On-The-Fly Control Design of Modular Robot Manipulators

Andrea Giusti and Matthias Althoff

***Abstract*—We consider the problem of automatically designing model-based controllers of modular robot manipulators in a systematic way. Modular robots are especially useful in flexible manufacturing, where one wishes to quickly assemble robots from a set of modules for temporary tasks. The evaluation of all possible dynamical models is typically impractical, undermining the implementation of model-based control laws. Contrary to most other work that approached this challenge by designing decentralized controllers, we generate on-the-fly model-based centralized controllers after a new robot has been assembled. In this brief we extend the applicability of our previous work on this subject by considering both link and joint modules, assembly-dependent friction effects and we finally present experiments that validate the overall approach.**

***Index Terms*—Automatic controller design, motion control, kinematics, dynamics, modular robots.**

I. Introduction

MODULAR reconfigurable robot manipulators [1] are promising robots for enhancing the flexibility of next generation manufacturing systems [2]. These robots challenge current manipulators in industry which are typically designed for a set of predetermined tasks. The flexibility of modular robots compared to their fixed-structure counterparts is especially remarkable in flexible environments such as space operations and exploration [3], search and rescue [4], service robots and robots for human-robot cooperation [5]. The versatility of modular robot manipulators leads to several technical challenges, especially for the design of the control system. In particular, considering arbitrary assemblies of a nonuniform set of modules, a large number of different dynamic systems can be obtained, undermining the implementation of model-based controllers [6].

Previous works that aim at solving the control problem of modular robot manipulators mainly focus on decentralized schemes. In [7] the dynamics are considered as those of a set of interconnected subsystems, for which an adaptive control action is designed to cancel the couplings. A method that addresses the position control problem of modular robots can be found in [8], where the authors introduce a scheme based on fuzzy gain tuning of distributed proportional-integral-differential (PID) controllers. The authors of [9] have developed a decentralized control method based on joint torque sensing for compensating coupling effects. A decentralized and robust control method is presented in [10], where authors consider the presence of harmonic drives and claim suitability for modular robot applications due to the simplicity of the resulting controller structure. A control concept based on communication between neighboring modules to provide model-based distributed control is in [11], where authors stress that they achieve unprecedented tracking performance. However, in that work a comparison with the performance of a PID controller is discussed, and high performance model-based trajectory tracking controllers such as the inverse dynamics control [12, Sec. 8.5.2] or passivity-based control [13, Sec. 8.4] were not considered. Controller architectures for modular robot manipulators have also been proposed (see e.g. [14]), where decentralized approaches are considered for motion control.

The above-mentioned methods are mainly motivated by the intrinsic difficulty of designing centralized model-based control laws for modular systems due to the large number of possible combinations of modules. Thanks to our previous work in [15] and this brief, we overcome this problem with a systematic approach for designing centralized model-based controllers on the fly. Our approach is based on distributed data stored in each module, describing the module characteristics for control design. After assembly, the stored module data are collected by the central control unit which automatically synthesizes model-based global tracking controllers. A similar idea has been previously used in [2] where the configuration-dependent gravity vector can be automatically computed from information stored in the modules. In contrast to that work, we propose the automatic generation of trajectory tracking controllers with a systematic and general approach for obtaining the module data and synthesizing them automatically into an assembled-robot description.

We design on-the-fly trajectory tracking controllers by incorporating a systematic method for automatic kinematic and dynamic modelling of the manipulator using module data. Previous works that consider the derivation of complete models from modules are described in [16]–[18]. However, these methods lack seamless general applicability since they consider modular structures with specific geometries [19]. Our proposed approach for obtaining the forward kinematics is based on the Denavit-Hartenberg (D-H) convention [20] and is therefore related to the previous work in [19], [21] and [22]. In [22] only revolute joints are considered and in [19], [21] special cases of consecutive joint axes that produce the typical non-uniqueness of the D-H convention [12, Sec. 2.8.2] are not considered. To overcome difficulties of other approaches, we introduced a notation that enables a systematic characterization of modules based on an extension of the standard D-H convention with the additional advantage that the automatic procedure for obtaining the relative parameters is simplified,

A. Giusti and M. Althoff are with the Department of Informatics, Technical University of Munich (TUM), 85748 Garching, Germany, e-mail: {giusti, althoff}@in.tum.de.

Manuscript received month day, year; revised month day, year.

especially when considering prismatic joints. Another benefit of extending the D-H notation instead of introducing an entirely different formalism is that it is already the most widely employed method for kinematic modelling of standard manipulators. Our proposed framework allows the kinematic description of the robot to be automatically derived. Well-established methods for solving the inverse kinematic problem numerically, which can take advantage of possible kinematic redundancies and avoid problematic velocity profiles near the kinematic singularities, can be implemented [23]. In this brief we can therefore focus on joint-space control only, without loss of generality.

This brief extends [15] as follows: *i)* we enhance the theoretical part for automatic modelling by also considering link modules, *ii)* we account for assembly-dependent friction effects and *iii)* we experimentally validate our proposed method using a real modular robot. In Sec. II we describe the control problem. In Sec. III and Sec. IV we present our complete approach for on-the-fly modelling and control synthesis. In Sec. V we show and discuss experimental results to draw conclusions in Sec. VI.

II. Problem Description

We consider modular robot manipulators with heterogeneous modules. We assume: 1. a joint module is composed of a rigid proximal part, a rigid distal part and a joint (see Fig. 1); 2. a link module is a rigid component (see Fig. 2); 3. each module has an input and an output connector that are standardized and allow the connection of subsequent modules at only one relative orientation. The above-mentioned assumptions imply that we consider joint modules that introduce one degree of freedom to the robotic structure and link modules that do not. However, the generality of our approach is not at risk because more complex modules (i.e. modules that have more than one joint element) can be modeled as a series of the joint modules considered. Assumption 3 is made for simplicity of description and without loss of generality because the application of our proposed method to modules with more than a pair of input and output connectors is straightforward as further clarified in Sec. III-A.

