

A geometric optimization method for the trajectory planning of flexible manipulators

Arthur Lismonde^{*1}, Valentin Sonneville² and Olivier Brûls¹

¹*Department of Aerospace and Mechanical Engineering, University of Liège,
{alismonde,o.bruls}@uliege.be*

²*Aerospace Engineering, University of Maryland, vspsonn@umd.edu*

Abstract

Lightweight and flexible robots offer an interesting answer to industrial needs for safety and efficiency. The control of such systems should be able to deal properly with the flexible behavior in the links and the joints. In this paper, a feedforward control action is computed by solving the inverse dynamics of the system. Flexibility in the system is modeled using finite elements formulated in the local frame. The inverse problem is then solved using a constrained optimization formulation. This local frame representation reduces the nonlinearity in the equations of motion and improves the convergence of the numerical scheme. To illustrate the method, numerical examples of a serial and a parallel 3D robot are shown.

Keywords: inverse dynamics, optimization, flexible robot, trajectory tracking

1 Introduction

Safety and process efficiency are important concerns in the production industry. Compared to yesterday's rigid and bulky robots, lightweight and collaborative automation has a high potential in this context. Indeed, lightweight robots are intrinsically safer for surrounding workers and require less energy to work. Although reliable lightweight robots are already on the market, controlling such lightweight structure is not an easy task and some progress still have to be made.

Robotic manipulators with reduced bulkiness and weight are more prone to vibration and elastic deformation. To control the elastic deformation of the robotic system, its controller should be designed to take the flexibility of the system into account. Flexible systems can also be classified as underactuated systems as they usually have less control inputs than degrees of freedom (dof). In order to have precise end-effector motions, elastic deflections and vibrations of the arm can be monitored and fed back to the controller so that appropriate compensation can be done [14]. Such feedback methods require the use of sensors to measure the states of the system see, e.g., [17]. Based on an analysis of the system, a feedforward component can also be considered to generate a motion with reduced deflections. The input command designed using input shaping methods [30] is able to cancel out vibrations for some selected frequencies. Inverse dynamics methods can also be used to compute feedforward control inputs based on a model of the manipulator. A feedforward input and a feedback loop can be combined to further improve robustness [15, 22, 35].

This work focuses on the off-line computation of the feedforward command of flexible 3D manipulators based on inverse dynamics methods. The tasks to be achieved are the tracking of a specified trajectory, i.e., the end-effector has to follow a trajectory defined over time. The flexible robot is a multibody system (MBS) for which a model is assumed to be available. Based on this model, the feedforward command is computed by solving the inverse dynamics of the flexible MBS. For underactuated manipulators, the system exhibits some internal dynamics that has to be well

^{*}The first author would like to acknowledge the Belgian Fund for Research training in Industry and Agriculture for its financial support (FRIA grant).

managed. This internal dynamics can be unstable and as a consequence, the solution of the inverse dynamics can be unbounded. Such systems are called non-minimum phase systems and require particular inverse dynamics methods. For linear systems, provided that the system has no pole on the imaginary axis, the time domain inverse dynamics method [19] separates the dynamics of the system into a causal and an anticausal part before the inverse dynamics is solved. The causal part leads to some postactuation in the resulting inputs while the anti-causal part leads to some preactuation in the system. For nonlinear systems, the stable inversion method can be used [16]. Its application to underactuated and flexible manipulators is described in [28, 29]. In this method, the inverse dynamics problem is stated as a two-point boundary value problem (BVP) where initial and final conditions have to be met. The final condition, assures that the system ends on a stable manifold i.e., a configuration space where all trajectories converge to a given configuration as time goes forward. On the other side, the initial condition forces the system to start on an unstable manifold i.e., a configuration space where all trajectories converge to a given configuration as time goes backward. This method is able to solve the inverse dynamics either in the form of an ODE or in the form of a differential-algebraic equation (DAE), as shown in [12]. The construction of such boundary values is based on the eigen properties of the system and can be quite cumbersome to compute. In order to avoid the definition of boundary values, an optimization approach was proposed in [3]. This optimization method also leads to a bounded solution which converges to the solution of the two-point boundary value problem when the length of the pre- and post-actuation phases increase [4]. A few numerical methods to solve the inverse dynamics of flexible systems in DAE form are compared in [26]. Eventually, these methods all lead to bounded but non-causal solutions involving a preactuation and a postactuation phase.

The choice of an appropriate modelling approach for the 3D flexible manipulator is also a central question in this paper. Lumped mass models are used in [25, 35] to model flexibility with a limited number of parameters. The assumed modes method used by [8, 24, 31], requires the selection of a limited number of mode shapes determined using a recursive method. The finite element method is a general method that can easily deal with parallel kinematic topologies and nonlinear elastic phenomena at the expense of a higher dimensional model. The formulations of beam and shell finite elements often involve 3D rotation variables which should be properly represented [18, 5]. Here, a geometric approach is selected for the treatment of rotation so that parameterization nonlinearities and singularities can be avoided. More precisely, a finite element formulation on the *special Euclidean* group $SE(3)$ is used [11, 33, 34] so that velocities and strains are represented in the local frame and the geometric nonlinearities are reduced.

In this work, an inverse dynamics solver able to deal with general 3D flexible MBS is presented. The spatial discretization is performed according to the finite element formulation on $SE(3)$ and the inverse dynamics is formulated as a constrained optimization problem [9, 10] where the objective is to minimize a measure of the amplitude of the internal dynamics of the system over the trajectory. The equations of motion of the MBS and the specified end-effector trajectory are considered as constraints of the optimization process. The optimization problem is then discretized in time and solved using a direct transcription method [6]. This results in a large but sparse nonlinear programming problem (NLP) with some variables defined on the Lie group $SE(3)$. The solution of optimization problems on manifolds and Lie groups is addressed in [1, 23]. In this paper, we propose to reformulate the problem in the Lie algebra using incremental displacements with respect to the so-called rigid-trajectory. This leads to a NLP defined in a vector space which can be solved using standard optimization algorithms.

The examples of one serial and one parallel flexible system are shown to illustrate the method.

