On real-time smooth interconnection of online synthesized controllers in the behavioral framework

Tushar Jain\textsuperscript{a}\textsuperscript{*} and Joseph Julien Yamé\textsuperscript{b}

\textsuperscript{a}Aalto University, School of Chemical Technology, P.O. Box 16100, 00076-AALTO, Finland; \textsuperscript{b}Université de Lorraine, Centre de Recherche en Automatique de Nancy, CNRS, UMR 7039, Campus Sciences, 54506 Vandoeuvre-lès-Nancy, France

(Received 21 October 2013; final version received 3 April 2014)

In this paper, we present a real-time algorithm for synthesizing an online controller and its implementation in the closed-loop. The novelty of this algorithm lies in the fact that we do not use any a priori knowledge of the model of the plant in real-time. We use the mathematical framework of behavioral theory to demonstrate the online synthesizing and implementation mechanism.

Keywords: control synthesis; behavioral theory; smooth interconnection; linear systems; data-driven controller

1. Introduction

Online synthesis of a controller without using an a priori knowledge of the plant in real-time has found applications in the field of adaptive control, for example, under the unfalsified framework (Safonov & Tsao, 1997), and particularly, in dealing with reconfigurable fault-tolerant control systems (Jain, Yamé, & Sauter, 2012a). Synthesizing controllers methods without using a model of the plant are generally referred as “data-driven design” methods. No doubt these methods offer practical advantages from the fact that scarcely the precise model of the plant is available. Within the unfalsified framework, we are already provided with a set of pre-designed controllers and the rest of the job is to select the “right” controller that can deliver the performance specifications. However, it becomes more challenging whenever at some point there does not exist any right control within this pre-designed set and have to synthesize the right controller in real-time without using any a priori information of the plant model.

In the behavioral setting, controlling a plant is viewed as the interconnection of two dynamical systems, namely the plant and the controller. A linear time-invariant dynamical system \(\Sigma\) is described by a triple \(\Sigma = (T, S, B)\) where \(T \subseteq \mathbb{R}\) is the time axis, \(S \subseteq \mathbb{R}^s\) is the signal space, with \(\mathbb{R}^s\) denoting the \(s\)-dimensional real Euclidean vector space over the field of real numbers \(\mathbb{R}\), and \(B \subseteq S^T\) is the behavior. The set \(S\) is the space in which the system variables take on their values and the behavior \(B\) is a family of \(S\)-valued time trajectories, where a trajectory is a function \(s: T \rightarrow S, t \mapsto s(t)\) and \(s\) denotes the number of components in \(s\). Let \(S(\xi) \in \mathbb{R}^{s \times s}[\xi]\), and consider the following system of constant coefficient differential equations:

\[
S \left( \frac{d}{dt} \right) s = 0,
\]

where \(\mathbb{R}^{s \times s}[\xi]\) is the set of polynomial matrices with indeterminate \(\xi\) having an unspecified number of rows (of course, finite) and \(s\) number of columns. Then the behavior \(B\) is the set of solutions of the finite system (1) which is defined as

\[
B = \{ s \in (\mathbb{R}^s)^T \mid \text{Equation (1) satisfies} \}.
\]

Representation (1) is called a kernel representation of \(B\) and sometimes, we denote the behavior as \(B = ker(\Sigma(d/dt))\). The behavior of a dynamical system can also be described using the latent/auxiliary variable that either appears naturally in the modeling process or can be artificially introduced. They are often termed as state variables. These variables have the property that they parametrize the memory of the system, i.e. they “split” the past (\(p\)) and future (\(f\)) of the behavior (Polderman & Willems, 1997, Chapter 4).

