Direct Data-Driven Control of Constrained Systems

Dario Piga, Simone Formentin, and Alberto Bemporad

***Abstract*— In model-based control design, one often has to describe the plant by a linear model. Deriving such a model poses issues of parameterization, estimation, and validation of the model before designing the controller. In this paper, a direct data-driven control method is proposed for designing controllers that can handle constraints *without deriving a model of the plant* and *directly from data*. A hierarchical control architecture is used, in which an inner *linear time-invariant* or *linear parameter-varying* controller is first designed to match a simple and *a priori* specified closed-loop model. Then, an outer model predictive controller is synthesized to handle input/output constraints and to enhance the performance of the inner loop. The effectiveness of the approach is illustrated by means of a simulation and an experimental example. Practical implementation issues are also discussed.**

***Index Terms*— Constrained control, data-driven control, linear parameter-varying (LPV) systems, model predictive control.**

I. Introduction

MOST model-based control design methods rely on the availability of a linear dynamical model of the open-loop process. Even in those applications where gathering data to identify and validate a model of the plant is not costly or time-consuming, finding a mathematical linear time-invariant (LTI) or linear parameter-varying (LPV) description of the plant, which is good for control design purposes, is not an easy task. In fact, when deriving a model of the plant, one always trades off accuracy versus complexity, and, most of the times, one is not able to decide *a priori* how accurate the model should be to achieve a satisfactory closed-loop performance.

Recently, a data-driven method has been proposed for directly designing LTI/LPV controllers from data, thus avoiding to parameterize, identify, and transform an LPV model of the open-loop system [1]. This approach sounds appealing and shows many interesting features (e.g., the mapping with respect to the scheduling signal does not need to be defined *a priori*). However, this approach cannot be always considered as a competitor of other state-of-the-art control design techniques, since constraints on input and output variables cannot be considered. Furthermore, being a model-reference design method, it requires the desired closed-loop model to be defined, and the choice of an adequate (i.e., practically achievable) reference model without knowing the process dynamics may not be easy. These are well-known and open problems in the direct data-driven control literature, both in the LTI and in the LPV framework [2].

In this paper, we propose an extension of the data-driven control design method in [1]. The controller is split into two components, organized in a hierarchical fashion: an inner controller, which accounts for matching a given simple reference model, and an outer model predictive controller (MPC) acting as a command governor [3], [4], aiming at enhancing the closed-loop performance and ensuring that the constraints are not violated. The main rationale behind this architecture is that the reference model for the inner loop is chosen only to reduce model complexity and uncertainty, but it is decoupled from the desired closed-loop behavior, which is instead taken care of by the outer part of the controller. Hence, the problem of selecting an achievable reference model becomes less critical than in [1], in that low-performance models are more easily achieved than the desired closed-loop behavior. Moreover, such low-performance models are often achieved by low-order controller structures, which eases the identification of the inner control law from data. Then, the outer model-based controller manipulates the reference signal in such a way that the constraints on inputs (rate and magnitude) and outputs are fulfilled and closed-loop performance is improved, without complicating the data-driven design procedure. We will show that also the whole control design procedure does not rely on the plant knowledge, according to the direct data-driven philosophy of the method. To the best of our knowledge, this is the first work addressing the problem of handling constraints in a direct data-driven control design. The overall control scheme can be seen as a predictive controller for constrained systems directly designed from data.

The effectiveness of the hierarchical control architecture is illustrated by means of two examples: 1) the simulation case study of [1], which best highlights the improvements with respect to [1] and 2) an experimental case study concerning the control of an *RC* circuit with switching load to test the performance of the method when dealing with real-world data.

Other design procedures for two-degree-of-freedom control architectures have been proposed in [5]–[7]. Nonetheless, in [5] and [6], the objective is to exploit the two-degree of freedom scheme to match both the sensitivity and the complementary sensitivity functions in the *iterative feedback tuning* and *virtual reference feedback tuning* method, respectively. Instead, in [7], the goal of one block is to linearize the system around an operating point, while the other control block is used to boost the performance of the linearized system. The latter contribution is closer to what is proposed here, with the

Manuscript received November 9, 2016; revised March 4, 2017; accepted May 1, 2017. Date of publication May 23, 2017; date of current version June 11, 2018. Manuscript received in final form May 4, 2017. This work was supported in part by the European Commission through the Project DISIRE-Distributed In-Situ Sensors Integrated into Raw Material and Energy Feedstock under Grant H2020-636834 and through the Project DAEDALUS - Distributed control and simulation platform to support an ecosystem of digital automation developers under Grant H2020-723248. Recommended by Associate Editor S. Tarbouriech. *(Corresponding author: Dario Piga.)*

D. Piga is with the IDSIA Dalle Molle Institute for Artificial Intelligence, Scuola Universitaria Professionale della Svizzera italiana, 6928 Manno, Switzerland (e-mail: dario.piga@supsi.ch).

S. Formentin is with the Politecnico di Milano, 20133 Milan, Italy.

A. Bemporad is with the IMT School for Advanced Studies Lucca, 55100 Lucca, Italy.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCST.2017.2702118

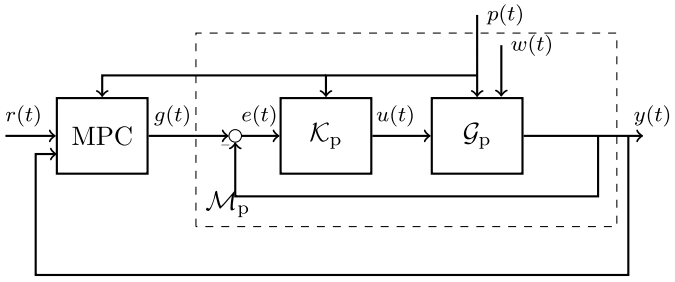

Fig. 1. Proposed hierarchical control architecture: the inner controller $\mathcal{K}_p$ provides minimal tracking capabilities for the unconstrained system $\mathcal{G}_p$, whereas the outer MPC controller enhances the performance and guarantees that the constraints are not violated. $\mathcal{K}_p$ is designed from data, so that the inner loop matches as much as possible $\mathcal{M}_p$.

major differences that, in [7]: 1) the controller is not derived directly from data but a model of the system is first identified and 2) the framework is deterministic with prior assumptions on the boundedness of the noise, while a stochastic setting is considered in this paper.

The paper is organized as follows. In Section II, the control problem is formally stated and the additional requirements with respect to [1] are discussed. The hierarchical architecture of the proposed approach is introduced in Section III, and the design of the inner and outer controllers is discussed. The two case studies are illustrated in Section IV.