Within the proposed setting, modules can be serially connected to constitute a manipulator with possibly frequently changing assemblies. Each assembly is an open kinematic chain with N degrees of freedom, where N is the number of used joint modules. Using bold symbols for vectors and matrices, the mathematical model describing the dynamics of such a system can be expressed as follows [12, Ch. 7]:

$$\mathbf{M}(\mathbf{q})\,\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}} + \mathbf{f}(\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \mathbf{u}, \qquad (1)$$

where $\mathbf{q} \in \mathbb{R}^N$ is the vector of the generalized coordinates, $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{N\times N}$ is the symmetric and positive definite inertia matrix, $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}}$ (with $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^{N\times N}$) is the vector of the Coriolis and centrifugal terms, $\mathbf{f}(\dot{\mathbf{q}}) \in \mathbb{R}^N$ is the vector of friction terms, $\mathbf{g}(\mathbf{q}) \in \mathbb{R}^N$ is the vector of gravity terms, and $\mathbf{u} \in \mathbb{R}^N$ is the vector of the actuation forces/torques. For each assembled manipulator, we face the problem of automatically and quickly (on-the-fly) designing model-based control laws that guarantee tracking of (at least) twice-differentiable trajectories in joint space $\mathbf{q}_d(t)$ with global asymptotic stability:

$$\lim_{t\to\infty} \|\mathbf{e}(t)\| = 0,$$

where $\|\mathbf{e}(t)\| = \|\mathbf{q}_d(t) - \mathbf{q}(t)\|$. All norms in this brief are Euclidean norms.

III. Automatic Modelling

The basic idea of our proposed method starts with the characterization of each module to extract parameters and store them within the modules. The data obtained from characterization consist of a unique identification number and the kinematic/dynamic parameters of the module (referred to as module data in this brief). After the robot is assembled, the central control unit collects the module data and processes them for automatic generation of model-based control laws.[1] In this section we describe our systematic way for characterizing both joint/link modules (Sec. III-A1, III-A2 for kinematics and Sec. III-B1 for dynamics) and the procedures for synthesizing the module data into an assembled-robot description after the data have been collected (Sec. III-A3 for kinematics and Sec. III-B2 for dynamics).

A. Kinematic Modelling using Module Data

As previously introduced, our proposed approach for kinematic modelling is based on the D-H convention [20]. Therefore, to define the relative transformation of coordinates of subsequent link-fixed frames, the four standard D-H parameters are considered: a_i, d_i, α_i, θ_i (see e.g. [12]). To address the automatic nature of our approach, it is important to resolve the problem that the standard D-H convention is not unique. In fact, for some relative orientations of subsequent joint axes (i.e. parallel, intersect or overlapped) the modeler has partial freedom to place the link-fixed frames. We set the frame deterministically by extending the standard D-H convention and by using two additional parameters p_i and n_i as described in detail in our previous work in [15]. As shown in that work, it is not difficult to notice that given a_i, α_i, γ_i, p_i, n_i and the type of the joint actuation for each link of the manipulator, the parameters d_i and θ_i of the standard D-H convention can be easily computed, while a_i and α_i remain the same. Our extension of the D-H convention is also applicable to standard manipulators and can be used to model the kinematics in a systematic and unique way.

1) Characterization of Joint Modules for Kinematics: Let us consider an exemplary joint module represented in Fig. 1(a). To characterize both the proximal and the distal part with a set of parameters we can apply a similar approach of the extended D-H convention. We obtain four parameters each for the proximal part (a^{pl}, α^{pl}, p^{pl}, n^{pl}) and for the distal part (a^{dl}, α^{dl}, p^{dl}, n^{dl}) (see Fig. 1(b) and (c)). Finally, to complete the characterization of a joint module and parametrize the complete transformation of coordinates from the input frame to the output one, three additional parameters are required: δ^{pl},

[1] Further details on network solutions that can support this approach can be found in [15].

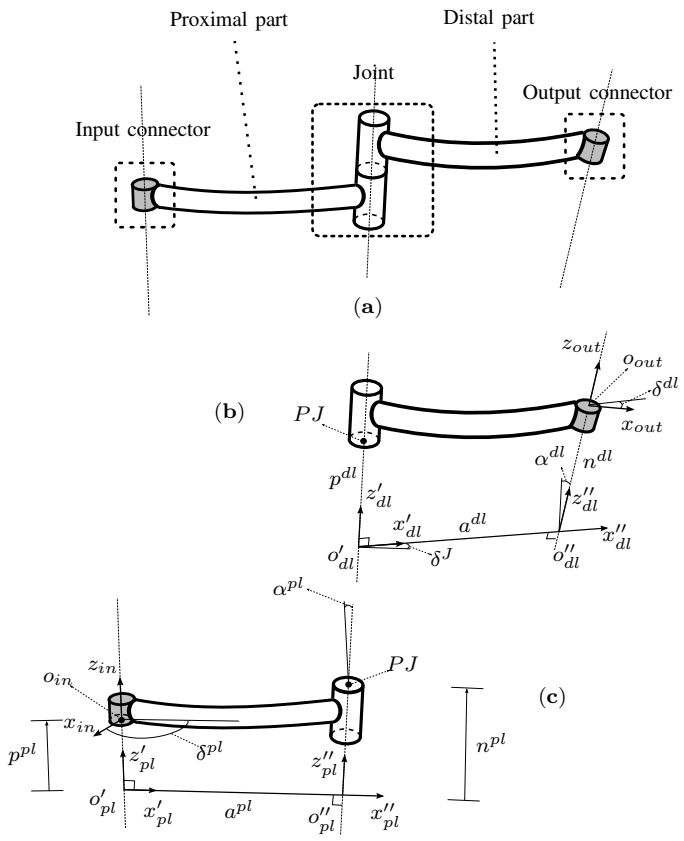

Fig. 1. Kinematic notation for joint module characterization. The connectors are indicated in light-grey. (a) is the entire module, (b) the distal part and (c) the proximal part.

δ^{dl}, δ^{J} (all the angles of this notation are positive counter-clockwise). These parameters depend on the geometry of the module and their meaning is further detailed in [15].

2) Characterization of Link Modules for Kinematics: Similarly to the joint modules, we can characterize link modules with a set of parameters using the same systematic approach. Considering the exemplary link module of Fig. 2, the first step to characterize it is to set the input and output frames at the respective connectors. Then, the four parameters for a link module (a^l, α^l, p^l, n^l) can be obtained from the module geometry. The meaning of the parameters is easy to infer observing Fig. 2 and considering either the characterization of the distal part or the proximal part of joint modules. In conclusion, to complete the parametrization of the transformation of coordinates from the input frame to the output frame, the angle $\delta^{l,in}$ between x'_l and x_{in}, and the angle $\delta^{l,out}$ between x'_l and x_{out} are also required. It is worth noting that end-effector modules can be considered as link modules with the output frame placed in any user-defined pose of interest.

The parameters required to characterize a module for kinematics using our proposed notation are listed in Tab. I. Note that the extension of the proposed approach to modules with multiple input and/or output connectors is straightforward, because in this case a set of parameters for each possible pair of connectors can be obtained.

