c} \times f}$ is the Jacobian matrix of the geometric constraints, and $\boldsymbol{\lambda} \in \mathbb{R}^{n_\mathrm{c}}$ are the corresponding Lagrange multipliers, which represent the reactions enforcing these constraints. For the constraints, the position of the point L, where the cut loop shall be closed, is described with respect to the two serial subsystems via the 2D position vectors $\boldsymbol{r}_{\mathrm{L},1}(\boldsymbol{q})$ and $\boldsymbol{r}_{\mathrm{L},2}(\boldsymbol{q})$. Thus here the

$n_c = 2$ geometric constraints can be written as

$$\boldsymbol{c}(\boldsymbol{q}) = \boldsymbol{r}_{\mathrm{L},1}(\boldsymbol{q}) - \boldsymbol{r}_{\mathrm{L},2}(\boldsymbol{q}) = \boldsymbol{0}. \tag{3}$$

For further details on the modeling of flexible multibody systems, we refer to the books [30, 32].

3.2 Model inversion and projections

For the later discussed end-effector trajectory tracking controllers, we need an inverse model. As indicated in Fig. 2, the inverse dynamics is simply obtained by adding n_ς algebraic servo constraints $\boldsymbol{\varsigma} = \boldsymbol{0}$ to Eqs. (2a), (2b). This gives

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} = \boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{B}(\boldsymbol{q})\boldsymbol{u} + \boldsymbol{C}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{\lambda}, \tag{4a}$$

$$\boldsymbol{c}(\boldsymbol{q}) = \boldsymbol{0}, \tag{4b}$$

$$\boldsymbol{\varsigma}(t, \boldsymbol{q}) = \boldsymbol{r}_{\mathrm{o}}(\boldsymbol{q}) - \boldsymbol{r}_{\mathrm{d}}(t) = \boldsymbol{0}. \tag{4c}$$

Here the output $\boldsymbol{r}_{\mathrm{o}} \in \mathbb{R}^{f_{\mathrm{o}}}$ is restricted to a desired trajectory $\boldsymbol{r}_{\mathrm{d}} \in \mathbb{R}^{f_{\mathrm{o}}}$ through the servo constraints. For the system of Fig. 1, the 2D end-effector position or an approximation of it is the output of interest, i.e., $f_{\mathrm{o}} = 2$, and accordingly $n_\varsigma = 2$ servo constraints are used. Servo constraints [3] are similar to geometric constraints but are not enforced by the Lagrange multipliers $\boldsymbol{\lambda}$ but by the control inputs $\boldsymbol{u}$. For the considered system type, where the geometric constraints lead to DAEs with differentiation index 3, the inversion via servo constraints does not change the problem structure. The solution of Eqs. (4a)–(4c) is a major focus of this research. It further represents the considered problem class, input-affine flexible multibody systems with holonomic geometric and servo constraints with a vector relative degree of $\boldsymbol{r}_{\mathrm{deg}} = \{2, \ldots, 2\}$. For details on the relative degree, we refer to [14, 28], and for further information on the relation between the relative degree and the differentiation index, we refer to [7].

To simplify the solution of the inverse model of Eqs. (4a)–(4c), the idea is to transform the set of DAEs to ODEs, i.e., to reduce the differentiation index from 3 to 0. This will be realized with a projection that cancels the unknown Lagrange multipliers $\boldsymbol{\lambda}$ and the inputs $\boldsymbol{u}$ from Eq. (4a). This method is also called the null-space method.

Initially, the constraint equations (4b) and (4c) need to be differentiated twice with respect to time:

$$\dot{\boldsymbol{c}} = \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{c}}{\partial t} = \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{c}' = \boldsymbol{0}, \tag{5a}$$

$$\ddot{\boldsymbol{c}} = \boldsymbol{C}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{C}}\dot{\boldsymbol{q}} + \dot{\boldsymbol{c}}' = \boldsymbol{C}\ddot{\boldsymbol{q}} + \boldsymbol{c}'' = \boldsymbol{0}, \tag{5b}$$

$$\dot{\boldsymbol{\varsigma}} = \frac{\partial \boldsymbol{r}_{\mathrm{o}}}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{r}_{\mathrm{o}}}{\partial t} - \dot{\boldsymbol{r}}_{\mathrm{d}} = \boldsymbol{H}\dot{\boldsymbol{q}} + \boldsymbol{h}' - \dot{\boldsymbol{r}}_{\mathrm{d}} = \boldsymbol{0}, \tag{5c}$$

$$\ddot{\boldsymbol{\varsigma}} = \boldsymbol{H}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{H}}\dot{\boldsymbol{q}} + \dot{\boldsymbol{h}}' - \ddot{\boldsymbol{r}}_{\mathrm{d}} = \boldsymbol{H}\ddot{\boldsymbol{q}} + \boldsymbol{h}'' - \ddot{\boldsymbol{r}}_{\mathrm{d}} = \boldsymbol{0}. \tag{5d}$$

Even though the terms $\boldsymbol{c}'$ and $\boldsymbol{h}'$ vanish for the considered constraints, they are kept for generality. It is worth mentioning that $'$ and $''$ are not mathematical operators but are used for a compact notation throughout this paper. The Jacobian matrices of the constraints and the input matrix can be summarized as

$$\boldsymbol{\Gamma}^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{C}^{\mathrm{T}} & \boldsymbol{H}^{\mathrm{T}} \end{bmatrix}, \tag{6a}$$

$$\boldsymbol{G}^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{C}^{\mathrm{T}} & \boldsymbol{B} \end{bmatrix}. \tag{6b}$$

Here both matrices $\boldsymbol{\Gamma}^{\mathrm{T}} \in \mathbb{R}^{f \times (n_{\mathrm{c}}+n_{\varsigma})}$ and $\boldsymbol{G}^{\mathrm{T}} \in \mathbb{R}^{f \times (n_{\mathrm{c}}+n_{\varsigma})}$ have the same size, which is important for the following computations. Thus the number of servo constraints needs to match the number of utilized inputs. If more inputs are available, then a submatrix of the input matrix $\boldsymbol{B}$ should be used within $\boldsymbol{G}^{\mathrm{T}}$.

Next, we discuss systematic approaches to arrive at the needed projection and the ODEs.

3.2.1 Manual selection via coordinate partitioning

Firstly, we extend the coordinate partitioning approach [30] to the case with servo constraints. We manually split up the chain coordinates $\boldsymbol{q} \in \mathbb{R}^{f}$ into independent coordinates $\boldsymbol{q}_{\mathrm{i}} \in \mathbb{R}^{f_{\mathrm{i}}}$ with $f_{\mathrm{i}} = f - n_{\mathrm{c}} - n_{\varsigma}$ and into dependent coordinates $\boldsymbol{q}_{\mathrm{d}} \in \mathbb{R}^{f_{\mathrm{d}}}$ with $f_{\mathrm{d}} = n_{\mathrm{c}} + n_{\varsigma}$. Thus we can write the chain coordinates as $\boldsymbol{q} = [\boldsymbol{q}_{\mathrm{i}}^{\mathrm{T}}, \boldsymbol{q}_{\mathrm{d}}^{\mathrm{T}}]^{\mathrm{T}}$. Note 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 flexibility, the elastic coordinates are a possible choice of the independent coordinates.

Based on the partitioning of the coordinates $\boldsymbol{q}$ and Eq. (6a), Eqs. (5a) and (5c) can be rewritten as

$$\begin{bmatrix} \dot{\boldsymbol{c}} \\ \dot{\boldsymbol{\varsigma}} \end{bmatrix} = \boldsymbol{\Gamma}\dot{\boldsymbol{q}} + \boldsymbol{\gamma}' = \boldsymbol{\Gamma}_{\mathrm{i}}\dot{\boldsymbol{q}}_{\mathrm{i}} + \boldsymbol{\Gamma}_{\mathrm{d}}\dot{\boldsymbol{q}}_{\mathrm{d}} + \boldsymbol{\gamma}' = \boldsymbol{0}, \tag{7}$$

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

$$\boldsymbol{\Gamma} = \frac{\partial \boldsymbol{\gamma}}{\partial \boldsymbol{q}}, \quad \boldsymbol{\Gamma}_{\mathrm{i}} = \frac{\partial \boldsymbol{\gamma}}{\partial \boldsymbol{q}_{\mathrm{i}}}, \quad \boldsymbol{\Gamma}_{\mathrm{d}} = \frac{\partial \boldsymbol{\gamma}}{\partial \boldsymbol{q}_{\mathrm{d}}} \tag{8}$$

are Jacobian matrices with $\boldsymbol{\gamma} = [\boldsymbol{c}^{\mathrm{T}},\ \boldsymbol{\varsigma}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{n_{\mathrm{c}}+n_{\varsigma}}$. Solving Eq. (7) for $\dot{\boldsymbol{q}}_{\mathrm{d}}$ gives

$$\dot{\boldsymbol{q}} = \begin{bmatrix} \dot{\boldsymbol{q}}_{\mathrm{i}} \\ \dot{\boldsymbol{q}}_{\mathrm{d}} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{I}_{f_{\mathrm{i}}} \\ -\boldsymbol{\Gamma}_{\mathrm{d}}^{-1}\boldsymbol{\Gamma}_{\mathrm{i}} \end{bmatrix}}_{\boldsymbol{J}_{\mathrm{r}}} \dot{\boldsymbol{q}}_{\mathrm{i}} + \underbrace{\begin{bmatrix} \boldsymbol{0} \\ -\boldsymbol{\Gamma}_{\mathrm{d}}^{-1}\boldsymbol{\gamma}' \end{bmatrix}}_{\boldsymbol{\theta}'} \tag{9}$$

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, by Eqs. (5b) and (5d) on acceleration level it follows that

$$\ddot{\boldsymbol{q}} = \begin{bmatrix} \ddot{\boldsymbol{q}}_{\mathrm{i}} \\ \ddot{\boldsymbol{q}}_{\mathrm{d}} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{I}_{f_{\mathrm{i}}} \\ -\boldsymbol{\Gamma}_{\mathrm{d}}^{-1}\boldsymbol{\Gamma}_{\mathrm{i}} \end{bmatrix}}_{\boldsymbol{J}_{\mathrm{r}}} \ddot{\boldsymbol{q}}_{\mathrm{i}} + \underbrace{\begin{bmatrix} \boldsymbol{0} \\ -\boldsymbol{\Gamma}_{\mathrm{d}}^{-1}\boldsymbol{\gamma}'' \end{bmatrix}}_{\boldsymbol{\theta}''} \tag{10}$$

with $\boldsymbol{\gamma}'' = [(\boldsymbol{c}'')^{\mathrm{T}},\ (\boldsymbol{h}'' - \ddot{\boldsymbol{r}}_{\mathrm{d}})^{\mathrm{T}}]^{\mathrm{T}}$. To transform the DAEs (4a)–(4c) 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 Eq. (4a). A possible choice is

$$\boldsymbol{G}\boldsymbol{J}_{\ell} = \begin{bmatrix} \boldsymbol{G}_{\mathrm{i}} & \boldsymbol{G}_{\mathrm{d}} \end{bmatrix} \underbrace{\begin{bmatrix} \mathbf{I}_{f_{\mathrm{i}}} \\ -\boldsymbol{G}_{\mathrm{d}}^{-1}\boldsymbol{G}_{\mathrm{i}} \end{bmatrix}}_{\boldsymbol{J}_{\ell}} = \boldsymbol{0}. \tag{11}$$