II. Problem Statement

Let the output signal $y(t) \in \mathbb{R}$, $t \in \mathbb{Z}$, be generated by an unknown *single-input single-output* system $\mathcal{G}_p$, driven by the manipulated input $u(t) \in \mathbb{R}$, a measured exogenous signal $p(t) \in \mathbb{P} \subseteq \mathbb{R}^{n_p}$, and an unmeasured disturbance $w(t) \in \mathbb{R}^{n_w}$. From now on, we assume $n_p = 1$ to keep the notation simple. The system $\mathcal{G}_p$ is assumed to be *bounded-input bounded-output* stable. Assume that a collection of data $\mathcal{D}_N = \{u(t), y(t), p(t); t \in \mathcal{I}_1^N\}$, $\mathcal{I}_1^N = \{1, \ldots, N\}$ generated by the system $\mathcal{G}_p$ is available.

We aim at synthesizing a controller, such that any user-defined (admissible) reference signal can be accurately tracked by the output, without possibly violating the following constraints on inputs and outputs:

$$u_{\min} \leq u(t) \leq u_{\max}, \quad \Delta u_{\min} \leq u(t) - u(t-1) \leq \Delta u_{\max} \tag{1a}$$

$$y_{\min} \leq y(t) \leq y_{\max} \quad \forall t \in \mathbb{Z}, \quad t \geq 0. \tag{1b}$$

Note that the constraints on the input are generally imposed by actuator limitations, while the constraints on the output might reflect, for instance, performance specifications or safety conditions. Considering such constraints is, therefore, of primary importance for many critical engineering applications.

Rather than attempting at deriving a model of the open-loop plant $\mathcal{G}_p$, we aim at designing a tracking controller directly from the available data set $\mathcal{D}_N$.

III. Hierarchical Approach

The proposed control design approach relies on the hierarchical (two degrees of freedom) architecture shown in Fig. 1, which integrates the following.

1) An *inner linear controller* $\mathcal{K}_p(\theta)$ described by

$$A_K(p, t, q^{-1}, \theta)u(t) = B_K(p, t, q^{-1}, \theta)(g(t) - y(t)) \tag{2}$$

where

$$A_K(p, t, q^{-1}, \theta) = 1 + \sum_{i=1}^{n_{a_K}} a_i^K(p, t, \theta)q^{-i} \tag{3}$$

$$B_K(p, t, q^{-1}, \theta) = \sum_{i=0}^{n_{b_K}} b_i^K(p, t, \theta)q^{-i}. \tag{4}$$

Note that controller (2) is of LPV nature for maximum generality. LTI controllers can be used by simply dropping the dependence on the external signal p.

The dynamical order of controller $\mathcal{K}_p(\theta)$, defined by the parameters n_{a_K} and n_{b_K}, is *a priori* specified by the user, while $a_i^K(p, t, \theta)$ and $b_i^K(p, t, \theta)$ are nonlinear (possibly dynamic) functions of the scheduling variable sequence p and depend on the design parameter vector θ. For instance, if the coefficient function $a_i^K(p, t, \theta)$ is parameterized as a two-degree polynomial in $p(t)$, that is

$$a_i^K(p, t, \theta) = \theta_0^{(i)} + \theta_1^{(i)}p(t) + \theta_2^{(i)}p^2(t) \tag{5}$$

the coefficients $\theta_0^{(i)}, \theta_1^{(i)}$, and $\theta_2^{(i)}$ are the design parameters of the inner controller $\mathcal{K}_p(\theta)$, and thus they are elements of the design parameter vector θ.

The inner controller (i.e., the parameter vector θ) is designed to achieve a desired LPV (or LTI) closed-loop behavior $\mathcal{M}_p$, *a priori* specified by the user and described by the state-space model

$$\begin{aligned} x_M(t+1) &= \bar{A}_M(p, t)x_M(t) + \bar{B}_M(p, t)g(t) \\ y_d(t) &= \bar{C}_M(p, t)x_M(t) \end{aligned} \tag{6}$$

where y_d denotes the desired closed-loop output for a given reference signal g. The controller parameters θ achieving the chosen reference model $\mathcal{M}_p$, as well as the functional dependence on p, are estimated directly from the training data set $\mathcal{D}_N$, without first identifying a model for the plant $\mathcal{G}_p$. Such a data-driven procedure for LPV control design was originally introduced in [1], and it will be reviewed in Section III-A.

2) An *outer linear MPC*, designed based on the desired closed-loop model $\mathcal{M}_p$. The MPC controller selects on-line, and, according to a *receding horizon* strategy, the optimal reference supplied to the inner closed-loop system in order to fulfill the constraints (1), thus acting as a reference governor. Besides constraint fulfillment, the outer MPC enhances the performance of the inner closed-loop system.

By merging the two controllers together in the above hierarchical fashion, one can choose a low-demanding (e.g., with slow dynamics and low damping factor) inner closed-loop behavior $\mathcal{M}_p$, which is known to be easily achievable by the inner controller $\mathcal{K}_p(\theta)$ (for this, only a rough knowledge of the process dynamics is required). The tasks of optimizing the closed-loop performance and fulfilling the input/output

constraints are then left to the outer MPC, which can be designed based on the (known) closed-loop dynamics $\mathcal{M}_p$.

A. Inner Controller Design

The main ideas behind the direct data-driven approach introduced in [1] and employed in this paper to design the inner controller $\mathcal{K}_p(\theta)$ are briefly recalled here. The design of the outer MPC-based controller is discussed in Section III-B.

Based on the available training data set $\mathcal{D}_N$, the objective is to design $\mathcal{K}_p(\theta)$, achieving a desired closed-loop behavior $\mathcal{M}_p$ *a priori* specified by the user and described by the state-space equations (6). Unlike [1], no specific requirement on the performance of the (inner) closed-loop behavior $\mathcal{M}_p$ is needed, as the outer MPC will handle the performance requirements. The only assumption that needs to be satisfied by $\mathcal{M}_p$ is that such a behavior is practically achievable. This assumption is barely satisfied when a closed loop $\mathcal{M}_p$ with high performance (e.g., systems exhibiting a high bandwidth and a low overshoot) is chosen. In other words, the chosen parameterization for $\mathcal{K}_p(\theta)$ might not be flexible enough to achieve the desired closed-loop behavior. It is then advisable to impose a low-performance closed-loop behavior $\mathcal{M}_p$.

Remark 1: The above observation can be further clarified by considering a simple LTI example. Consider a model matching problem for a nonminimum phase plant, in which the reference model does not contain the nonminimum phase zeroes of the plant. If the desired bandwidth is high, it is well known that the optimal controller will be likely to destabilize the system in closed loop [8]. However, a reference model with a lower bandwidth could still be achieved, as far as the nonminimum phase zeroes are left beyond the desired cutoff frequency.