3) Synthesis of Module Data for Kinematics: Let us consider the generic connection in Fig. 3 that is representative

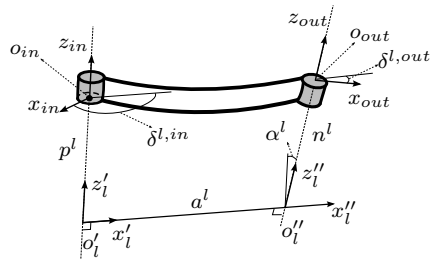

Fig. 2. Kinematic notation for link module characterization. The connectors are indicated in light-grey.

TABLE I
INFORMATION TO STORE IN EACH MODULE FOR KINEMATICS

Joint Module	Proximal	a^{pl}	α^{pl}	p^{pl}	n^{pl}	δ^{pl}	
	Distal	a^{dl}	α^{dl}	p^{dl}	n^{dl}	δ^{dl}	
	Joint	δ^J		Joint type			
Link Module		a^l	α^l	p^l	n^l	$\delta^{l,in}$	$\delta^{l,out}$

of a link of a modular manipulator. In this case two joint modules are connected through an arbitrary number $k \geq 0$ of link modules. In order to automatically obtain the parameters that describe the transformation of coordinates of each constituted link of the manipulator, we consider a synthesis matrix that represents (with reference to Fig. 3) the homogeneous transformation matrix of a frame oriented as the D-H one and located at PJ_i from a frame parallel to the first auxiliary frame of the distal part of module$_{j-1}$ (denoted with superscript $'$ in Fig. 1(b)) with origin PJ_{i-1}. To obtain such a synthesis matrix we first compute an auxiliary homogeneous transformation matrix $\mathbf{F}'_i$ describing the pose of a frame parallel to the second auxiliary frame of the proximal part of module$_{j+k}$ (denoted with superscript $''$ in Fig. 1(c)), with respect to the frame parallel to the first auxiliary frame of the distal part of module$_{j-1}$ and located at PJ_{i-1}.

The matrix $\mathbf{F}'_i$ can be simply computed using subsequent multiplications of homogeneous transformation matrices for elementary translations/rotations for the kinematic data of the modules that compose the link. Considering Fig. 3 and using elementary translations/rotations we first obtain the homogeneous transformation matrix for the distal part ($\mathbf{A}^{dl}_{j-1}$), then the one for the k link modules ($\mathbf{A}^{l}_{j,j+k}$) and finally the one for the proximal part of the module that completes the link ($\mathbf{A}^{pl}_{j+k}$). Denoting by $T_\rho(\cdot)/R_\rho(\cdot)$ the homogeneous transformation of the elementary translation/rotation along/around the ρ axis, the auxiliary matrix is computed as:

$$\mathbf{F}'_i = \mathbf{A}^{dl}_{j-1}\,\mathbf{A}^{l}_{j,j+k}\,\mathbf{A}^{pl}_{j+k} = \begin{bmatrix} \mathbf{R}'_i & \mathbf{U}'_i \\ \mathbf{0}^T & 1 \end{bmatrix},$$

where

$$\mathbf{A}^{dl}_{j-1} = T_z(-p^{dl}_{j-1})T_x(a^{dl}_{j-1}) \\ R_x(\alpha^{dl}_{j-1})T_z(n^{dl}_{j-1})R_z(\delta^{dl}_{j-1}),$$

$$\mathbf{A}^{pl}_{j+k} = R_z(-\delta^{pl}_{j+k})T_z(-p^{pl}_{j+k}) \\ T_x(a^{pl}_{j+k})R_x(\alpha^{pl}_{j+k})T_z(n^{pl}_{j+k}),$$

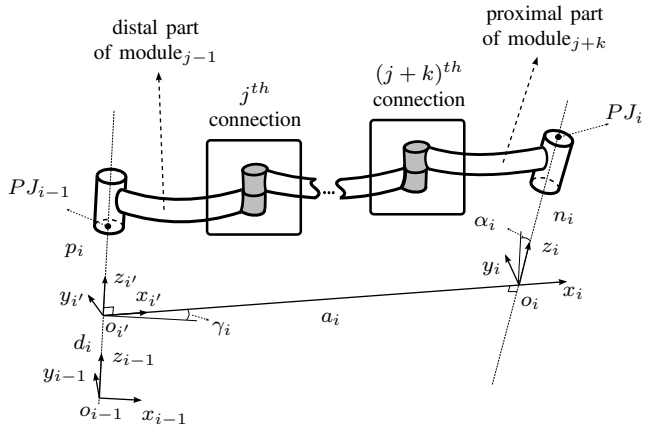

Fig. 3. Synthesis of kinematic parameters for the i^{th} constituted link with j being the cumulative number of modules connected.

$$\mathbf{A}^l_{j,j+k} = \begin{cases} \prod_{h=j}^{j+k-1} R_z(-\delta^{l,in}_h)T_z(-p^l_h)T_x(a^l_h) \\ \qquad R_x(\alpha^l_h)T_z(n^l_h)R_z(\delta^{l,out}_h), & \text{if } k>0, \\ \mathbf{I}_{4\times 4}, & \text{if } k=0. \end{cases}$$

The obtainment of the parameters of the extended D-H convention for the i^{th} assembled link (a_i, α_i, γ_i, p_i, n_i) now follows as in [15].

B. Automatic Modelling of the Dynamics using Module Data

To obtain the dynamical model of the modular manipulator we exploit results from the large body of research on modelling standard robots. Most efficient numerical methods are based on the recursive Newton-Euler (N-E) formulation; their computational efficiency makes them a suitable choice for our application. In particular, we make use of the standard recursive N-E algorithm and its modified version for passivity based control. We do not recall the details of these algorithms because this would exceed the purpose of this brief; we redirect the interested reader to [12], [24] for further details.

The fundamental information of the robot assembly needed to run these algorithms is the mass of each link, its inertia tensor, the coordinates of the center of gravity and the coordinates of the application point of forces on the link itself. It should be noted that after collecting the module data, the central control unit only knows the parameters of the *modules* that compose each link and not those of the *actual link* of the arm. Therefore, after defining what information of the modules for dynamics should be stored within them, one needs a procedure to process these data and obtain the dynamic parameters of the assembled manipulator. More specifically, observing the exemplary link of the assembled manipulator shown in the top of Fig. 4, we consider a frame located at the joint connection point with origin in $\mathbf{D}_i$. With respect to this frame (assumed to be parallel to the resulting D-H one), we need the coordinates of the center of mass $\mathbf{r}^i_{D_i,C_i}$ and the coordinates of the connection point with the previous joint $\mathbf{r}^i_{D_{i-1},D_i}$. In addition we need the mass of the link m_i and its inertia tensor $\mathbf{I}^i_i$. Superscript of vectors indicates the frame in which they are defined. We are now ready to define the data for dynamics that must be stored in the modules and the automatic procedure to obtain the dynamic parameters of the assembled manipulator.

