Received February 21, 2015, accepted March 22, 2015, date of publication March 31, 2015, date of current version April 10, 2015.

Digital Object Identifier 10.1109/ACCESS.2015.2418157

Closed-Loop Control of Variable Stiffness Actuated Robots via Nonlinear Model Predictive Control

ALTAY ZHAKATAYEV, (Member, IEEE), MATTEO RUBAGOTTI, (Member, IEEE), AND HUSEYIN ATAKAN VAROL, (Member, IEEE)

Department of Robotics and Mechatronics, Nazarbayev University, Astana 010000, Kazakhstan

Corresponding author: H. A. Varol (ahvarol@nu.edu.kz)

This work was supported partially by the Ministry of Education and Science of Kazakhstan through the project Optimal Design and Control of Variable Impedance Actuated Robots.

ABSTRACT Variable stiffness actuation has recently attracted great interest in robotics, especially in areas involving a high degree of human–robot interaction. After investigating various design approaches for variable stiffness actuated (VSA) robots, currently the focus is shifting to the control of these systems. Control of VSA robots is challenging due to the intrinsic nonlinearity of their dynamics and the need to satisfy constraints on input and state variables. Contrary to the partially open-loop state-of-the-art approaches, in this paper, we present a close-loop control framework for VSA robots leveraging recent increases in computational resources and advances in optimization algorithms. In particular, we generate reference trajectories by means of open-loop optimal control, and track these trajectories via nonlinear model predictive control in a closed-loop manner. In order to show the advantages of our proposed scheme with respect to the previous (partially open-loop) ones, extensive simulation and real-world experiments were conducted using a two link planar manipulator for a ball throwing task. The results of these experiments indicate that the closed-loop scheme outperforms the partially open loop one due to its ability to compensate for model uncertainties and external disturbances, while satisfying the imposed constraints.

INDEX TERMS Robot manipulation, variable stiffness actuation, model predictive control, optimization algorithms, embedded optimization.

I. INTRODUCTION

Despite recent technological advances, robots are still outperformed by humans in tasks requiring dexterity, safety and efficiency. Human-like performance would allow robots to be further utilized in areas such as medical, search and rescue, and social robotics. It would also allow industrial robotics to shift to a new paradigm, where robots and humans are working together collaboratively and safely without the need of protective barriers. The potential of achieving human level performance inspired researchers to design anthropomorphic robots [1]. After initial attempts focusing on the kinematic configuration of the robots, the research focused on their dynamic behavior in situations such as collisions, interaction with humans, objects and the environment. One approach is the design of light-weight robots with integrated torque sensors and active torque control [2]. In spite of the presence of successful and already commercialized examples, these robots have inherent limitations [3]. Firstly, the actuators in the joints are not decoupled from the links, and compliant behavior is achieved by active control: therefore, short-lasting impacts might exceed the torque limits of the gearbox and actuators, and presumably damage the system. Secondly, on the contrary to the human musculoskeletal system, these robots do not have the ability to store energy induced in their link-side structure, which limits their velocity and dynamic force in tasks such as throwing, jumping and running.

Variable stiffness actuated (VSA) robots [4]–[10] have recently emerged in order to cope with the challenges faced by the actively torque-controlled light-weight robots. VSA robots are usually intrinsically compliant, and elastic elements in the joints are employed to store energy. In this way, the robots can decrease energy consumption in repetitive tasks, increase maximum force and velocity capabilities, and easily absorb impacts. Moreover, the variable stiffness behavior of the robot can protect both the robot joints and

the human. However, these benefits are also accompanied by certain disadvantages: the design gets more complicated due to the need of incorporating elastic elements and their associated additional actuators, and the absolute position accuracy and mechanical bandwidth are reduced due to the higher elasticity. The parameter identification, task planning and control of VSA robots are also challenging problems: due to coupled nonlinear dynamics, actuation constraints and high number of control inputs, intuitive tuning of the controllers cannot utilize the full potential of VSA robots. This latter can be achieved by exploiting the natural dynamics of the system, for which the controller should optimally modulate the stiffness of the actuators during operation. Controllers based on this idea were used to decrease energy consumption, improve human-robot interaction safety, and increase performance in explosive movement tasks [11]–[14]. A general framework for the optimal control of robots driven by compliant actuators was presented in [15], in which firstly the motors with fast dynamics are controlled in closed-loop. Then, the closed-loop motor dynamics with actuation constraints are incorporated into the complete dynamic representation of the VSA robot, and a sequence of reference motor positions is determined by solving an optimal control problem (OCP [16]) *offline*, with the aim of carrying out a specific task.

As pointed out in [15], full closed-loop control would be needed to deal with the presence of external disturbances and imperfect knowledge of the model parameters (hereafter, we will refer to both of them together simply as *uncertainties*). A closed-loop control law for VSA robots should be able to directly handle multivariable coupled nonlinear systems, at the same time taking all the constraints into account, while updating the control law at each sampling time based on sensors measurements. The best candidate with such characteristics is model predictive control (MPC) [17]. Given the intrinsic nonlinear dynamics of VSA robots, nonlinear MPC (NMPC) is needed, which, however, has the drawback of requiring the solution of a non-convex optimization problem at each sampling time. Convex optimization problems, for which faster and more reliable solvers are available, can instead be formulated in case of linear MPC. For a general overview on convex and non-convex optimization problem, the reader is referred to [18]. Since solving a numerical optimization problem is typically a computationally expensive task, linear and nonlinear MPC have been traditionally used to control slow processes, such as chemical plants, with sampling intervals in the order of seconds or minutes (see, e.g., the survey [19]). On the other hand, controlling robotic systems typically requires much shorter sampling intervals. The difficulties related to the combined presence of computational complexity and short sampling interval probably constitute the main reason why NMPC, to the best of our knowledge, has never been experimentally implemented for VSA robots.

Due to recent advances in efficient solvers and formulations for fast MPC, and to the availability of more powerful microprocessors, the described situation is changing. The use of MPC (especially for the simpler case of linear systems) is now been extended from the classical process control applications to applications requiring faster sampling rates in areas such as mechatronics, automotive, and power electronics [20]–[26].

In this paper, our main contribution is the development and experimental verification of an NMPC-based framework for controlling VSA robots. As a first step, after obtaining a dynamical model of the robot, an OCP is generated similarly to [15], in order to find the sequence of input and state values that minimize the cost function associated to the considered task. However, instead of directly feeding the control sequence into the system in an open-loop fashion, the time evolution of the robot links generated via OCP (referred to as *ideal trajectory* in the following) is used as a reference to be tracked by an NMPC controller. The overall block scheme of the proposed control law (the details of which will be explained in the remainder of the paper) is shown in Fig. 1, also in comparison with the partially open-loop approach proposed in [15]. The NMPC controller, based on real-time data from sensors, generates the closed-loop control action by running an a-priori fixed number of iterations of a finite-horizon optimal control problem (FHOCP) at each sampling instant. The implementation of a fixed number of iterations of the numerical solver, leading to inexact (typically, sub-optimal) solutions, is usually sufficient to obtain a satisfying performance. This evidence is also supported by theoretical results on closed-loop stability using inexact numerical solutions [27]–[29].

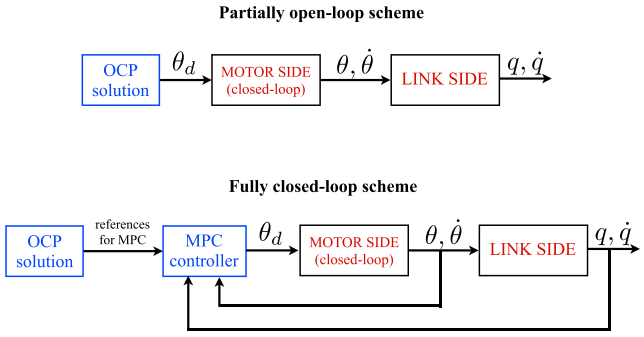

FIGURE 1. Comparison between the partially open-loop scheme of [15] and the proposed NMPC-based fully closed-loop scheme.

In order to provide a general approach for running NMPC in a framework that can be easily used for different VSA robots, we specifically refer to the ACADO tool [30], [31]. This choice is dictated by different reasons: ACADO is open source, implements high-performance numerical solvers, provides a rather intuitive syntax in C++, and allows the designer to generate the controller routine in C++.

As a case study, we consider the control of a two-link variable stiffness actuated manipulator actuated by four motors via four nonlinear elastic elements, with the goal of maximizing the distance at which a ball attached to the end effector is thrown, given a fixed time interval available for

IEEE Access
The journal for rapid open access publishing

the robot motion. Simulations and real-world experiments, in which external disturbances or parameter variations are introduced, are conducted to assess the effectiveness of our approach compared to the scheme proposed in [15].

The paper is organized as follows: Section II introduces the main notation used throughout the paper, while Section III briefly recalls the main concepts related to the modeling of VSA robots. The OCP is described in Section IV, while the MPC problem is formulated and discussed in Section V. Section VI presents the case study, while conclusions are drawn in Section VII.

II. NOTATION

Let $\mathbb{R}$, $\mathbb{R}_{>0}$, $\mathbb{Z}_{\geq 0}$, and $\mathbb{Z}_{>0}$ denote the sets of reals, positive reals, non-negative integers, and positive integers, respectively. Given a vector $a \in \mathbb{R}^n$, a' denotes its transpose, and $\|a\|$ its Euclidean norm. Given a square matrix $A \in \mathbb{R}^{n\times n}$, A' denotes its transpose. Given a symmetric matrix $S = S'$, its positive and semi-positive definiteness are indicated as $S \succ 0$ and $S \succeq 0$, respectively. Given $a \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n\times n}$, the weighted Euclidean norm is defined as $\|a\|_A \triangleq \left(a'Aa\right)^{\frac{1}{2}}$. A diagonal matrix $M \in \mathbb{R}^{n\times n}$ whose diagonal elements are $m_1, m_2, \ldots, m_n$ is referred to as $M = \text{diag}(m_1, m_2, \ldots, m_n)$.