In the following, the operator $M(p,t,q^{-1})$ will be used as a shorthand form to indicate the mapping of g to y_d via the reference model $\mathcal{M}_p$. Formally, M is such that $y_d(t) = M(p,t,q^{-1})g(t)$ for all trajectories of p and g. Furthermore, we define the left inverse of $M(p,t,q^{-1})$ as the LPV mapping $M^\dagger(p,t,q^{-1})$ that gives g as output when fed by y_d, for any trajectory of p, i.e., $M^\dagger(p,t,q^{-1})M(p,t,q^{-1}) = 1$.[1]

Let $\varepsilon = y_d - y$ be the error between the desired and actual outputs in response to g. According to Fig. 2, we have

$$\begin{aligned} g(t) &= M^\dagger(p,t,q^{-1})y_d(t) \\ &= M^\dagger(p,t,q^{-1})(\varepsilon(t) + y(t)) \qquad (7a) \\ A_K(p,t,\theta)u(t) &= B_K(p,t,\theta)(g(t) - y(t)) \qquad (7b) \end{aligned}$$

$\forall t \in \mathcal{I}_1^N$. Thus, the controller parameters θ are computed by minimizing the 2-norm of the error ε subject to (7a) and (7b), that is

$$\begin{aligned} &\min_{\theta,\varepsilon} \ \frac{1}{N}\sum_{t=1}^{N} \varepsilon^2(t) \\ &\text{s.t. } A_K(p(t),t,\theta)u(t) = B_K(p(t),t,\theta) \\ &\quad \times (M^\dagger(p(t),t)\varepsilon(t) + M^\dagger(p(t),t)y(t) - y(t)) \qquad (8) \end{aligned}$$

[1]For reference maps given in the state-space form (6), the left inverse $M^\dagger(p,t,q^{-1})$ can be computed as indicated in [1, Proposition 1].

with $\{u(t), y(t), p(t)\} \in \mathcal{D}_N$. Note that problem (8) is a purely (nonconvex) data-based problem, independent of $\mathcal{G}_p$. By introducing the residual

$$\begin{aligned} &\varepsilon_u(\theta,t) \\ &= B_K(p(t),t,\theta)M^\dagger(p(t),t)\varepsilon(t) \\ &= A_K(p(t),t,\theta)u(t) - B_K(p(t),t,\theta)(M^\dagger(p(t),t)y(t) - y(t)) \end{aligned}$$

the original (nonconvex) problem (8) is replaced with the following (convex) problem:

$$\begin{aligned} \min_{\theta} \ \frac{1}{\gamma}\|\theta\|^2 + \frac{1}{N}\sum_{t=1}^{N} |A_K(p(t),t,\theta)u(t) \\ - B_K(p(t),t,\theta)(M^\dagger(p(t),t)y(t) - y(t))|^2 \qquad (9) \end{aligned}$$

$\{u(t), y(t), p(t)\} \in \mathcal{D}_N$, where $\gamma > 0$ is a regularization parameter. Besides the regularization term, the difference between problems (8) and (9) is that the norm of $\varepsilon(t)$ is minimized in (8), while the norm of the filtered error $\varepsilon_u(\theta,t) = B_K(p(t),t,\theta)M^\dagger(p(t),t)\varepsilon(t)$ is minimized in (9). Unlike (8), the solution of (9) is given by simple least-squares, provided that the controller coefficients $a_i^K(p,t,\theta)$ and $b_i^K(p,t,\theta)$ are parameterized as linear functions of θ [e.g., like in (5)]. However, since the residuals $\varepsilon_u(\theta,t)$ are not white, the final estimate of the least-squares problem (9) is not consistent [i.e., the final estimate θ is not guaranteed to converge to the optimal parameters solving the original problem (8)] and the bias can be significant in the case of noise $w(t)$ of large variance. According to [1, Sec. 4], in order to overcome this problem, the following slight modification of problem (9), based on instrumental variables, can be solved instead of (9):

$$\min_{\theta,\varepsilon_u} \frac{1}{\gamma}\|\theta\|^2 + \frac{1}{N^2}\left\|\sum_{t=1}^{N} z(t)\varepsilon_u(\theta,t)\right\|^2 \qquad (10)$$

$\{u(t), y(t), p(t)\} \in \mathcal{D}_N$, where $z(t)$ is the so-called instrument, chosen by the user, so that $z(t)$ is not correlated with the noise $w(t)$. In [1, Proposition 2], it is shown that, in the case $w(t)$ is zero-mean and the output $y(t)$ depends linearly on $w(t)$ [e.g., $w(t)$ is a measurement noise], the final estimate provided by (10) converges to the solution of problem (8).

In case the controller coefficients $a_i^K(p,t,\theta)$ and $b_i^K(p,t,\theta)$ in (3) and (4) are parameterized as a linear combination of the known basis functions of p [like in (5)], problem (10) is a parametric quadratic programming (QP) problem. In case the dependence of $a_i^K(p,t,\theta)$ and $b_i^K(p,t,\theta)$ on p is not *a priori* specified, the dual version of (10) can be formulated and the *kernel-based* approaches described in [1, Sec. 5.2] can be used to compute a nonparametric estimate of the controller coefficients $a_i^K(p,t,\theta)$ and $b_i^K(p,t,\theta)$. When Gaussian kernels are used, only the hyperparameter σ, representing the width of the kernels $\kappa(t,j) = e^{(((p(t)-p(j))^2)/\sigma)}$, is specified by the user. In this case, the design parameter vector θ contains the kernel width σ and all the parameters used to describe the coefficients $a_i^K(p,t,\theta)$ and $b_i^K(p,t,\theta)$ in terms of kernels. The reader is referred to [1, (55)] for a kernel representation of $a_i^K(p,t,\theta)$ and $b_i^K(p,t,\theta)$.

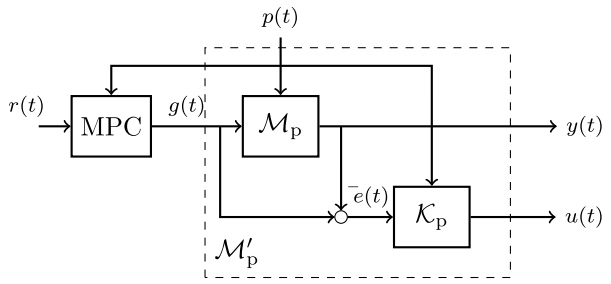

Fig. 2. Equivalent single-input two-output LPV model describing the relationship between the MPC output $g(t)$ and the plant input and output signals.

B. Outer Controller Design

The outer MPC controller, acting as a reference governor, is designed based on the equivalent single-input two-output model $\mathcal{M}'_p$ shown in Fig. 2, where the dynamics of the inner closed-loop system are now described by the (known) model $\mathcal{M}_p$. The augmented LPV model $\mathcal{M}'_p$ thus describes the relationship between $g(t)$ and $u(t)$, $y(t)$. Within this framework, the role of the inner LPV controller $\mathcal{K}_p(\theta)$ is to transform the behavior of the unknown plant $\mathcal{G}_p$ into that of a known, usually simpler, and *a priori* specified LPV model $\mathcal{M}_p$.