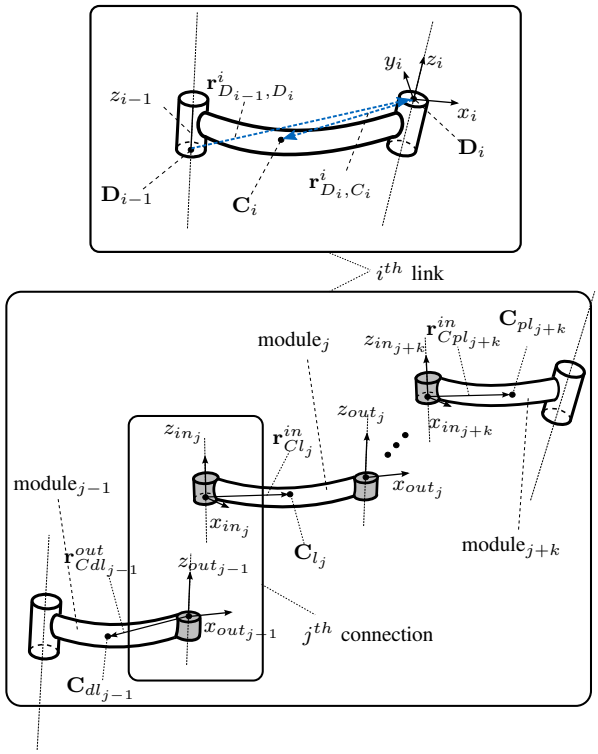

Fig. 4. Representation of a connection of modules that constitutes the i^{th} link of the manipulator involving parameters for dynamics. Connectors are indicated in light-grey.

1) Characterization of Modules for Dynamics: To define the part of the module data for dynamics, we consider a general connection as shown in the exploded view of Fig. 4. This exemplary connection involves module$_{j-1}$ (joint module), k link modules and a subsequent module$_{j+k}$ (joint module). When the connection is performed, the i^{th} link of the manipulator is realized. Since the algorithms for modelling dynamics of robotic arms require the dynamical parameters of each i^{th} link of the manipulator, we need to know the dynamical parameters of the rigid bodies that compose it. Therefore, the module data required for dynamics are the corresponding mass, coordinates of the center of mass, and the inertia tensor of the distal part (m^{dl}, $\mathbf{r}^{out}_{Cdl}$,$\mathbf{I}^{out}_{dl}$), proximal part (m^{pl}, $\mathbf{r}^{in}_{Cpl}$,$\mathbf{I}^{in}_{pl}$) and of the link modules (m^l, $\mathbf{r}^{in}_{Cl}$,$\mathbf{I}^{in}_{l}$) that compose each link of the manipulator. As it can be noticed by the superscripts, the coordinates of the center of mass and the inertia tensor are expressed in the input frame for proximal parts and link modules, while for distal parts they are expressed in the output frame. This makes it easy to implement a recursive procedure to synthesize the parameters of the modules and obtain those of the link of the manipulator that they compose.

To have a complete description of the system dynamics for control purposes, the friction model should also be considered. The friction model represented by $\mathbf{f}(\dot{\mathbf{q}})$ in (1) is usually assumed to be composed of a static and a viscous component. For each joint-module, its function $f(\dot{q})$ (or the corresponding coefficients) must be included in the module data. A typical friction model that constitutes a trade-off between modelling

TABLE II
INFORMATION TO STORE IN EACH MODULE FOR DYNAMICS

Joint Module	Proximal	m^{pl}	$\mathbf{I}_{pl}^{in}$	$\mathbf{r}_{Cpl}^{in}$	
	Distal	m^{dl}	$\mathbf{I}_{dl}^{out}$	$\mathbf{r}_{Cdl}^{out}$	
	Joint	$f(\dot{q})$	$\tau(c)$	I_m	κ_r
Link Module		m^l	$\mathbf{I}_l^{in}$	$\mathbf{r}_{Cl}^{in}$	

accuracy and simplicity for control purposes for the i^{th} joint of a robotic arm is the following [12, Ch. 7]:

$$f_i(\dot{q}_i) = \beta_{v,i}\,\dot{q}_i + \beta_{c,i}\,\text{sign}(\dot{q}_i), \tag{2}$$

where $\beta_{v,i}$ and $\beta_{c,i}$ are the viscous and static friction coefficients, respectively.

To map the control signal c onto the torque/force applied at the joint axis and vice-versa, we also need to include the respective actuator-dependent function $\tau(c)$. It is worth mentioning that when electric motors and gears are employed a non-negligible contribution to the dynamics is typically due to the inertia of the rotor I_m of the actuators [25], [26], which we have considered by adding it to the diagonal of the inertia matrix through the square of the gear ratio κ_r. In summary, the module data to be stored in the modules for dynamics are listed in Tab. II.

2) Synthesis of Module Data for Dynamics: We can now introduce the procedure to obtain the parameters for each constituted link of a modular manipulator using module data. With reference to Fig. 4, the procedure starts from the synthesis of the parameters of the connection between the distal part of the module$_{j-1}$ and the first link module through the j^{th} connection. This connection constitutes a new component that is essentially an auxiliary distal part whose mass m_j^a, coordinate of center of mass $\mathbf{r}_{Ca_j}^{out}$ and inertia tensor $\mathbf{I}_{a_j}^{out}$ are synthesized in the following. We denote the matched input-output frame as "io" when a connection is realized:

$$m_j^a = m_{j-1}^{dl} + m_j^l, \quad \mathbf{I}_{a_j}^{io} = \mathbf{I}_{dl_{j-1}}^{out} + \mathbf{I}_{l_j}^{in},$$
$$\mathbf{r}_{Ca_j}^{io} = \frac{m_{j-1}^{dl}\,\mathbf{r}_{Cdl_{j-1}}^{out} + m_j^l\,\mathbf{r}_{Cl_j}^{in}}{m_j^a}.$$