2 Dynamics of a flexible multibody system

The finite element formalism on the special Euclidean group $SE(3)$ can be used to model the dynamics of flexible MBS including rigid and flexible bodies interconnected by kinematic joints [33, 34]. In this way, the position and orientation of each node I of the finite element mesh is represented as a 4×4 transformation matrix $\mathbf{H}_I$ with a rotation $\mathbf{R}_I \in SO(3)$ and a position

$\mathbf{p}_I \in \mathbb{R}^3$ component

$$\mathbf{H}_I = \begin{bmatrix} \mathbf{R}_I & \mathbf{p}_I \\ \mathbf{0} & 1 \end{bmatrix} \in SE(3)$$

where $I = 1, \dots, N$ and N is the number of nodes of the mesh. The nodal configuration $\mathbf{H}$ can then be represented as a block diagonal matrix that gathers each nodal variable described above

$$\mathbf{H} = \text{diag}(\mathbf{H}_1, \dots, \mathbf{H}_N)$$

The nodal variable $\mathbf{H}_I$ and the corresponding nodal velocity $\mathbf{v}_I = (\mathbf{v}_{I,R}^T, \mathbf{v}_{I,p}^T)^T$ are linked by the *compatibility equation*

$$\dot{\mathbf{H}}_I = \mathbf{H}_I \tilde{\mathbf{v}}_I = \begin{bmatrix} \mathbf{R}_I & \mathbf{p}_I \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{v}}_{I,R} & \mathbf{v}_{I,p} \\ \mathbf{0} & 0 \end{bmatrix} \quad (1)$$

where $\mathbf{v}_{I,R}$ and $\mathbf{v}_{I,p}$ are respectively the absolute angular and translational velocities represented in the local frame. Notice that the $\tilde{\bullet}$ operator maps an element of any Lie group to its corresponding Lie algebra. In the case of a 3 component vector $\mathbf{v}_{I,R}$, the corresponding Lie algebra can be represented by a skew-symmetric matrix

$$\tilde{\mathbf{v}}_{I,R} = \begin{bmatrix} 0 & -\mathbf{v}_{I,R3} & \mathbf{v}_{I,R2} \\ \mathbf{v}_{I,R3} & 0 & -\mathbf{v}_{I,R1} \\ -\mathbf{v}_{I,R2} & \mathbf{v}_{I,R1} & 0 \end{bmatrix}$$

whereas in the case of a 6 component vector $\mathbf{v}_I$, it can be represented by 4×4 matrices, as in Eq. (1). These notations are similar to references [18, 33, 27, 21]. Thanks to r control inputs $\mathbf{u} \in \mathbb{R}^r$, the MBS is driven so that the end effector position $\mathbf{y}_{eff} \in \mathbb{R}^r$ follows a prescribed trajectory $\mathbf{y}_{presc}(t)$. The latter is evolving in time and is defined by r scalar components. In inverse dynamics problems, the internal dynamics is defined as the remaining dynamics when the system outputs are prescribed. It is thus governed by the equations of motion of the MBS supplemented with the *servo* constraints [7] prescribing the end effector trajectory

$$\dot{\mathbf{H}}_I = \mathbf{H}_I \tilde{\mathbf{v}}_I \quad \text{with } I = 1, \dots, N \quad (2)$$

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{g}(\mathbf{H}, \mathbf{v}) + \mathbf{B}^T \boldsymbol{\lambda} = \mathbf{A}\mathbf{u} \quad (3)$$

$$\boldsymbol{\Phi}(\mathbf{H}) = \mathbf{0} \quad (4)$$

$$\mathbf{y}_{eff}(\mathbf{H}) - \mathbf{y}_{presc}(t) = \mathbf{0} \quad (5)$$

where $\mathbf{M}$ is the system symmetric mass matrix, $\mathbf{v} = (\mathbf{v}_1^T, \dots, \mathbf{v}_N^T)^T$ is the vector of nodal velocities, $\mathbf{g}$ is the vector of internal and complementary inertia forces, $\mathbf{B}$ is the gradient of the kinematic constraints $\boldsymbol{\Phi}$, which are used to represent the connections imposed by the kinematic joints. The matrix $\mathbf{A}$ is a matrix that applies the controls $\mathbf{u}$ on the system. The m dimensional vector $\boldsymbol{\lambda}$ is the Lagrange multipliers related to the m kinematic constraints $\boldsymbol{\Phi}$. The last equation is called the servo constraint and fixes a part of the motion. It assures that the end-effector position $\mathbf{y}_{eff}$ follows the prescribed trajectory $\mathbf{y}_{presc}(t)$.

This paper focuses on dynamic systems such that the matrix

$$\begin{bmatrix} \mathbf{B} \\ \mathbf{D} \end{bmatrix} \mathbf{M}^{-1} [\mathbf{B}^T - \mathbf{A}] \quad (6)$$

where $\mathbf{D}$ is the gradient of the servo constraints, is nonsingular, which means that the relative degree is equal to 2 for each output i.e., systems described by an index 3 DAE. This condition comes from [4], as it was a necessary condition to prove the convergence of the BVP formulation to the optimization formulation. Then, the system is underactuated if $6N - m - r > 0$ which is the dimension of the internal dynamics which is represented by Eqs. (2)-(5). The trajectory would be completely specified if some initial conditions were provided for the internal dynamics. However, if the internal dynamics is unstable, the forward propagation of the initial conditions would lead to an unbounded solution requiring very large control efforts $\mathbf{u}$. A bounded solution can be defined using an optimization formulation, in which the initial conditions on the dynamics are left free.

3 Optimization problem

The inverse dynamics problem is formulated as a constrained optimization problem where a measure of the amplitude of the internal dynamics over the trajectory is to be minimized. Considering that such a measure can be represented based on an instantaneous function $\mathbb{V}(\mathbf{H}(t))$ depending on the nodal configurations, the optimization problem is the minimization of the objective function J on the time lap $T = t_f - t_i$.

Mathematically,

$$\min_{\mathbf{H}} J = \min_{\mathbf{H}} \frac{1}{T} \int_{t_i}^{t_f} \mathbb{V}(\mathbf{H}(t)) dt \quad (7)$$

subjected to the equality constraints $\mathbf{c}$ defined by the equation of motion of the flexible MBS Eqs. (2)-(5) for $t \in [t_i, t_f]$. As mentioned in the introduction, one may observe that, in this fomulation, no initial and final values of $\mathbf{H}$ and $\mathbf{v}$ are defined. They are determined by the optimization algorithm itself.