Here $\boldsymbol{G}_\mathrm{i} \in \mathbb{R}^{(n_\mathrm{c}+n_\varsigma)\times f_\mathrm{i}}$ and $\boldsymbol{G}_\mathrm{d} \in \mathbb{R}^{(n_\mathrm{c}+n_\varsigma)\times f_\mathrm{d}}$ can be obtained by a partitioning of $\boldsymbol{G}$ analogously to $\boldsymbol{\Gamma}$ in Eq. (7). The transposed projection matrix $\boldsymbol{J}_\ell^\mathrm{T}$ is now multiplied from the left to the system dynamics (4a), where also the accelerations $\ddot{\boldsymbol{q}}$ are replaced by Eq. (10), yielding

$$\boldsymbol{J}_\ell^\mathrm{T}\boldsymbol{M}\left(\boldsymbol{J}_\mathrm{r}\ddot{\boldsymbol{q}}_\mathrm{i}+\boldsymbol{\theta}''\right)=\boldsymbol{J}_\ell^\mathrm{T}\boldsymbol{f}+\underbrace{\boldsymbol{J}_\ell^\mathrm{T}\boldsymbol{G}^\mathrm{T}}_{\boldsymbol{0}}\begin{bmatrix}\boldsymbol{\lambda}\\ \boldsymbol{u}\end{bmatrix}. \tag{12}$$

Then the equation of motion (12), which is an ODE for the independent accelerations, is rearranged to

$$\ddot{\boldsymbol{q}}_\mathrm{i}=\left(\boldsymbol{J}_\ell^\mathrm{T}\boldsymbol{M}\boldsymbol{J}_\mathrm{r}\right)^{-1}\boldsymbol{J}_\ell^\mathrm{T}(\boldsymbol{f}-\boldsymbol{M}\boldsymbol{\theta}''). \tag{13}$$

Now, solving Eq. (9) for $\dot{\boldsymbol{q}}_\mathrm{i}$ via a left multiplication with the transposed nonsquare matrix $\boldsymbol{J}_\mathrm{r}$ yields the independent velocities

$$\dot{\boldsymbol{q}}_\mathrm{i}=\left(\boldsymbol{J}_\mathrm{r}^\mathrm{T}\boldsymbol{J}_\mathrm{r}\right)^{-1}\boldsymbol{J}_\mathrm{r}^\mathrm{T}(\dot{\boldsymbol{q}}-\boldsymbol{\theta}'). \tag{14}$$

Plugging Eqs. (13) and (14) back into Eqs. (10) and (9) gives the equations of motion in chain coordinates

$$\dot{\boldsymbol{q}}=\boldsymbol{J}_\mathrm{r}\left(\boldsymbol{J}_\mathrm{r}^\mathrm{T}\boldsymbol{J}_\mathrm{r}\right)^{-1}\boldsymbol{J}_\mathrm{r}^\mathrm{T}\left(\dot{\boldsymbol{q}}-\boldsymbol{\theta}'\right)+\boldsymbol{\theta}', \tag{15a}$$

$$\ddot{\boldsymbol{q}}=\boldsymbol{J}_\mathrm{r}\left(\boldsymbol{J}_\ell^\mathrm{T}\boldsymbol{M}\boldsymbol{J}_\mathrm{r}\right)^{-1}\boldsymbol{J}_\ell^\mathrm{T}\left(\boldsymbol{f}-\boldsymbol{M}\boldsymbol{\theta}''\right)+\boldsymbol{\theta}''. \tag{15b}$$

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 that 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 Eq. (15a). 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 the velocity level. Consequently, the drift through integration is reduced to being only linear within the constraint equations (4b) and (4c) on the position level.

For systems where the underactuation is due to the flexibilities, neglecting these flexibilities yields an equivalent fully actuated rigid system. Then the projected parts of Eqs. (15a), (15b) vanish as no independent coordinates exist, which gives $\dot{\boldsymbol{q}}=\boldsymbol{\theta}'$ and $\ddot{\boldsymbol{q}}=\boldsymbol{\theta}''$. Thus, the inverse is purely algebraic, and instead of time integration, it can also be solved by finding the roots of Eqs. (4b) and (4c), as well as with Eq. (7), to obtain the state trajectories. This corresponds to inverse kinematics.

3.2.2 Automatic selection via QR decomposition

A manual selection of the independent and dependent coordinates as in the preceding coordinate partitioning approach can cause 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 [17, 31], this approach relies on the QR decompositions

$$\boldsymbol{\Gamma}^\mathrm{T}=\begin{bmatrix}\boldsymbol{C}^\mathrm{T} & \boldsymbol{H}^\mathrm{T}\end{bmatrix}=\underbrace{\begin{bmatrix}\boldsymbol{Q}_\mathrm{r} & \boldsymbol{J}_\mathrm{r,q}\end{bmatrix}}_{\boldsymbol{Q}_\Gamma}\begin{bmatrix}\boldsymbol{R}_\mathrm{r}\\ \boldsymbol{0}\end{bmatrix}=\boldsymbol{Q}_\mathrm{r}\boldsymbol{R}_\mathrm{r}, \tag{16a}$$

$$\boldsymbol{G}^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{C}^{\mathrm{T}} & \boldsymbol{B} \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{Q}_\ell & \boldsymbol{J}_{\ell,\mathrm{q}} \end{bmatrix}}_{\boldsymbol{Q}_\mathrm{G}} \begin{bmatrix} \boldsymbol{R}_\ell \\ \boldsymbol{0} \end{bmatrix} = \boldsymbol{Q}_\ell \boldsymbol{R}_\ell, \tag{16b}$$

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 orthogonal matrices $\boldsymbol{Q}_\Gamma \in \mathbb{R}^{f\times f}$ and $\boldsymbol{Q}_\mathrm{G} \in \mathbb{R}^{f\times f}$, it follows that

$$\boldsymbol{\Gamma}\boldsymbol{J}_{\mathrm{r,q}} = \boldsymbol{R}_\mathrm{r}^{\mathrm{T}} \underbrace{\boldsymbol{Q}_\mathrm{r}^{\mathrm{T}}\boldsymbol{J}_{\mathrm{r,q}}}_{\boldsymbol{0}} = \boldsymbol{0}, \tag{17a}$$

$$\boldsymbol{G}\boldsymbol{J}_{\ell,\mathrm{q}} = \boldsymbol{R}_\ell^{\mathrm{T}} \underbrace{\boldsymbol{Q}_\ell^{\mathrm{T}}\boldsymbol{J}_{\ell,\mathrm{q}}}_{\boldsymbol{0}} = \boldsymbol{0}. \tag{17b}$$

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

$$\boldsymbol{q} = \boldsymbol{Q}_\Gamma \begin{bmatrix} \boldsymbol{q}_{\mathrm{d,q}} \\ \boldsymbol{q}_{\mathrm{i,q}} \end{bmatrix} = \boldsymbol{J}_{\mathrm{r,q}}\boldsymbol{q}_{\mathrm{i,q}} + \boldsymbol{Q}_\mathrm{r}\boldsymbol{q}_{\mathrm{d,q}} \tag{18}$$

with $\boldsymbol{q}_{\mathrm{d,q}} \in \mathbb{R}^{f_\mathrm{d}}$ and $\boldsymbol{q}_{\mathrm{i,q}} \in \mathbb{R}^{f_\mathrm{i}}$ of 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 matrix $\boldsymbol{\Gamma}$. 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

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_{\mathrm{r,q}}\dot{\boldsymbol{q}}_{\mathrm{i,q}} + \boldsymbol{Q}_\mathrm{r}\boldsymbol{q}'_{\mathrm{d,q}} = \boldsymbol{J}_{\mathrm{r,q}}\dot{\boldsymbol{q}}_{\mathrm{i,q}} + \boldsymbol{\theta}'_\mathrm{q}, \tag{19a}$$

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}_{\mathrm{r,q}}\ddot{\boldsymbol{q}}_{\mathrm{i,q}} + \boldsymbol{Q}_\mathrm{r}\boldsymbol{q}''_{\mathrm{d,q}} = \boldsymbol{J}_{\mathrm{r,q}}\ddot{\boldsymbol{q}}_{\mathrm{i,q}} + \boldsymbol{\theta}''_\mathrm{q} \tag{19b}$$

with $\boldsymbol{q}'_{\mathrm{d,q}}$ and $\boldsymbol{q}''_{\mathrm{d,q}}$ to be determined. Inserting Eqs. (16a) and (19a) into the constraint equation (7) on the velocity level gives

$$\begin{bmatrix} \dot{\boldsymbol{c}} \\ \dot{\boldsymbol{\varsigma}} \end{bmatrix} = \boldsymbol{\Gamma}\dot{\boldsymbol{q}} + \boldsymbol{\gamma}' = \boldsymbol{R}_\mathrm{r}^{\mathrm{T}} \underbrace{\boldsymbol{Q}_\mathrm{r}^{\mathrm{T}}\boldsymbol{J}_{\mathrm{r,q}}}_{\boldsymbol{0}}\dot{\boldsymbol{q}}_{\mathrm{i,q}} + \boldsymbol{R}_\mathrm{r}^{\mathrm{T}} \underbrace{\boldsymbol{Q}_\mathrm{r}^{\mathrm{T}}\boldsymbol{Q}_\mathrm{r}}_{\boldsymbol{\mathrm{I}}_{f_\mathrm{d}}}\boldsymbol{q}'_{\mathrm{d,q}} + \boldsymbol{\gamma}' = \boldsymbol{0}. \tag{20}$$

This yields $\boldsymbol{q}'_{\mathrm{d,q}} = -\boldsymbol{R}_\mathrm{r}^{-\mathrm{T}}\boldsymbol{\gamma}'$, where $\boldsymbol{R}_\mathrm{r} \in \mathbb{R}^{(n_\mathrm{c}+n_\varsigma)\times(n_\mathrm{c}+n_\varsigma)}$ is a square matrix. Analogously, on acceleration level, it follows that $\boldsymbol{q}''_{\mathrm{d,q}} = -\boldsymbol{R}_\mathrm{r}^{-\mathrm{T}}\boldsymbol{\gamma}''$. Based on Eq. (17b), 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 Eq. (12). With analogous steps to arrive at Eqs. (15a), (15b), we have the following equations of motion in chain coordinates:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_{\mathrm{r,q}}\boldsymbol{J}_{\mathrm{r,q}}^{\mathrm{T}}\left(\dot{\boldsymbol{q}} - \boldsymbol{\theta}'_\mathrm{q}\right) + \boldsymbol{\theta}'_\mathrm{q}, \tag{21a}$$

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}_{\mathrm{r,q}}\left(\boldsymbol{J}_{\ell,\mathrm{q}}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{J}_{\mathrm{r,q}}\right)^{-1}\boldsymbol{J}_{\ell,\mathrm{q}}^{\mathrm{T}}\left(\boldsymbol{f} - \boldsymbol{M}\boldsymbol{\theta}''_\mathrm{q}\right) + \boldsymbol{\theta}''_\mathrm{q}. \tag{21b}$$

These are again ODEs in state-space representation with the same structure 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 Eq. (21a).