To express the coordinates of the center of mass and the inertia tensor with respect to the output frame, we use homogeneous transformations and Steiner's theorem [12, App. B.2]. Since kinematic parameters are known, the coordinates of the center of mass of the auxiliary distal part, with respect to the output frame, are computed as:

$$\begin{bmatrix} \mathbf{r}_{Ca_j}^{out} \\ 1 \end{bmatrix} = \left[\mathbf{A}_{out,a_j}^{io,a_j}\right]^{-1} \begin{bmatrix} \mathbf{r}_{Ca_j}^{io} \\ 1 \end{bmatrix}, \tag{3}$$

with

$$\mathbf{A}_{out,a_j}^{io,a_j} = \begin{bmatrix} \mathbf{R}_{out,a_j}^{io,a_j} & \mathbf{U}_{out,a_j}^{io,a_j} \\ \mathbf{0}^T & 1 \end{bmatrix} = R_z(-\delta_j^{l,in})T_z(-p_j^l)T_x(a_j^l) R_x(\alpha_j^l)T_z(n_j^l)R_z(\delta_j^{l,out}).$$

The matrix $\mathbf{A}_{out,a_j}^{io,a_j}$ in (3) is the homogeneous transformation matrix of the coordinate transformation between the input and output frame of the j^{th} link module, which becomes part of the auxiliary distal part once connected. In addition we compute:

$$\mathbf{I}_{a_j}^{out} = \left(\mathbf{R}_{out,a_j}^{io,a_j}\right)^T \left(\mathbf{I}_{a_j}^{io} - m_j^a\,\mathbf{S}^T(\mathbf{r}_{Ca_j}^{io})\mathbf{S}(\mathbf{r}_{Ca_j}^{io})\right)\mathbf{R}_{out,a_j}^{io,a_j} + m_j^a\,\mathbf{S}^T(\mathbf{r}_{Ca_j}^{out})\mathbf{S}(\mathbf{r}_{Ca_j}^{out}),$$

where $\mathbf{S}(\cdot)$ denotes an anti-symmetric matrix of the type:

$$\mathbf{S}(\mathbf{U}) = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix},\ \mathbf{U} = \begin{bmatrix} u_x & u_y & u_z \end{bmatrix}^T.$$

Assuming that an additional module$_{j+1}$ (link module) is connected to the auxiliary distal part that we have, it is now trivial to see that a new auxiliary distal part is created and that the synthesis of the corresponding parameters is performed with the same procedure described above. This recursion is performed for the k link modules. Once this recursion is completed, or if it was not necessary (e.g. for $k = 0$), the last connection is made between a distal part and a proximal part of two joint modules. Therefore, the last operations for obtaining the dynamical parameters of each i^{th} constituted link of the manipulator (m_i, $\mathbf{I}_i^i$, $\mathbf{r}_{D_i,C_i}^i$, $\mathbf{r}_{D_{i-1},D_i}^i$) now follow as in [15].[2]

IV. MODEL-BASED CONTROLLER SYNTHESIS

The central control unit collects the module data after each assembly and processes them to automatically obtain the assembled-robot description (for kinematics and dynamics) using the procedures described in Sec. III-A3 and Sec. III-B2. Subsequently, the implementation of model-based control laws follows directly. Since the control problem we face is the tracking of trajectories with global asymptotic stability, we consider two of the most effective model-based tracking control methods: inverse dynamics control [12, Sec. 8.5.2] and passivity-based control [13, Sec. 8.4]. We implement the inverse dynamics control law using:

$$\mathbf{u}_{ID} = \mathbf{M}(\mathbf{q})\left(\ddot{\mathbf{q}}_d + \mathbf{K}_P\mathbf{e} + \mathbf{K}_D\dot{\mathbf{e}}\right) + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}} + \mathbf{f}(\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}),$$

where $\mathbf{K}_P$, $\mathbf{K}_D$ are gain matrices of proper dimensions. In our work, the computations are performed numerically, directly using the standard recursive N-E algorithm [12].

The second popular model-based trajectory tracking controller we consider is the passivity-based control law. The control command of such a controller is computed as:

$$\mathbf{u}_{PB} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_a + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}_a + \mathbf{f}(\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \mathbf{K}\,\mathbf{r}, \tag{4}$$

with $\dot{\mathbf{q}}_a = \dot{\mathbf{q}}_d + \mathbf{K}_V\mathbf{e}$, $\mathbf{r} = \dot{\mathbf{e}} + \mathbf{K}_V\mathbf{e}$ and where $\mathbf{K}_V$ and $\mathbf{K}$ are diagonal positive definite matrices of proper dimensions. For computing this control law numerically we adopt the modification to the standard N-E algorithm proposed in [24].

As described in [27], the load at the joints of a robot may affect the friction model significantly. This aspect deserves

[2]Please note that the implementation of the recursive N-E algorithm typically requires $\mathbf{I}_i^i$ to be expressed in a barycentric frame (in this case oriented as the corresponding D-H one), which can be straightforwardly obtained by applying Steiner's theorem.

Fig. 5. Assemblies of the robot considered for tests. The assembly I is composed of all available modules and in assembly II we remove the end-effector module. It is worth mentioning that the end-effector module has non-negligible dynamical parameters with respect to other modules of the available set. Finally, we test assembly III, in which the last three modules are removed. Both assemblies I and II constitute a six degrees of freedom arm, while assembly III is a robotic arm with four degrees of freedom.

special attention considering that in our approach the modules are characterized once and that they may compose arbitrary assemblies successively. To solve this problem we foster a method that relies on the introduction of an adaptive term to the passivity-based feedback control law.[3] We introduce a term to the feedback control law in (4) for having an adaptive friction compensation by tailoring the work of [29] to our purpose. We consider the friction model of (2) and assume that only nominal parameters $\beta_{0v,i}$ and $\beta_{0c,i}$ for each i^{th} joint are available e.g. from initial module characterization. Given this nominal information, the passivity-based control command with adaptive friction compensation is computed with

$$\mathbf{u}_{PBAFC} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_a + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}_a + \hat{\mathbf{f}}(\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \mathbf{K}\,\mathbf{r},$$

where $\hat{f}_i(\dot{q}_i) = \hat{\beta}_{v,i}(t)\dot{q}_i + \hat{\beta}_{c,i}(t)\text{sign}(\dot{q}_i)$, and $\hat{\beta}_{v,i}(0) = \beta_{0v,i}$, $\hat{\beta}_{c,i}(0) = \beta_{0c,i}$. Now, by employing the following adaptive law

$$\left[\begin{array}{cccc}\dot{\hat{\beta}}_{v,1}(t) & \dot{\hat{\beta}}_{c,1}(t) & \cdots & \dot{\hat{\beta}}_{v,N}(t) \;\; \dot{\hat{\beta}}_{c,N}(t)\end{array}\right]^T = \mathbf{K}_{\Delta_\beta}^{-1}\mathbf{Y}(\dot{\mathbf{q}})^T\mathbf{r},$$

where $\mathbf{K}_{\Delta_\beta}$ is a positive definite matrix of proper dimensions and

$$\mathbf{Y}(\dot{\mathbf{q}}) = \left[\begin{array}{cccc}\dot{q}_1 & \text{sign}(\dot{q}_1) & \dots & 0 \quad 0 \\ 0 & 0 & \ddots & 0 \quad 0 \\ 0 & 0 & \dots & \dot{q}_N \;\; \text{sign}(\dot{q}_N)\end{array}\right],$$

global asymptotic stability can be easily shown using a similar argument as in [12, Sec. 8.4.5].