An additional assumption is now introduced. For that purpose, an equivalent rigid system is first defined by replacing every compliant body by a rigid body with the same total mass, rotational inertia and geometry as in the undeformed configuration. We assume that the rigid system is fully actuated and has no internal dynamics. This means that the number of actuators r is equal to the number of dofs of the rigid system and that, when the output trajectory is specified, the rigid trajectory is completely determined. The internal dynamics of the flexible systems is thus interpreted as a purely elastic motion about the rigid trajectory. For that reason, the function $\mathbb{V}(\mathbf{H}(t))$ can be defined either as the strain energy of the system or based on the strain measures of the finite element model. For example, if the flexibility in the system is modeled thanks to beam elements formulated on the $SE(3)$ group [34] and that each beam element is defined between two end nodes, denoted as A and B , $\mathbb{V}(\mathbf{H}(t))$ can simply be defined as

$$\mathbb{V}(\mathbf{H}(t)) = \frac{1}{2} \sum_{j=1}^{n_{el}} L_j \boldsymbol{\epsilon}_j^T \mathbf{K}_j \boldsymbol{\epsilon}_j \quad (8)$$

where n_{el} is the number of beam elements, L_j is the length of the beam element and $\mathbf{K}_j$ is the beam element stiffness matrix. The strains $\boldsymbol{\epsilon}_j$ are defined using the 6×1 *relative configuration vector* $\mathbf{h}_{AB}$

$$\mathbf{h}_{AB} = \log_{SE(3)}(\mathbf{H}_A^{-1} \mathbf{H}_B) \quad (9)$$

where $\mathbf{H}_A$ and $\mathbf{H}_B$ are the nodal variables of the two end nodes A and B respectively. The *logarithmic* mapping $\log_{SE(3)}(\bullet)$ is used to represent the matrix $\mathbf{H}_A^{-1} \mathbf{H}_B$, which represents the relative motion between nodes A and B , using a 6×1 vector $\mathbf{h}_{AB}$. The strains $\boldsymbol{\epsilon}_j$ are then evaluated by comparing $\mathbf{h}_{AB}$ in the current configuration and $\mathbf{h}_{AB}^0$ in the reference undeformed configuration

$$\boldsymbol{\epsilon}_j = \frac{\mathbf{h}_{AB} - \mathbf{h}_{AB}^0}{L_j} \quad (10)$$

4 Direct transcription method

4.1 Time discretization

The optimization is carried out using a direct transcription method i.e., the time interval is first discretized in s time steps t^k ($k = 1, \dots, s$) so that the optimal control problem is reformulated as a discrete Nonlinear Programming (NLP) problem.

In the discrete settings, the minimization of the objective function, previously given by Eq. (7), can therefore be written in its discrete form

$$J = \frac{1}{T} \sum_{k=1}^s \mathbb{V}(\mathbf{H}^k) h \quad (11)$$

subjected to the discrete constraints at each time step t^k

$$\dot{\mathbf{H}}_I^k - \mathbf{H}_I^k \tilde{\mathbf{v}}_I^k = \mathbf{0} \quad (12)$$

$$\mathbf{M}^k \dot{\mathbf{v}}^k + \mathbf{g}(\mathbf{H}^k, \mathbf{v}^k) + \mathbf{B}^{k,T} \boldsymbol{\lambda}^k - \mathbf{A} \mathbf{u}^k = \mathbf{0} \quad (13)$$

$$\Phi(\mathbf{H}^k) = \mathbf{0} \quad (14)$$

$$\mathbf{y}_{eff}(\mathbf{H}^k) - \mathbf{y}_{presc}(t^k) = \mathbf{0} \quad (15)$$

where h is the time step size, $I = 1, \dots, N$ and $k = 1, \dots, s$. Additional time integration constraints are required to connect the discrete nodal configurations, velocities and accelerations.

In the present work, the Lie group generalized- α time integration scheme [13] is used. As it is able to deal with the equations of motion formulated in the Lie group $SE(3)$. For each time step t^k ,

$$\Delta \mathbf{Q}^k - h \mathbf{v}^k - \left(\frac{1}{2} - \beta\right) h^2 \mathbf{a}^k - \beta h^2 \mathbf{a}^{k+1} = \mathbf{0} \quad (16)$$

$$\mathbf{v}^{k+1} - \mathbf{v}^k - (1 - \gamma) h \mathbf{a}^k - \gamma h \mathbf{a}^{k+1} = \mathbf{0} \quad (17)$$

$$(1 - \alpha_m) \mathbf{a}^{k+1} + \alpha_m \mathbf{a}^k - (1 - \alpha_f) \dot{\mathbf{v}}^{k+1} - \alpha_f \dot{\mathbf{v}}^k = \mathbf{0} \quad (18)$$

where $\mathbf{a}^k$ can be seen as an estimation of the actual acceleration, β , γ , α_m and α_f are parameters that define the numerical behavior of the method.

Vector $\Delta \mathbf{Q}^{k,T} = (\Delta \mathbf{Q}_1^{k,T}, \dots, \Delta \mathbf{Q}_N^{k,T})$ is a time incremental variable representing the change of the nodal variables between two consecutive time steps:

$$\mathbf{H}_I^{k+1} = \mathbf{H}_I^k \exp_{SE(3)}(\widetilde{\Delta \mathbf{Q}_I^k}) \quad (19)$$

with $I = 1, \dots, N$. The exponential map $\exp_{SE(3)}(\bullet)$ maps a 6×1 vector to an element of $SE(3)$. The vector $\Delta \mathbf{Q}_I^k$ represent the incremental motion of node I between steps k and $k + 1$.

After discretization, the unknown variables of the optimization problem are

$$(\mathbf{H}^1, \mathbf{v}^1, \dot{\mathbf{v}}^1, \mathbf{a}^1, \boldsymbol{\lambda}^1, \mathbf{u}^1, \dots, \mathbf{H}^s, \mathbf{v}^s, \dot{\mathbf{v}}^s, \mathbf{a}^s, \boldsymbol{\lambda}^s, \mathbf{u}^s)$$

with $\mathbf{H}^k = \text{diag}(\mathbf{H}_1^k, \dots, \mathbf{H}_N^k)$ and each $\mathbf{H}_I^k \in SE(3)$. Obviously, some components of this set of variables belong to the non-linear $SE(3)$ group, for $k = 1, \dots, s$. In order to solve this problem using classical techniques, meaning techniques that deal with variables defined on a linear space, a reformulation on the tangential space of the Lie group based on incremental configuration variables is thus proposed.

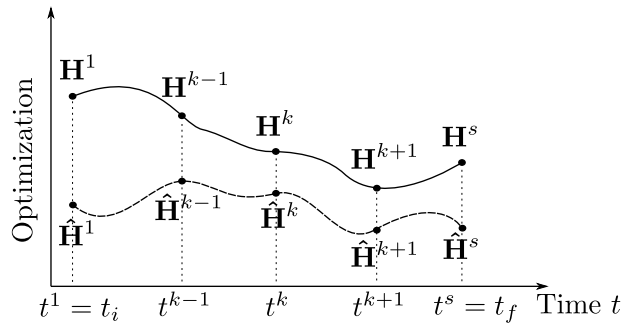


Figure 1: Direct transcription method: optimization starting from the initial guess $(\hat{\mathbf{H}}^1, \dots, \hat{\mathbf{H}}^s)$ that leads to the optimal trajectory $(\mathbf{H}^1, \dots, \mathbf{H}^s)$.