3.2.3 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 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 Sects. 3.2.1 and 3.2.2 is possible to benefit from the advantages of both methods. Here we denote this mix as a mixed coordinate partitioning. Based on Eq. (13), the equation of motion in minimal form then follows as

$$\ddot{\boldsymbol{q}}_\mathrm{i} = \left(\boldsymbol{J}_{\ell,\mathrm{q}}^\mathrm{T}\boldsymbol{M}\boldsymbol{J}_\mathrm{r}\right)^{-1}\boldsymbol{J}_{\ell,\mathrm{q}}^\mathrm{T}(\boldsymbol{f}-\boldsymbol{M}\boldsymbol{\theta}''). \tag{22}$$

3.2.4 Lagrange multipliers and inputs

Next, we can compute the previously canceled Lagrange multipliers $\boldsymbol{\lambda}$ and the control inputs $\boldsymbol{u}$ independently of the coordinate partitioning or the QR decomposition. For underactuated systems, the matrix $\boldsymbol{G}^\mathrm{T}$ cannot be inverted, and Eq. (4a) cannot be directly solved for $\boldsymbol{\lambda}$ and $\boldsymbol{u}$. Nevertheless, they can be obtained in a form independent of $\ddot{\boldsymbol{q}}$. These accelerations are eliminated by combining the system dynamics (4a) with the second time derivatives of the constraint equations (5b) and (5d). This gives the intermediate result

$$\boldsymbol{C}\boldsymbol{M}^{-1}\left(\boldsymbol{f}+\boldsymbol{B}\boldsymbol{u}+\boldsymbol{C}^\mathrm{T}\boldsymbol{\lambda}\right)+\boldsymbol{c}''=\boldsymbol{0}, \tag{23a}$$

$$\boldsymbol{H}\boldsymbol{M}^{-1}\left(\boldsymbol{f}+\boldsymbol{B}\boldsymbol{u}+\boldsymbol{C}^\mathrm{T}\boldsymbol{\lambda}\right)+\boldsymbol{h}''-\ddot{\boldsymbol{r}}_\mathrm{d}=\boldsymbol{0}. \tag{23b}$$

Analogously to [5], solving for $\boldsymbol{\lambda}$ and $\boldsymbol{u}$ leads to

$$\begin{bmatrix}\boldsymbol{\lambda}\\ \boldsymbol{u}\end{bmatrix} = \underbrace{\begin{bmatrix}\boldsymbol{C}\boldsymbol{M}^{-1}\boldsymbol{C}^\mathrm{T} & \boldsymbol{C}\boldsymbol{M}^{-1}\boldsymbol{B}\\ \boldsymbol{H}\boldsymbol{M}^{-1}\boldsymbol{C}^\mathrm{T} & \boldsymbol{H}\boldsymbol{M}^{-1}\boldsymbol{B}\end{bmatrix}}_{\boldsymbol{\Delta}}{}^{-1}\begin{bmatrix}-\boldsymbol{c}''-\boldsymbol{C}\boldsymbol{M}^{-1}\boldsymbol{f}\\ \boldsymbol{v}-\boldsymbol{h}''-\boldsymbol{H}\boldsymbol{M}^{-1}\boldsymbol{f}\end{bmatrix} \tag{24}$$

with $\boldsymbol{v}=\ddot{\boldsymbol{r}}_\mathrm{d}$ here. 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. The matrix $\boldsymbol{\Delta}=\boldsymbol{\Gamma}\boldsymbol{M}^{-1}\boldsymbol{G}^\mathrm{T}$ needs to be regular and is here denoted as an 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 [30].

Note that for the considered exemplary system of Fig. 1, there are constellations where $\boldsymbol{\Delta}$ becomes singular. This is the case when α_1 is 90°, or α_2 is −90°, or when they are close to such an angle depending on the elastic deformation, since then the actuators can move the end-effector only in the x-direction. However, as this only happens close to the actuator limits, it does not affect typical operations. Thus we further assume that $\boldsymbol{\Delta}$ is regular, leading to a differentiation index of 3 and to a vector relative degree of $\boldsymbol{r}_\mathrm{deg}=\{2,\ldots,2\}$ as mentioned before.

3.2.5 Solving the inverse model

For underactuated systems such as flexible multibody systems, the inverse model from Eqs. (4a)–(4c) has a dynamics corresponding to the so-called (driven) internal dynamics, which is commonly known from feedback linearization [14]. The transformation of the inverse model from DAEs to ODEs, as discussed in the previous sections, will be used to simplify the solution process of this dynamic inverse. In this regard, two solution approaches are

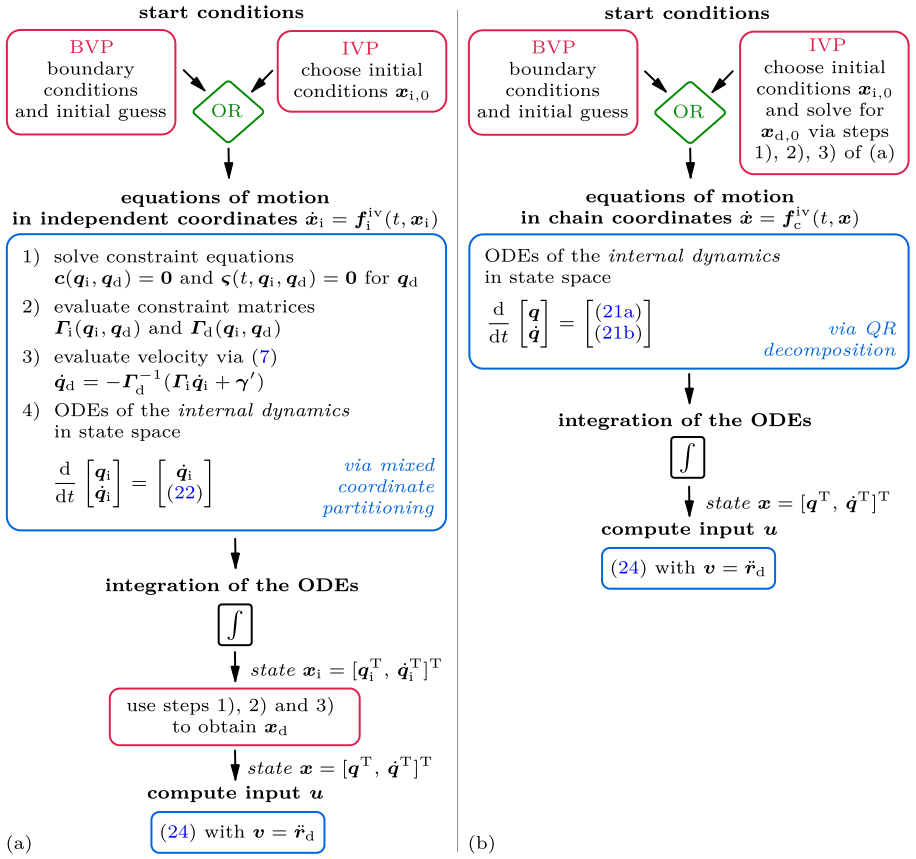

Fig. 3 Solution process of the model inversion for flexible multibody systems with geometric and servo constraints (**a**) in independent coordinates (minimal form) and (**b**) in chain coordinates

now discussed and summarized in Fig. 3. One approach uses a formulation in independent coordinates with the independent state $\boldsymbol{x}_{\mathrm{i}}$, and one uses a formulation in chain coordinates with the state $\boldsymbol{x}$. These state variables are defined as

$$\boldsymbol{x} = \begin{bmatrix}\boldsymbol{q}\\ \dot{\boldsymbol{q}}\end{bmatrix}, \quad \boldsymbol{x}_{\mathrm{i}} = \begin{bmatrix}\boldsymbol{q}_{\mathrm{i}}\\ \dot{\boldsymbol{q}}_{\mathrm{i}}\end{bmatrix}, \quad \boldsymbol{x}_{\mathrm{d}} = \begin{bmatrix}\boldsymbol{q}_{\mathrm{d}}\\ \dot{\boldsymbol{q}}_{\mathrm{d}}\end{bmatrix} \tag{25}$$

with $\boldsymbol{x}_{\mathrm{d}}$ being the dependent state.

For the minimal form, i.e., in independent coordinates, we consider only a manual selection of the independent coordinates $\boldsymbol{q}_{\mathrm{i}}$. This ensures that the coordinate selection does not change throughout operation, which simplifies the solution process. The proposed process via the mixed coordinate partitioning approach is shown in Fig. 3(a). Here we use a QR decomposition for the projection to prevent singularities within $\boldsymbol{G}_{\mathrm{d}}$. The superscript "iv" in the function symbol $\boldsymbol{f}_{\mathrm{i}}^{\mathrm{iv}}$ denotes that an inverse model is used. 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 (4b) and (4c) on the position level. For the formulation in chain coordinates, illustrated in Fig. 3(b), we propose to purely use the QR decomposition since it also prevents singularities within the coordinate selection.

Boundary value problem The stability of the internal dynamics determines how the solution process can be performed. If it is unstable, then the solution process is much more complicated. This is usually the case when tracking the exact end-effector of flexible link robots, i.e., when we use $\boldsymbol{r}$ as output $\boldsymbol{r}_\mathrm{o}$, such as for the exemplary system of Fig. 1. Then the classical causal inversion, which originates from [12, 13], via forward time integration leads to unbounded states and inputs. This can be prevented with the concept of stable inversion [9], applied in this research to obtain bounded results. Within this framework a two-point BVP needs to be solved instead of an IVP, typically much easier to handle. This means that we cannot predefine the initial state, but for the considered system type, a preactuation is necessary to drive the system to the required state while keeping the output constant. Within the preactuation phase an actuator force or torque, which can be regarded as the output of the inverse model, acts before the end-effector moves, which corresponds to the input of the inverse model. Consequently, the inverse has anticausal characteristics and needs to be calculated offline, since the desired end-effector trajectory needs to be known in advance and cannot be applied online. Also, a postactuation is needed at the trajectory end, where the actuators and the internal dynamics come to rest after the end-effector already stopped moving. This is a causal characteristics besides the explained anticausal behavior, meaning that the inverse system incorporates a combination of both characteristics denoted as noncausal [19].