Consider the following, not necessarily minimal, state-space realization of $\mathcal{M}'_p$:

$$\begin{cases} \xi(t+1) = & A_M(p(t))\xi(t) + B_M(p(t))g(t) \\ \begin{bmatrix} y(t) \\ u(t) \end{bmatrix} & = C_M(p(t))\xi(t) + \begin{bmatrix} 0 \\ D_M(p(t)) \end{bmatrix} g(t) \end{cases} \tag{11}$$

where the matrices $A_M(p(t)), B_M(p(t)), C_M(p(t))$, and $D_M(p(t))$ can be derived from the description of the reference model $\mathcal{M}_p$ [see (6)] and the inner controller $\mathcal{K}_p$ [see (2)].

Based on the prediction model (11), the outer MPC controller is designed both to impose input/output constraints and to possibly improve the tracking quality of the reference signal r. As shown in the equivalent scheme of Fig. 2, only the reference model $\mathcal{M}_p$ and the model of the controller $\mathcal{K}_p(\theta)$ are needed to predict the behavior of $u(t)$ and $y(t)$. Then, also in this second step, a model of $\mathcal{G}_p$ is not required.

The design method is as follows. By assuming that the state vector $\xi(t)$ of the inner-loop model $\mathcal{M}_p$ is fully accessible or, alternatively, estimated from the measurements of u, y, and p, for example, by means of a linear time-varying Kalman filter, at each time instant t, the reference tracking MPC problem can be formulated, at each time instant t, as in (12), as shown at the bottom of the this page, where N_p and N_u denote the prediction and control horizon, respectively, Q_y, Q_u, $Q_{\Delta u}$, Q_g, and Q_ϵ are nonnegative weights, $u_{\rm ref}$ is a desired input reference (that is typically generated from the output reference r by means of static optimization), and V_y, V_u, and $V_{\Delta u}$ are positive vectors that are used to soften the constraints, so that (12) always admits a solution, which can be computed via *QP*.

In the MPC formulation (12), the following terms are penalized: 1) the tracking error between the reference signal r and the output y; 2) the tracking error between the input reference signal $u_{\rm ref}$ and the manipulated variable u; 3) the increments of the plant input u (the larger the weight $Q_{\Delta u}$, the less aggressive the control action); 4) the error between the reference signal r and the MPC output g; and 5) the violation of the constraints. From a practical point of view, the goal of the penalty on $g - r$ is to guarantee that the reference signal g of the inner closed-loop system does not differ too much from the reference signal r, so as to avoid to excite unmodeled (nonlinear) dynamics. In case $p(t+k)$ is known at time t for the future N_p steps, we set $p(t+k|t) = p(t+k)$ and call the MPC formulation (12) *linear time-varying MPC* (LTV-MPC). In case future values of p are not known, we set $p(t+k|t) \equiv p(t)$ and call the formulation *linear parameter-varying MPC* (LPV-MPC), in which the prediction model is LTI but depends on $p(t)$, and therefore, the MPC controller itself is LPV. Alternatively, the LPV-MPC scheme in [9] can be used to design a robust LPV-MPC-based controller. In such an approach, the future values of the scheduling variable are assumed to be uncertain and to vary within a polytope.

When both the nominal reference model $\mathcal{M}_p$ and the inner controller $\mathcal{K}_p$ are LTI, problem (12) is a more standard LTI-MPC problem, which has computational advantages over LTV-MPC and LPV-MPC, in that the QP problem matrices can be precomputed offline, and an explicit MPC approach [10], [11] may reduce the upper control layer to a piecewise affine function. However, having $\mathcal{M}'_p$ LTI barely happens in practice, in particular when the behavior of the

$$\min_{\{g(t+k|t)\}_{k=1}^{N_u}} \quad Q_y \sum_{k=1}^{N_p} (y(t+k|t) - r(t+k))^2 + Q_u \sum_{k=1}^{N_p} (u(t+k|t) - u_{\rm ref}(t+k))^2$$
$$+ Q_{\Delta u} \sum_{k=1}^{N_p} (u(t+k|k) - u(t+k-1|t))^2 + Q_g \sum_{k=1}^{N_u} (r(t+k) - g(t+k|t))^2 + Q_\epsilon \epsilon^2 \tag{12a}$$

$$\text{s.t.} \quad \xi(t+k+1|t) = A_M(p(t+k|t))\xi(t+k|t) + B_M(p(t+k|t))g(t+k|t), \quad k = 0, \ldots, N_p - 1 \tag{12b}$$

$$\begin{bmatrix} y(t+k|t) \\ u(t+k|t) \end{bmatrix} = C_M(p(t+k|t))\xi(t+k|t) + \begin{bmatrix} 0 \\ D_M(p(t+k|t)) \end{bmatrix} g(t+k|t), \quad k = 1, \ldots, N_p \tag{12c}$$

$$-V_y\epsilon + y_{\min} \le y(t+k|t) \le y_{\max} + V_y\epsilon, \quad -V_u\epsilon + u_{\min} \le u(t+k|t) \le u_{\max} + V_u\epsilon, \quad k = 1, \ldots, N_p \tag{12d}$$

$$-V_{\Delta u}\epsilon + \Delta u_{\min} \le u(t+k|t) - u(t+k-1|t) \le \Delta u_{\max} + V_{\Delta u}\epsilon, \quad k = 1, \ldots, N_p \tag{12e}$$

$$g(t+N_u+j|t) = g(t+N_u|t), \quad \xi(t|t) = \xi(t), \quad g(t) = g(t|t)\, j = 1, \ldots, N_p - N_u \tag{12f}$$

true plant $\mathcal{G}_p$ is strongly influenced by the scheduling signal p. In this context, even when the selected reference model $\mathcal{M}_p$ is LTI, a parameter-varying controller $\mathcal{K}_p$ may be needed to achieve the desired behavior.

IV. Case Studies

The effectiveness of the proposed hierarchical control approach is shown in this section on two case studies. The first one is the simulation example (concerning the control of a servo positioning system) used in [1] to illustrate the direct data-driven LPV control method. The second case study is an experimental application, addressing the control of the output voltage in an *RC* electric circuit with switching load. These examples show that complex dynamics of quasi-LPV and switching systems can be dealt with using the approach of this paper. All computations are carried out on an i7 2.40-GHz Intel core processor with 4 GB of RAM running MATLAB R2014b, and the *MPC Toolbox* [12] is used to design the outer MPC.

A. Simulation Case Study (The Servo Positioning System

As a first case study, we consider the control of a voltage-controlled dc motor with an additional mass mounted on the rotation disk. In what follows, we show that the hierarchical control structure in Fig. 1 may significantly improve the results of [1], besides allowing us to impose constraints on the input/output signals.