III. MODELING OF VSA ROBOTIC SYSTEMS

In order to model VSA robots, two sets of coordinates have to be considered. A first set $q \in \mathbb{R}^{n_q}$ represents the joint angles of the robot, like in standard rigid manipulators. In addition to that, another set of coordinates $\theta \in \mathbb{R}^{n_\theta}$ accounts for the motor angles of the compliant actuators, reflected through gear reduction. Vectors q and θ are referred to as the *link-side* and *motor-side* coordinates, respectively [15]. The link-side dynamics is described, using the Lagrangian formalism, by

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + D\dot{q} + G(q) = \tau_E(q,\theta) \tag{1}$$

where $M(q) \in \mathbb{R}^{n_q\times n_q}$ is the inertia matrix of the rigid part of the robot ($M = M' \succ 0$), $C(q,\dot{q}) \in \mathbb{R}^{n_q\times n_q}$ represents the contribution of normal inertial forces and Coriolis forces, $D\dot{q} \in \mathbb{R}^{n_q}$ and $G(q) \in \mathbb{R}^{n_q}$ account for the viscous friction and gravity force terms, respectively. On the right-hand side, $\tau_E(q,\theta) \in \mathbb{R}^{n_q}$ are the joint torques generated by the elastic elements that affect the link-side dynamics. Since the motor-side dynamics is typically faster than the link-side dynamics, and since the stiffness modulation depends on the value of θ, a separate control loop is usually employed for the position control of the motor side. The torque generated by each motor, which would be the "physical" input to the system, becomes a function of the motor variables θ and $\dot{\theta}$, and of the reference angular positions, namely $\theta_d \in \mathbb{R}^{n_\theta}$. Under the standard assumption of high gear reduction, and/or of high-gain feedback position controllers, it is well known that the closed-loop dynamics of each motor can be represented by a linear second-order model, as

$$\ddot{\theta}_i + 2\zeta_i\dot{\theta}_i + \kappa_i^2\theta_i = \kappa_i^2\theta_{d,i},\ i = 1,\ldots,n_\theta \tag{2}$$

where θ_i and $\theta_{d,i}$ are the i-th components of θ and θ_d, respectively, while $\zeta_i, \kappa_i \in \mathbb{R}_{>0}$ are constants associated to each motor dynamics. Their values can be related to the parameters of the motors and of their position controllers (see [15, Sec. III]). Typically, position control is already implemented in the available commercial servomotors, and it is tuned such that the closed-loop dynamics is critically-damped, i.e., $\zeta = \kappa$, in order to obtain the fastest possible response with no overshoot.

In order to obtain an overall nonlinear state-space model of the system, we define the state vector

$$x \triangleq \begin{bmatrix} q' & \dot{q}' & \theta' & \dot{\theta}' \end{bmatrix}' \in \mathbb{R}^{n_x},\ n_x \triangleq 2n_q + 2n_\theta,$$

and the input vector $u \triangleq \theta_d \in \mathbb{R}^{n_\theta}$. Let $B \triangleq \text{diag}\{2\zeta_i\} \in \mathbb{R}^{n_\theta\times n_\theta}$ and $K \triangleq \text{diag}\{\kappa_i^2\} \in \mathbb{R}^{n_\theta\times n_\theta}$. Overall, the state-space model of the system is

$$\begin{aligned} \dot{x} &= f(x,u) \\ &= \begin{bmatrix} \dot{q} \\ -M^{-1}\left(C(q,\dot{q})\dot{q} + D\dot{q} + G(q) - \tau_E(q,\theta)\right) \\ \dot{\theta} \\ -B\dot{\theta} - K\theta + K\theta_d \end{bmatrix} \end{aligned} \tag{3}$$

where $f(\cdot,\cdot)$ is in general a continuously differentiable function with respect to both its arguments.

IV. OCP FOR VSA ROBOTIC SYSTEMS

For the control of VSA robots, it is natural to define tasks formulated as OCPs. The goal of the OCP is to determine the optimal realization $u_{ol}^*(t)$ of the desired open-loop input sequence $u_{ol}(t) \in \mathbb{R}^{n_\theta}$ in a fixed time interval $[0,\tilde{T})$, $\tilde{T} \in \mathbb{R}_{>0}$. In our approach, $u_{ol}(t)$ is assumed to be a piecewise-constant signal with fixed discretization step $T_s \in \mathbb{R}_{>0}$. More precisely, given $k \in \mathbb{Z}_{\geq 0}$,

$$u_{ol}(t) = u_{ol}(kT_s),\ \forall\, t \in [kT_s, (k+1)T_s). \tag{4}$$

It is assumed that $\tilde{T}$ is an integer multiple of T_s, i.e., there exists $\tilde{N} \in \mathbb{Z}_{>0}$ such that $\tilde{T} = \tilde{N}T_s$. As a consequence, the whole information contained in $u_{ol}(t)$ for $t \in [0,\tilde{T})$ can be summarized by the finite sequence

$$\mathbf{u_{ol}} \triangleq \{u_{ol}(0), u_{ol}(T_s), \ldots, u_{ol}((\tilde{N}-1)T_s)\}. \tag{5}$$

By denoting the corresponding optimal sequence by $\mathbf{u_{ol}^*}$, the OCP can be written as

$$\mathbf{u_{ol}^*} = \arg\min_{x(0),\mathbf{u_{ol}}} J(x(\cdot),\mathbf{u_{ol}}) \tag{6a}$$

$$\text{subject to } x(t) = x(0) + \int_0^t f(x(\tau),u_{ol}(\tau))d\tau \tag{6b}$$

$$x(t) \in \tilde{\mathcal{X}},\quad \forall t \in [0,\tilde{T}] \tag{6c}$$

$$u_{ol}(t) \in \tilde{\mathcal{U}},\quad \forall t \in [0,\tilde{T}). \tag{6d}$$

The cost function is typically defined as

$$J(x(\cdot),\mathbf{u_{ol}}) = h(x(\tilde{T})) + \int_0^{\tilde{T}} g(x(t),u_{ol}(t))dt. \tag{6e}$$

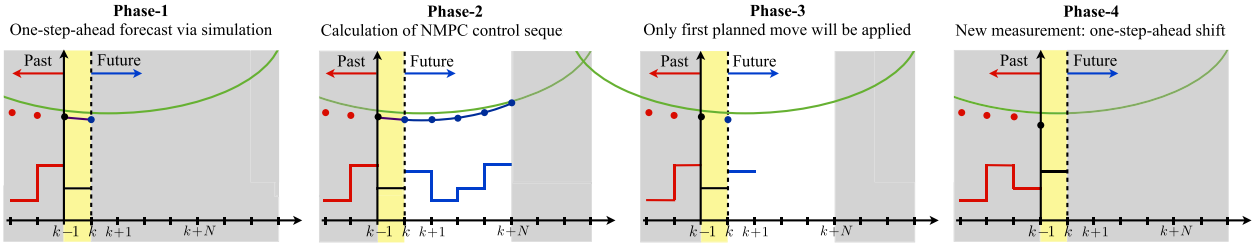

FIGURE 2. Different phases of the NMPC control law computation for closed-loop control. Detailed explanations of the phases are presented in Section V-A.

The term $h(\cdot)$, called *Mayer term*, is related to the terminal state: it can be used, for instance, to impose the minimization of the distance at time $\tilde{T}$ of the end effector with respect to a reference value. On the other hand, $g(\cdot,\cdot)$, called *Lagrange term*, accounts for the behavior of the robot during the task, and can be used in order, for instance, to minimize the energy consumption, or the actuators wear. The sets $\tilde{\mathcal{X}}$ and $\tilde{\mathcal{U}}$ are in general constructed in order to take into account limits on angular positions and velocities of the motors, displacement of the elastic elements, angular positions and velocities of the links. The guarantee of satisfying such constraints is of paramount importance for VSA robots: a control strategy which did not consider them could easily lead to permanent damage of the robot components. The optimal open-loop input signal $u^*_{ol}(t)$ is obtained for $t \in [0, \tilde{T})$ from the sequence $\mathbf{u}^*_{\mathbf{ol}}$, together with the corresponding evolution of the state variables $x_{ol}(t)$.

Remark 1: In order to numerically solve the OCP, the evolution of the state is obtained by discretizing its continuous evolution with a given number of sub-intervals, and by employing numerical integration algorithms, such as Runge-Kutta methods [32], which are implemented in ACADO. A nonlinear optimization problem needs to be solved in order to obtain the OCP solution, for which different approaches exist [16]. In this paper, exploiting the structure of the discretized nonlinear optimization problem, the so-called *multiple shooting* approach is used, together with condensing techniques [33]. Even though different numerical solvers could be employed, we limit our brief description to those implemented in ACADO [30], which make use of sequential quadratic programming (SQP) based on quasi-Newton Hessian approximations. The underlying quadratic programs are solved using the tool qpOASES [34].

V. CLOSED-LOOP CONTROL FRAMEWORK

A. NMPC FORMULATION

The solution here proposed consists of introducing a closed-loop controller for the overall robot dynamics. The evolution of the variables of interest obtained with the OCP (ideal trajectory) is used as reference by the NMPC controller. After reading the measured values of all the state variables, the controller runs a finite number of iterations of a FHOCP at each of the fixed time instants $t = kT_s$, where T_s is the sampling interval (coinciding with the discretization step of the OCP), while $k \in \mathbb{Z}_{\geq 0}$. The generated control variable $u(t)$, as in the case of the OCP control variable, is assumed to be a piecewise-constant signal, i.e.,

$$u(t) = u(kT_s), \quad \forall\, t \in [kT_s, (k+1)T_s). \tag{7}$$

A conceptual diagram of how the NMPC control law is computed is shown in Fig. 2, in which all time indices are normalized by T_s and a single state and control variable are present for clarity. A detailed explanation of each phase is given in the following.