V. Experimental Results

In this section we present experimental results of our proposed approach on a real modular robot. Our robot test bed is composed of the Schunk LWA-4P (see Fig. 5) and a Speed-Goat real-time target machine equipped with a dedicated CAN-bus communication module. From the set of modules available to us we can test three different assemblies that we denote hereafter by I, II, III (see Fig. 5). The required module data for kinematics and dynamics are obtained with the previously discussed procedures using CAD data and data-sheets from the robot manufacturer.[4] We use MATLAB/Simulink Real-Time 2015b to implement our approach. The interface with the modules is based on CANopen. With the maximum number of axes available to us (6 axes), we obtain a stable closed-loop communication when running at $t_{sample} = 2\,ms$ and therefore we set a sampling rate of $1/2\,ms = 500\,Hz$ for all controllers we test. The tests are performed using the following test trajectory, which provides zero initial desired joint position/velocity for all axes and is twice differentiable: $\mathbf{q}_d(t) = \boldsymbol{\rho}\sin\left(\frac{\pi\,t}{10}\right)\sin\left(\frac{\pi\,t}{5}\right)$, where $\boldsymbol{\rho} = \frac{1}{10}[\,-6\pi\;\; 2\pi\;\; -3\pi\;\; 5\pi\;\; -7\pi\;\; 7\pi\,]^T$. The location of each element in $\mathbf{q}_d(t)$ corresponds to the desired trajectory of the respective joint variable starting from the base of the robot. The same trajectory is used for testing the robot with assembly III by properly reducing its dimension. The controllers under test require the measurement of joint velocities which are not directly available with our test bed. To estimate the joint velocities we implement a kinematic Kalman filter based on [30] because of its tuning and implementation simplicity. A practical aspect due to the noisy estimate of the joint velocities is that for compensation of the friction we observe better tracking performance when using a feed-forward compensation based on the desired joint velocity. Therefore, the friction has been compensated accordingly for the experiments shown.

The first experiment we present aims at validating the correctness of the model obtained on-the-fly using module data. The results of this experiment are collected in Fig. 6. In the first plot from the top of Fig. 6, we show the closed-loop tracking performance of the inverse-dynamics controllers for assembly I using both the model obtained on-the-fly from module data and the model obtained with a standard modelling procedure (e.g. ignoring modularity and using standard D-H convention). As expected, the results show proper matching. Additionally, in the second plot from the top of Fig. 6 we compare the torque prediction of the on-the-fly generated model with the measured torque applied for executing the motion during the experiment and a remarkably good match is observed. The gains used for this experiment have been selected such that the closed-loop error dynamics are in principle equivalent to a set of second-order linear systems, each with natural frequency $\omega_n = 55\,(rad/s)$ and damping ratio $\zeta = 0.65$ (i.e. $\mathbf{K}_P = \omega_n^2\,\mathbf{I}$ and $\mathbf{K}_D = 2\zeta\omega_n\,\mathbf{I}$).

We show the benefit of the adaptive friction compensation by performing a test with a non-finely-tuned version of the automatically generated passivity-based controller and a spoiled initial guess for the friction parameters (setting them to zero, which is not the case in reality since transmissions are employed). The results of this test are shown in Fig. 7, where we can clearly observe the benefit due to the action of the

[3]It is worth noting that well-established methods for full dynamics adaptation could also be considered (see e.g. [28]). However, in this context the dynamic parameters of the modules are not configuration dependent and can be accurately known at the time of the module development with modern CAD softwares. This is typically not the case for the friction model.

[4]The software for processing the module data according to our proposed approach, which additionally contains the data of the modules we use, can be downloaded at: https://github.com/AndreaGiusti/OTFCtrlModRob.

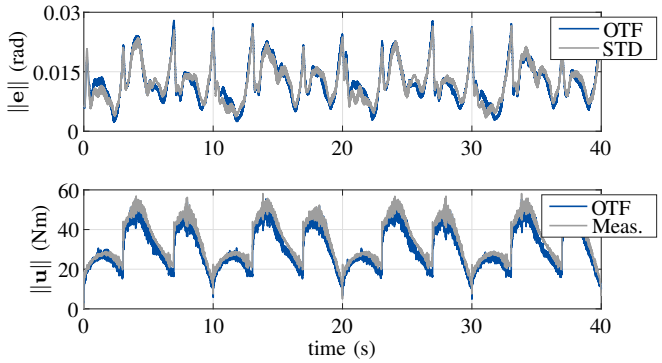

Fig. 6. Demonstration of the model accuracy. The first plot shows tracking performance of inverse dynamics control (with assembly I) using the model obtained with standard modelling approach (STD) and on-the-fly (OTF). In the second plot we show the matching between the measured torque for executing the motion and the prediction using the model generated on-the-fly.

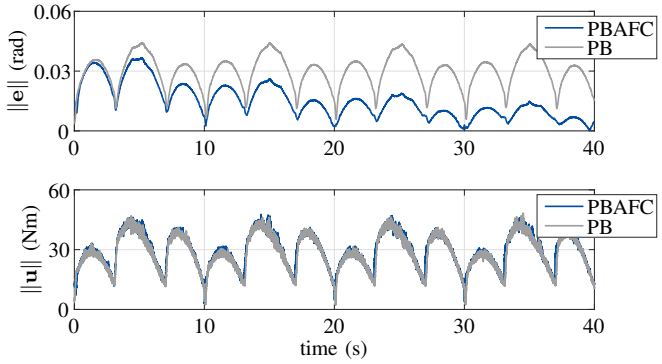

Fig. 7. Demonstration of the adaptive friction compensation. We show the action of the passivity-based controller with adaptive friction compensation labeled as PBAFC with respect to a standard passivity-based controller labeled as PB. We use assembly I with non-finely-tuned controllers and assume wrongly known friction model coefficients (by setting them to zero). In this experiment we used the following gains: $\mathbf{K}_V = \mathbf{K} = 30\mathbf{I}$, $\mathbf{K}_{\Delta_\beta} = \mathbf{I}$.

adaptive term from the reduction over time of the tracking error, still maintaining relatively smooth control commands. This has an important implication since assembly dependent friction effects may not be negligible and rough tuning of the automatically generated controller could happen when the fast commissioning of a newly assembled robot is required.