The mathematical model of the dc motor, used to simulate the behavior of the system, is represented by the continuous-time state-space equations

$$\begin{bmatrix} \dot{\alpha}(\tau) \\ \dot{\omega}(\tau) \\ \dot{I}(\tau) \end{bmatrix} = \begin{bmatrix} 0 & 1+\dfrac{\sin(\alpha(\tau))}{\alpha(\tau)} & 0 \\ \dfrac{mgl}{J}\dfrac{\sin(\alpha(\tau))}{\alpha(\tau)} & -\dfrac{b}{J} & \dfrac{K}{J} \\ 0 & -\dfrac{K}{L} & -\dfrac{R}{L} \end{bmatrix} \begin{bmatrix} \alpha(\tau) \\ \omega(\tau) \\ I(\tau) \end{bmatrix} + \begin{bmatrix} 0 & 0 & \dfrac{1}{L} \end{bmatrix}^{\top} V(\tau)$$

$$y(\tau) = [1 \quad 0 \quad 0] \begin{bmatrix} \alpha(\tau) \\ \omega(\tau) \\ I(\tau) \end{bmatrix}$$

where $V(\tau)$ [V] is the control input voltage over the armature, $I(\tau)$ [mA] is the current, $\alpha(\tau)$ [rad] is the shaft angle, and $\omega(\tau)$ [rad/s] is the angular velocity of the motor. The nomenclature of the parameters characterizing the dc motor is reported in Table I, along with their values. The output signal is observed with a sampling time $T_s = 10$ ms.

To gather data, the plant is excited with a discrete-time filtered zero-mean white noise voltage (followed by a zero-order hold block) with a Gaussian distribution and standard deviation of 20 V. The input filter is a first-order digital filter with a cutoff frequency of 1.6 Hz. The output measurements are corrupted by an additive colored noise $w(\tau)$ with zero mean and variance, such that the signal-to-noise ratio, namely the ratio between the signal and noise variances, is 19 dB. A second experiment with the same input is also performed to build the instruments $z(t)$ used in (10).

TABLE I
Physical Parameters of the DC Motor [13]

	Description	Value
R	Motor resistance	9.5 Ω
L	Motor inductance	$0.84 \cdot 10^{-3}$ H
K	Motor torque constant	$53.6 \cdot 10^{-3}$ Nm/A
J	Complete disk inertia	$2.2 \cdot 10^{-4}$ Nm^2
b	Friction coefficient	$6.6 \cdot 10^{-5}$ Nms/rad
M	Additional mass	0.07 kg
l	Mass distance from the center	0.042 m

1) Design of the Inner LPV Controller $\mathcal{K}_p$: A training data set $\mathcal{D}_N$ with $N = 1500$ input/output measurements is used to identify the inner LPV controller $\mathcal{K}_p$ through the procedure discussed in Section III-A. The chosen reference model $\mathcal{M}_p$ is described by the state-space equations

$$\begin{aligned} x_M(t+1) &= 0.99 x_M(t) + 0.01 g(t) \\ \alpha_M(t) &= x_M(t) \end{aligned} \tag{13}$$

that is, the desired (inner) closed-loop behavior $\mathcal{M}_p$ is a simple discrete-time first-order LTI model, with a cutoff frequency of about 6 Hz.

The chosen structure for the inner controller $\mathcal{K}_p$ is a fourth-order LPV system with an integral action described by

$$u(t) = \sum_{i=1}^{4} a_i^K(\Pi(t), \theta) u(t-i) + \sum_{j=0}^{4} b_j^K(\Pi(t), \theta) e_{\text{int}}(t-j)$$
$$e_{\text{int}}(t) = e_{\text{int}}(t-1) + (g(t) - y(t))$$

where $\Pi(t) = p(t-1)$, with $p(t) = \alpha(t) = y(t)$ (i.e., the output signal measurement is chosen as a scheduling variable).

An *a priori* parameterization of the coefficient functions $a_i^K(\Pi(t), \theta)$ and $b_j^K(\Pi(t), \theta)$ is not specified, and Gaussian kernels with width $\sigma = 2.4$ are used to estimate the functions $a_i^K(\Pi(t), \theta)$ and $b_j^K(\Pi(t), \theta)$ achieving the desired closed-loop behavior $\mathcal{M}_p$ in (14) (the reader is referred to [1, Sec. 5] for an explicit representation of the coefficient functions in terms of kernels). The hyperparameter γ in (10) is set to 180. The values of γ and the kernel width σ are found through cross-validation as follows. A 2-D grid search over the hyperparameters σ and γ is performed. For each value of σ and γ, a controller $\mathcal{K}_p$ is estimated and the one providing the best performance, in terms of mean square (MS) error, is chosen. The controller performance is measured on a calibration set $\mathcal{D}_{N_C} = \{u(t), y(t), p(t)\}_{t=1}^{N_C}$ of $N_C = 500$ samples generated from an open-loop experiment, in terms of its capabilities to reconstruct the (open-loop) input sequence $u(t)$, given the scheduling variable observations $p(t)$ and the *virtual tracking error* $e(t) = M^{\dagger} y(t) - y(t)$. To give an idea of the estimated p-dependent coefficients, the plots of $a_1^K(\Pi(t), \theta)$ and $b_3^K(\Pi(t), \theta)$ are provided in Fig. 3.

Once $\mathcal{K}_p$ is designed, a closed-loop experiment is performed using a piecewise constant signal as a reference excitation. The response of the inner closed-loop system is shown in Fig. 4, and compared with the output $y_d = \alpha_M$ of the desired closed-loop model $\mathcal{M}_p$ (computed for the same reference excitation). The input voltage $u(t) = V(t)$ provided by the controller $\mathcal{K}_p$ and applied to the motor is plotted in Fig. 5. Results

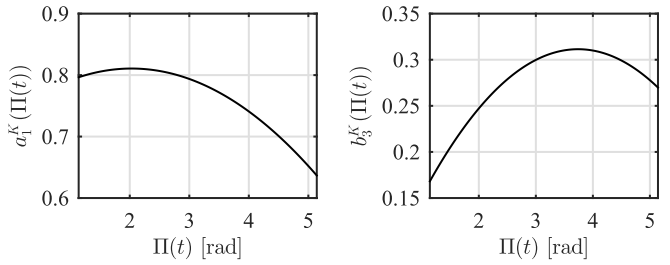

Fig. 3. Example 1: estimated coefficient functions $a_1^K(\Pi(t))$ (left) and $b_3^K(\Pi(t))$ (right).

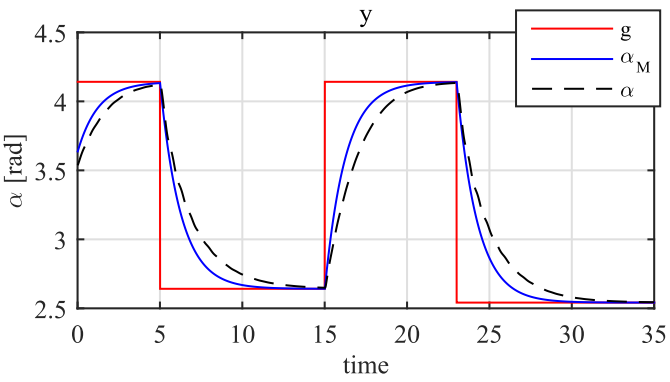

Fig. 4. Example 1: inner loop behavior. Red solid line: reference signal $g(\tau)$. Blue solid line: desired step response of the shaft angle $y_d(\tau)$. Black dashed line: actual controlled output $y(\tau)$.