1) ONE-STEP AHEAD FORECAST VIA SIMULATION

For robotics applications, the time intervals needed to obtain the state value from the sensors (T_{sens}), determine the value of the next control move (T_{solv}), and transmit the computed input command to the actuators (T_{tran}), are not negligible with respect to the sampling interval T_s. Therefore, it is not possible to assume that $u(kT_s)$ be computed based on the measurement of $x(kT_s)$ and instantaneously applied to the system at time kT_s. Instead, we propose that the control law, computed using the state value at time $(k-1)T_s$, be applied at time kT_s under the reasonable assumption that

$$T_{sens} + T_{solv} + T_{tran} \leq T_s. \tag{8}$$

Directly applying the control law with a delay of T_s without taking into account the evolution of the system states during the sampling time would degrade the system performance. Instead, we follow the solution shown in Fig. 2 (Phase 1). The green line represents the ideal state trajectory (from OCP) to be tracked, shown as a solid line. The grey band on the left represents the past: the red dots are the measured past values of the state variable, while the piecewise-constant red signal represents the past control moves applied to the system. When time enters the yellow band, which represents the current sampling interval, a control move is applied (black constant signal within the yellow band), which had been computed during the previous sampling interval through NMPC. As a particular case, at $k = 0$, since no NMPC move was computed before, one can apply $u(t) = u_{ol}(t)$ for $t \in [0, T_s)$, in the reasonable assumption that $x(0) \approx x_{ol}(0)$. In general, at the beginning of each sampling interval, a measurement of the state variable $x((k-1)T_s)$ is acquired (black dot in Fig. 2 (Phase 1)). Instead of starting computing the NMPC law that will be applied at time kT_s from the measured

state value, the control routine simulates the system dynamics for one sampling interval, which is typically a computationally negligible effort. The simulation is represented as a purple line in Fig. 2 (Phase 1), which ends at the blue dot representing the estimate of $x(kT_s)$, computed as

$$\hat{x}(kT_s) \triangleq x((k-1)T_s) + \int_{(k-1)T_s}^{kT_s} f(x(\tau), u((k-1)T_s))d\tau. \tag{9}$$

This is a very simple strategy to effectively handle time delays in NMPC (the reader is referred to the introduction of [35] for a discussion of the topic).

2) CALCULATION OF NMPC CONTROL SEQUENCE

The goal of the FHOCP, based on $\hat{x}(kT_s)$ as initial condition, is to determine a realization of $u(t)$, namely $u^*(t)$, in the interval $t \in [kT_s, (k+N)T_s)$, where $N \in \mathbb{Z}_{>0}$ ($N \ll \tilde{N}$) is called *prediction horizon*. Given the piecewise-constant nature of the control law, the whole information contained in $u(t)$ for $t \in [kT_s, (k+N)T_s)$ can be summarized by the finite sequence

$$\mathbf{u}(kT_s) \triangleq \{u(kT_s), u((k+1)T_s), \ldots, u((k+N)T_s)\}. \tag{10}$$

Using a standard notation, the corresponding control sequence $\mathbf{u}^*(kT_s)$ is determined by running a finite number of iterations of the FHOCP defined in the following:

$$\underset{\hat{x}(kT_s), \mathbf{u}(kT_s)}{\text{minimize}} \quad \Phi(x(\cdot), \mathbf{u}(kT_s)) \tag{11a}$$

$$\text{subject to } x(t) = \hat{x}(kT_s) + \int_{kT_s}^{t} f(x(\tau), u(\tau))d\tau, \tag{11b}$$

$$x(t) \in \mathcal{X}, \quad \forall t \in [kT_s, (k+N)T_s], \tag{11c}$$

$$u(t) \in \mathcal{U}, \quad \forall t \in [kT_s, (k+N)T_s). \tag{11d}$$

The sequence $\mathbf{u}^*(kT_s)$ can be seen in Fig. 2 (Phase 2) as a piecewise-constant blue signal, while the corresponding predicted evolution of the state is represented by the solid blue line starting at $\hat{x}(kT_s)$, in which the blue dots represent the values of the state corresponding to the changes of sampling intervals. In order to better react to uncertainties, it is advisable to impose slightly looser constraints in the FHOCP as compared to the OCP, i.e., $\mathcal{X} \supseteq \tilde{\mathcal{X}}$ and $\mathcal{U} \supseteq \tilde{\mathcal{U}}$. In this way, the NMPC controller can use the additional freedom to reduce the deviations from the ideal trajectory. The cost function in (11a) is defined as

$$\begin{aligned}\Phi(x(\cdot), \mathbf{u}(kT_s)) &\triangleq \|x((k+N)T_s) - x_d((k+N)T_s)\|_S^2 \\ &+ \int_{kT_s}^{(k+N)T_s} \Big(\|x(t) - x_d(t)\|_Q^2 + \|u(t) - u_d(t)\|_R^2\Big)dt\end{aligned} \tag{12}$$

where $S \in \mathbb{R}^{n_x \times n_x}$, $Q \in \mathbb{R}^{n_x \times n_x}$, $R \in \mathbb{R}^{n_\theta \times n_\theta}$ are chosen such that $S, Q \succeq 0$, and $R \succ 0$. The signals $x_d(t)$ and $u_d(t)$, defined for $t \in [0, T]$, represent the desired evolution of the system variables. A possibility could be to completely draw them from the OCP solution, i.e. $x_d(t) \equiv x_{ol}(t)$, and $u_d(t) \equiv u_{ol}(t)$. As an alternative, we propose the following approach:

- The state variables of the link-side dynamics (i.e., q and $\dot{q}$) are typically related to the task performance, and their evolution determined in the OCP should be tracked as closely as possible. Therefore, the values of the first $2n_q$ components of x_d (the reference values for q and $\dot{q}$) should coincide with the first $2n_q$ components of $x_{ol}(t)$.
- The state variables related to the motor-side dynamics (i.e., θ and $\dot{\theta}$) are usually "intermediate variables" between the control inputs and the link-side state variables. For this reason, the values of the last $2n_\theta$ components of x_d (the reference values for θ and $\dot{\theta}$) can be set as constant signals, corresponding to those leading to the minimum wear of the robot elements. For instance, the reference position for a motor can be the one for which a spring connected to it is uncompressed, while the corresponding reference velocity can be set to zero.
- The reference control variables u_d (reference angular positions given to the servomotors) can also be set as those which cause the minimum wear and/or energy consumption of the system.

The last ingredients of (12) to be described are the weight matrices Q, R, and S. It is a common practice to choose them as diagonal matrices, and we follow this approach. For example, $Q \triangleq \text{diag}\{\bar{q}_1, \ldots, \bar{q}_{n_x}\}$ implies

$$\|x(t) - x_d(t)\|_Q^2 = \sum_{i=1}^{n_x} \bar{q}_i \left(x_i(t) - x_{d,i}(t)\right)^2$$

meaning that every element of the diagonal determines the weight of the corresponding state variable. The elements of the diagonal matrix R are referred to as $\bar{r}_i$, and those of S as $\bar{s}_i$. Since the main purpose of the NMPC controller is to track the ideal evolution of the link-side state variables, then $\bar{q}_i$ for $i = 1, \ldots, 2n_q$, should be chosen much larger than $\bar{q}_i$ for $i = 2n_q + 1, \ldots, n_x$, in order to obtain small tracking errors. The same values of $\bar{q}_i$, $i = 2n_q + 1, \ldots, 2n_q + n_\theta$ (i.e., those relative to θ) can be replicated in $\bar{r}_i$, $i = 1, \ldots, n_\theta$. Larger values of $\bar{s}_i$ would lead to a greater importance of the tracking error at the end of the prediction horizon, whereas larger values of $\bar{q}_i$ would reduce the tracking error during the prediction horizon. In order to weight the tracking error at $(k+N)T_s$ as the tracking error in one of the sampling intervals of the prediction horizon, one can impose $S = T_s Q$, but different formulations can be chosen depending on the specific task.

3) ONLY FIRST PLANNED MOVE WILL BE APPLIED

The whole determined sequence is not directly used: instead, following the so-called *receding horizon* principle, only the first control move calculated for $t \in [kT_s, (k+1)T_s)$, namely $u^*_{RH}(kT_s)$, is applied to the system (Fig. 2 (Phase-3)).

4) NEW MEASUREMENT: ONE-STEP-AHEAD SHIFT

As the new measurement is acquired, the control move just computed by NMPC is applied, and the time window used for the prediction is shifted one sampling interval ahead (as can be seen by the time indices in Fig. 2 (Phase-4)). A new one-step simulation will be run, together with a new FHOCP. The name *receding horizon*, that provides the name for what is realized in steps 3 and 4, derives from the one-step-ahead shift of the prediction window. This is a key concept, that makes MPC a closed-loop technique: for this reason, MPC is also referred to as "receding horizon control". The black dot representing the measured state in Fig. 2 (Phase-4) has been placed on purpose at a different value than its corresponding value expected from the one-step simulation: this means that the actual evolution of the system dynamics, due to the presence of uncertainties, is slightly different than expected.

Remark 2: In order to solve the FHOCP, the employed method is in principle similar to the one used for solving the OCP, i.e., the state evolution is discretized, and a fixed number of iterations of the resulting nonlinear optimization problem is run by multiple shooting using SQP, running, in turn, a fixed number of iterations of the underlying quadratic program with qpOASES. However, while in the OCP case the optimization procedure could take, in principle, as long as required, in a closed-loop NMPC implementation for robotic systems, few milliseconds would be typically available. However, by warm-starting the FHOCP (i.e., using information on the solution at the previous sampling time to initialize the optimization problem) and using relatively small values of N, an acceptable solution can be obtained in the available computation time. In case a solution of the FHOCP is not provided within the sampling interval, due to problems in reading the sensors, or computational issues, the second NMPC move computed at the previous sampling time can be applied to the system. This constitutes an intrinsic fault tolerance of the control scheme, which can run in open loop for a few consecutive iterations. We refer the reader to [35]–[37] for further details on the real-time iteration scheme implemented in ACADO.

B. REMARKS ON THEORETICAL GUARANTEES

In order to use a simple and non-conservative NMPC formulation, we follow a standard practice in MPC applications, that is applying a strategy that does not give any theoretical guarantees on recursive feasibility and closed-loop stability in the presence of uncertainties, and experimentally test its effectiveness (as it will be shown in the case study). Nonetheless, we prove in the following a simple result, which guarantees perfect tracking in the ideal case.