In this mathematical framework, the control problem is posed in the following way. Given the behavior of the plant, and the desired behavior that “captures” the performance specifications, find a controller such that after interconnecting it with the plant, the interconnected system satisfies these specifications. The behavioral description of generating this controller was proposed in van der Schaft (2003, Section 3). This controller is termed as the canonical controller. In van der Schaft (2003), the controller is constructed for general systems without imposing any

\*Corresponding author. Email: tushar.jain@aalto.fi

© 2014 The Author(s). Published by Taylor & Francis.
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. The moral rights of the named author(s) have been asserted.
realizability requirements. This issue is of utmost practical importance for a possible implementation of the controller in the closed-loop. Theoretically, in Julius, Willems, Belur, and Trentelman (2005, Theorem 16), the so-called regularity of interconnection is imposed for the “design” of the canonical controller. Trentelman, Yoe, and Praagman (2007) established algorithms for the computation of controllers that regularly implement a given desired behavior. While solutions, by designing controllers in the behavioral framework, of the so-called asymptotic tracking and regulation problem are presented in Fiaz, Takaba, and Trentelman (2011). Nevertheless, the reported computation procedure in the aforesaid literature requires the knowledge of plant parameters in real-time. On the other hand, algorithms for the design of data-driven control within the behavioral framework are reported in Markovsky and Rapisarda (2008) and Markovsky (2010). However, these algorithms construct the offline controller and compute implicitly the impulse response of the plant.

The contributions of this paper are two-fold. Firstly, the aim is to design an online controller based on the real-time measurements generated by the plant. The subsequent step is to implement this online synthesized controller in the closed-loop by replacing the previous controller through switching. Nevertheless, an instant switching of the controller leads to a deterioration of the system’s performance (Yamé & Kinnairt, 2007). In this context, the second objective is to guarantee that the online designed controller makes a smooth interconnection with the plant in the sense that it does not affect the closed-loop performance at the time of interconnection. In the course of achieving the above objectives, we do not use any a priori information of the plant in real-time. The problems considered here are quite significant for systems dealing with active fault-tolerant control problems where the closed-loop is required to satisfy the performance specifications at anytime (Jain, Yamé, & Sauter, 2011).

2. Feedback Interconnection in behavioral context

The interconnection of two dynamical systems is often dealt in two contexts, namely the full interconnection and the partial interconnection. In the former, all system variables take part in an interconnection while in the latter only a few of them take part. Our point of discussion revolves around the case of full interconnection. Consider two dynamical systems \( \Sigma_1 = (\mathcal{T}, \mathcal{S}, \mathcal{B}_1) \) and \( \Sigma_2 = (\mathcal{T}, \mathcal{S}, \mathcal{B}_2) \) with the common time axis \( \mathcal{T} \), and the common signal space \( \mathcal{S} \). The interconnection of \( \Sigma_1 \) and \( \Sigma_2 \) is defined by \( \Sigma = (\mathcal{T}, \mathcal{S}, \mathcal{B}_1 \cap \mathcal{B}_2) \), where \( \mathcal{B}_1 \cap \mathcal{B}_2 = \{ s | s \in \mathcal{B}_1 \text{ and } s \in \mathcal{B}_2 \} \). Here, \( \cap \) denotes the interconnection of two systems, while \( \cup \) denotes the intersection of the behaviors of the two systems. In terms of kernel representations, let \( \Sigma_1 \) be described by \( R_1(\xi)s = 0 \), and \( \Sigma_2 \) by \( R_2(\xi)s = 0 \). Then, \( \Sigma_1 \cap \Sigma_2 \) is described by \( \begin{bmatrix} R_1(\xi) \\ R_2(\xi) \end{bmatrix} s = 0 \). In this way, a dynamical system imposes restrictions on another dynamical system such that the interconnected system satisfies the laws of both systems.

It is shown in Willems (1997) that by the sole interconnection one may not impose restrictions on the trajectories to achieve the control specifications. See Jain, Yamé, and Sauter (2012b) for a motivational example. Further restrictions are needed to achieve the so-called regular interconnection. Whenever a dynamical system makes an interconnection with another, the laws which are already present in one system should not be repeated in another. Such interconnections in which only new laws are imposed are termed as regular. It turns out that imposing the regularity on the interconnection is completely satisfied by considering the control problem in the feedback configuration as illustrated in Figure 1. This configuration allows us to partition the system variables into inputs and outputs. Consequently, the system in Equation (1) can equivalently be represented by the system having the following special form:

\[
P \left( \frac{d}{dt} \right) w_1 = Q \left( \frac{d}{dt} \right) w_2, \quad s = \Pi \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}
\]

with \( P \in \mathbb{R}^{s \times w}[\xi], Q \in \mathbb{R}^{s \times w}[\xi], w_1 + w_2 = s, \) and \( \Pi \) an \( s \times s \) permutation matrix. It is even possible to choose the permutation matrix such that \( P^{-1}Q \), with \( \det(P) \neq 0 \) is a matrix of proper rational functions. In this respect, the variable \( w_2 \) acts as the input and \( w_1 \) as the output in Equation (2).