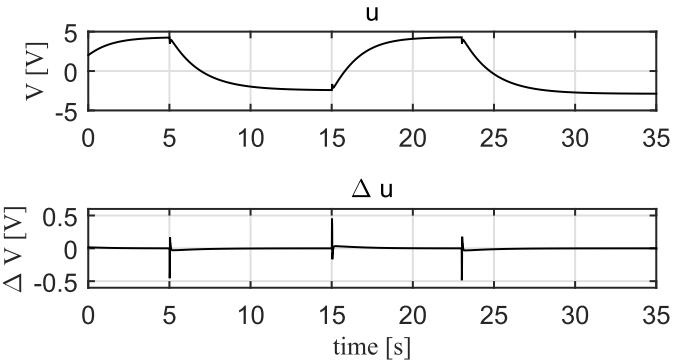

Fig. 5. Example 1: inner loop behavior. Plant input $V(\tau)$ and input increments $\Delta V(tT_s) = V(tT_s) - V((t-1)T_s)$.

in Fig. 4 show a good matching between the actual output y and the output y_d of the desired reference model $\mathcal{M}_p$. However, the closed-loop system exhibits slow dynamics, with a 10%–90% rise time of about 4 s and a 2% settling time (defined as the time elapsed by the output to enter and remain within a 2% error band) of about 6 s. Due to the limited degrees of the freedom in the controller structure, it has not been possible to achieve desired reference models $\mathcal{M}_p$ with faster dynamics. A sensitivity analysis with respect to different reference models $\mathcal{M}_p$ is reported in Table II, which shows the cutoff frequencies of different desired reference models $\mathcal{M}_p$ versus the *MS* of the differences between the desired closed-loop output y_d and the actual one y, for the same reference signal in Fig. 4. Note that, on the one hand, as the cutoff frequency of the reference model $\mathcal{M}_p$ decreases, the mismatch between desired and actual closed-loop output decreases, at the price of achieving slower dynamics. On the other hand, for reference models with a cutoff frequency larger than 10 Hz, the actual output y diverges.

TABLE II

CUTOFF FREQUENCY OF DIFFERENT REFERENCE MODELS $\mathcal{M}_p$ VERSUS MS OF THE DIFFERENCE BETWEEN DESIRED AND ACTUAL CLOSED-LOOP OUTPUT. THE MS IS NOT REPORTED WHEN THE ACHIEVED CLOSED-LOOP SYSTEM IS UNSTABLE

Cut-off frequency [Hz]	1	3	6	10	20
MS	0.0092	0.0571	0.1080	–	–

2) Design of the Outer MPC: Based on the chosen reference model $\mathcal{M}_p$ (which is used to describe the behavior of the inner closed-loop system) and the designed LPV controller $\mathcal{K}_p$, an outer MPC is designed in order to achieve the following objectives: 1) improve the performance of the inner loop, in terms of rise time and settling time and 2) enforce the following constraint on the input voltage rate: $V(tT_s) - V((t-1)T_s) \leq 0.2V$, $t = 1, 2, \ldots$

The MPC horizons and the weights defining the MPC cost function (12) are tuned through closed-loop simulation, by using model$\mathcal{M}_p$ to simulate the behavior of the inner closed-loop system. We stress that this step is very application-dependent, nevertheless no additional knowledge about the process $\mathcal{G}_p$ is required, being totally based on the chosen reference model $\mathcal{M}_p$. The chosen values are equal to $N_p = 10$, $N_u = 10$, $Q_y = 8.5$, $Q_u = 0$, $Q_{\Delta u} = 0.2$, and $Q_g = 1.7$.

The response of the closed-loop system for the same reference signal used in Section IV-A1 is shown in Fig. 6, while the input voltage applied to the motor is shown in Fig. 7. For the sake of comparison, the output of the inner loop achieved without the proposed hierarchical structure is shown in Fig. 6. The obtained results show that, although constraints on the variation of the input voltage are enforced, the hierarchical MPC structure allows us to achieve a faster reference tracking than the inner-loop system, with a 10%–90% rise time of about 1.3 s (about 3× smaller than the inner-loop rise time) and a 2% settling time of about 2.2 s (about 2.7× smaller than the inner-loop settling time). The obtained results show that the proposed hierarchical control architecture makes the choice of the reference model $\mathcal{M}_p$ less critical than in [1]. In fact, with the chosen structure of the controller $\mathcal{K}$ and the reference model $\mathcal{M}_p$, closed-loop systems with a cutoff frequency larger than 6 Hz were not achieved in [1] (see Table II). On the other hand, with the approach discussed in this paper, the (low performance) reference model $\mathcal{M}_p$ is achievable with the selected inner controller $\mathcal{K}_p$. Thus, the inner controller design phase does not guarantee the desired performance, but returns an accurate model of the inner loop. The task of enhancing the performance is then left to the outer MPC loop, whose design is based on $\mathcal{M}_p$. Within this setting, it cannot be stated *a priori* whether an LTI model is better than an LPV one, in that what is important here is the matching between the desired and the actual output for a given reference.

The computation time of the MPC layer is 18 ms (including various MATLAB overheads) on the used i7 Intel processor, based on the code generated by the MPC Toolbox, which is already in the order of magnitude of the sampling time

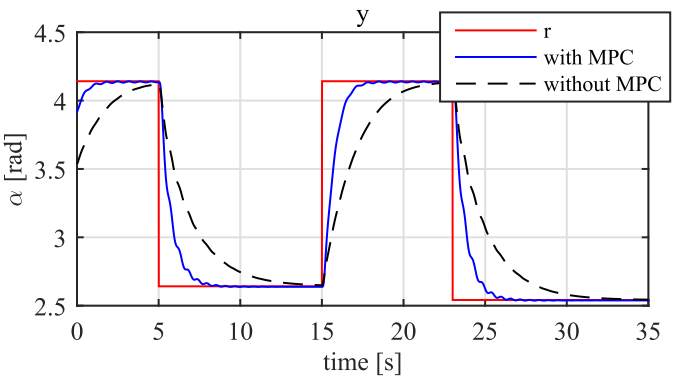

Fig. 6. Example 1: closed-loop behavior. Red solid line: reference signal $r(\tau)$. Blue solid line: controlled output $y(\tau)$. Black dashed line: inner-loop output achieved without outer MPC.