Proposition 1: Assume that the solution of the OCP (6), namely $u_{ol}(t)$ and $x_{ol}(t)$ for $t \in [0, \tilde{T}]$, has been obtained for system (3). An NMPC controller is then used to track the link-side dynamics, by solving the FHOCP (11) online, for $k = 0, \ldots, \tilde{N} - N$. Assume, for the sake of simplicity, that $x_d(t) \equiv x_{ol}(t)$, $u_d(t) \equiv u_{ol}(t)$, and that the number of iterations of the FHOCP is large enough so that its optimal solution is achieved at each sampling time. Also, $\mathcal{X} \supseteq \tilde{\mathcal{X}}$ and $\mathcal{U} \supseteq \tilde{\mathcal{U}}$. If $x(0) = \tilde{x}(0)$, and $x(t)$ is generated by solving the FHOCP (11), then $x(t) \equiv \tilde{x}(t)$, for all $t \in [0, \tilde{T}]$.

Proof: During the time interval $t \in [0, T_s)$, the input $u(t) \equiv u_{ol}(t)$ is applied to the system. Since the system dynamics is exactly represented by (3), one has $x(t) \equiv x_{ol}(t)$ for $t \in [0, T_s]$. Being $\hat{x}(T_s) = x^*_{ol}(T_s)$ and $\mathcal{X} \supseteq \tilde{\mathcal{X}}$, $\mathcal{U} \supseteq \tilde{\mathcal{U}}$, then $u^*_{RH}(T_s) = u^*_{ol}(T_s)$ is a feasible solution of (11) for $k = 1$. Moreover, setting $u^*_{RH}(T_s) = u^*_{ol}(T_s)$ leads to $\Phi(x(\cdot), \mathbf{u}^*(0)) = 0$, meaning that $u^*_{ol}(T_s)$ is the global minimizer for the FHOCP that will be ideally determined by the optimization algorithm. The application of this solution leads to perfect tracking of the state variables in $t = [T_s, 2T_s]$, and, in particular, yields $x(2T_s) = \tilde{x}(2T_s)$. The same reasoning can be applied in sequence to all subsequent steps. The proposition is therefore proved by induction. ■

In case only the link-side components of the state are taken as references from the OCP solution, and the weight terms $\bar{q}_i$ and $\bar{s}_i$ associated with them are much larger than the other weight terms, we expect a nearly perfect tracking of the link-side variables in the nominal case, while the evolution of the motor-side variables and the input signals can follow a different pattern as compared to the OCP solution. Also, in the perturbed case, we expect that the tracking performance be slowly degrading as the importance of the uncertainties increases, but we also expect better performance than applying the OCP control sequence in an open-loop fashion.

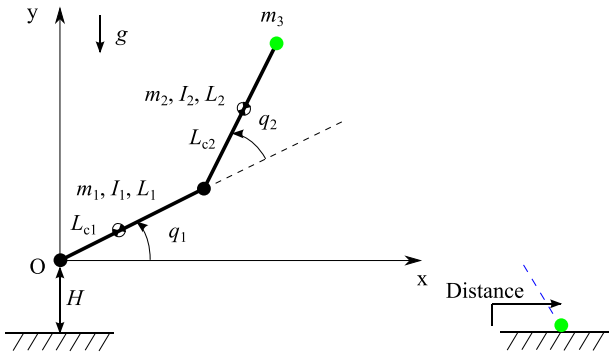

FIGURE 3. Schematic drawing of the two-link planar manipulator.

VI. CASE STUDY

A. SYSTEM MODELING

The considered robot is a two-link planar manipulator, with a ball attached to the end effector (Fig. 3) actuated by four servomotors connected to an equal number of nonlinear elastic elements (NEEs). The link-side dynamics (1) is analyzed first, and the vector of link-side angles is defined as $q = [q_1 \; q_2]' \in \mathbb{R}^2$. The inertia matrix $M(q)$ has the following form

$$M(q) = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \end{bmatrix} \tag{13}$$

with

$$m_{11} = I_1 + m_1 L_{c1}^2 + I_2 + m_2(L_1^2 + 2L_1 L_{c2}\cos q_2 + L_{c2}^2) + m_3(L_1^2 + 2L_1 L_2 \cos q_2 + L_2^2)$$
$$m_{12} = I_2 + m_2(L_{c2}^2 + L_1 L_{c2}\cos q_2) + m_3(L_2^2 + L_1 L_2 \cos q_2)$$
$$m_{22} = I_2 + m_2 L_{c2}^2 + m_3 L_2^2$$

where I_i, m_i, L_i, L_{ci} are correspondingly (for link $i = 1, 2$) its moment of inertia around the center of mass, its mass, its length and the length from the joint center of the link to the center of mass of the link, while m_3 is the mass of the ball. Also,

$$C(q, \dot{q}) = -(m_2 L_{c2} + m_3 L_2) L_1 \sin q_2 \begin{bmatrix} 2\dot{q}_2 & \dot{q}_2 \\ -\dot{q}_1 & 0 \end{bmatrix}, \quad (14)$$

and

$$D = \begin{bmatrix} b_1 & 0 \\ 0 & b_2 \end{bmatrix}, \quad (15)$$

where b_1 and b_2 represent viscous damping constants. Finally,

$$G(q) = \begin{bmatrix} g_1 \\ g_2 \end{bmatrix}, \quad (16)$$

where $g_1 \triangleq g(m_1 L_{c1} + m_2 L_1 + m_3 L_1)\cos q_1 + g(m_2 L_{c2} + m_3 L_2)\cos(q_1 + q_2)$, and $g_2 \triangleq g(m_2 L_{c2} + m_3 L_2)\cos(q_1 + q_2)$.

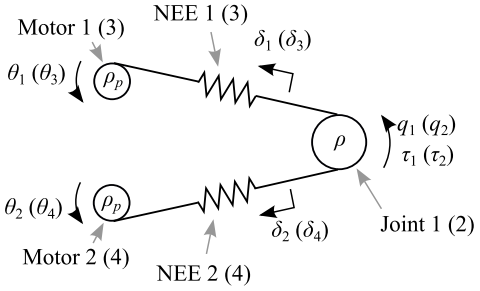

FIGURE 4. Schematic drawing of an antagonistic joint, connected to two motors via nonlinear elastic elements (NEEs). Parameters outside and inside of parentheses are for joint 1 and joint 2, respectively.

Antagonistic configuration of the NEEs is considered for each of the two joints in order to achieve variable-stiffness behavior: therefore, each joint has two NEEs (Fig. 4). The vector of elastic joint torques τ_E can be represented as

$$\tau_E = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = \rho \begin{bmatrix} (T_1 - T_2) \\ (T_4 - T_3) \end{bmatrix} \quad (17)$$

where T_i, $i = 1, .., 4$ is the tension in each of the four tendons, and ρ is the joint radius, which is the same for both joints. As can be seen from Fig. 4, NEE 1 and NEE 2 are attached to the first joint, while NEE 3 and NEE 4 are attached to the second joint. NEEs are designed such that the tendon tensions are quadratic polynomial functions of the tendon displacements: $T_i = \alpha_i {\delta_i}^2 + \beta_i \delta_i + \gamma_i$, $i = 1, \ldots, 4$, where $\alpha_i, \beta_i, \gamma_i$ are coefficients and δ_i is the tendon displacement. The values of δ_i can be calculated from joint geometry and system parameters as $\delta_1 = \delta_0 - \rho(q_1 + \frac{\pi}{2}) + \rho_p \theta_1$, $\delta_2 = \delta_0 + \rho(q_1 + \frac{\pi}{2}) - \rho_p \theta_2$, $\delta_3 = \delta_0 + \rho q_2 + \rho_p \theta_3$, $\delta_4 = \delta_0 - \rho q_2 - \rho_p \theta_4$, δ_0 and ρ_p being the initial displacement of NEEs and the radius of pulleys on motors (which is the same for all motors), respectively. Antagonistic configuration and nonlinear behavior of the NEEs ensure variable compliance actuation. Overall, the system in form (3) has 12 state variables and four input variables.

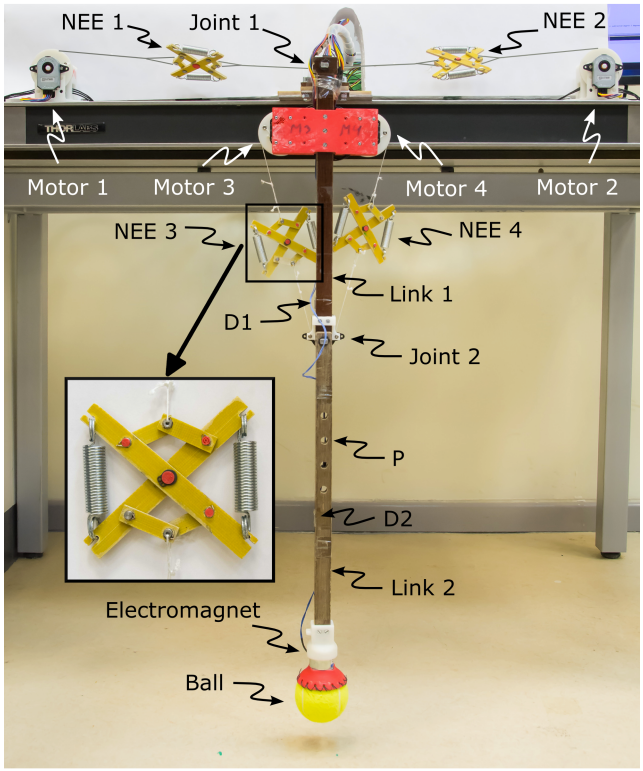

FIGURE 5. Experimental setup of the two link planar robot with variable stiffness actuators. D1 and D2 show the attachment points of the disturbance magnets to links 1 and 2, respectively. P indicates the extra mass connection points for parameter variation experiments.