In Figure 1(b), \( \mathcal{P} \) denotes the behavior of the plant, and \( \mathcal{C} \) denotes the behavior of the controller with system variable \( s = (r^T, y^T, u^T)^T \). Given their kernel representation, these

Figure 1. Feedback configuration in the case of full interconnection. (a) In classical setting and (b) in behavioral setting.
behaviors are defined as
\[ \mathcal{P} = \{ s \in (\mathbb{R}^2)^T | \mathcal{R}(s) = 0 \}, \quad R = [0 \quad d_p \quad -n_p], \]
\[ \mathcal{C} = \{ s \in (\mathbb{R}^2)^T | \mathcal{C}(s) = 0 \}, \quad C = [n_c \quad -n_c \quad -d_c], \quad (3) \]
where \( d_p(x) \in \mathbb{R}^{1 \times 1}[s] \), \( n_p(x) \in \mathbb{R}^{1 \times 1}[s] \) with \( d_p(x) \neq 0 \) and \( d_c(x) \in \mathbb{R}^{1 \times 1}[s] \), \( n_c(x) \in \mathbb{R}^{1 \times 1}[s] \) with \( d_c(x) \neq 0 \) for single-input single–output systems. All polynomials in the kernel representation of the behavior are assumed to be co-prime. The interconnection of \( \mathcal{P} \) and \( \mathcal{C} \) yields the controlled behavior \( \mathcal{K} \), which is defined as \( \mathcal{K} = \{ s \in (\mathbb{R}^2)^T | s \in \mathcal{P} \} \). In this case, we say that for a given \( \mathcal{P} \), \( \mathcal{K} \) is implemented by \( \mathcal{C} \) or \( \mathcal{C} \) implements \( \mathcal{K} \). The following result from Rocha and Wood (2001, Lemma 3.5) gives the implementability condition on \( \mathcal{K} \).

**THEOREM 2.1** The behavior \( \mathcal{K} \) is implementable w.r.t. \( \mathcal{P} \) by the full interconnection if and only if \( \mathcal{K} \subset \mathcal{P} \).

### 3. Online controller design

Control problems are always accompanied by certain criteria, which single out specific sub-behaviors as desirable. Here, such a criterion is defined in terms of the system variable, and we call it the desired behavior \( \mathcal{D} \in (\mathbb{R}^2)^T \). In terms of a kernel representation, it is defined by \( \mathcal{D} = \ker(D(\xi)) \), or more explicitly,
\[ D(\xi)s = \begin{bmatrix} n_{r_T} & -d_T & 0 \\ n_{r_s} & 0 & -d_T \end{bmatrix} \begin{bmatrix} r \\ y \\ u \end{bmatrix} = 0, \quad (4) \]
where \( d_T(\xi) \in \mathbb{R}^{1 \times 1}[s], n_{r_T}(\xi) \in \mathbb{R}^{1 \times 1}[s], n_{r_s}(\xi) \in \mathbb{R}^{1 \times 1}[s] \) with \( d_T(\xi) \neq 0 \) are given. Therefore, the control problem now turns into synthesizing a controller \( \mathcal{C} \) for the given \( \mathcal{P} \) and \( \mathcal{D} \) such that when it interconnects with the plant, the interconnected system satisfies the desired behavior.