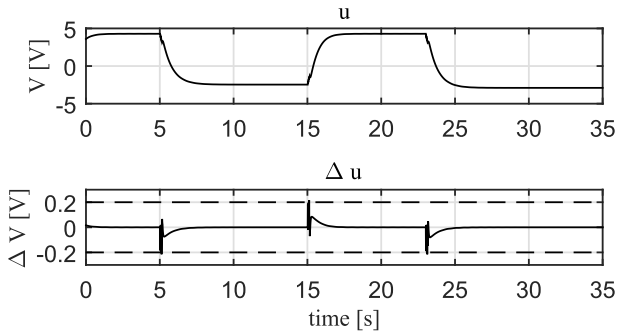

Fig. 7. Example 1: closed-loop behavior. Input voltage $V(\tau)$ and input increments $\Delta V(tT_s) = V(tT_s) - V((t-1)T_s)$, constrained between ± 0.2 V (dashed lines).

$T_s = 10$ ms. Although computational feasibility is not the main aim of this case study, it is realistic to assume that the controller could be implemented in real-time to control the motor by adopting a fast C implementation of the QP constructor of problem (12) and QP solver (see also [14]).

B. Experimental Case Study (Switching RC Circuit)

We address the problem of controlling the output voltage of an *RC* circuit with switching load. The aim of this example is not to improve the current state of the art in electric network control, but to show that the proposed method provides good performance also in real-world setups, with a physical plant and measurements gathered from real analog-to-digital (A/D) converters.

The schematic of the network is shown in Fig. 8. An *Arduino UNO* board is used for: 1) measuring the output voltage V_{out} [namely, the output $y(t)$]; 2) generating the input voltage V_{in} [namely, the input $u(t)$] applied to the circuit; 3) turning ON and OFF the switch [whose driving signal is the exogenous scheduling signal $p(t)$].

All the computations (including those related to inner and outer control laws) are carried out in MATLAB. The data are transmitted from the Arduino board to MATLAB, and viceversa, via a serial communication at a rate of 9600 Bd.

In order to gather the training set $\mathcal{D}_N$ used to identify $\mathcal{K}_p$, the following open-loop experiment is performed.

1) A piecewise-constant signal is applied as an input voltage $V_{\text{in}}(t)$ to the electronic circuit.

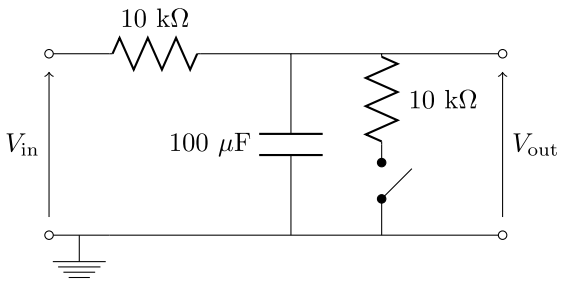

Fig. 8. Example 2: schematic of the electronic circuit. The ON/OFF switch is implemented using an MOSFET.

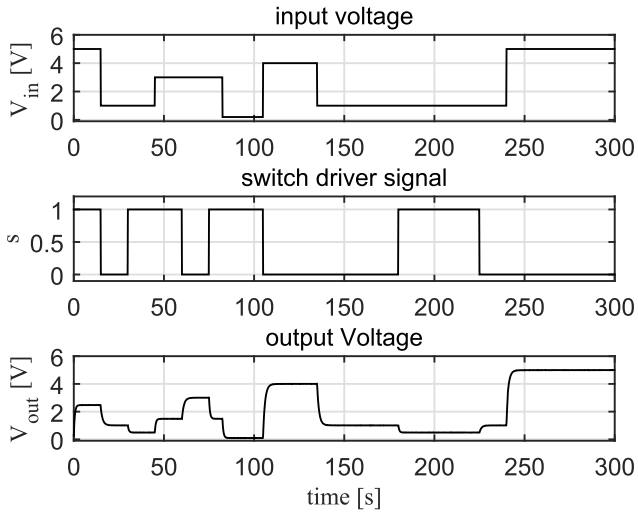

Fig. 9. Example 2: open-loop experiment. Top: input voltage $V_{\text{in}}(\tau)$. Middle: switching signal $s(\tau)$. Bottom: output voltage $V_{\text{out}}(\tau)$.

2) An exogenous piecewise-constant Boolean signal $s(t)$ drives the switch as follows: $s(t) = 1$ for Switch ON and $s(t) = 0$ for Switch OFF.
3) The voltage across the capacitor $V_{\text{out}}(t)$ is measured at a sampling time of $T_s = 150$ ms with an *A/D* converter available on Arduino.[2] 2000 samples are acquired. A second measurement of $V_{\text{out}}(t)$ is taken from another A/D converter to build the instruments.

The signals $V_{\text{in}}(t)$, $s(t)$, and $V_{\text{out}}(t)$ are plotted in Fig. 9.

A new data set with 500 samples is also built for tuning the hyperparameters γ and σ via cross-validation.

1) Inner LPV Controller Design: The following first-order LTI model is chosen as a reference model $\mathcal{M}_p$ for the inner loop:

$$\begin{aligned} x_M(t+1) &= 0.95x_M(t) + 0.05g(t) \\ y_{\text{d}}(t) &= x_M(t). \end{aligned} \tag{14}$$

A first-order LPV controller $\mathcal{K}_p$ with an integral action and static dependence on the scheduling variable $p(t)$ is used, that is

$$u(t) = a_1^K(p(t-1))u(t-1) + \sum_{j=0}^{1} b_j^K(p(t-1))e_{\text{int}}(t-j)$$
$$e_{\text{int}}(t) = e_{\text{int}}(t-1) + (g(t) - y(t)).$$

The parameters a_1^K, b_1^K, and b_2^K defining the LPV controller $\mathcal{K}_p$ are identified through the procedure discussed in

[2]The A/D converter has an input rage of 0–5 V and a resolution of 10 b;

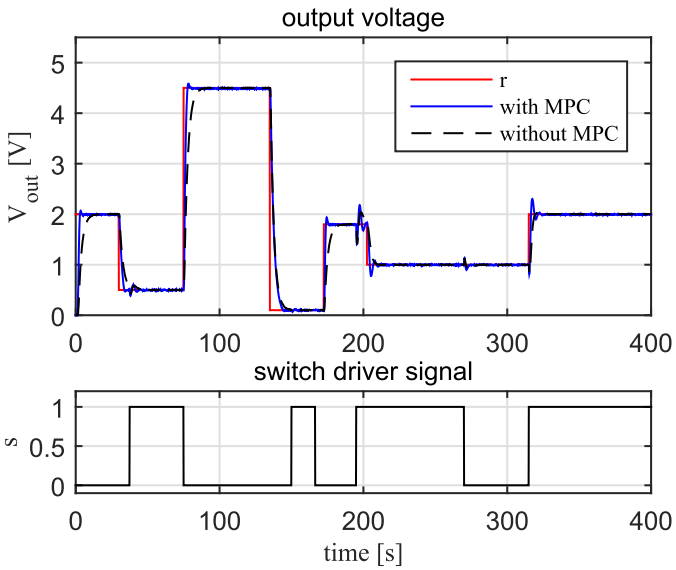

Fig. 10. Example 2: closed-loop experiment. Top: reference signal (red line), controlled output $V_{\text{out}}(\tau)$ (blue solid line), and inner-loop output achieved without the outer MPC (black dashed line). Bottom: switching signal $s(\tau)$.