4.2 Incremental formulation of the NLP problem

As illustrated in Fig. 1, we propose to represent the trajectory in terms of increments with respect to the rigid trajectory. As the equivalent rigid system is fully actuated, the rigid trajectory, denoted

as $\hat{\mathbf{H}}(t)$, can be evaluated using standard inverse dynamics methods. Then, we introduce the vector of iteration incremental variables $\Delta \mathbf{q}^T = (\Delta \mathbf{q}_1^T, \dots, \Delta \mathbf{q}_N^T)$, which determines the change between the rigid trajectory $\hat{\mathbf{H}}$ and the current value $\mathbf{H}$.

At time step k , the relation between the configuration $\mathbf{H}^k = \text{diag}(\mathbf{H}_1^k, \dots, \mathbf{H}_N^k)$ and the iteration incremental variables $\Delta \mathbf{q}^{k,T} = (\Delta \mathbf{q}_1^{k,T}, \dots, \Delta \mathbf{q}_N^{k,T})$ is

$$\mathbf{H}_I^k = \hat{\mathbf{H}}_I^k \exp_{SE(3)}(\widetilde{\Delta \mathbf{q}_I^k}) \quad (20)$$

where $\hat{\mathbf{H}}_I^k$ represents the position and orientation of node I at time step k for the initial guess.

It is important to notice the difference between Eq. (19) and Eq. (20). The former relates the nodal configuration of the system at the same iteration but at two consecutive time steps, i.e. $\mathbf{H}^k = \text{diag}(\mathbf{H}_1^k, \dots, \mathbf{H}_N^k)$ and $\mathbf{H}^{k+1} = \text{diag}(\mathbf{H}_1^{k+1}, \dots, \mathbf{H}_N^{k+1})$. The latter relates the nodal configuration of the system before and after optimization at a given time step, i.e. $\hat{\mathbf{H}}^k = \text{diag}(\hat{\mathbf{H}}_1^k, \dots, \hat{\mathbf{H}}_N^k)$ and $\mathbf{H}^k = \text{diag}(\mathbf{H}_1^k, \dots, \mathbf{H}_N^k)$. The relation between the relevant variables is illustrated in Fig. 2. Each arrow represents an exponential mapping $\exp_{SE(3)}(\tilde{\bullet})$ with either time incremental or iteration incremental arguments, i.e. $\Delta \mathbf{Q}^k$ and $\Delta \mathbf{q}^k$ respectively.

The actual design variables $\mathbf{x}$ are thus

$$\mathbf{x} = (\Delta \mathbf{q}^1, \mathbf{v}^1, \dot{\mathbf{v}}^1, \mathbf{a}^1, \boldsymbol{\lambda}^1, \mathbf{u}^1, \dots, \Delta \mathbf{q}^s, \mathbf{v}^s, \dot{\mathbf{v}}^s, \mathbf{a}^s, \boldsymbol{\lambda}^s, \mathbf{u}^s)$$

The optimization problem has now vectorial design variables and can be solved using a classical NLP algorithm. The initial guess for the design variables is the rigid trajectory. This means that velocities, accelerations, Lagrange multipliers and control inputs values are initialized with the rigid trajectory values and since, initially, there is no elastic deformation, the initial guess for the incremental variables $\Delta \mathbf{q}$ is zero.

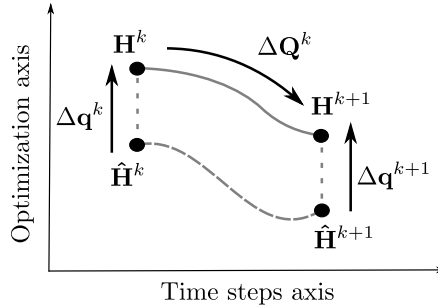


Figure 2: Relation between variables $\Delta \mathbf{q}^k$, $\Delta \mathbf{Q}^k$ and $\mathbf{H}^k$.

4.3 Optimization algorithm

The combination of the finite element formalism and the direct transcription method leads to a large optimization problem with a high number of design variables and constraints. However, the problem is sparse as coupling terms in the discrete constraints are limited to consecutive time steps (see Eq. (16)-(18)) and nodes directly connected in the finite element mesh. Some nonlinear optimization solvers, such as KNITRO, IPOPT and FMINCON, were tested with appropriate options and algorithms to deal with large scale and sparse problems. Eventually, the *interior point* algorithm in the FMINCON solver of Matlab[®] seemed to perform more efficiently for the present test cases and was used to solve the inverse dynamics problem. The tolerance for the satisfaction of the constraints were fixed to $tol = 10^{-6}$ in order to guarantee that both the servo-constraints and the holonomic constraints are satisfied with a sufficient level of accuracy by the numerical solution.

For computational efficiency, the gradients of the objective function and the constraint equations are computed analytically and provided to the solver.

The gradient of the objective function (11) is

$$\nabla J = \frac{1}{T} \sum_{k=0}^s \left[\sum_{j=1}^{n_{el}} L_j \epsilon_j^{k,T} \mathbf{K}_j \frac{\partial \epsilon_j^k}{\partial \Delta \mathbf{q}^k} \right] h \quad (21)$$

and the coupling terms in $\frac{\partial \epsilon_j}{\partial \Delta \mathbf{q}}$ are limited to the pair of nodes connected by beam elements, which further reinforces the sparsity pattern. In this case, the derivative of the strains with respect to the optimization variable $\Delta \mathbf{q}$ given by the relation

$$\frac{\partial \epsilon_j^k}{\partial \Delta \mathbf{q}^k} = \left[-\mathbf{T}^{-1}(\tilde{\mathbf{h}}_{AB}^k) \mathbf{T}(\Delta \mathbf{q}_A^k) ; \mathbf{T}^{-1}(\tilde{\mathbf{h}}_{AB}^k) \mathbf{T}(\Delta \mathbf{q}_B^k) \right] \quad (22)$$

where the operators $\mathbf{T}^{-1}(\bullet)$ and $\mathbf{T}(\bullet)$ represent the differentiation of the logarithmic map $\log_{SE(3)}(\bullet)$ in Eq. (9) and the exponential map $\exp_{SE(3)}(\bullet)$ in Eq. (20) [33]. The Hessian matrix is computed numerically by the solver using the *lbfgs* quasi-Newton approximation.