In our approach, we do not have any access to a priori plant parameters, i.e. the matrix \( R(\xi) \) is not available during the design process, and we do not even estimate it. That is why, we have shadowed the plants’ behavior in Figure 1. Precisely, our interest lies in synthesizing an online controller without using the polynomials \( d_p(\xi) \) and \( n_p(\xi) \). It is a fact that the given desired behavior should be implementable otherwise no controller can achieve the desired specifications. So from Theorem 2.1, we have the following implementability condition on \( \mathcal{D} \) as \( \mathcal{D} \subset \mathcal{P} \).

**PROPOSITION 3.1** Let \( \mathcal{D} \) be an implementable desired behavior. Then, controller \( \mathcal{C} \) implements \( \mathcal{D} \), if and only if \( \mathcal{C} \) is designed using the synthesis equation
\[ n_c \ddot{y} + d_c \ddot{u} = 0. \quad (7) \]

**Proof** First, we will show that if \( \mathcal{D} \) is implementable then there exists a controller \( \mathcal{C} \) that implements \( \mathcal{D} \). Later, we will derive the explicit relation to compute the controller polynomials. From the inclusion (5), there exists a polynomial matrix \( F(\xi) \in \mathbb{R}^{1 \times 2}[s] \) such that
\[ R = FD \quad (8) \]
with \( F = [f_1, f_2] \). Let \( V(\xi) = [n_c(\xi), n_d(\xi)] \in \mathbb{R}^{1 \times 2}[s] \) be a polynomial matrix such that the matrix \( \begin{bmatrix} F \\ V \end{bmatrix} \), noted \( \text{column}(F, V) \) for conciseness, is unimodular. Define the controller as
\[ C = VDs = 0. \quad (9) \]
The controlled behavior \( \mathcal{K} \) obtained from the interaction of controller \( \mathcal{C} \) defined by Equation (9) with the plant \( \mathcal{P} \) is then given by
\[ \mathcal{K} = \mathcal{P} \cap \mathcal{C} = \ker \left( \begin{bmatrix} R \\ VD \end{bmatrix} \right). \quad (10) \]

Since \( \mathcal{D} \) is implementable, we have Equation (8) and by putting this in Equation (10), we get
\[ \mathcal{K} = \ker \left( \begin{bmatrix} F \\ V \end{bmatrix} D \right). \]

By assumption, \( \text{column}(F, V) \) is unimodular. From Polderman and Willems (1997, Theorem 2.5.4), we conclude, \( \mathcal{K} = \mathcal{D} \). This proves that the controller \( \mathcal{C} \) defined in Equation (9) implements the desired behavior \( \mathcal{D} \). Now we proceed to derive relationship (7). Writing Equation (9) explicitly, we have
\[ \begin{bmatrix} n_c & d_c \\ n_{r_T} & -d_T & 0 \\ n_{r_s} & 0 & -d_T \end{bmatrix} \begin{bmatrix} r \\ y \\ u \end{bmatrix} = 0. \quad (11) \]

From the above equation, it is only required to compute the polynomials \( n_c(\xi) \) and \( d_c(\xi) \).

Write Equation (11) as
\[ \begin{bmatrix} n_c n_{r_T} + d_c n_{r_s} & -n_c d_T & -d_c d_T \end{bmatrix} \begin{bmatrix} r \\ y \\ u \end{bmatrix} = 0. \]

Owing to the structure of the controller as defined in Equation (3), we should have \( n_c n_{r_T} + d_c n_{r_s} = n_c d_T \). Thus,
we can write the last equation as

$$\begin{bmatrix} n_c d_T & -d_c d_T & -d_c d_T \\ n_T y & -d_T & -d_T & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r \\ y \\ u \end{bmatrix} = 0.$$  

The trajectories belonging to the above equation also satisfy the equation given by the first row of Equation (4), that is,

$$\begin{bmatrix} n_c d_T & -d_c d_T & -d_c d_T \\ n_T y & -d_T & -d_T & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r \\ y \\ u \end{bmatrix} = 0. \tag{12}$$