B. EXPERIMENTAL SETUP

In order to conduct real-world experiments to show the performance of our scheme, the robot described in the previous section was built as a prototype and is shown in Fig. 5. The design was accomplished using SolidWorks. Links and joints were machined from textolite (a type of composite epoxy material) and steel, respectively. Motor holders, motor and joint pulleys were printed using Objet Connex 260 3D printer. Four Dynamixel MX-28T motors with 2.4 Nm stall torque and 67 revolutions per minute no-load speed were used as actuators. Communication between the motors and the control computers was accomplished via a USB2Dynamixel module connected to the USB port of the control computer. Six AMT-102V capacitive incremental encoders with a resolution of 1024 pulses per revolution were used to measure joint and motor angular positions. The encoder measurements were acquired by three National Instruments (NI) PCI-6221 cards. BK Precision 1761 DC Power supplies were

used to provide 12 V to the motors and the electromagnet and 5 V to the encoders. A simple circuit containing a power transistor was used to drive the electromagnet for attaching and releasing the ball. This circuit is controlled via a digital output of one of the PCI-6221 cards. The experimental setup was assembled on an aluminium breadboard (Thorlabs PBH51506). The connections between the motors and the NEEs, and between the NEEs and the joint pulleys were made using polyethylene fibers (Spectra) rated for a load of 400 N.

In order to determine the parameters of the system, all the components of the robot were weighed using electronic weight scale before the assembly. SolidWorks was used to determine the dimensions, volume, moment of inertia and center of mass of different parts of the robot. The friction coefficients (b_1 and b_2) of the joints were determined experimentally. Experiments were performed to characterize each NEE. Specifically, in these experiments each NEE was being pulled by the Dynamixel MX-28T motor by fixed small step-size increments, while a force gauge sensor (Extech Instruments 475055) was used to record force measurements at each step. The obtained experimental force-displacement measurements were used to fit a quadratic curve, the parameters of which were used in OCP and MPC formulations.

Accordingly, the values of the link-side system parameters were found to be: $L_1 = 0.330$ m, $L_2 = 0.433$ m, $L_{c1} = 0.116$ m, $L_{c2} = 0.222$ m, $m_1 = 0.674$ kg, $m_2 = 0.307$ kg, $m_3 = 0.074$ kg, $I_1 = 5.08 \cdot 10^{-3}$ kg·m^2, $I_2 = 6.92 \cdot 10^{-3}$ kg·m^2, $b_1 = 0.0124$ N·m·s, $b_2 = 0.0064$ N·m·s, $\alpha_1 = 12400$ N/m^2, $\beta_1 = 1360$ N/m, $\alpha_2 = 13600$ N/m^2, $\beta_2 = 1350$ N/m, $\alpha_3 = 5320$ N/m^2, $\beta_3 = 1500$ N/m, $\alpha_4 = 13700$ N/m^2, $\beta_4 = 1410$ N/m, $\gamma_i = 0$, $i = 1, .., 4$, $\rho = 0.013$ m, $\rho_p = 0.026$ m, $L_{max} = 0.025$ m, $\delta_0 = 0.005$ m. As for the motor-side dynamics, the parameters in (2) for the Dynamixel MX-28T motors were determined by system identification to be $\zeta_i = \kappa_i = 40.0$.

C. OCP FORMULATION

The main objective of the OCP is to maximize the distance d at which the ball is thrown, given a fixed time interval $\tilde{T} = 4$ s for the motion (which, with the given sampling time $T_s = 20$ ms, leads to $\tilde{N} = 200$). The value of d for any time instant during the robot motion is

$$d = x_r + \dot{x_r}\frac{\dot{y_r} + \sqrt{\dot{y_r}^2 + 2g(y_r + H)}}{g}, \tag{18}$$

where $H = 0.813$ m is the height from the ground to the frame origin, $g = 9.81$ m/s^2, and

$$\begin{aligned} x_r &= L_1\cos(q_1) + L_2\cos(q_1+q_2) \\ y_r &= L_1\sin(q_1) + L_2\sin(q_1+q_2) \\ \dot{x_r} &= -L_1\sin(q_1)\dot{q_1} - L_2\sin(q_1+q_2)(\dot{q_1}+\dot{q_2}) \\ \dot{y_r} &= L_1\cos(q_1)\dot{q_1} + L_2\cos(q_1+q_2)(\dot{q_1}+\dot{q_2}). \end{aligned}$$

While trying to maximize $d(\tilde{T})$, we would like to limit the energy consumption during the robot motion. As each motor velocity $\dot{\theta}_i$ and torque $\mathcal{T}_i = T_i\rho_p$ increases, the energy consumption of a motor also increases, since the elastic potential energy of each NEE is $\mathcal{E}_i = \int T_i d\delta_i$. Therefore, quadratic terms relative to motor velocities and torques are also considered in the OCP cost function (6e), as follows

$$J(x(\cdot), \mathbf{u_{ol}}) = \int_0^{\tilde{T}} \left(\sum_{i=1}^{4} \left(\epsilon_\tau \mathcal{T}_i^2(t) + \epsilon_\omega \dot{\theta}_i^{\,2}(t) \right) \right) dt - d(\tilde{T}) \tag{19}$$

where we set $\epsilon_\tau = \epsilon_\omega = 5 \cdot 10^{-2}$ in order to suppress high-frequency oscillations of the links (higher values for these parameters would lead to more conservative solutions, while lower values would lead to energetic and impulsive behavior). The sets $\tilde{\mathcal{X}}$ in (6c) and $\tilde{\mathcal{U}}$ in (6d) are expressed in the following, where we directly refer to the physical quantities. As for the state constraints,

$$\tilde{\mathcal{X}} \triangleq \left\{ x \in \mathbb{R}^{12} : \begin{array}{rl} -170\pi/180 \le q_1 \le -10\pi/180 & \text{[rad]} \\ -90\pi/180 \le q_2 \le 90\pi/180 & \text{[rad]} \\ 0 \le \mathcal{T}_i \le 1.4 & \text{[Nm]} \\ -5 \le \dot{\theta}_i \le 5, & \text{[rad/s]} \\ 0.005 \le \delta_i \le 0.025, & \text{[m]} \end{array} \right\}$$

where $i = 1, \ldots, 4$. The inequalities are imposed to prevent exceeding physical joint motion limits, to limit motor torques and velocities (the imposed values are compatible with the Dynamixel MX-28T motor, and verified experimentally), and to limit spring compression. Notice that all these inequalities can be directly expressed as only depending on the state vector. The set of input constraints

$$\tilde{\mathcal{U}} \triangleq \left\{ u \in \mathbb{R}^4 \ : \ 0 \le \theta_{d,i} \le 2\pi \ [rad], i = 1, .., 4 \right\}$$

is formulated to avoid multiple motor rotations. The initial condition $x(0)$ for the OCP was obtained by setting vertical hanging position of the two links ($q_1 = -\frac{\pi}{2}, q_2 = 0$), zero link and motor velocities, initial motor and reference motor positions corresponding to initial NEE displacements. For our task, we used the ACADO toolkit (Version 1.2.0) with the solver options described in Remark 1, setting the KKT tolerance (used for the termination criterion of the SQP algorithm) to 10^{-5}. The computation time of the OCP on a desktop computer with 2.4 GHz Intel Xeon E5620 processor and 16 GB of memory was approximately 22 minutes.

D. NMPC FORMULATION

As described in Section V-A, the primary objective of the NMPC controller is to track the OCP evolution of the link-side variables. In order to formulate the NMPC law, we set a prediction horizon $N = 10$, which implies a prediction time of 200 ms. The desired evolution of state and control variables needed for the definition of the cost function (12) has been defined as follows. The components of x_{ol} relative to q and $\dot{q}$ for the two links are used as reference for the corresponding variables of the MPC problem,

since the evolution of the link-side variables is strictly related to the throwing task. As for the other state variables, the references for θ_d and θ were set to values corresponding to minimum spring compression, while the reference values for $\dot{\theta}$ were set to zero, in order to limit energy consumption and wear of the springs. The weight matrices have been defined as $Q = \text{diag}\{1, 1, 0.1, 0.1, 10^{-6}, 10^{-6}, 10^{-6}, 10^{-6}, 10^{-4}, 10^{-4}, 10^{-4}, 10^{-4}\}$, $R = \text{diag}\{10^{-6}, 10^{-6}, 10^{-6}, 10^{-6}\}$, and $S = T_s Q$.

The OCP determines an evolution of input and state variables that is often close to the boundaries of sets $\tilde{\mathcal{U}}$ and $\tilde{\mathcal{X}}$. Due to the presence of uncertainties, these bounds can be violated in practice. Therefore, in order to give additional flexibility to the MPC controller to compensate the effect of uncertainties, some of the boundaries used for the OCP solution were expanded in the MPC formulation, as mentioned in Section V-A2:

$$\mathcal{X} \triangleq \left\{ x \in \mathbb{R}^{12} : \begin{array}{r} -210\pi/180 \le q_1 \le 30\pi/180 \quad \text{[rad]}, \\ -120\pi/180 \le q_2 \le 120\pi/180 \ \text{[rad]}, \\ 0 \le \mathcal{T}_i \le 1.6 \ \text{[Nm]}, \\ -5 \le \dot{\theta}_i \le 5, \ \text{[rad/s]}, \\ 0.003 \le \delta_i \le 0.025, \ \text{[m]}. \end{array} \right\}$$

Using these settings, the MPC controller is generated through ACADO in the form of C++ code and Matlab/Simulink block variants. In both cases, the MPC controller is synthesized according to Remark 2, and the fixed number of iterations of the FHOCP solver in each sampling interval is set to 10.

E. SIMULATION RESULTS

In order to compare the proposed approach with the framework proposed in [15], we test both schemes of Fig. 1 in Simulink environment. For the open-loop case, we feed the input signals obtained from the OCP solution to a Simulink model of the plant. For the closed-loop case, a specific Simulink block generated by ACADO implements the MPC controller, which interacts with the Simulink block modeling the planar VSA manipulator. In both cases, the ODE4 (Runge-Kutta) solver of Simulink is employed with fixed-step size set to 0.2 ms, while the piecewise-constant control signals are defined with $T_s = 20$ ms.