Regarding the optimization constraints $\mathbf{c}$, they will be denoted as $\mathbf{c}_m$ and $\mathbf{c}_\alpha$ for the equation of motion (13)-(15) and the time integration equations (16)-(18) respectively. At each time step t^k , these constraints only involve a subset $\mathbf{x}^k = (\Delta \mathbf{q}^k, \mathbf{v}^k, \dot{\mathbf{v}}^k, \mathbf{a}^k, \boldsymbol{\lambda}^k, \mathbf{u}^k)$ of the design variables $\mathbf{x}$. We have

$$\frac{\partial \mathbf{c}_m^k}{\partial \mathbf{x}^k} = \begin{bmatrix} \mathbf{K}^k \mathbf{T}^k(\Delta \mathbf{q}^k) & \mathbf{C}^k & \mathbf{M}^k & \mathbf{0} & \mathbf{B}^{k,T} & -\mathbf{A} \\ \mathbf{B}^k & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{D}^k & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (23)$$

where $\mathbf{K}$ and $\mathbf{C}$ are the tangent stiffness and damping matrix respectively and $\mathbf{D}$ is the gradient of the servo constraints,

$$\frac{\partial \mathbf{c}_\alpha^k}{\partial \mathbf{x}^k} = \begin{bmatrix} -\mathbf{T}^k(\Delta \mathbf{Q}^k) & -h\mathbf{I} & \mathbf{0} & -(\frac{1}{2} - \beta)h^2\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{I} & \mathbf{0} & -(1 - \gamma)h\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\alpha_f\mathbf{I} & \alpha_m\mathbf{I} & \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (24)$$

and

$$\frac{\partial \mathbf{c}_\alpha^k}{\partial \mathbf{x}^{k+1}} = \begin{bmatrix} \mathbf{T}^{*k}(\Delta \mathbf{Q}^k) & \mathbf{0} & \mathbf{0} & -\beta h^2\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & -\gamma h\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -(1 - \alpha_f)\mathbf{I} & (1 - \alpha_m)\mathbf{I} & \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (25)$$

where $\mathbf{I}$ is the identity matrix, the matrix $\mathbf{T}^k$ and $\mathbf{T}^{*k}$ are block diagonal matrices which appear after differentiating Eq. (14) and Eq. (19) that have underlying exponential maps $\exp_{SE(3)}(\bullet)$.

Finally, the resulting constraint gradient matrix $\frac{\partial \mathbf{c}}{\partial \mathbf{x}}$ is made up of these matrices at each time step, stacked on each other to build a block diagonal sparse matrix

$$\frac{\partial \mathbf{c}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial \mathbf{c}_m^1}{\partial \mathbf{x}^1} & \mathbf{0} & \dots & \mathbf{0} \\ \frac{\partial \mathbf{c}_\alpha^1}{\partial \mathbf{x}^1} & \frac{\partial \mathbf{c}_\alpha^1}{\partial \mathbf{x}^1} & & \\ \mathbf{0} & \frac{\partial \mathbf{c}_m^2}{\partial \mathbf{x}^2} & & \\ & \frac{\partial \mathbf{c}_\alpha^2}{\partial \mathbf{x}^2} & \frac{\partial \mathbf{c}_\alpha^2}{\partial \mathbf{x}^3} & \\ & & \ddots & \\ \vdots & & & \frac{\partial \mathbf{c}_m^{s-1}}{\partial \mathbf{x}^{s-1}} & \mathbf{0} \\ & & & \frac{\partial \mathbf{c}_\alpha^{s-1}}{\partial \mathbf{x}^{s-1}} & \frac{\partial \mathbf{c}_\alpha^{s-1}}{\partial \mathbf{x}^s} \\ \mathbf{0} & \dots & \mathbf{0} & \frac{\partial \mathbf{c}_m^s}{\partial \mathbf{x}^s} \end{bmatrix} \quad (26)$$

It is important to note that, although the tangent stiffness $\mathbf{K}^k$ matrices and tangent damping $\mathbf{C}^k$ matrices are nonlinear and should be re-evaluated for the constraint gradient at each iteration, the optimization process may still converge if they are kept constant over the iterations. Indeed, thanks to the local representation coming from the use of the $SE(3)$ formalism, these matrices do not vary much through the optimization process. Hence, for simple systems, this assumption could lead to improved computational time.

However, in the more complex systems studied here, using constant $\mathbf{K}^k$ and $\mathbf{C}^k$ in the constraint gradient leads to more iterations of the optimization and an increased computational time. In the following examples, the whole constraint gradient is thus computed at every iteration.

5 3D serial robot

The inverse dynamics of a flexible 3D system is now solved using the proposed approach. A serial 3 dof manipulator, as shown in Fig. 3, is first considered [20]. It is composed of two links: an upper arm and a forearm. The end-effector is modeled as a point mass m_{end} at the tip of the forearm. The upper arm and the forearm both have length l and a tubular square cross section. The upper arm has a side length a_1 and an edge thickness e_1 . The forearm has a side length a_2 and an edge thickness e_2 . While the former has a greater cross section and is considered as a rigid body element, the forearm is considered flexible and is modeled using 4 beam elements. The description of the beam formulation on $SE(3)$ can be found in [34]. The upper arm connects the first two hinge joints, controlled using input torques u_1 and u_2 , to the third one, controlled using input torque u_3 . The outputs of the system are the x , y and z components of the end-effector position $\mathbf{y}_{eff}$. The first hinge joint has its axis along axis z . The second and the third hinge joints initially have their axis along axis y . In the initial position, each link makes a 45° angle with respect to the x axis.

The trajectory the end-effector has to follow is a planar circular arc in the yz plane. The motion profile is built using a seventh order polynomial in order to insure continuity of the position, velocities, accelerations and jerks over time. The end-effector starts from position $[2l \cos(\frac{\pi}{4}) \ 0 \ 0]$ and goes to position $[2l \cos(\frac{\pi}{4}) \ l \ 0]$. The radius of the circular arc is thus $l/2$. The trajectory is covered in 1.1 s and the pre- and post-actuation phases both last 0.2 s: the total simulation time is 1.5 s. The material parameters and dimensions of the 3D flexible arm can be found in Table 1.

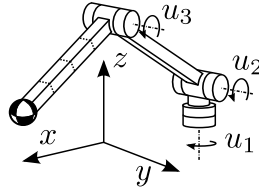


Figure 3: Serial 3D arm system with one rigid body and 4 beam elements.

Table 1: Parameters of the serial arm system.