From this, we can now eliminate the $r$-variable. Premultiplying Equation (12) by the matrix $\begin{bmatrix} n_T & -n_c d_T & 0 & 1 \end{bmatrix}$, it yields

$$\begin{bmatrix} 0 & (n_c d_T - n_c n_T d_T) d_T & -d_n n_T d_T \\ n_T y & -d_T & -d_T & 0 \end{bmatrix} \begin{bmatrix} r \\ y \\ u \end{bmatrix} = 0. \tag{13}$$

Clearly, the trajectory $s$ belonging to the behavior described by Equation (12) implies that $s$ belongs to the behavior described by Equation (13) (see Polderman and Willems (1997, Theorem 5.2.14)). The controller polynomials can now be evaluated by the first row of the above equation which is given as

$$\begin{bmatrix} n_c d_T & -d_c d_T & -d_c d_T \\ n_T y & -d_T & -d_T & 0 \end{bmatrix} \begin{bmatrix} y \\ u \end{bmatrix} = 0. \tag{14}$$

Simplifying it further, we can write it as

$$\begin{bmatrix} n_c & d_c \\ n_T y & -d_T \\ 0 \end{bmatrix} \begin{bmatrix} y \\ u \end{bmatrix} = 0 \Leftrightarrow n_c y + d_c u = 0. \tag{15}$$

with $\ker(d_T(\xi).I_n)$, where $I_n$ is the identity matrix of dimension $n$, describing the stable autonomous behavior with Equation (15) representing the controllable part of the behavior described by the trajectories $(y, u)$ in Equation (14) (see Polderman & Willems, 1997, Theorem 5.2.14 for the decomposition of behaviors).

Equation (15) assists us in synthesizing the controller polynomials $n_c(\xi)$ and $d_c(\xi)$ for the given desired behavior and observed trajectories generated from the plant. Interestingly, solving the above equation now becomes solving a system identification problem. This can be solved using various methods listed in the literature (Isermann & Münchhof, 2011), where the controller is identified directly instead of identifying the plant primarily. Note that on fixing the degree of controllers’ polynomials, the controller synthesized using the tools borrowed from the system identification community becomes an approximated controller. For realizing the controller we enforce its structure to be a bi-proper rational function. The significance of imposing the bi-properness will be discussed in Section 4. Once the controller that can achieve the performance specifications is synthesized, the next step is to implement this controller in real-time. This requires to replace the previously operating controller by switching the online synthesized controller within the closed-loop.

4. Real-time implementation of the controller

From now on, we will make reference to a time-dependent subset of the behavior $\mathcal{P}$. This time-dependent subset, denoted as $\mathcal{P}^t_m$, is characterized by the set of signals experimentally measured on the time interval $(-\infty, t]$ where $t$ is the current time. More precisely, let $(u_m, y_m)$ be the input/output (of the real plant) actually measured in an experimental setting, as e.g. in an actual closed-loop setting, then this subset $\mathcal{P}^t_m$ of the behavior $\mathcal{P}$ is defined by

$$\mathcal{P}^t_m = \left\{ s = (r, u, y) \in \Sigma^T \text{ s.t. } \begin{bmatrix} u(\tau) - u_m(\tau) \\ y(\tau) - y_m(\tau) \end{bmatrix} = 0 \text{ for all } \tau \in (-\infty, t) \right\}. \tag{16}$$

With this definition, it is clear that $\mathcal{P}^t_m \subseteq \mathcal{P}$ for all $t \in (-\infty, \infty)$

Consider the scenario illustrated in Figure 2, where $\mathcal{C}_p$ is the past controller with which the measurement set $\mathcal{P}^t_m$ is formed and a new controller $\mathcal{C}_f$, i.e. the future controller, is synthesized online using Proposition 3.1. It is well known that whenever an instantaneous switching of a controller is performed in real-time at $t = t_{\text{inter}}$, undesirable transients might appear in the closed-loop that significantly deteriorates the control performance (Graebe & Ahlén, 1996). It is argued in Yamé and Kinnaert (2007), Kinnaert, Delwiche, and Yamé (2009) that the main cause behind the appearance of these undesirable transients, called bumps, is the lack of dynamical consistency between the “state trajectory” of the controller $\mathcal{C}_f$ before and after the switching or interconnection instant $t = t_{\text{inter}}$. Note that these undesirable transients appearing at the time of interconnection are not taken into account while synthesizing the controller. Our next objective is therefore to implement $\mathcal{C}_f$ in the running closed-loop system in a way such that the overall controlled behavior still satisfies the desired behavior despite