As a first example, both schemes are applied to the system, which is simulated using exactly the same state equations (3) employed in the OCP and in the MPC controller. These results are not shown, but in this nominal case the open-loop scheme perfectly tracks the link-side references, while the closed-loop scheme generates a negligible error. Indeed, perfect tracking even in the nominal case would be obtained only if $x_d(t) \equiv x_{ol}(t)$ and $u_d(t) \equiv u_{ol}(t)$: in our implementation, instead, the reference x_d for the motor-side variables in the MPC controller does not coincide with the OCP solution. This also leads to obtaining reference positions $u_i, i = 1,..,4$, which are not the same as in the OCP, but this occurs because a different solution is obtained, which leads to a similar evolution of the link-side variables. For the open-loop scheme, we obtain the thrown distance $d = 3.80$ m, while for the closed-loop scheme $d = 3.77$ m is obtained (due to the imperfect tracking).

As a second example, the action of additive disturbance terms is considered. In particular, two disturbance torques $D_1(t)$ and $D_2(t)$, of constant but different magnitude, are applied simultaneously in the interval $t = [0.2, \ 0.6]$ s to joints 1 and 2, respectively. The amplitude of the disturbance torques were varied for different simulation instances in order to get a map of d as a function of disturbance magnitudes and direction. Each external torque magnitude is varied from -0.5 N·m to 0.5 N·m in 21 intermediate values. The magnitude of these disturbances is in the same range as the joint torques generated by the actuation system, and can represent for instance the interaction of the robot with an external object during the task. The results of d versus disturbance torques are shown in the upper row of Fig. 6. Despite the fact that, in the open-loop case (Fig. 6a), the thrown distance slightly exceeds that of the nominal case for a few disturbance combinations, d is in general lower than the nominal value of 3.80 m. Moreover, in some simulations the link positions exceeded the imposed constraints, which in real life might damage the system. As shown in Fig. 6b, the closed-loop system achieves the nominal d for most combinations of disturbances. This is especially apparent for combinations of disturbances with the same direction and with values close to maximum ± 0.5 N·m, where d is 3.80 m, while for the open-loop system it is close to 1.50 m. Only for disturbance combinations with opposite sign and with magnitude close to 0.5 N·m the performance of the closed-loop system is slightly worse than the performance of the open-loop system.

As a third example, a parameter variation was taken into account: the masses of the first link (m_1) and of the second link (m_2) were varied in a range of $\pm 25\%$ with respect to their nominal value with 21 intermediate values in order to obtain d as a function of parameters. Fig. 6c, obtained for the open-loop case, shows distance values lower than the ideal d for most combinations. For some values of parameters (m_1 is large while m_2 is close to nominal), d achieves the ideal value or even exceeds it slightly. This might be explained as increasing m_1 leads to higher inertia of the first link, and thus to higher kinetic energy at the last swing before ball release. Fig. 6d shows d obtained using the MPC controller, where in most conditions the nominal distance is achieved, except for parameter values when m_1 is at the lower and m_2 is at the higher end of the range, for which d decreases to 3 m. Similar to the second example, it was observed that the open-loop control caused constraint violation for some parameter combinations, whereas this was not the case for the closed-loop control.

As expected, the proposed scheme manages to limit the performance degradation as compared to the open-loop scheme. For each point of the graphs in Fig. 6 the difference between the nominal distance from OCP and the distance obtained in the simulation was computed. The root mean

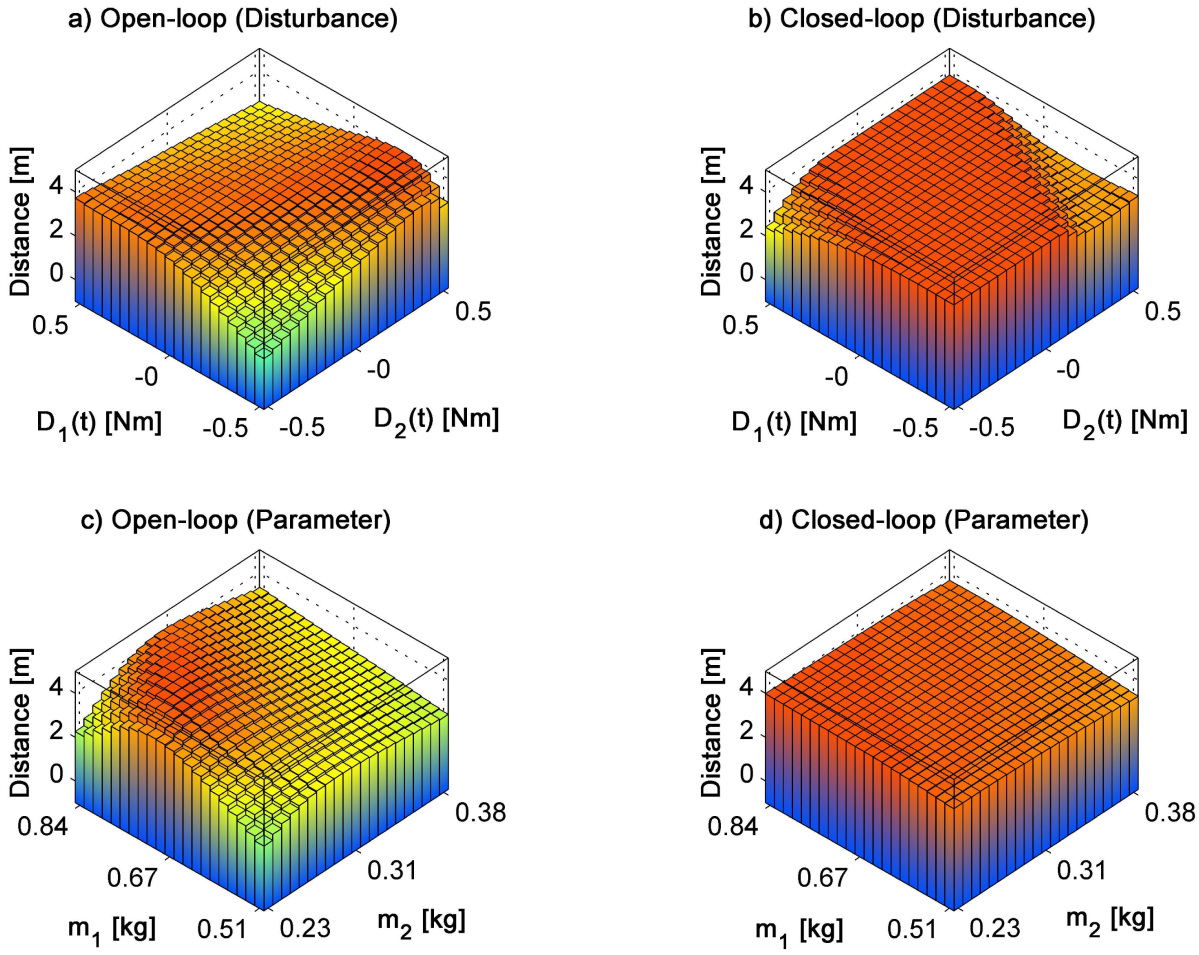

FIGURE 6. Distance thrown as a function of external disturbances applied at two joints (a,b) and of added parameter variation (c,d).

squares of this differences are 0.62 m, 0.29 m, 0.66 m, 0.16 m for the results shown in Figs. 6a, 6b, 6c, and 6d, respectively.

F. EXPERIMENTAL RESULTS

Open-loop and closed-loop implementations of the robot task were written in C++ for real-world experiments. In both programs, the control task is run every 20 ms, which is enforced using high-resolution native timer routines of Microsoft Windows (QueryPerformanceCounter API). In the open-loop case, the input signals obtained from the OCP solution are read at every time step and sent directly over the USB2Dynamixel module to the motors. At each sampling time of the closed-loop experiments, firstly motor and link encoders are read, secondly the robot model is simulated for one sampling time to compute the one-step ahead forecast (Section V-A1), thirdly the NMPC control sequence is computed using the ACADO generated code block (Section V-A2), fourthly the first move of this sequence (Section V-A3) is applied to the system by sending the motor commands via USB2Dynamixel module. In both cases, the electromagnet is turned off to release the ball at $\tilde{T} = 4$ s. The system can maintain the 20 ms control loop, if the condition in (8) is satisfied. In order to check this condition during closed-loop experiments, T_{sens}, T_{solv} and T_{tran} were recorded at each sampling time of six experimental trials resulting in 1200 instances of these three time intervals. Their sum constitutes the total execution time (i.e. $T_{sens} + T_{solv} + T_{tran}$) of the closed-loop controller at each sampling time. A histogram generated using these data is shown in Fig. 7. The maximum execution time is less than 20 ms, which satisfies (8), and shows the real-time feasibility of our approach for the specific task.

Open-loop and closed-loop controllers were tested for three scenarios: nominal case (i.e., without uncertainties except those intrinsically present in the modeling inaccuracies), with external disturbances and with added parameter variation. In the nominal case, the thrown distances for the open-loop and closed-loop control were 2.05 m and 3.22 m, respectively, compared to the 3.80 m ideal distance. Reference, open-loop and closed-loop control joint positions and velocities along with the corresponding motor positions for

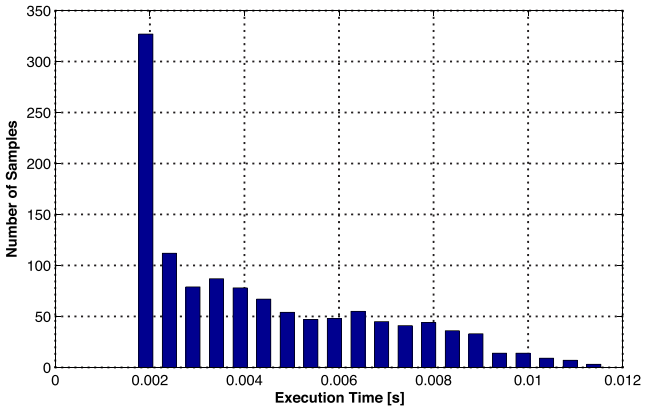

FIGURE 7. Histogram of execution times needed to run the NMPC control routine in the real world experiment. Notice that the maximum execution time is significantly smaller than the sampling interval $T_s = 20$ ms, which constitutes the upper bound.