For the stable inversion approach, it is required 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 require 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 [28]. Since these manifolds are typically difficult to handle, they are now locally approximated by the eigenspaces, which has also been done in [36]. 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 so-called zero dynamics, i.e., the internal dynamics with constant output $\boldsymbol{r}_\mathrm{o}$, 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 $\dot{\boldsymbol{x}} = \boldsymbol{f}_\mathrm{c}^\mathrm{iv}(\boldsymbol{x})$; compare Fig. 3. This allows us 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 the autonomous system

$$\dot{\boldsymbol{\eta}} = \boldsymbol{A}_\eta \boldsymbol{\eta}. \tag{26}$$

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 the 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

$$\boldsymbol{A}_\eta = \boldsymbol{T}\boldsymbol{D}_\eta\boldsymbol{T}^{-1}. \tag{27}$$

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

$$\dot{\boldsymbol{\varphi}} = \boldsymbol{D}_\eta \boldsymbol{\varphi}. \tag{28}$$

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 $\boldsymbol{T}^{-1}$ related to the stable eigenvalues and $\boldsymbol{T}_\mathrm{u}$ being the rows related to the unstable eigenvalues, we can construct the boundary conditions. At the trajectory start time $t = t_0$ the boundary condition

$$\boldsymbol{T}_\mathrm{s}(t_0)\boldsymbol{\eta}(t_0) = \boldsymbol{0} \tag{29}$$

keeps the internal dynamics on the approximated unstable manifold, and

$$\boldsymbol{T}_\mathrm{u}(t_\mathrm{F})\boldsymbol{\eta}(t_\mathrm{F}) = \boldsymbol{0} \tag{30}$$

is used for the approximated stable manifold at the trajectory end time $t = t_\mathrm{F}$ [28, 30]. 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 we apply the discussed approach to a formulation in chain coordinates, then 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 $\boldsymbol{\varsigma} = \boldsymbol{0}$ may be used at one boundary and $\dot{\boldsymbol{c}} = \boldsymbol{0}$ and $\dot{\boldsymbol{\varsigma}} = \boldsymbol{0}$ at the other.

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. Further details on stable inversion can be found, e.g., in the book [30].

Initial value problem Even if a bounded noncausal solution can be computed, often a stable internal dynamics is desired, which can be computed online by forward integration of an IVP to obtain a bounded causal solution. In the case of sufficient computational power, this enables even a real-time capable model inversion, and for the later applied feedback linearization, a minimum phase system is needed.

Since for flexible link robots, the unstable internal dynamics is usually introduced through the structural flexibility, such as for the now considered system of Fig. 1, the idea is to weight the corresponding deformation. Thus a redefined end-effector output close to the exact end-effector position will be identified, which leads to a small tracking error and a stable internal dynamics. The 2D position of the exact end-effector can be described by

$$\boldsymbol{r} = \begin{bmatrix} s_1 \\ d/2 \end{bmatrix} + \begin{bmatrix} -\cos\alpha_{1,\mathrm{r}} & \sin\alpha_{1,\mathrm{r}} \\ -\sin\alpha_{1,\mathrm{r}} & -\cos\alpha_{1,\mathrm{r}} \end{bmatrix} \begin{bmatrix} \ell_2 - u_{x'}^{\mathrm{GN}}(\boldsymbol{\phi}(\ell_2)\boldsymbol{q}_\mathrm{e}) \\ \boldsymbol{\phi}(\ell_2)\boldsymbol{q}_\mathrm{e} - u_{y'}^{\mathrm{GN}}(\boldsymbol{\phi}(\ell_2)\boldsymbol{q}_\mathrm{e}) \end{bmatrix}. \tag{31}$$

Here $u_{x'}^{\mathrm{GN}}$ and $u_{y'}^{\mathrm{GN}}$ account for geometric nonlinearity and are functions of the linear deformation $\boldsymbol{\phi}(\ell_2)\boldsymbol{q}_\mathrm{e}$ of the flexible link in the y'-direction, and $\boldsymbol{\phi}$ is the corresponding row vector of the shape functions. Also, ℓ_2 is the undeformed length of the long link 2, and $\alpha_{1,\mathrm{r}}$ is the rotation of the floating frame of reference. As opposed to this, α_1 describes the complete rotation of the joint on the first slider including the part coming from the elastic rotation.

The insignificant deformations in the x'-direction are neglected within $\boldsymbol{r}$. As mentioned before, only the first bending eigenmode is essential. Therefore the elastic deformation $\boldsymbol{\phi}(\ell_2)\boldsymbol{q}_\mathrm{e}$ is replaced by $\phi_1(\ell_2)q_{\mathrm{e},1}$, where ϕ_1 is the displacement shape function related to the elastic coordinate $q_{\mathrm{e},1}$ of the first bending eigenmode. If we additionally weight this deformation by the output redefinition weight w according to $w\phi_1(\ell_2)q_{\mathrm{e},1}$, then we obtain the redefined end-effector output

$$\boldsymbol{r}_\mathrm{re} = \begin{bmatrix} s_1 \\ d/2 \end{bmatrix} + \begin{bmatrix} -\cos\alpha_{1,\mathrm{r}} & \sin\alpha_{1,\mathrm{r}} \\ -\sin\alpha_{1,\mathrm{r}} & -\cos\alpha_{1,\mathrm{r}} \end{bmatrix} \begin{bmatrix} \ell_2 - u_{x'}^{\mathrm{GN}}(w\phi_1(\ell_2)q_{\mathrm{e},1}) \\ w\phi_1(\ell_2)q_{\mathrm{e},1} - u_{y'}^{\mathrm{GN}}(w\phi_1(\ell_2)q_{\mathrm{e},1}) \end{bmatrix}. \tag{32}$$

Table 1 Model inversion cases

case	elast. DOFs	tracked output	equations of motion	solution	intern. dyn.
rigid	none	*rigid* end-effector position	algebraic constraints (4b), (4c), (7), and (24) with $\boldsymbol{v} = \ddot{\boldsymbol{r}}_{\mathrm{d}}$	root finding	none
redefined output	first bending eigenmode	*redefined* end-effector position	in chain coordinates based on QR decomposition (Fig. 3(b))	forward integration of an IVP	stable with $w = 0.9625$
exact end-effector	first bending eigenmode	*exact* end-effector position	minimal form with $\boldsymbol{x}_{\mathrm{i}} = [q_{\mathrm{e},1},\ \dot{q}_{\mathrm{e},1}]^{\mathrm{T}}$ via mixed coordinate partitioning (Fig. 3(a))	stable inversion of a two-point BVP	unstable

For the weight $w = 1$, it corresponds to the exact end-effector position, and for $w = 0$, it corresponds to the end-effector position of an equivalent rigid system. Consequently, a weight w close to 1 needs to be identified, which results in a stable internal dynamics. Then for such w, tracking of the redefined output allows us to use standard ODE integrators to solve the dynamic inverse model via forward integration.

4 End-effector trajectory tracking control

In this section, we present the design and experimental application of different end-effector trajectory tracking controllers, which are the main result of this research. All control approaches are based on an inverse flexible multibody model obtained via servo constraints, as discussed in Sect. 3.2. Here the stability of the resulting internal dynamics decides which control approach is used; compare the overview in Fig. 2.

4.1 Feedforward-based control

In the following, we apply the model inversion discussed in Sect. 3.2 experimentally as feedforward control for the exemplary system of Fig. 1. We implement it via three different cases, which are summarized in Table 1. Here the rigid case corresponds to inverse kinematics for an equivalent rigid system, where the system state, without elastic terms, is obtained via root finding. The input $\boldsymbol{u}$ is computed in a subsequent step.

For the redefined output case, we select a weight w that yields a stable internal dynamics. Then we solve an IVP for a formulation in chain coordinates via QR decomposition. This prevents possible singularities, which is especially important if the model inversion will be performed in real-time.

When tracking the exact end-effector position, a BVP is solved due to the unstable internal dynamics. Here the equations of motion are formulated in independent coordinates, which makes the problem more compact. For the considered system, the only retained elastic coordinate $q_{\mathrm{e},1}$ is selected as independent. It turns out that a rigid initial guess, i.e., with zero deformation, can be used to solve the BVP for the considered trajectory. Since the BVP needs to be solved offline, possible singularities within the underlying mixed coordinate partitioning approach can be caught. These, however, do not occur here.

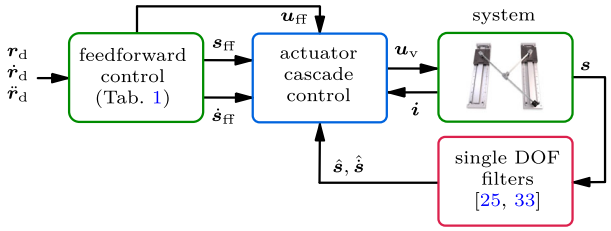

Fig. 4 Control structure of the feedforward control based on model inversion with actuator feedback

In Sect. 4.1.1 the actuator trajectories resulting from these three different model inversions are sent to an actuator cascade control. In Sect. 4.1.2 we use an additional LQR to also control the elastic deformation within a feedback loop.

4.1.1 Feedforward control only with actuator feedback control

A straightforward implementation of the model-based feedforward control within experiments is to only apply the corresponding actuator-related trajectories. This means that only the feedforward trajectories for the actuator motions $\boldsymbol{s}, \dot{\boldsymbol{s}}$ and for the input currents $\boldsymbol{u}$ are used and feedback controlled. The resulting control structure is shown in Fig. 4. Here "ff" denotes the corresponding feedforward quantities, and the hat symbol denotes estimated quantities. The necessary time derivatives of the position measurements $\boldsymbol{s}$ can be efficiently obtained by a standard Kalman filter, considering these signals as linear single-degree-of-freedom (DOF) systems where the neglected dynamics is included as a process noise [25, 33]. This method is here denoted as single DOF filtering. Then the obtained quantities are sent to an actuator cascade control, which compensates disturbances such as friction to a large extent. It consists of three loops, with a P control for the position loop, a PI control for the velocity loop, and another PI control for the loop of the current $\boldsymbol{i}$. With these loops, we obtain the voltage $\boldsymbol{u}_{\mathrm{v}}$.

Errors within the elastic deformation are not feedback controlled. Nevertheless, for cases with an accurate model-based feedforward control, these errors will be small. Moreover, not using feedback of the elastic link deformations ensures a very stable control behavior also in the presence of significant disturbances.

Now experiments are conducted for the exemplary system of Fig. 1. The results for the three model inversion cases from Table 1 are shown in Fig. 5. Here e_{rms} is the root-mean-square error, and e_{max} is the maximum absolute error of the 2D end-effector position. The bars here and of all following measurements represent the mean values of three different experiments. The smooth desired end-effector trajectory is a line of length $0.2\sqrt{2}$ m to be tracked in 0.5 s, which will also be used in all following experiments. This trajectory, i.e., the desired end-effector motion starts after 1 s, which allows a preactuation phase for the BVP. For this BVP, the model inversion control tracks the exact end-effector position and therefore introduces only numerical tracking errors in simulations, in contrast to the other two cases, which track end-effector approximations. Still, Fig. 5 shows that the redefined output case in experiments performs very similarly to the exact end-effector case with very small absolute error differences. On the one hand, this is caused by experimental errors, which introduce additional deviations for all cases. On the other hand, this is due to the high value of $w = 0.9625$, which gives a redefined end-effector output $\boldsymbol{r}_{\mathrm{re}}$ close to the exact end-effector $\boldsymbol{r}$. This value also leads to a stable internal dynamics.