![Figure 2](image_url)
the switching commands. When a real-time interconnection is achieved in this way, we qualify it as a *smooth* real-time interconnection. To make precise this notion of smoothness, let \( K_{\text{pic}} \) and \( K_{\text{fic}} \) denote the past interconnected system and the future interconnected system, respectively, and \( t_{\text{inter}} \) is the time at which the interconnection takes place,

\[
K_{\text{pic}} = \{(r,y,u)(r,u,y) \in \mathcal{P} \text{ and } (r,y,u) \in C_P\}
\]

\[
\forall t < t_{\text{inter}},
\]

\[
K_{\text{fic}} = \{(r,y,u)(r,u,y) \in \mathcal{P} \text{ and } (r,y,u) \in C_F\}
\]

\[
\forall t \geq t_{\text{inter}},
\]

**Definition 4.1** The interconnection between the controller \( C_f \) and the running plant \( P \) is said to be a real-time smooth interconnection whenever \( K_{\text{fic}} \subseteq D \).

Clearly, the real-time interconnection is not smooth whenever controller switching leads to closed-loop signals which no longer belong to the desired behavior, such signals being usually termed as bumpy signals. The next proposition, which is one of the main result of this paper, is of paramount importance and will be the basis for the derivation of the algorithm in achieving a real-time smooth interconnection.

**Proposition 4.2** Let \( D \) be an implementable desired behavior and consider the controller switching scenario as depicted in Figure 2. Then, controller \( C_f \) can be smoothly interconnected to \( P \) in real-time if there exists a trajectory \( \tilde{y} \) such that \( \tilde{s} = (\tilde{r}, \tilde{u}_m, \tilde{y}_m) \in \mathcal{P}_m \cap C_f \subseteq D \) for all \( t < t_{\text{inter}} \).

**Proof** Let \( s_f \) be the trajectory of the closed-loop switched-mode system after \( t_{\text{inter}} \), i.e. after switching \( C_f \) in the loop. Then, under the stated condition, the achievability of a real-time smooth interconnection of \( C_f \) with \( P \) at time \( t_{\text{inter}} \) is trivially equivalent to the fact that the following concatenated signal, denoted \( s \), and defined by

\[
s = \tilde{s} \circ_{t_{\text{inter}}} s_f \iff s(t) = \begin{cases} 
\tilde{s}(t), & t < t_{\text{inter}}, \\
{s_f(t)}, & t \geq t_{\text{inter}}
\end{cases} \tag{17}
\]

should belongs to the desired behavior, i.e.

\[
D \ni s = \tilde{s} \circ_{t_{\text{inter}}} s_f. \tag{18}
\]

Clearly, the membership relationship (18) implies \( s_f \in D \), which means that the interconnection is actually smooth.

**Remark 1** First, it is worth noticing that Proposition 4.2 implicitly introduces a “virtual” behavior, i.e. a set of signals \( \tilde{s} = (\tilde{r}, \tilde{u}_m, \tilde{y}_m) \) corresponding to a loop in which \( C_f \) and \( P \) are fictively connected before time \( t_{\text{inter}} \) (though in reality \( P \) is connected with \( C_P \) for all \( t < t_{\text{inter}} \)). Second, the significance of Proposition 4.2 is that in order to achieve a smooth interconnection when \( C_f \) will be effectively switched in the loop for all \( t \geq t_{\text{inter}} \), this virtual loop should already satisfies to the desired behavior \( D \).

In the proof of Proposition 4.2, the membership relationship (18) is exhibited as the main requirement for achieving a real-time smooth interconnection. Therefore, the question arises as: under what condition, the relationship (18) holds?