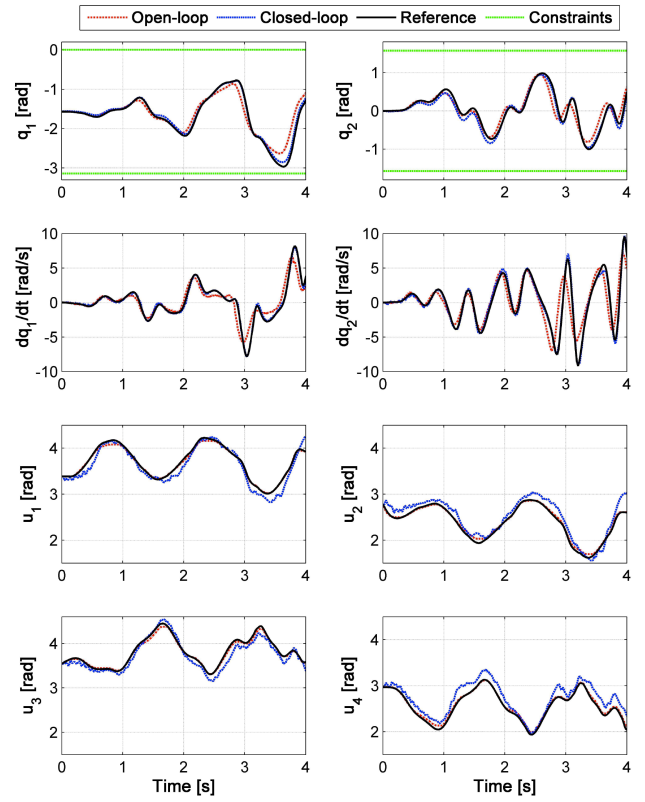

FIGURE 8. Reference, open-loop and closed-loop joint positions and velocities, and motor positions in the nominal case.

the nominal case are shown in Fig. 8. It can be observed that the closed-loop controller tracks the joint positions and velocities, which are the link-side parameters enforced by the controller and also directly effecting the task performance, closer than the open-loop variant. On the other hand, the motor positions in the closed-loop case deviate from their references (not directly affecting the task performance) more than in the open-loop case, so as to achieve better link-side tracking under uncertainties. The superior performance of the closed-loop system can be attributed to its inherent robustness to these uncertainties.

For the experiments aiming to assess the performance of both schemes in the presence of external disturbances, both links were disturbed independently in the first 1.5 s of the experiments. The disturbances were generated using neodymium magnets. One magnet was rigidly attached to each link. Each of these magnets was connected to another identical magnet inside a plastic structure. The plastic structure was connected to the rigid part of the experimental setup using a tendon similar to the ones used for motor connections. The slack on these tendons was used to tune the timing of the disturbances. The application of the disturbances using the magnets is shown in the supplemental material video. The force at which the magnets detach from the link was measured between 1.2 N and 1.6 N, which results in maximum disturbance torques of nearly 0.5 Nm applied to the joints. The weight of the cubic neodymium magnets with 5 mm edge dimension is around 1 g, which can be considered negligible compared to the weights of the links. The performance of both schemes degrades due to the disturbance. Specifically, the thrown distance in the open-loop experiment was 1.63 m (0.42 m less than the real-world experiment in the nominal case), while this distance is 3.04 m for closed-loop (0.19 m less than the real-world experiment in the nominal

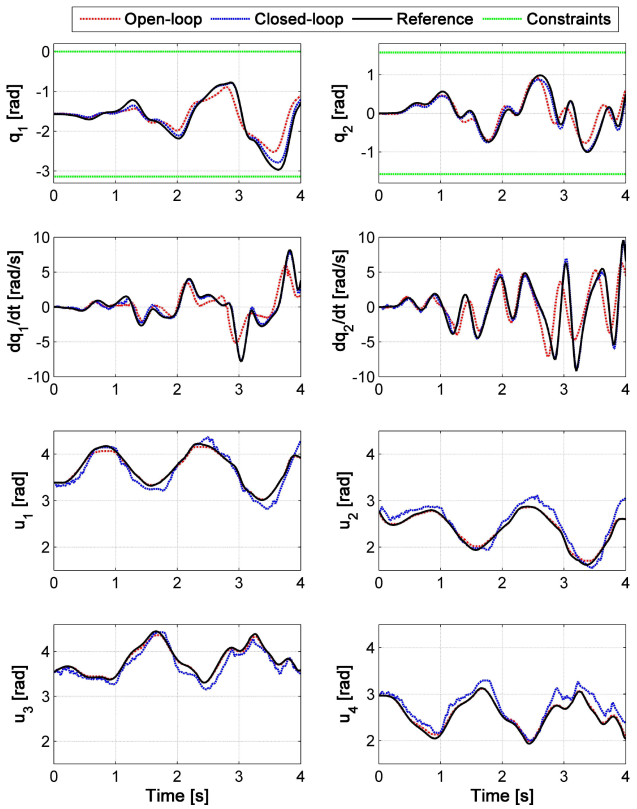

FIGURE 9. Reference, open-loop and closed-loop joint positions and velocities, and motor positions when external torque disturbances are applied at the joints from 0.2 s to 0.6 s.

case) almost doubling the open-loop case. Figure 9 shows the reference, open-loop and closed-loop control joint positions and velocities along with the corresponding motor positions for the disturbance case. After the disturbance is applied, the deviation of the open-loop controlled system from the reference link-side trajectories is higher than the closed-loop one. The compensatory action of the closed-loop controller against the disturbance can be observed from the motor positions. This example demonstrates the advantage of NMPC for disturbance rejection over the open-loop control.

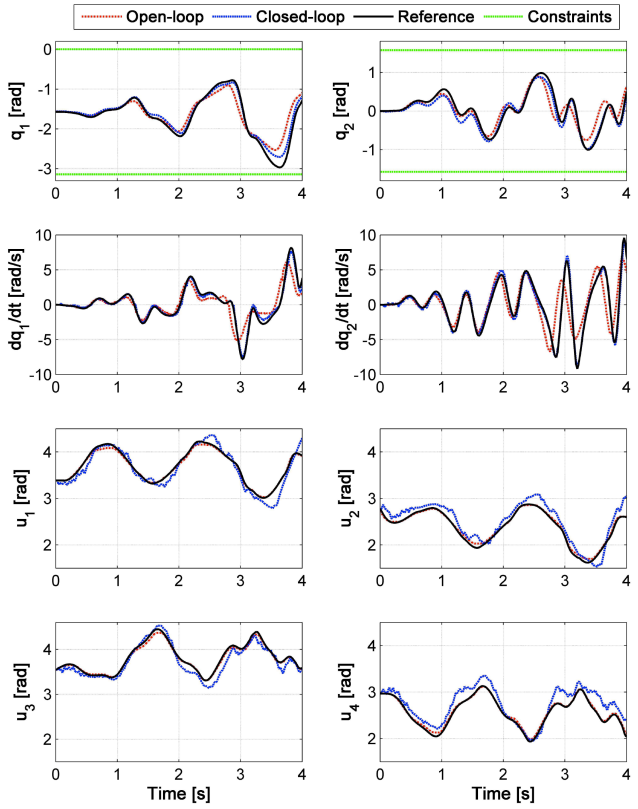

FIGURE 10. Reference, open-loop and closed-loop joint positions and velocities, and motor positions when extra mass is added to the second link as parameter uncertainty.

Finally, both schemes were tested to determine their performance in the presence of parameter variation. The weight of the second link was increased by attaching four M8 screws with nuts to the specially designed slots (see Fig. 5). Each screw and nut couple weights around 18 g, resulting in 72 g weight increase of the second link (which is approximately 23% of the original link weight). The open-loop experiment with this parameter variation results in $d = 1.57$ m, while for the closed-loop experiment $d = 2.71$ m is obtained. Link and motor side trajectories for the open-loop case display increased deviations from the optimal trajectory, compared to the nominal case (Fig. 10). For the closed-loop experiment, link and motor positions still closely follow the reference trajectories. Compared to the nominal case, the decrease of the thrown distance due to added parameter uncertainty is more than the one due to external disturbances. This can be explained by the fact that the external disturbances act only for a finite period of time, and, given enough time to compensate for it, the MPC controller steers the system back to the nominal trajectory. On the other hand, the added parameter variation is similar to the effect of disturbances that act all time when the robot motion is happening.

TABLE 1. Open-loop and closed-loop thrown distance d (m) for the ideal case (from simulation), and the three real-world experiments (nominal, external disturbance and parameter variation).

	Ideal	Nominal	Disturbance	Parameter
Open loop	3.80	2.05	1.63	1.57
Closed loop	3.77	3.22	3.04	2.71

Table 1 summarizes the thrown distances for the considered scenarios. One can notice that, even though the closed-loop MPC scheme achieves larger thrown distances than the partially open-loop approach, the ideal distance (obtained from simulation) is not reached. We speculate that this might be due to oversimplification of the system model (e.g., the intrinsic dynamics of the NEEs, such as friction and hysteresis, are not considered), or due to errors in obtaining the parameters of the model. These issues might be alleviated developing system identification methods geared for VSA robots, and/or adaptive control schemes for on-line parameter estimation.

A video containing the real-world experiments is included in the supplemental material, which also qualitatively conveys the performance of our approach.

VII. CONCLUSIONS

A framework for the closed-loop control of VSA robots based on NMPC has been introduced and analyzed. It has been shown that the proposed scheme presents a lower sensitivity to uncertainties with respect to the state-of-the-art approach [15]. Basic theoretical results in the nominal case have been proved, and a case study including simulation and real-world experiments has been presented to demonstrate the practical applicability of the presented work.

ACKNOWLEDGMENT

The authors would like to express their thanks to Michael Lewis and Bauyrzhan Aubakir for their assistance in multimedia material preparation, Olzhas Adiyatov for his assistance in the real-time implementation of the algorithms and Iliyas Tursynbek for his assistance in the preparation of the experimental setup.