Note that an unscented Kalman filter (UKF) is used for the end-effector position estimation, which is based on a very similar model as the flexible controllers. The implementation

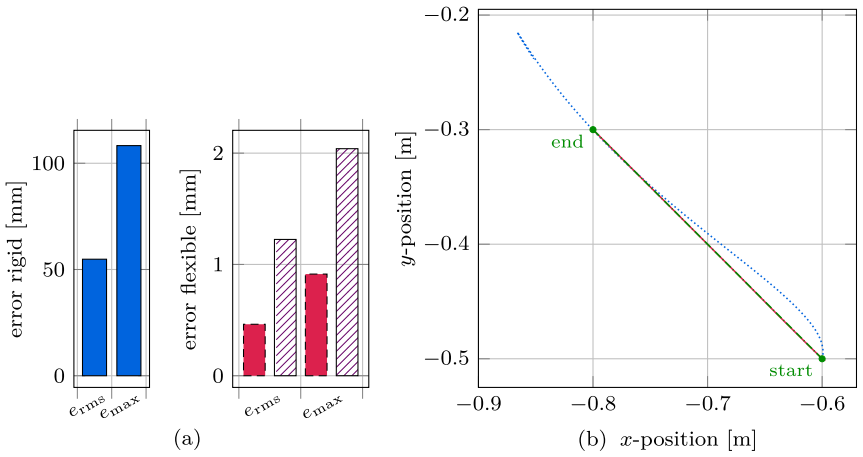

Fig. 5 Exact end-effector (**a**) trajectory tracking errors and (**b**) path for experiments for the feedforward control with actuator feedback based on a rigid model (,) and based on a flexible model for the exact (,) as well as for the redefined end-effector using $w = 0.9625$ ( ), being compared to the desired trajectory ( )

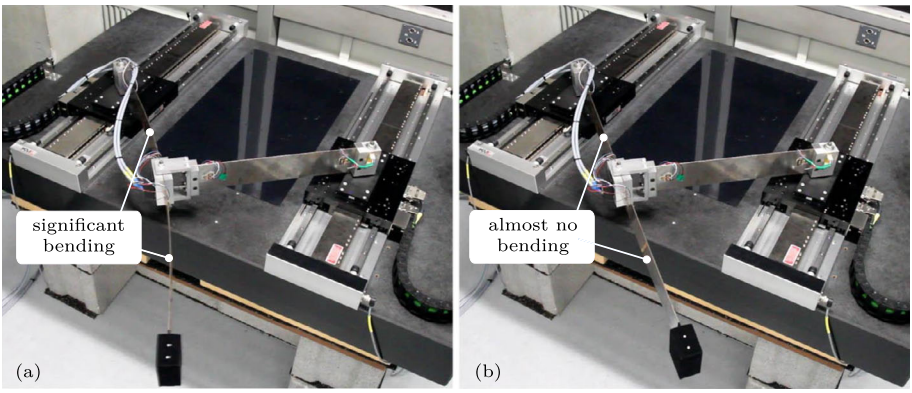

Fig. 6 Snapshots of the test rig for feedforward control with actuator feedback at the end of the line trajectory: (**a**) based on a rigid model and (**b**) based on a flexible model for the exact end-effector

details of the UKF can be found in [25]. Nevertheless, both flexible cases clearly outperform the classical rigid case, since the considered motion excites significant bending. Snapshots of the experiments in Fig. 6 confirm a huge improvement introduced through using a flexible model. Due to the poor performance of the rigid case, it will not be further considered in this paper.

4.1.2 Feedforward control with an LQR

In the previous Sect. 4.1.1, only using measurement feedback for the actuators performs very effectively since negligible disturbances on the link deformations exist. Still, scenarios are possible where significant deviations occur in the elastic deformations. Then the state

trajectories obtained by a model inversion can be systematically tracked, for example, using an LQR, which uses state feedback. This enables flexible link robots to also react to deviations within the trajectories of the elastic coordinates to ensure a close end-effector trajectory tracking.

The LQR formalism (see, e.g., [18]) is designed for linear forward models in state-space representation. Therefore, in the first step the DAEs (2a), (2b), which describe a general forward model of flexible link parallel robots, will be 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 the exemplary system of Fig. 1 to be not stabilizable. The forward model needed for the LQR simply corresponds to the inverse model without servo constraints, i.e., $n_\varsigma = 0$. Therefore the equations introduced in Sect. 3.2 can be analogously applied when omitting the terms coming from servo constraints. The matrices

$$\boldsymbol{J}_\ell = \boldsymbol{J}_\mathrm{r} = \begin{bmatrix} \mathbf{I}_{f_\mathrm{i}} \\ -\boldsymbol{C}_\mathrm{d}^{-1}\boldsymbol{C}_\mathrm{i} \end{bmatrix} = \boldsymbol{J}_\mathrm{c} \tag{33}$$

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

$$\boldsymbol{C}_\mathrm{i} = \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}_\mathrm{i}}, \quad \boldsymbol{C}_\mathrm{d} = \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}_\mathrm{d}} \tag{34}$$

are needed, which have the dimensions $n_\mathrm{c} \times f_\mathrm{i}$ 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 (2a). With a derivation analogous to that of Eq. (13), the independent accelerations follow as

$$\ddot{\boldsymbol{q}}_\mathrm{i} = \left(\boldsymbol{J}_\mathrm{c}^\mathrm{T}\boldsymbol{M}\boldsymbol{J}_\mathrm{c}\right)^{-1}\boldsymbol{J}_\mathrm{c}^\mathrm{T}\left(\boldsymbol{f} + \boldsymbol{B}\boldsymbol{u} - \boldsymbol{M}\boldsymbol{\theta}_\mathrm{c}''\right). \tag{35}$$

Here the control input $\boldsymbol{u}$ still occurs since a forward model is considered in contrast to Eq. (13). Also, the terms related to servo constraints within $\boldsymbol{\theta}''$ vanish, which gives

$$\boldsymbol{\theta}'' = \begin{bmatrix} \mathbf{0} \\ -\boldsymbol{C}_\mathrm{d}^{-1}\boldsymbol{c}'' \end{bmatrix} = \boldsymbol{\theta}_\mathrm{c}''. \tag{36}$$

With Eq. (35), the function for the equations of motion in independent coordinates can be established analogously to Fig. 3(a) 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 Fig. 7. Here, "fw" denotes the forward model.

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 $\boldsymbol{u}$ 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 $\boldsymbol{x}_{\mathrm{i,s}} = [\boldsymbol{q}_{\mathrm{i,s}}^\mathrm{T},\ \mathbf{0}^\mathrm{T}]^\mathrm{T}$, and assuming a planar scenario, the input $\boldsymbol{u}$ is chosen as $\boldsymbol{u}_\mathrm{s} = \mathbf{0}$. For the considered exemplary system, $\boldsymbol{q}_\mathrm{i}$ is now of dimension

$$f_\mathrm{i} = f - n_\mathrm{c} - n_\varsigma = 5 - 2 - 0 = 3. \tag{37}$$

Here the actuator positions and the elastic coordinates, containing only the first bending eigenmode, are chosen as independent, i.e., $\boldsymbol{q}_\mathrm{i} = [q_{\mathrm{e},1},\ s_1,\ s_2]^\mathrm{T}$. This is different from the previous Sect. 4.1.1, where only the elastic coordinate is selected as independent within the model inversion formulated as a BVP. For $\boldsymbol{q}_{\mathrm{i,s}}$, different actuator positions are chosen at each

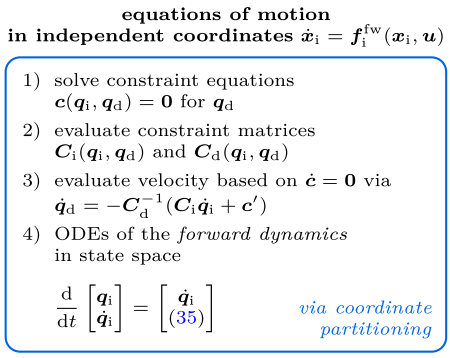

Fig. 7 Forward model for flexible multibody systems with geometric constraints in independent coordinates (minimal form)

linearization point with zero link deformation. Using such steady-state points represents an effective trade-off, since flexible links oscillate around these points, and typically the nonlinearity comes mainly from the underlying rigid body poses. This also ensures that the LQR design is applicable to a variety of scenarios, such as tracking of initially unknown trajectories, i.e., without linearization around the considered trajectory.

Analogously to $\boldsymbol{f}_\mathrm{i}^\mathrm{fw}$ within Fig. 7, the output $\boldsymbol{y} = \boldsymbol{r}_\mathrm{o}(\boldsymbol{q}) \in \mathbb{R}^{f_\mathrm{o}}$ can also be written as a function $\boldsymbol{y} = \boldsymbol{r}_\mathrm{o}(\boldsymbol{q}_\mathrm{i})$, which relies on 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 follows as

$$\dot{\tilde{\boldsymbol{x}}}_\mathrm{i} = \left.\frac{\partial \boldsymbol{f}_\mathrm{i}^\mathrm{fw}}{\partial \boldsymbol{x}_\mathrm{i}}\right|_{\substack{\boldsymbol{x}_\mathrm{i,s}\\ \boldsymbol{u}_\mathrm{s}}} \tilde{\boldsymbol{x}}_\mathrm{i} + \left.\frac{\partial \boldsymbol{f}_\mathrm{i}^\mathrm{fw}}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}_\mathrm{i,s}\\ \boldsymbol{u}_\mathrm{s}}} \boldsymbol{u} = \tilde{\boldsymbol{A}}_\mathrm{i}\tilde{\boldsymbol{x}}_\mathrm{i} + \tilde{\boldsymbol{B}}_\mathrm{i}\boldsymbol{u}, \tag{38a}$$

$$\tilde{\boldsymbol{y}} = \left.\frac{\partial \boldsymbol{r}_\mathrm{o}}{\partial \boldsymbol{x}_\mathrm{i}}\right|_{\boldsymbol{x}_\mathrm{i,s}} \tilde{\boldsymbol{x}}_\mathrm{i} = \tilde{\boldsymbol{C}}_\mathrm{i}\tilde{\boldsymbol{x}}_\mathrm{i} \tag{38b}$$

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 $\tilde{\boldsymbol{A}}_\mathrm{i} \in \mathbb{R}^{2f_\mathrm{i}\times 2f_\mathrm{i}}$ is the constant system matrix, $\tilde{\boldsymbol{B}}_\mathrm{i} \in \mathbb{R}^{2f_\mathrm{i}\times f_\mathrm{a}}$ is the constant input matrix, and $\tilde{\boldsymbol{C}}_\mathrm{i} \in \mathbb{R}^{f_\mathrm{o}\times 2f_\mathrm{i}}$ 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

$$J = \int_0^\infty \tilde{\boldsymbol{x}}_\mathrm{i}^\mathrm{T}\tilde{\boldsymbol{Q}}_\mathrm{i}\tilde{\boldsymbol{x}}_\mathrm{i} + \boldsymbol{u}^\mathrm{T}\tilde{\boldsymbol{R}}_\mathrm{i}\boldsymbol{u}\,\mathrm{d}t. \tag{39}$$