The third experiment we discuss is about the comparison of the performance of the considered model-based controllers, with a standard approach based on decentralized proportional-integral-derivative controllers and decentralized feed-forward actions (PIDs). We test these controllers for the three different assemblies shown in Fig. 5. The classical structure of these decentralized controllers allows the designer to see the closed-loop for each axis as a second-order linear system with user-defined natural frequency ω_n and damping ratio ζ, by considering the actuator dynamics only and by leaving the couplings as disturbance to be rejected (see e.g. in [12, Sec. 8.3.1]). We can therefore provide a fair comparison by setting $\omega_n = 50\,(rad/s)$ and $\zeta = 0.65$ for both inverse-dynamics control and the PIDs. We tuned the passivity-based control law with adaptive friction compensation such that the tracking performance with assembly III are comparable with the other controllers: $\mathbf{K} = 50\mathbf{I}$, $\mathbf{K}_V = 50\mathbf{I}$, $\mathbf{K}_{\Delta_\beta} = \mathbf{I}$. The results of this experiment are collected in Fig. 8. The first column of plots of this figure shows comparable performance when using the simplest robot assembly of the considered ones. The other two columns show the tracking performance when changing the robot assembly without re-tuning, mimicking a quick required reconfiguration of the robot. While instabilities are observed when using the decentralized PIDs, as expected the performance of the automatically generated model-based controllers are not significantly affected by changing the assembly. Additionally, we report that with the use of the recursive N-E algorithm and our experimental setup, the model-based control commands are computed in less than 20 microseconds.[5]

VI. Conclusion

A systematic method for model-based control of modular robot manipulators is presented and successfully applied to a real modular robot. The on-the-fly design of model-based tracking controllers greatly simplifies tuning procedures compared to conventional methods, reducing the time for commissioning the robot after a new assembly and preserving the advantages of flexibility. Indeed, the instabilities experienced in our experiments after reconfiguration show that non-trivial retuning of simple decentralized control schemes such as PIDs, is typically required. This may lead to time-consuming re-tuning phases that undermine the flexibility introduced by frequent reconfiguration capabilities of this class of robots.

Our proposed framework has also been successfully applied to a different reconfigurable robot test bed with elastic joint modules in [31]. Additionally, this framework can easily incorporate robust control approaches [32], [33], when model uncertainties, unknown loads and disturbances have a significant impact. Another interesting extension of this work may involve the consideration of assemblies with closed kinematic chains, for which e.g. the corresponding extension of the D-H convention described in [12, Sec. 2.8.3] and related modern control approaches such as [34], [35] may be considered.

Acknowledgment

The research leading to these results has received funding from the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme FP7/2007-2013/ under REA grant agreement number 608022.

References

[1] D. Schmitz, P. Khosla, and T. Kanade, "The CMU reconfigurable modular manipulator system," Inst. Software Res., Carnegie Mellon Univ., Pittsburgh, PA, USA, CMU-RI-TR-88-7, Tech. Rep., 1988.

[2] C. J. J. Paredis, H. B. Brown, and P. K. Khosla, "A rapidly deployable manipulator system," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 2, 1996, pp. 1434–1439.

[3] M. Yim, K. Roufas, D. Duff, Y. Zhang, C. Eldershaw, and S. Homans, "Modular reconfigurable robots in space applications," *Autonomous Robots*, vol. 14, no. 2-3, pp. 225–237, 2003.

[4] C. Wright, A. Buchan, B. Brown, J. Geist, M. Schwerin, D. Rollinson, M. Tesch, and H. Choset, "Design and architecture of the unified modular snake robot," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2012, pp. 4347–4354.

[5]It is important to mention that the PIDs can be made stable again by manual re-tuning which, however, would require additional user intervention contrary to our proposed approach.

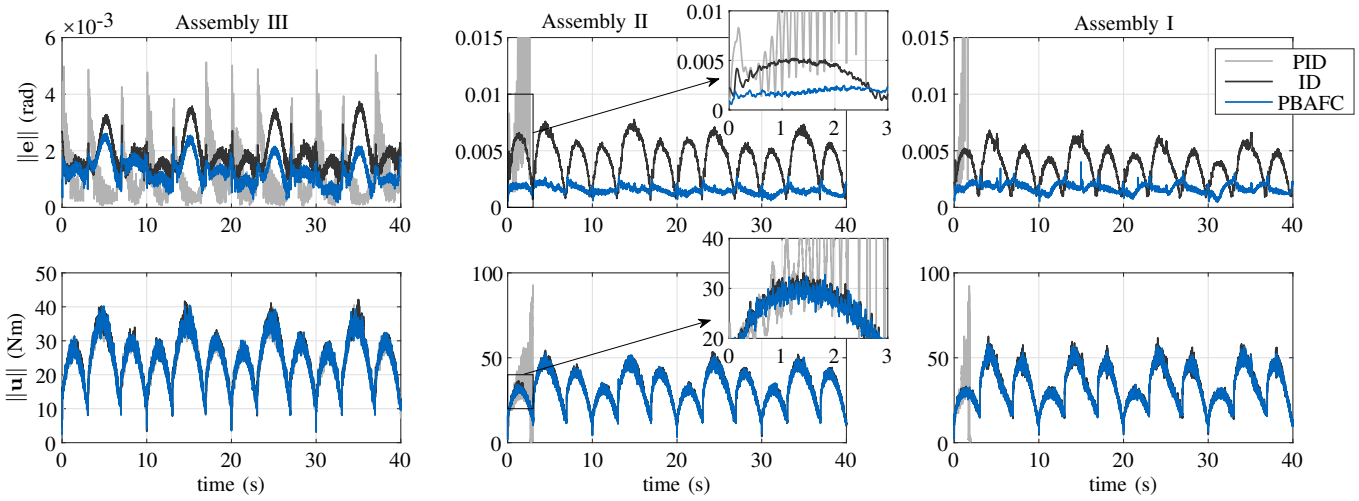

Fig. 8. Experimental comparison of different controllers for a changed assembly of the robot. We denote with ID the inverse dynamics control, with PBAFC the passivity-based control with adaptive friction compensation and with PID the scheme with decentralized PID controllers. PID control tests with assembly II and I had to be stopped due to instability.