REFERENCES

[1] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*. New York, NY, USA: Springer-Verlag, 2008.
[2] G. Hirzinger, J. Bals, M. Otter, and J. Stelter, "The DLR-KUKA success story: Robotics research improves industrial robots," *IEEE Robot. Autom. Mag.*, vol. 12, no. 3, pp. 16–23, Sep. 2005.
[3] A. Albu-Schaffer *et al.*, "Soft robotics," *IEEE Robot. Autom. Mag.*, vol. 15, no. 3, pp. 20–30, Sep. 2008.

[4] C. English and D. Russell, "Implementation of variable joint stiffness through antagonistic actuation using rolamite springs," *Mech. Mach. Theory*, vol. 34, no. 1, pp. 27–40, 1999.
[5] K. Koganezawa, Y. Watanabe, and N. Shimizu, "Antagonistic muscle-like actuator and its application to multi-d.o.f. forearm prosthesis," *Adv. Robot.*, vol. 12, nos. 7–8, pp. 771–789, 1997.
[6] G. Tonietti, R. Schiavi, and A. Bicchi, "Design and control of a variable stiffness actuator for safe and fast physical human/robot interaction," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2005, pp. 526–531.
[7] S. Wolf and G. Hirzinger, "A new variable stiffness design: Matching requirements of the next robot generation," in *Proc. Int. Conf. Robot. Autom.*, May 2008, pp. 1741–1746.
[8] S. Wolf, O. Eiberger, and G. Hirzinger, "The DLR FSJ: Energy based design of a variable stiffness joint," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 5082–5089.
[9] A. Jafari, N. G. Tsagarakis, and D. G. Caldwell, "A novel intrinsically energy efficient actuator with adjustable stiffness (AwAS)," *IEEE/ASME Trans. Mechatronics*, vol. 18, no. 1, pp. 355–365, Feb. 2013.
[10] A. Jafari, N. G. Tsagarakis, I. Sardellitti, and D. G. Caldwell, "A new actuator with adjustable stiffness based on a variable ratio lever mechanism," *IEEE/ASME Trans. Mechatronics*, vol. 19, no. 1, pp. 55–63, Feb. 2014.
[11] M. Garabini, A. Passaglia, F. Belo, P. Salaris, and A. Bicchi, "Optimality principles in variable stiffness control: The VSA hammer," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 3770–3775.
[12] D. Braun, M. Howard, and S. Vijayakumar, "Optimal variable stiffness control: Formulation and application to explosive movement tasks," *Auto. Robots*, vol. 33, no. 3, pp. 237–253, 2012.
[13] S. Haddadin, F. Huber, and A. Albu-Schaffer, "Optimal control for exploiting the natural dynamics of variable stiffness robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 3347–3354.
[14] I. Sardellitti, G. A. Medrano-Cerda, N. Tsagarakis, A. Jafari, and D. G. Caldwell, "Gain scheduling control for a class of variable stiffness actuators based on lever mechanisms," *IEEE Trans. Robot.*, vol. 29, no. 3, pp. 791–798, Jun. 2013.
[15] D. J. Braun *et al.*, "Robots driven by compliant actuators: Optimal control under actuation constraints," *IEEE Trans. Robot.*, vol. 29, no. 5, pp. 1085–1101, Oct. 2013.
[16] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 4th ed. Belmont, MA, USA: Athena Scientific, 2012.
[17] J. B. Rawlings and D. Q. Mayne, *Model Predictive Control: Theory and Design*. Madison, WI, USA: Nob Hill Pub., 2009.
[18] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Belmont, MA, USA: Athena Scientific, 1999.
[19] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Eng. Pract.*, vol. 11, no. 7, pp. 733–764, 2003.
[20] J. H. Lee, "Model predictive control: Review of the three decades of development," *Int. J. Control, Autom. Syst.*, vol. 9, no. 3, pp. 415–424, 2011.
[21] C.-Y. Lin and Y.-C. Liu, "Precision tracking control and constraint handling of mechatronic servo systems using model predictive control," *IEEE/ASME Trans. Mechatronics*, vol. 17, no. 4, pp. 593–605, Aug. 2012.
[22] M. A. Stephens, C. Manzie, and M. C. Good, "Model predictive control for reference tracking on an industrial machine tool servo drive," *IEEE Trans. Ind. Informat.*, vol. 9, no. 2, pp. 808–816, May 2013.
[23] S. Di Cairano, H. E. Tseng, D. Bernardini, and A. Bemporad, "Vehicle yaw stability control by coordinated active front steering and differential braking in the tire sideslip angles domain," *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 4, pp. 1236–1248, Jul. 2013.
[24] J. Rodriguez *et al.*, "State of the art of finite control set model predictive control in power electronics," *IEEE Trans. Ind. Informat.*, vol. 9, no. 2, pp. 1003–1016, May 2013.
[25] F. Debrouwere *et al.*, "Time-optimal path following for robots with convex–concave constraints using sequential convex programming," *IEEE Trans. Robot.*, vol. 29, no. 6, pp. 1485–1495, Dec. 2013.
[26] F. Debrouwere, M. Vukov, R. Quirynen, M. Diehl, and J. Swevers, "Experimental validation of combined nonlinear optimal control and estimation of an overhead crane," in *Proc. IFAC World Congr.*, 2014, pp. 9617–9622.
[27] G. Pannocchia, J. B. Rawlings, and S. J. Wright, "Conditions under which suboptimal nonlinear MPC is inherently robust," *Syst. Control Lett.*, vol. 60, no. 9, pp. 747–755, 2011.
[28] P. O. M. Scokaert, D. Q. Mayne, and J. B. Rawlings, "Suboptimal model predictive control (feasibility implies stability)," *IEEE Trans. Autom. Control*, vol. 44, no. 3, pp. 648–654, Mar. 1999.
[29] M. Rubagotti, P. Patrinos, and A. Bemporad, "Stabilizing linear model predictive control under inexact numerical optimization," *IEEE Trans. Autom. Control*, vol. 59, no. 6, pp. 1660–1666, Jun. 2014.
[30] B. Houska, H. J. Ferreau, and M. Diehl, "ACADO toolkit—An open-source framework for automatic control and dynamic optimization," *Opt. Control Appl. Methods*, vol. 32, no. 3, pp. 298–312, 2011.
[31] B. Houska, H. J. Ferreau, and M. Diehl, "An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range," *Automatica*, vol. 47, no. 10, pp. 2279–2285, 2011.
[32] U. M. Ascher and L. R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Philadelphia, PA, USA: SIAM, 1998.
[33] H. G. Bock and K. J. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems," in *Proc. IFAC World Congr.*, Aug. 1984, pp. 243–247.
[34] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Math. Program. Comput.*, vol. 6, no. 4, pp. 327–363, 2014.
[35] M. Diehl, R. Findeisen, and F. Allgöwer, "A stabilizing real-time implementation of nonlinear model predictive control," in *Real-Time PDE-Constrained Optimization* (Computational Science & Engineering). Philadelphia, PA, USA: SIAM, 2007, pp. 25–52.
[36] M. Diehl, H. G. Bock, J. P. Schlöder, R. Findeisen, Z. Nagy, and F. Allgöwer, "Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations," *J. Process Control*, vol. 12, no. 4, pp. 577–585, 2002.
[37] M. Diehl, R. Findeisen, F. Allgöwer, H. G. Bock, and J. P. Schlöder, "Nominal stability of real-time iteration scheme for nonlinear model predictive control," *IEE Proc.-Control Theory Appl.*, vol. 152, no. 3, pp. 296–308, May 2005.

ALTAY ZHAKATAYEV (M'15) received the B.S. degree in aerospace engineering from Texas A&M University, College Station, TX, USA, in 2010, and the M.S. degree in mechanical engineering from University College London, London, U.K., in 2012.

He was with the Advanced Robotics and Mechatronics Systems Laboratory, Nazarbayev University, Astana, Kazakhstan, in 2012, as a Junior Research Assistant. His current research interests include the optimal control and design of variable impedance actuated robots using nonlinear optimization algorithms.

MATTEO RUBAGOTTI (S'07–M'11) received the B.S. and M.S. degrees in computer engineering and the Ph.D. degree in electronics, computer science, and electrical engineering from the University of Pavia, Italy, in 2004, 2006, and 2010, respectively. As a Ph.D. student, he held visiting positions with the Center for Automotive Research, The Ohio State University, Columbus, OH, USA, and the Automatic Control Laboratory, Swiss Federal Institute of Technology, Zurich, Switzerland.

He was a Post-Doctoral Fellow with the Research Group of Prof. A. Bemporad, University of Trento, Italy, and then with the IMT Institute for Advanced Studies, Lucca, Italy, from 2010 to 2012. Since 2012, he has been an Assistant Professor with the Department of Robotics and Mechatronics, Nazarbayev University, Astana, Kazakhstan. He has co-authored over 35 papers in international journals and conferences. His research interests are in the area of control theory (linear/nonlinear model predictive control and sliding mode control), with applications to robotics, mechatronics, and energy systems.

Dr. Rubagotti has served on the Conference Editorial Board of the IEEE Control Systems Society since 2014.

HUSEYIN ATAKAN VAROL (S'01–M'09) received the B.S. degree in mechatronics engineering from Sabanci University, Istanbul, Turkey, in 2005, and the M.S. and Ph.D. degrees in electrical engineering from Vanderbilt University, Nashville, TN, USA, in 2007 and 2009, respectively. From 2009 to 2011, he was a Post-Doctoral Research Associate and then a Research Assistant Professor with the Department of Mechanical Engineering, Center for Intelligent Mechatronics, Vanderbilt University.

He joined as a Faculty Member with Nazarbayev University, Astana, Kazakhstan, in 2011, as an Associate Professor of Robotics, where he currently directs the Advanced Robotics and Mechatronics Systems Laboratory. His research interests include design and control of biomechatronic systems, variable impedance actuation, machine learning, and embedded systems.He has authored over 40 technical papers on related topics in international journals and conferences.

Dr. Varol was a finalist for the KUKA Innovation Award in 2014. He was a recipient of the IEEE International Conference on Rehabilitation Robotics Best Paper Award in 2009, and IEEE Engineering in Medicine and Biology Society Outstanding Paper Award in 2013.