Selecting $\tilde{\boldsymbol{Q}}_\mathrm{i} = \tilde{\boldsymbol{C}}_\mathrm{i}^\mathrm{T}\tilde{\boldsymbol{C}}_\mathrm{i} \in \mathbb{R}^{2f_\mathrm{i}\times 2f_\mathrm{i}}$ minimizes $\tilde{\boldsymbol{y}}^\mathrm{T}\tilde{\boldsymbol{y}} \geq 0$, which guarantees that $\tilde{\boldsymbol{Q}}_\mathrm{i}$ is positive semidefinite. Since the output $\boldsymbol{r}_\mathrm{o}$ does not depend on the velocities $\dot{\boldsymbol{q}}_\mathrm{i}$, the $\tilde{\boldsymbol{Q}}_\mathrm{i}$-matrix contains nonzero elements only in the upper left block. Additionally using the exact end-effector $\boldsymbol{r}$ as output, i.e., $\boldsymbol{y} = \boldsymbol{r}_\mathrm{o} = \boldsymbol{r}$, ensures that its error is kept small, which is crucial for the considered end-effector trajectory tracking. Note that since the LQR is still no output controller, no problems occur when using the exact end-effector. The matrix $\tilde{\boldsymbol{R}}_\mathrm{i} \in \mathbb{R}^{f_\mathrm{a}\times f_\mathrm{a}}$ 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

$$\boldsymbol{u} = -\boldsymbol{K}_\mathrm{lqr}\tilde{\boldsymbol{x}}_\mathrm{i} \tag{40}$$

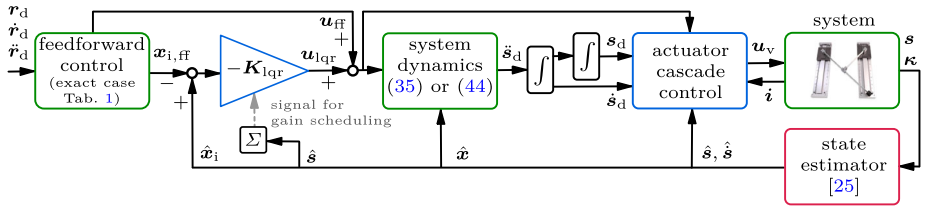

Fig. 8 Control structure of the LQR for end-effector trajectory tracking of flexible link parallel robots

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. For the considered exemplary robot, $\boldsymbol{K}_{\mathrm{lqr}}$ only depends on the sum of the two actuator positions. The reason is that linearization points with the same sum lead to an identical kinematic setup only with a possible shift in the x-direction of the complete robot.

The currents obtained via the LQR are then added to the currents obtained via the feedforward control for the exact end-effector as described in Table 1. These currents are denoted as $\boldsymbol{u}_{\mathrm{lqr}}$ and $\boldsymbol{u}_{\mathrm{ff}}$, respectively; see Fig. 8. Also, in this control structure, we can see that the standard LQR control law from Eq. (40) is slightly adapted by feeding back a trajectory tracking error. This is the error between the independent state $\hat{\boldsymbol{x}}_{\mathrm{i}}$, estimated by a UKF, and the precalculated independent state $\boldsymbol{x}_{\mathrm{i,ff}}$ of the feedforward control. This ensures that the state trajectories $\boldsymbol{x}_{\mathrm{i,ff}}$ are closely tracked. Since these trajectories are obtained via a model inversion for the desired end-effector trajectory $\boldsymbol{r}_{\mathrm{d}}$, the LQR can even improve the end-effector output trajectory tracking performance.

Directly applying the input currents $\boldsymbol{u}_{\mathrm{lqr}} + \boldsymbol{u}_{\mathrm{ff}}$ is usually not meaningful, e.g., because of friction effects within the actuators. Therefore the currents are converted to actuator motions via the projected system dynamics and the estimated state to realize these motions via the actuator cascade control; compare Fig. 8. Here Eq. (35) 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 obtain the projection matrix $\boldsymbol{J}_{\mathrm{c,q}} \in \mathbb{R}^{f \times f_{\mathrm{i}}}$ based on a QR decomposition. This prevents possible singularities within $\boldsymbol{C}_{\mathrm{d}}$, which can occur through a manual selection of independent coordinates. With the matrices from Eqs. (6a), (6b), which reduce to

$$\boldsymbol{G} = \boldsymbol{\Gamma} = \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{q}} = \boldsymbol{C} \tag{41}$$

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

$$\boldsymbol{C}^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{Q}_{\mathrm{c}} & \boldsymbol{J}_{\mathrm{c,q}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}_{\mathrm{c}} \\ \boldsymbol{0} \end{bmatrix} = \boldsymbol{Q}_{\mathrm{c}} \boldsymbol{R}_{\mathrm{c}}. \tag{42}$$

This also gives $\boldsymbol{Q}_{\mathrm{c}} \in \mathbb{R}^{f \times n_{\mathrm{c}}}$ and the square matrix $\boldsymbol{R}_{\mathrm{c}} \in \mathbb{R}^{n_{\mathrm{c}} \times n_{\mathrm{c}}}$. Also, the unphysical independent accelerations $\ddot{\boldsymbol{q}}_{\mathrm{i,q}}$ need to be transformed back to the accelerations $\ddot{\boldsymbol{q}}$ in chain coordinates. Since here

$$\boldsymbol{J}_{\ell,\mathrm{q}} = \boldsymbol{J}_{\mathrm{r,q}} = \boldsymbol{J}_{\mathrm{c,q}}, \tag{43}$$

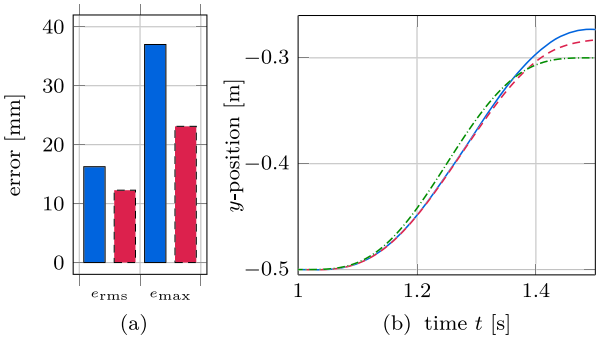

Fig. 9 Exact end-effector (**a**) trajectory tracking errors and (**b**) y-position for experiments with an unmodeled mass of 0.4 kg: for the flexible feedforward control for the exact end-effector with actuator feedback (▬,——) and additional LQR (▬,- - - -), compared to the desired trajectory (- - - -)

in a similar way as for Eq. (21b), we obtain the relation in chain coordinates as

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}_{\mathrm{c,q}} \left(\boldsymbol{J}_{\mathrm{c,q}}^{\mathrm{T}} \boldsymbol{M} \boldsymbol{J}_{\mathrm{c,q}} \right)^{-1} \boldsymbol{J}_{\mathrm{c,q}}^{\mathrm{T}} \left(\boldsymbol{f} + \boldsymbol{B}\boldsymbol{u} - \boldsymbol{M}\boldsymbol{\theta}_{\mathrm{c,q}}'' \right) + \boldsymbol{\theta}_{\mathrm{c,q}}'' \tag{44}$$

with $\boldsymbol{\theta}_{\mathrm{c,q}}'' = -\boldsymbol{Q}_{\mathrm{c}} \boldsymbol{R}_{\mathrm{c}}^{-\mathrm{T}} \boldsymbol{c}''$. From $\ddot{\boldsymbol{q}}$, resulting from evaluating the right side of Eq. (44), the actuator accelerations $\ddot{\boldsymbol{s}}$ can be selected, which are then integrated by the simple forward Euler method. Finally, the obtained desired actuator motions $\boldsymbol{s}_{\mathrm{d}}$ and $\dot{\boldsymbol{s}}_{\mathrm{d}}$ can be accurately reproduced via the actuator cascade control.

Now experiments are conducted for the exemplary system of Fig. 1, where the LQR is used together with the feedforward control for the desired line trajectory. Here 101 steady-state points are used for the linearization for the LQR design. The linearization is performed in a preprocessing step. Also, both diagonal elements of the diagonal matrix $\tilde{\boldsymbol{R}}_{\mathrm{i}}$ are chosen as 10^{-4}, which is found via tuning. Note that the SI units for $\tilde{\boldsymbol{R}}_{\mathrm{i}}$ and for further control gains are omitted for easier readability.

It turns out that the experimental end-effector trajectory tracking performance with an additional LQR is very similar when only using actuator feedback control such as in the previous Sect. 4.1.1. Nevertheless, the LQR can be highly advantageous for cases with larger disturbances or modeling errors. In the experiments presented in Fig. 9 an unmodeled mass of 0.4 kg is attached to the end-effector, which reduces the first bending eigenfrequency by around 15%. Here the feedforward control, the LQR, and the state estimator, on which the LQR relies, all lack knowledge of the added mass. Therefore, to still evaluate the control performance of the exact end-effector, a second informed UKF is used, which knows the added mass. The results in Fig. 9 are completely based on this informed UKF. They clearly show the improved end-effector trajectory tracking performance with an additional LQR in the presence of significant modeling errors.

4.2 Feedback linearization

Up to now the actuator motions and the system state have been controlled in a feedback loop to provide end-effector trajectory tracking. Still, the error of the actual tracking output $\boldsymbol{r}_{\mathrm{o}}$ has not been directly feedback controlled yet. Therefore the concept of feedback linearization [14] is now considered to obtain a linear input–output relation that is easy to control. Here an approach is proposed for feedback linearization of flexible multibody systems with

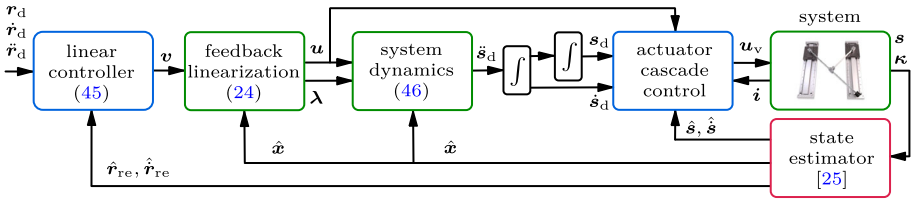

Fig. 10 Control structure of the input–output feedback linearization for flexible link robots with geometric and servo constraints

geometric and servo constraints. Due to the existence of an internal dynamics when using a flexible model, full state feedback linearization is not possible. Therefore in this section, we focus on input–output feedback linearization using a flexible multibody model.

Tracking of a redefined output, i.e., $\boldsymbol{r}_o = \boldsymbol{r}_{re}$, ensures that a minimum phase system is obtained. For the considered system of Fig. 1, this results in two output components, which will be controlled by two inputs, being the currents of the linear motors. Also, as explained in Sect. 3.2.4, the extended decoupling matrix $\boldsymbol{\Delta}$ is regular for the investigated scenarios. This leads to the fulfilled sufficient conditions for input–output feedback linearization with geometric and servo constraints via a static state feedback law:

(1) The system is minimum phase, due to tracking of an appropriate redefined output $\boldsymbol{r}_{re}$.
(2) The numbers of inputs and outputs, introduced via servo constraints, are equal.
(3) The vector relative degree related to the geometric and servo constraints, $\boldsymbol{r}_{deg} = \{2, \ldots, 2\}$, is well-defined with a regular extended decoupling matrix $\boldsymbol{\Delta}$ of Eq. (24).