Upper arm	$l = 1 \text{ m}$	$a_1 = 0.05 \text{ m}$	$e_1 = 0.01 \text{ m}$ $\rho = 2700 \text{ kg/m}^3$
Forearm	$l = 1 \text{ m}$ $E = 70 \text{ GPa}$	$a_2 = 0.0075 \text{ m}$ $\nu = 0.3$	$e_2 = 0.0015 \text{ m}$ $\rho = 2700 \text{ kg/m}^3$
End-eff.	$m_{end} = 0.1 \text{ kg}$		

The convergence of the optimization process is quite sensitive to the initial guess of the problem. To compute it, a complete rigid system is considered. The beams of the forearm are replaced with a rigid body with the same geometrical and material properties. The numerical parameters of the model are set to capture the dynamics of the system with a bandwidth of 100 Hz: in the initial position, the first seven modes of the flexible structure are within this frequency and the first unstable pole of the inverse dynamics problem is located at the frequency of 13 Hz. Therefore, the system is discretized into $s = 300$ steps, leading to a time step size h of 0.005 s (200 Hz), which is twice the bandwidth and more than ten times the first unstable frequency. For the generalized- α method, a spectral radius of $\rho_\infty = 0$ is considered ($\beta = 1$, $\gamma = 1.5$, $\alpha_m = -1$ and $\alpha_f = 0$) to eliminate higher frequency content.

With the tolerance $tol = 10^{-6}$, the optimization is completed after 5 iterations and lasts 10 minutes (using a x64 bits i7-4600u CPU with 16 Gb RAM memory). The command inputs $\mathbf{u}$ and $\mathbf{u}_{rigid}$, with and without flexibility considerations respectively, are compared in Fig. 4. Some visible differences can be observed but the pre- and post-actuation in the input commands are hardly

visible. These pre- and post- actuation phases can be observed by looking at the acceleration profile of the three joints in Fig. 5. One can see that after 1.3 s, the acceleration of the third joint is still varying. Although the torque u_3 is nearly equal to the gravity compensation torque in the post-actuation phase, it still results in some internal motion in the arm. It is important to point out that although some motion is present in the joints, the end-effector does not actually move during the post-actuation.

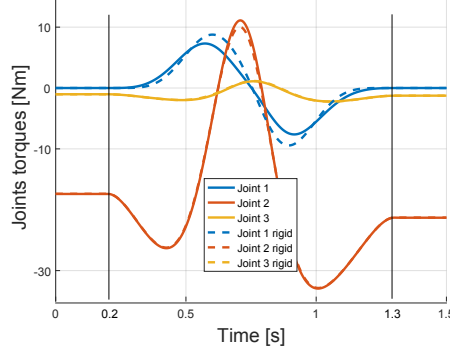


Figure 4: Input commands of a flexible 3D arm.

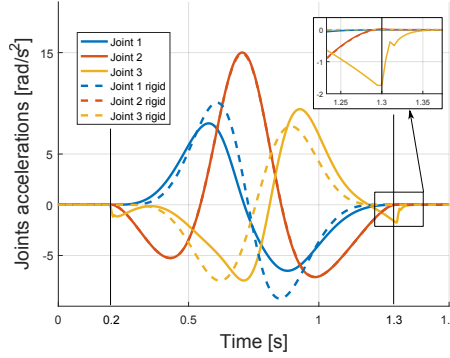


Figure 5: Acceleration of the three joint of a flexible 3D arm.

In order to verify the computed optimal inputs $\mathbf{u}$, both inputs $\mathbf{u}$ and $\mathbf{u}_{rigid}$ are applied to the flexible system and a direct dynamic analysis is performed. These inputs lead to end-effector trajectories shown in Fig. 6. Since the trajectory should be circular, the radius of these resulting trajectories are shown in Fig. 7. One can observe that the rigid input $\mathbf{u}_{rigid}$ leads to some oscillation of the end-effector around the nominal circular trajectory. These oscillations are strongly reduced when flexibility is considered. The relative tracking error e^k resulting from both direct simulations can be calculated at each time step t^k using

$$e^k = \frac{\|\mathbf{y}_{presc} - \mathbf{y}_{eff}\|}{\|\mathbf{y}_{presc}\|} \quad (27)$$

where $\|\bullet\|$ is the classical Euclidean norm or L_2 norm. The relative rms error is then calculated as

$$e_{rms} = \sqrt{\frac{1}{s} \sum_{k=1}^s (e^k)^2} \quad (28)$$

The relative rms error e_{rms} is equal to 10^{-2} when $\mathbf{u}_{rigid}$ is used as input and drops down to $e_{rms} = 2 \times 10^{-5}$ when $\mathbf{u}$ is used with $tol = 10^{-6}$. In this second case, the remaining error is only caused by finite numerical tolerances that are used by the algorithm. For example, if the tolerance is reduced to $tol = 10^{-8}$, this error becomes $e_{rms} = 3.3 \times 10^{-7}$.

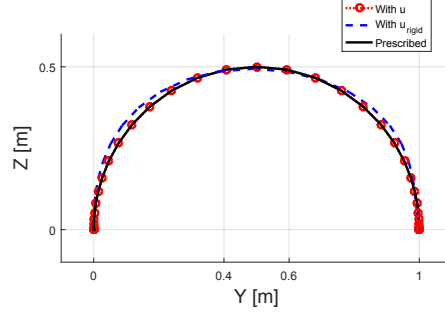


Figure 6: End-effector trajectory of a flexible 3D arm using $\mathbf{u}$ and $\mathbf{u}_{rigid}$.

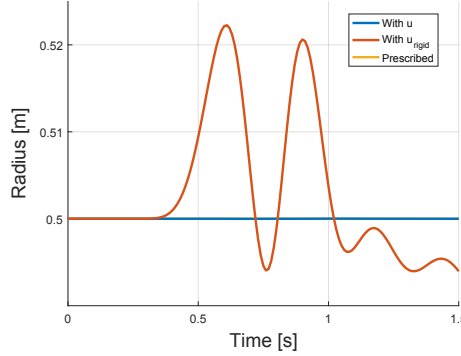


Figure 7: Radius of the resulting trajectory compared to the desired trajectory radius of $l/2 = 0.5\text{m}$.

In this example, the end-effector is directly connected to the flexible component of the system: the second link. If the first link was considered flexible and the second was considered rigid instead, there would be no difference in the way the present method handles the problem. However, it can be noted that in that case, the resulting inverse dynamics problem has its lowest unstable pole located above the bandwidth set by the numerical parameters. Therefore, the inverse dynamics can be solved numerically without needing stable inversion techniques.