[5] A. De Santis, B. Siciliano, A. De Luca, and A. Bicchi, "An atlas of physical human-robot interaction," *Mechanism and Machine Theory*, vol. 43, no. 3, pp. 253–270, 2008.

[6] E. Icer, A. Giusti, and M. Althoff, "A task-driven algorithm for configuration synthesis of modular robots," in *Proc. IEEE Int. Conf. on Robotics and Automation*, May 2016, pp. 5203–5209.

[7] M. Zhu and Y. Li, "Decentralized adaptive fuzzy control for reconfigurable manipulators," in *Proc. IEEE Conference on Robotics, Automation and Mechatronics*, Sept. 2008, pp. 404–409.

[8] W. Melek and A. Goldenberg, "Neurofuzzy control of modular and reconfigurable robots," *IEEE/ASME Transactions on Mechatronics*, vol. 8, no. 3, pp. 381–389, Sept. 2003.

[9] G. Liu, S. Abdul, and A. Goldenberg, "Distributed modular and reconfigurable robot control with torque sensing," in *Proc. IEEE Int. Conf. on Mechatronics and Automation*, June 2006, pp. 384–389.

[10] W. Li, Z. Melek and C. Clark, "Decentralized robust control of robot manipulators with harmonic drive transmission and application to modular and reconfigurable serial arms," *Robotics*, vol. 27, pp. 291–302, 2009.

[11] W.-H. Zhu, T. Lamarche, E. Dupuis, D. Jameux, P. Barnard, and G. Liu, "Precision control of modular robot manipulators: The VDC approach with embedded FPGA," *IEEE Transactions on Robotics*, vol. 29, no. 5, pp. 1162–1179, Oct. 2013.

[12] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer, 2009.

[13] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. Wiley, 2006.

[14] D. Wang, A. Goldenberg, and G. Liu, "Development of control system architecture for modular and re-configurable robot manipulators," in *Proc. of the Int. Conf. on Mechatronics and Autom.*, 2007, pp. 20–25.

[15] A. Giusti and M. Althoff, "Automatic centralized controller design for modular and reconfigurable robot manipulators," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Sept. 2015, pp. 3268–3275.

[16] I.-M. Chen, S. Yeo, G. Chen, and G. Yang, "Kernel for modular robot applications: Automatic modeling techniques," *Int. J. Robot. Res.*, vol. 18, no. 2, pp. 225–242, 1999.

[17] I.-M. Chen and G. Yang, "Automatic model generation for modular reconfigurable robot dynamics," *Journal of Dyn. Sys., Meas., Control*, vol. 120, no. 3, pp. 346–352, 1998.

[18] E. Meister, E. Nosov, and P. Levi, "Automatic onboard and online modelling of modular and self-reconfigurable robots," in *Proc. IEEE Conference on Robotics, Automation and Mechatronics*, 2013, pp. 91–96.

[19] Z. Bi, W. Zhang, I. Chen, and S. Lang, "Automated generation of the D-H parameters for configuration design of modular manipulators," *Robotics and Computer-Integrated Manufacturing*, vol. 23, pp. 553–562, 2007.

[20] J. Denavit and R. Hartenberg, "A kinematic notation of lower-pair mechanisms based on matrices," *ASME Journal of Applied Mechanics*, vol. 22, pp. 215–221, 1955.

[21] B. Benhabib, G. Zak, and M. Lipton, "A generalized kinematic modeling method for modular robots," *Journal of Robotic Systems*, vol. 6, no. 5, pp. 545–571, 1989.

[22] L. Kelmar and P. Khosla, "Automatic generation of kinematics for a reconfigurable modular manipulator system," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 2, Apr. 1988, pp. 663–668.

[23] B. Siciliano, "Kinematic control of redundant robot manipulators: A tutorial," *Journal of intelligent and robotic systems*, vol. 3, no. 3, pp. 201–212, 1990.

[24] A. De Luca and L. Ferrajoli, "A modified Newton-Euler method for dynamic computations in robot fault detection and control," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2009, pp. 3359–3364.

[25] L. Sciavicco, B. Siciliano, and L. Villani, "Lagrange and Newton-Euler dynamic modelling of a gear-driven rigid robot manipulator with inclusion of motor inertia effects," *Advanced Robotics*, vol. 10, no. 3, pp. 317–334, 1996.

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

[27] M.-W. Ueberle, "Design, control, and evaluation of a family of kinesthetic haptic interfaces," Ph.D. dissertation, Technische Universität München, 2006.

[28] R. Ortega and M. W. Spong, "Adaptive motion control of rigid robots: a tutorial," in *Proc. 27th IEEE Conference on Decision and Control*, vol. 2, Dec. 1988, pp. 1575–1584.

[29] J.-J. E. Slotine and W. Li, "On the adaptive control of robot manipulators," *Int. J. Robot. Res.*, vol. 6, no. 3, pp. 49–59, 1987.

[30] P. Belanger, P. Dobrovolny, A. Helmy, and X. Zhang, "Estimation of angular velocity and acceleration from shaft-encoder measurements," *Int. J. Robot. Res.*, vol. 17, no. 11, pp. 1225–1233, 1998.

[31] A. Giusti, J. Malzahn, N. Tsagarakis, and M. Althoff, "Combined inverse-dynamics/passivity-based control for robots with elastic joints," in *Proc. IEEE International Conference on Robotics and Automation*, 2017, to be published.

[32] A. Giusti and M. Althoff, "Ultimate robust performance control of rigid robot manipulators using interval arithmetic," in *Proc. of the American Control Conference*, 2016, pp. 2995–3001.

[33] ——, "Efficient computation of interval-arithmetic-based robust controllers for rigid robots," in *Proc. First IEEE International Conference on Robotic Computing*, Apr. 2017, pp. 129–135.

[34] W. W. Shang, S. Cong, and Y. Ge, "Adaptive computed torque control for a parallel manipulator with redundant actuation," *Robotica*, vol. 30, no. 3, pp. 457–466, May 2012.

[35] W. W. Shang and S. Cong, "Motion control of parallel manipulators using acceleration feedback," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 1, pp. 314–321, Jan. 2014.