The implemented control structure is illustrated in Fig. 10 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 used. To apply the concept of input–output feedback linearization, nonetheless, these constraints are incorporated via their second time derivatives. Together with the system dynamics, Eq. (24) has been obtained. This equation already represents the static state feedback law of the input–output feedback linearization to obtain the required control inputs $\boldsymbol{u}$ and the Lagrange multipliers $\boldsymbol{\lambda}$. To evaluate Eq. (24), the state $\boldsymbol{x} = [\boldsymbol{q}^T, \dot{\boldsymbol{q}}^T]^T$ is needed, which is estimated via a UKF based on the measured actuator positions $\boldsymbol{s}$ and the link curvatures $\boldsymbol{\kappa}$. This means that also the current link deformations are considered in the controller.

Note that no transformation to the (Byrnes–Isidori) input–output normal form [6] is needed to obtain the input–output linearizing controller. Similarly, in Sect. 3.2 the internal dynamics is directly obtained via projections (compare Fig. 3) 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 has been used to find a redefined output $\boldsymbol{r}_{re}$ that renders this internal dynamics stable.

Applying the control input $\boldsymbol{u}$, obtained via Eq. (24), then theoretically cancels all nonlinear terms. Due to the incorporated servo constraints, this yields $\ddot{\boldsymbol{r}}_{re} = \boldsymbol{v}$, i.e., a linear differential input–output relation between the new control input $\boldsymbol{v}$ and the output $\boldsymbol{r}_{re}$. Selecting $\boldsymbol{v} = \ddot{\boldsymbol{r}}_d$ gives $\ddot{\boldsymbol{r}}_{re} = \ddot{\boldsymbol{r}}_d$ in an ideal scenario. Due to the input–output feedback linearization, occurring tracking errors within realistic scenarios can thus be compensated by a simple

linear controller of the form

$$\boldsymbol{v} = \ddot{\boldsymbol{r}}_{\mathrm{d}} + \begin{bmatrix} D & 0 \\ 0 & D \end{bmatrix} (\dot{\boldsymbol{r}}_{\mathrm{d}} - \dot{\boldsymbol{r}}_{\mathrm{re}}) + \begin{bmatrix} P & 0 \\ 0 & P \end{bmatrix} (\boldsymbol{r}_{\mathrm{d}} - \boldsymbol{r}_{\mathrm{re}}) \tag{45}$$

for the considered system. This leads to a stable output error dynamics for $P, D > 0$. Also, via Eq. (45) the desired end-effector trajectory is now applied in real-time. Regarding the Lagrange multipliers $\boldsymbol{\lambda}$ as additional control inputs, we can analogously stabilize the geometric constraints $\boldsymbol{c} = \boldsymbol{0}$ [10]. This is, however, not necessary when using a real system where these constraints cannot drift.

As a next step, the control inputs $\boldsymbol{u}$, i.e., the motor currents, are again transformed to actuator, i.e., slider accelerations $\ddot{\boldsymbol{s}}$, to guarantee, amongst others, an effective friction compensation via the utilized actuator cascade control. In contrast to the LQR of Sect. 4.1.2, where the unknown Lagrange multipliers are canceled by a projection leading to Eqs. (35) and (44), the $\boldsymbol{\lambda}$-vector is automatically calculated via Eq. (24) of the feedback linearization. This vector can then be simply plugged into Eq. (4a) together with the available control input $\boldsymbol{u}$ and the state $\boldsymbol{x}$, which gives

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}^{-1} \left(\boldsymbol{f} + \boldsymbol{B}\boldsymbol{u} + \boldsymbol{C}^{\mathrm{T}} \boldsymbol{\lambda} \right). \tag{46}$$

Here $\ddot{\boldsymbol{q}}$ contains the needed actuator accelerations $\ddot{\boldsymbol{s}}$. Evaluating Eq. (46) is computationally very efficient since the inverse of $\boldsymbol{M}$ is already known from Eq. (24) and the forward Euler method is accurate enough for the subsequent time integration. The resulting desired actuator motions $\boldsymbol{s}_{\mathrm{d}}$ and $\dot{\boldsymbol{s}}_{\mathrm{d}}$ 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, then the actuator accelerations can be obtained directly via Eq. (21b). Then a replacement of $\ddot{\boldsymbol{r}}_{\mathrm{d}}$ by $\boldsymbol{v}$ within $\boldsymbol{\theta}_{\mathrm{q}}''$ is necessary.

It is worth noting that a major difference from the previously discussed flexible feedforward controllers, which are dynamic controllers, is that the input–output feedback linearization only uses an algebraic, i.e., static control law, which is computationally much more efficient. Nevertheless, feedback linearization requires 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 input–output feedback linearization is applied experimentally for the desired line trajectory using the flexible link parallel robot of Fig. 1. Within the resulting plots of Fig. 11, we can observe that a larger weight w, i.e., choosing a redefined output $\boldsymbol{r}_{\mathrm{re}}$ closer to the exact end-effector position output $\boldsymbol{r}$, yields a more effective trajectory tracking behavior of $\boldsymbol{r}$. Also, the redefined output, which is in fact tracked by the controller, is kept very close to the desired end-effector trajectory that validates the control approach.

The corresponding errors are shown in Fig. 12 for different weights w and PD control gains. Here the small errors for the redefined output $\boldsymbol{r}_{\mathrm{re}}$ are comparable to the flexible feedforward results from Fig. 5. Therefore only a minor improvement is visible when using the nonzero 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 $\boldsymbol{r}$. Obviously, the main error within the exact end-effector comes from output redefinition needed to obtain a minimum phase system. This error can only be safely decreased if, e.g., disturbances are reduced to allow a higher value than the robust value of $w = 0.75$. The experimental setup under consideration turned out to become unstable roughly at $w = 0.85$. This is still somewhat away from the weight of $w = 0.9625$, which can be used within a flexible feedforward control in real-time. Thus

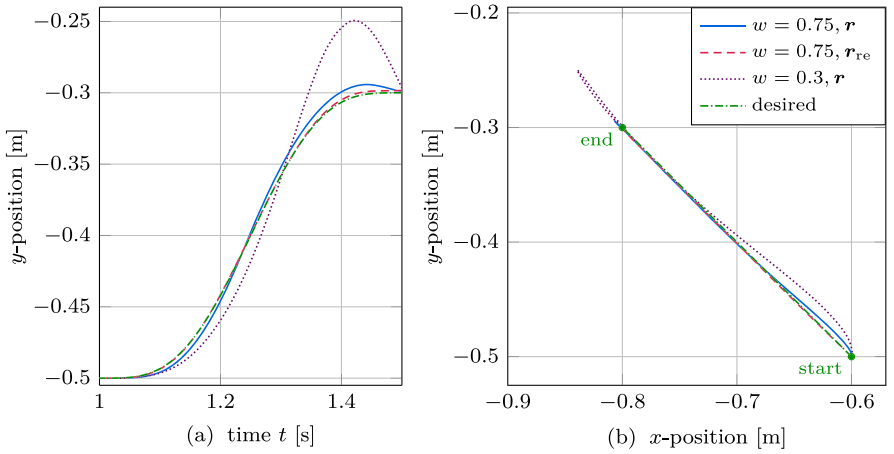

Fig. 11 End-effector trajectory tracking experiments for input–output feedback linearization with $P = D = 10$, showing (**a**) the end-effector y-position and (**b**) the end-effector path

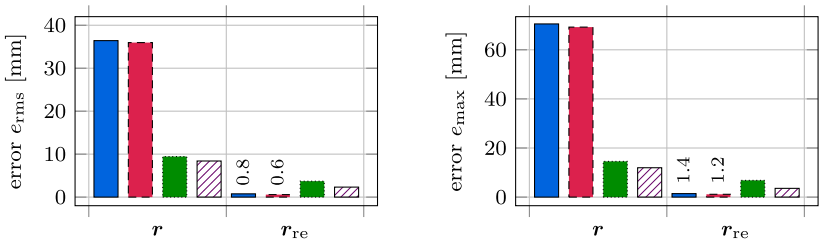

Fig. 12 End-effector trajectory tracking errors for input–output feedback linearization experiments with $w = 0.3$, $P = D = 0$ (), $w = 0.3$, $P = D = 10$ (), $w = 0.75$, $P = D = 0$ (), and $w = 0.75$, $P = D = 10$ ()

the redefinition error for the feedback linearization is significantly higher than for the feedforward control. 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 scenario in which the desired end-effector trajectory is known in advance, such as for repetitive tasks, the error introduced through output redefinition can be precompensated. With the offline calculated states from a model inversion for the exact end-effector case formulated as BVP, the corresponding redefined output motion can be evaluated. Applying this motion, denoted as adapted "line" trajectory, a new desired trajectory will ensure that a close tracking of $\boldsymbol{r}_{\mathrm{re}}$ of this adapted line results in a close tracking of $\boldsymbol{r}$ 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 Fig. 13. The performance is also close to the flexible feedforward control experiments from Fig. 5 with the possibility to compensate the disturbances on the flexible link. Furthermore, the PD control gains now improve the exact end-effector trajectory tracking error by a significant percentage. Interestingly, for such an adapted trajectory, the case $w = 0.3$ performs better than the case $w = 0.75$. However, as this cannot be confirmed by simulations, it is most likely an experimental effect occurring through the different weighting of the elastic coordinate.