6 3D parallel robot

This optimization method is now applied to a manipulator with a closed chain kinematics: a 3 dof Delta-like robot. It is made of an upper plate and a lower plate, both are equilateral triangles but the upper one has a side length l_{up} whereas the lower one is smaller and has a side length l_{low} . The point mass end-effector is located at the center of the lower plate as shown in Fig. 8. Three actuated hinge joints are located at each corner of the upper plate and have their axis parallel to the opposite side of the triangle. They are controlled using the input torques u_1 , u_2 and u_3 . To control the end-effector, these hinge joints are connected to the lower plate using three arms. Each one is made of two links, an upper arm and a forearm, and has two passive universal joints to connect them. The upper arms are square cross section tubes considered rigid and have dimensions $a_1 \times a_1 \times l_1$ and thickness e_1 . The forearms are considered flexible with dimensions $a_2 \times a_2 \times l_2$ and thickness e_2 and are represented using 2 beam finite elements. In the initial configuration, the center of gravity of each plate is coincident with the z axis and the three upper arms are horizontal. The initial vertical distance between the upper plate and the end-effector is z_{init} . The total mass of the the lower plate and the end-effector is m_{end} . The trajectory to be followed is a planar circular arc in the horizontal xy plane to be covered in 0.6 s; the duration of the pre- and post-actuation phases is fixed to 0.2 s. As for the serial example, the motion profile is constructed using a seventh order polynomial. The trajectory starts from position $[0 \ 0 \ -z_{init}]$ (the origin of the reference frame is located at the center of the upper plate) and stops at position $[0 \ 2l_{low} \ -z_{init}]$.

The geometrical and material parameters of the parallel manipulator are given in Table 2.

Table 2: Parameters of the parallel arm system.

Upper and lower plates	$l_{up} = 0.25$ m	$l_{low} = 0.1$ m	
Upper arms	$l_1 = 0.25$ m	$a_1 = 0.02$ m	$e_1 = 0.005$ m $\rho = 2700$ kg/m ³
Forearms	$l_2 = 0.5825$ m	$a_2 = 0.005$ m	$e_1 = 0.0015$ m $E = 70$ GPa $\nu = 0.3$ $\rho = 2700$ kg/m ³
End-effector	$z_{init} = 0.7$ m	$m_{end} = 0.1$ kg	

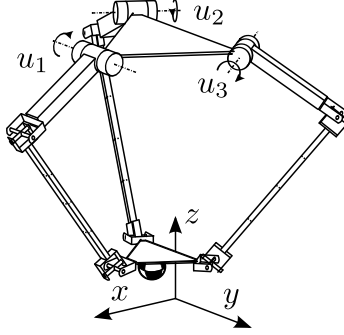


Figure 8: Parallel 3D arm system with 3 rigid bodies and 6 beam elements.

As for the previous example, the initial guess for this parallel system is the solution of the completely rigid system. The forearm is modeled using only rigid bodies. A spectral radius of $\rho_\infty = 0$ is considered and the system is divided into $s = 200$ steps. The optimization is completed after 6 iterations and lasts 30 minutes. The resulting accelerations in the joints is shown in Fig. 9. As in the previous example, a post actuation phase appears clearly when we zoom in at the end of the trajectory. Again, when these input commands are used for the direct dynamics problem, the improvements done on the tracking performance is visible (Fig. 10). When the rigid controls $\mathbf{u}_{rigid}$ are used, the radius of the end-effector trajectory tends to drift away from the nominal radius (Fig. 11). Again, this drift can be compensated for when flexibility is considered. The relative rms error e_{rms} drops from 5×10^{-2} to 4×10^{-4} when $\mathbf{u}$ is used. In this example the first eigen frequency is 15 Hz, meaning that the time step size of 0.005 s is also lower than a tenth of the first eigen frequency.

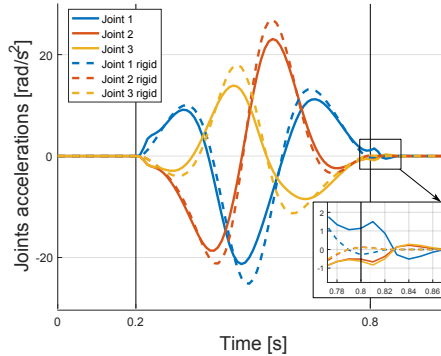


Figure 9: Joints accelerations of a flexible parallel 3D arm.

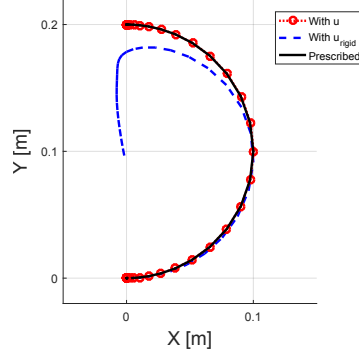


Figure 10: End-effector trajectory of a flexible parallel 3D arm using $\mathbf{u}$ and $\mathbf{u}_{rigid}$.

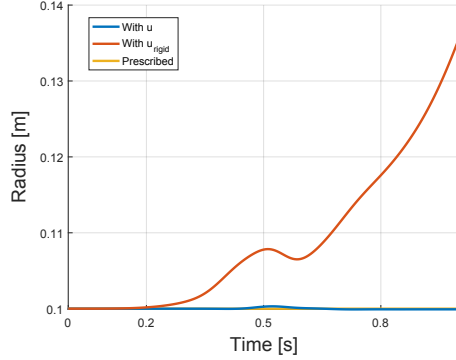


Figure 11: Radius of the resulting trajectory compared to the desired trajectory radius of $l_{low} = 0.1\text{m}$.

7 Conclusion

This paper presents a method to solve the inverse dynamics of flexible 3D MBS. The MBS is first modeled using a finite element approach with a special euclidean group $SE(3)$ formalism. The inverse dynamics problem is then formulated as a constrained optimization problem where the objective is to minimize the mean strain energy of the system, which gives some measure of the internal dynamics. The direct transcription method is used to discretize the optimization problem in time and additional time constraints have to be added. To improve computational efficiency, the gradients of the objective function and the constraints equations are provided to the solver. Thanks to the local frame representation that follows the $SE(3)$ formalism, the equations of motion can be represented with reduced non-linearity. 3D rotations can also be represented without singularities. The gradients of the optimization problem can benefit from these advantages. In order to use classical NLP solvers, configuration $\Delta \mathbf{q}^k$ and time $\Delta \mathbf{Q}^k$ incremental variables that belong to a vectorial linear space are introduced to avoid design variables defined on the non-linear space $SE(3)$. After optimization, the computed input commands manage to improve tracking precision for serial and parallel 3D systems.

The computing time is of the order of several minutes for the considered applications. Therefore, the method is currently intended for off-line evaluation of the input commands. There would clearly be an interest in improving the computational cost and further numerical improvements could widen the range of applications. One could try to reduce the dimension of the optimization problem using reduced order models based on superelement formulation [32]. Another way could be to use a multiple shooting method that combines forward time integration and optimization procedures [2]. Another perspective is to carry out tests on an experimental testbed for experimental validation, which would require the design of an additional feedback loop. The influence of imperfections in the joints, such as friction, that are present in any real applications, might also be investigated.