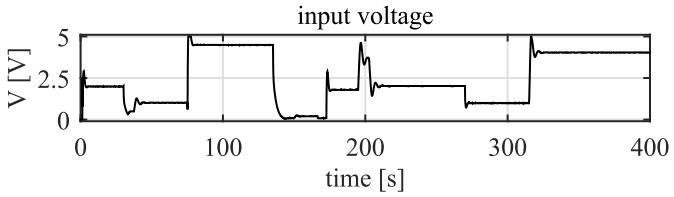

Fig. 11. Example 2: closed-loop experiment. Input voltage $V_{\text{in}}(\tau)$.

Section III-A. The values of the hyperparameter γ is 1000, while kernels' width is $\sigma = 1$.

2) Design of the Outer MPC: As the Arduino microcontroller can only provide voltage signals within the range 0–5 V, such a constraint on the signal $u(t) = V_{\text{in}}(t)$ is considered while computing the MPC law for generating $g(t)$. Furthermore, the controlled output $y(t) = V_{\text{out}}(t)$ is also constrained to belong to the interval $[0,\ 5]$ V, representing the input range of the A/D converters used in Arduino to measure the voltage $V_{\text{out}}(t)$.

The following values of the MPC parameters $N_p = 3$, $N_u = 3$, $Q_y = 0.45$, $Q_u = 0$ $Q_{\Delta u} = 0$, and $Q_g = 0.1$ are used. These parameters are tuned by means of closed-loop simulations, using the reference model $\mathcal{M}_p$ as the model of the inner loop.

The performance of the designed controllers is then tested by running a closed-loop experiment, with the trajectory of the switching driver signal $s(\tau)$ shown in Fig. 10 (bottom). The obtained controlled output voltage V_{out} is shown in Fig. 10 (top), along with the desired reference signal $r(\tau)$. For the sake of comparison, Fig. 10 also shows the output voltage V_{out} achieved by the inner closed-loop system, for the same reference, in the absence of the outer MPC. Such a comparison highlights an improvement in terms of raising time for the system with MPC. The trajectory of the input signal V_{in} is shown in Fig. 11. The obtained results show that the proposed hierarchical architecture allows us to efficiently track piecewise constant reference voltages in an *RC* circuit also in the presence of disturbance loads. Moreover, the obtained closed-loop dynamics turn out to be faster than the ones achieved by using only the inner LPV controller. Note that the sudden change of the output load causes only a negligible oscillation on the controlled output voltage V_{out} (see Fig. 10 at around $\tau = 90$ s and $\tau = 320$ s).

The CPU time required to compute the MPC law $g(t)$ at each time instant t ranges between 9 and 19 ms, significantly smaller than the sampling time $T_s = 150$ ms.

V. Conclusion

This paper has introduced a method for model-free design of feedback controllers for constrained linear systems directly from data. With respect to other existing direct control design methods, constraints on input and output variables can be handled. Moreover, the choice of the reference model is less critical, in that it is no longer related to the desired performance, but only to design a high-performance MPC outer loop. From a different perspective, the proposed approach allows one to design data-driven predictive controller directly from data, without explicitly modeling the open-loop plant dynamics, thus saving substantial development time. Future research will deal with: 1) extension to multivariable systems; 2) efficient online implementation of the outer MPC-based controller; and 3) design of robust controllers to consider a possible mismatch between the desired and actual inner closed-loop behavior.

References

[1] S. Formentin, D. Piga, R. Tóth, and S. M. Savaresi, "Direct learning of LPV controllers from data," *Automatica*, vol. 65, pp. 98–110, Sep. 2016.

[2] A. Bazanella, L. Campestrini, and D. Eckhard, *Data-Driven Controller Design: The H_2 Approach*. New York, NY, USA: Springer-Verlag, 2011.

[3] A. Bemporad, "Reference governor for constrained nonlinear systems," *IEEE Trans. Autom. Control*, vol. 43, no. 3, pp. 415–419, Mar. 1998.

[4] U. Kalabić, I. Kolmanovsky, and E. Gilbert, "Reduced order extended command governor," *Automatica*, vol. 50, no. 5, pp. 1466–1472, 2014.

[5] H. Hjalmarsson, M. Gevers, S. Gunnarsson, and O. Lequin, "Iterative feedback tuning: Theory and applications," *IEEE Control Syst.*, vol. 18, no. 4, pp. 26–41, Aug. 1998.

[6] A. Lecchini, M. Campi, and S. Savaresi, "Virtual reference feedback tuning for two degree of freedom controllers," *Int. J. Adapt. Control Signal Process*, vol. 16, pp. 355–371, Apr. 2002.

[7] C. Novara, S. Formentin, S. M. Savaresi, and M. Milanese, "Data-driven design of two degree-of-freedom nonlinear controllers: The D^2-IBC approach," *Automatica*, vol. 72, pp. 19–27, Sep. 2016.

[8] H. Nijmeijer and S. Savaresi, "On approximate model-reference control of siso discrete-time nonlinear systems," *Automatica*, vol. 34, no. 10, pp. 1261–1266, 1998.

[9] H. Abbas, R. Tóth, N. Meskin, J. Mohammadpour, and J. Hanema, "An MPC approach for LPV systems in input-output form," in *Proc. 54th Conf. Decision Control*, Osaka, Japan, 2015, pp. 91–96.

[10] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, Jan. 2002.

[11] A. Bemporad, "A multiparametric quadratic programming algorithm with polyhedral computations based on nonnegative least squares," *IEEE Trans. Autom. Control*, vol. 60, no. 11, pp. 2892–2903, Nov. 2015.

[12] A. Bemporad, M. Morari, and N. Ricker. (2015). Model predictive control toolbox for MATLAB 5.0. The Mathworks, Inc. [Online]. Available: http://www.mathworks.com/help/mpc/ug/adaptive-mpc.html

[13] B. Kulcsár, J. Dong, J. van Wingerden, and M. Verhaegen, "LPV subspace identification of a DC motor with unbalanced disc," in *Proc. IFAC Symp. Syst. Identification*, vol. 15. 2009, no. 1, pp. 856–861.

[14] G. Cimini, D. Bernardini, A. Bemporad, and S. Levijoki, "Online model predictive torque control for permanent magnet synchronous motors," in *Proc. IEEE Int. Conf. Ind. Technol.*, Seville, Spain, Mar. 2015, pp. 2308–2313.