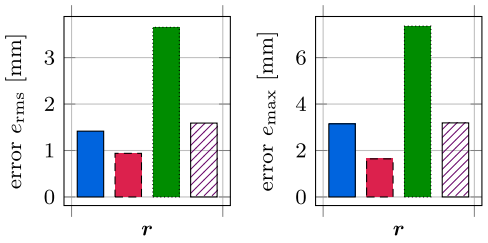

Fig. 13 End-effector trajectory tracking errors for input–output feedback linearization experiments with adapted line trajectory for $w = 0.3$, $P = D = 0$ (), $w = 0.3$, $P = D = 10$ (), $w = 0.75$, $P = D = 0$ (), and $w = 0.75$, $P = D = 10$ ()

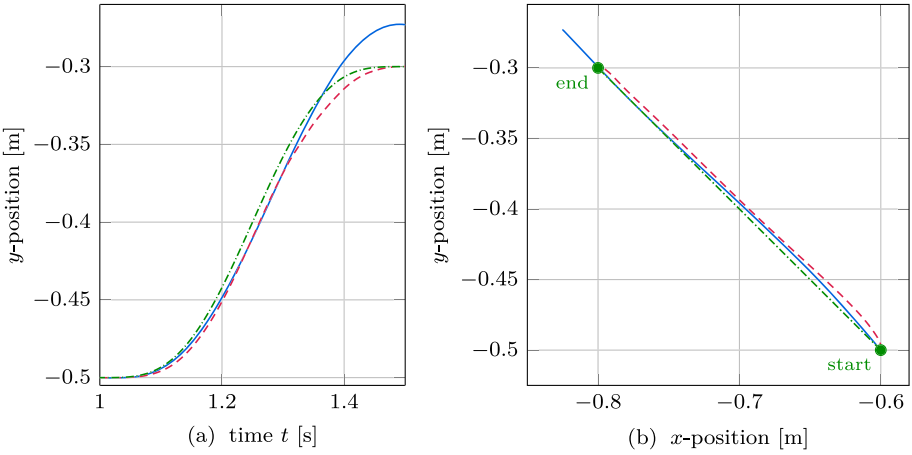

Fig. 14 Exact end-effector (**a**) y-position and (**b**) path for experiments with an unmodeled mass of 0.4 kg: for the flexible feedforward control for the exact end-effector with actuator feedback (——) and for the input–output feedback linearization with $w = 0.75$, $P = D = 10$ (- - - -), compared to the desired trajectory (—·—·—)

Finally, the additional end-effector mass from Sect. 4.1.2 is used to face the input–output feedback linearization controller with significant modeling errors. Both the controller and the state estimator, which it relies on, do not know this mass of 0.4 kg. Therefore, for quantification purposes of the actual control performance, the results in Figs. 14 and 15 are estimates of a second informed UKF, which knows the added mass. Also, the output redefinition is not compensated, i.e., no adapted line trajectory is used. The significant end-effector trajectory tracking error caused by the unmodeled mass for the feedforward control is taken from Fig. 9. The input–output feedback linearization without PD control, i.e., without feedback of the output error, leads to a reduced maximum error but to a larger root-mean-square error. In contrast, the input–output feedback linearization with active PD control improves both errors. Especially the reduction of the maximum error compared to the feedforward control is significant, caused by the compensation of the overshoot at the trajectory end; see Fig. 14. Also, the errors with active PD control are comparable to the results with an additional LQR shown in Fig. 9.

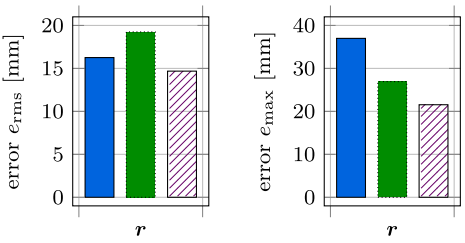

Fig. 15 End-effector trajectory tracking errors for experiments with an unmodeled mass of 0.4 kg: for the flexible feedforward control for the exact end-effector with actuator feedback ( ) and for the input–output feedback linearization with $w = 0.75$, $P = D = 0$ ( ) and $w = 0.75$, $P = D = 10$ ( )

5 Conclusions

This research presents an overview of computationally efficient modeling and end-effector trajectory tracking control approaches for flexible multibody systems with geometric constraints, e.g., due to kinematic loops, and servo constraints.

Initially, the flexible multibody model is inverted with the concept of servo constraints. To simplify the solution process, the resulting set of differential-algebraic equations is transformed into ordinary differential equations by projections. Typically, for flexible link robots, such as the considered exemplary system, an unstable internal dynamics occurs when tracking the exact end-effector. Then a two-point boundary value problem is solved offline within the framework of stable inversion. Initially, the resulting feedforward control based on flexible model inversion is applied only with actuator feedback. Here tracking of a redefined end-effector output, which leads to a stable internal dynamics that can be solved as initial value problem, performs similarly to tracking of the exact end-effector position within experiments. Also, both flexible model inversions show a significantly improved end-effector trajectory tracking performance compared to feedforward control based on classical rigid body modeling.

Subsequently, to also compensate errors within the elastic deformation, not only the actuator feedback is used, but also the state feedback. Here the well-known linear-quadratic regulator approach is adapted to trajectory tracking control for flexible multibody systems with geometric constraints. This leads to a superior end-effector trajectory tracking performance within experiments where an unmodeled mass is attached to the end-effector of the considered flexible link parallel robot. Finally, for a redefined output rendering the system minimum phase, the well-established input–output feedback linearization is applied. Here a systematic approach is proposed for flexible multibody systems with geometric and servo constraints. Experimental results validate the control design. Assuming a repetitive task allows us to adapt the desired trajectory offline to precompensate the error introduced by output redefinition. This further reduces the experimental end-effector trajectory tracking errors. The input–output feedback linearization also proves to be valuable for the case of an unmodeled mass at the end-effector.

Acknowledgements The authors would like to thank the German Research Foundation (DFG) for the financial support of the projects 187619583 and 396289190 and of the project EXC 2075 Simtech 390740016, subproject PN4-4.

Funding Note Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of Interest The authors declare that they have no conflict of interest.

References

1. Bai, M., Zhou, D.H., Schwarz, H.: Adaptive augmented state feedback control for an experimental planar two-link flexible manipulator. IEEE Trans. Robot. Autom. **14**(6), 940–950 (1998)
2. Benosman, M., Le Vey, G.: Control of flexible manipulators: a survey. Robotica **22**(5), 533–545 (2004)
3. Blajer, W., Kołodziejczyk, K.: A geometric approach to solving problems of control constraints: theory and a DAE framework. Multibody Syst. Dyn. **11**(4), 343–364 (2004)
4. Boyer, F., Khalil, W.: An efficient calculation of flexible manipulator inverse dynamics. Int. J. Robot. Res. **17**(3), 282–293 (1998)
5. Burkhardt, M., Seifried, R., Eberhard, P.: Experimental studies of control concepts for a parallel manipulator with flexible links. J. Mech. Sci. Technol. **29**(7), 2685–2691 (2015)
6. Byrnes, C.I., Isidori, A.: Local stabilization of minimum-phase nonlinear systems. Syst. Control Lett. **11**(1), 9–17 (1988)
7. Campbell, S.L.: High-index differential algebraic equations. J. Struct. Mech. **23**(2), 199–222 (1995)
8. Carusone, J., Buchan, K.S., D'Eleuterio, G.M.T.: Experiments in end-effector tracking control for structurally flexible space manipulators. IEEE Trans. Robot. Autom. **9**(5), 553–560 (1993)
9. Chen, D., Paden, B.: Stable inversion of nonlinear non-minimum phase systems. Int. J. Control **64**(1), 81–97 (1996)
10. Chiou, J.C., Wu, S.D.: Constraint violation stabilization using input–output feedback linearization in multibody dynamic analysis. J. Guid. Control Dyn. **21**(2), 222–228 (1998)
11. De Luca, A., Siciliano, B.: Trajectory control of a non-linear one-link flexible arm. Int. J. Control **50**(5), 1699–1715 (1989)
12. Hirschorn, R.: Invertibility of multivariable nonlinear control systems. IEEE Trans. Autom. Control **24**(6), 855–865 (1979)
13. Hirschorn, R.M.: Invertibility of nonlinear control systems. SIAM J. Control Optim. **17**(2), 289–297 (1979)
14. Isidori, A.: Nonlinear Control Systems, 3rd edn. Springer, London (1995)
15. Jiang, Z.H.: Workspace trajectory control of flexible robot manipulators using neural network and visual sensor feedback. In: IEEE Canadian Conference on Electrical and Computer Engineering, pp. 1502–1507 (2015)
16. Karande, S., Nataraj, P., Gandhi, P., Deshpande, M.M.: Control of parallel flexible five bar manipulator using QFT. In: IEEE International Conference on Industrial Technology, pp. 1–6 (2009)
17. Kim, S.S., Vanderploeg, M.J.: QR decomposition for state space representation of constrained mechanical dynamic systems. J. Mech. Transm. Autom. Des. **108**(2), 183–188 (1986)
18. Kwakernaak, H., Sivan, R.: Linear Optimal Control Systems. Wiley, New York (1972)
19. Kwon, D.S., Book, W.J.: A time-domain inverse dynamic tracking control of a single-link flexible manipulator. J. Dyn. Syst. Meas. Control **116**, 193–200 (1994)
20. Li, D.: Nonlinear control design for tip position tracking of a flexible manipulator arm. Int. J. Control **60**(6), 1065–1082 (1994)
21. Li, Q., Wang, T., Ma, X.: Geometric nonlinear effects on the planar dynamics of a pivoted flexible beam encountering a point-surface impact. Multibody Syst. Dyn. **21**(3), 249–260 (2009)
22. Lochan, K., Roy, B., Subudhi, B.: A review on two-link flexible manipulators. Annu. Rev. Control **42**, 346–367 (2016)
23. Moallem, M., Patel, R.V., Khorasani, K.: Nonlinear tip-position tracking control of a flexible-link manipulator: theory and experiments. Automatica **37**(11), 1825–1834 (2001)

24. Morlock, M., Burkhardt, M., Seifried, R.: Control concepts for a parallel manipulator with flexible links. PAMM **16**(1), 819–820 (2016)
25. Morlock, M., Schröck, C., Burkhardt, M., Seifried, R.: Nonlinear state estimation for trajectory tracking of a flexible parallel manipulator. IFAC-PapersOnLine **50**(1), 3449–3454 (2017)
26. Nowakowski, C., Fehr, J., Fischer, M., Eberhard, P.: Model order reduction in elastic multibody systems using the floating frame of reference formulation. IFAC Proc. Vol. **45**(2), 40–48 (2012)
27. Oakley, C.M., Cannon, R.H.: Anatomy of an experimental two-link flexible manipulator under end-point control. In: IEEE Conference on Decision and Control, pp. 507–513 (1990)
28. Sastry, S.: Nonlinear Systems: Analysis, Stability, and Control. Springer, New York (1999)
29. Sayahkarajy, M., Mohamed, Z., Mohd Faudzi, A.A.: Review of modelling and control of flexible-link manipulators. Proc. Inst. Mech. Eng., Part I, J. Syst. Control Eng. **230**(8), 861–873 (2016)
30. Seifried, R.: Dynamics of Underactuated Multibody Systems. Springer, Switzerland (2014)
31. Seifried, R., Burkhardt, M., Held, A.: Trajectory control of serial and parallel flexible manipulators using model inversion. In: Samin, J., Fisette, P. (eds.) Multibody Dynamics: Computational Methods and Applications. Computational Methods in Applied Sciences, vol. 28, pp. 53–75. Springer, Berlin (2013)
32. Shabana, A.A.: Dynamics of Multibody Systems, 3rd edn. Cambridge University Press, Cambridge (2005)
33. Singer, R.A.: Estimating optimal tracking filter performance for manned maneuvering targets. IEEE Trans. Aerosp. Electron. Syst. **AES-6**(4), 473–483 (1970)
34. Wallrapp, O.: Standardization of flexible body modeling in multibody system codes, part I: definition of standard input data. J. Struct. Mech. **22**(3), 283–304 (1994)
35. Zhang, Q., Mills, J.K., Cleghorn, W.L., Jin, J., Sun, Z.: Dynamic model and input shaping control of a flexible link parallel manipulator considering the exact boundary conditions. Robotica **33**(6), 1201–1230 (2015)
36. Zhao, H., Chen, D.: Tip trajectory tracking for multilink flexible manipulators using stable inversion. J. Guid. Control Dyn. **21**(2), 314–320 (1998)