References

- [1] Absil, P.A., Mahony, R., Sepulchre, R.: Optimization algorithms on Matrix Manifolds. Princeton University Press (2008)
- [2] Bastos, G.J.: Contribution to the inverse dynamics of flexible manipulators. Ph.D. thesis, University of Liège (2013)
- [3] Bastos, G.J., Seifried, R., Brüls, O.: Inverse dynamics of serial and parallel underactuated multibody systems using a DAE optimal control approach. *Multibody System Dynamics* **30**, 359–376 (2013). DOI 10.1007/s11044-013-9361-z
- [4] Bastos, G.J., Seifried, R., Brüls, O.: Analysis of stable model inversion methods for constrained underactuated mechanical systems. *Mechanism and Machine Theory* **111**, 99–117 (2017)
- [5] Bauchau, O.: Flexible multibody dynamics. Springer (2011)
- [6] Betts, J.T.: Practical Method for Optimal Control and Estimation Using Nonlinear Programming. Advances in Design and Control. SIAM (2010)
- [7] Blajer, W., Kolodziejczyk, K.: A geometric approach to solving problems of control constraints: Theory and a DAE framework. *Multibody System Dynamics* **11**, 343 – 364 (2004)
- [8] Book, W.J.: Recursive Lagrangian dynamics of flexible manipulator arms. *The International Journal of Robotics Research* **3**(3), 87–101 (1984)
- [9] Bottasso, C.L., Croce, A.: Optimal control of multibody systems using an energy preserving direct transcription method. *Multibody System Dynamics* **12**, 17–45 (2004)
- [10] Bottasso, C.L., Croce, A., Ghezzi, L., Faure, P.: On the solution of inverse dynamics and trajectory optimization problems for multibody systems. *Multibody System Dynamics* **11**, 1–22 (2004)
- [11] Brüls, O., Arnold, M., Cardona, A.: Two Lie group formulations for dynamic multibody systems with large rotations. In *Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference* (2011). Washington D.C., U.S.
- [12] Brüls, O., Bastos, G.J., Seifried, R.: A stable inversion method for feedforward control of constrained flexible multibody systems. *Journal of Computational and Nonlinear Dynamics* **9** (2014). DOI 10.1115/1.4025476
- [13] Brüls, O., Cardona, A., Arnold, M.: Lie group generalized- α time integration of constrained flexible multibody systems. *Mechanism and Machine Theory* **48**, 121–137 (2012)
- [14] Cannon, R., Schmitz, E.: Initial experiments on the end-point control of a flexible one-link robot. *The International Journal of Robotics Research* **3**(3), 62–75 (1984)
- [15] De Luca, A.: Feedforward/feedback laws for the control of flexible robots. In *Proceedings of the IEEE International Conference on Robotics & Automation* (2000)
- [16] Devasia, S., Chen, D., Paden, B.: Nonlinear inversion-based output tracking. *IEEE Transactions on Automatic Control* **41**(7) (1996)
- [17] Franke, R., Malzahn, J., Nierobisch, T., Hoffmann, F., Bertram, T.: Vibration control of a multi-link flexible robot arm with fiber-bragg-grating sensors. In *Proceedings of IEEE International Conference on Robotics and Automation* (2009)
- [18] G’eradin, M., Cardona, A.: Flexible multibody dynamics : a finite element approach. John Wiley (2001)

- [19] Kwon, D.S., Book, W.J.: A time-domain inverse dynamic tracking control of a single link flexible manipulator. *Journal of Dynamic Systems, Measurement and Control* **116**, 193–200 (1994)
- [20] Lismonde, A., Sonnevile, V., Bröls, O.: Trajectory planning of soft link robots with improved intrinsic safety. In *Proceedings of the 20th World Congress of the International Federation of Automatic Control* (2017)
- [21] Lynch, K.M., Park, F.C.: *Modern robotics : mechanics, planning, and control*. Cambridge University Press (2017)
- [22] Malzahn, J., Ruderman, M., Phung, A.S., Hoffmann, F., Bertram, T.: Input shaping and strain gauge feedback vibration control of an elastic robotic arm. In *Proceedings of IEEE Conference on Control and Fault Tolerant Systems* (2010)
- [23] Manara, S., Gabiccini, M., Artoni, A., Diehl, M.: On the integration of singularity-free representations of $SO(3)$ for direct optimal control. *Nonlinear Dynamics* **90**(2), 1223–1241 (2017)
- [24] Martins, J., Botto, M.A., Costa, J.S.D.: Modeling of flexible beams for robotic manipulators. *Multibody System Dynamics* **7**, 79–100 (2002)
- [25] Moberg, S.: *Modeling and control of flexible manipulators*. Ph.D. thesis, Linköping University (2010)
- [26] Moberg, S., Hanssen, S.: Inverse dynamics of flexible manipulators. In *Proceedings of the Multibody dynamics, ECCOMAS Thematic Conference* (2009)
- [27] Murray, R.M., Li, Z., Sastry, S.S.: *A mathematical introduction to robotic manipulation*. CRC Press (1994)
- [28] Seifried, R.: *Dynamics of Underactuated Multibody Systems: Modeling, Control and Optimal Design*. Solid Mechanics and its Applications. Springer (2014)
- [29] Seifried, R., Eberhard, P.: Design of feed-forward control for underactuated multibody systems with kinematic redundancy. *Motion and Vibration Control: Selected papers from MOVIC 2008* (2009)
- [30] Singer, N., Seering, W.P.: Preshaping command inputs to reduce system vibration. *Journal of Dynamic Systems, Measurement and Control* **112**, 76–82 (1990). DOI 10.1115/1.2894142
- [31] Solis, J.F.P., Navarro, G.S., Linares, R.C.: Modeling and tip position control of a flexible link robot: Experimental results. *Computacion y Sistemas* **12**(4), 421–435 (2009)
- [32] Sonnevile, V.: *A geometric local frame approach for flexible multibody systems*. Ph.D. thesis, University of Liège (2015)
- [33] Sonnevile, V., Bröls, O.: A formulation on the special Euclidean group for dynamic analysis of multibody systems. *Journal of Computational and Nonlinear Dynamics* **9** (2014). DOI 10.1115/1.4026569
- [34] Sonnevile, V., Cardona, A., Bröls, O.: Geometrically exact beam finite element formulated on the special euclidean group $SE(3)$. *Computer Methods in Applied Mechanics & Engineering* **268**, 451–474 (2014). DOI 10.1016/j.cma.2013.10.008
- [35] Stauffer, P., Gattringer, H.: *Multibody System Dynamics, Robotic and Control*, chap. Passivity-Based Tracking Control of a Flexible Link Robot, pp. 95–112. Springer Vienna (2013)