To answer this question, we need more information on the behavior of the system. Observe that, thanks to Remark 1, the trajectory (17) might be viewed as that of the closed-loop system consisting of \( C_f \) being constantly in the loop since ever and forever. This trajectory is in fact splitted at \( t = t_{\text{inter}} \) in two parts, that is, a past trajectory and a future trajectory. Although these manifest trajectories may be the main signals of immediate interest, there may be additional independent variables in the system which allow a more complete description of the behavior. In particular, for a future trajectory in the behavior \( D \) to be a continuation of a past trajectory in \( D \) obviously would require some boundary conditions to be met at the splitting time. These boundary conditions can be expressed through the aforementioned independent variables which are usually called the state of the system. The manifest trajectories might therefore be explicitly parametrized with these new independent variables (i.e. the state) so as to catch all the information about the past which are relevant to do a continuation of the trajectory in the future. It can be proved that such a parametrization always exists (Polderman & Willems, 1997). Let us denote the state of the virtual closed-loop and future behavior of the system, respectively, by \( \ell \) and \( \ell_f \) and parametrize explicitly the manifest trajectory of the virtual closed-loop and future manifest trajectory with respect to their states as \( \tilde{s}(\ell) \) and \( s_f(\ell_f) \). The following lemma which is a direct consequence of the property of the state answers to the raised question above.

**Lemma 4.3** Let the manifest trajectories \( \tilde{s}(\ell) \) and \( s_f(\ell_f) \) be elements of the behavior \( D \), then their concatenation at time \( t = t_{\text{inter}} \) also belongs to \( D \) if \( \ell(t^{-}_{\text{inter}}) = \ell(t^{+}_{\text{inter}}) \).

As we are dealing with behaviors of controlled systems consisting in the interconnection of a plant \( P \) and a controller \( C \), the state \( \ell \) of the controlled behavior \( P \cap C \) is a vector which can be partitioned explicitly as \( \ell = [(\xi^P)^T (\xi^C)^T]^T \) where \( \xi^P, \xi^C \) are, respectively, the states of the plant and the controller. Thanks to this partitioning, the state of the virtual closed-loop behavior \( P^m_m \cap C_f \) for \( t \in (-\infty, t_{\text{inter}}) \) is written as \( \tilde{\ell} = [(\xi^P)^T (\xi^C)^T]^T \) and the state of the future behavior as \( \ell_f = [(\xi^P)^T (\xi^C)^T]^T \) for all \( t > t_{\text{inter}} \). Introduce the (past) state \( \ell_p = [(\xi^P)^T (\xi^C_m)^T]^T \) of the actual loop \( P^m_m \cap C_p \) evolving on the time axis \((-\infty, t_{\text{inter}})\), then clearly \( \xi^P = \xi^P_p \). It is a well-known fact (Bellman & Cooke, 1995) that despite a possible discontinuity of the manifest variables (attached to \( P \)) at the
switching instant \( t_{inter} \), the state of the plant is continuous at \( t = t_{inter} \), i.e. \( \zeta^{P'}(t_{inter}^+) = \zeta^{P'}(t_{inter}^-) \) which implies that \( \dot{\zeta}^{P'}(t_{inter}^-) = \dot{\zeta}^{P'}(t_{inter}^+) \). From the above, it turns out that the boundary condition \( \tilde{\ell}(t_{inter}^-) = \tilde{\ell}(t_{inter}^+) \) in Lemma 4.3 is achieved if and only if \( \dot{\zeta}^{C'}(t_{inter}^-) = \dot{\zeta}^{C'}(t_{inter}^+) \). This means that the state of the to-be-switched controller \( C'_f \) should be initialized at the switching instant \( t = t_{inter} \) with the value \( \dot{\zeta}^{C'}(t_{inter}^-) \) of the state that controller \( C'_f \) would have achieved, had it been in the loop on the (past) time interval \((-\infty, t_{inter})\).

Next, the question arises of how to compute explicitly the state \( \zeta \) of a controller when the controller is given by its kernel representation as in Equation (3). A general result, yielding an algorithm for computing the state parameterizing the manifest behavior of a dynamical system from Rapisarda and Willems (1997), is specialized here to a controller \( C \) by the following theorem.

**Theorem 4.4** Given the manifest variable \( s \in \mathbb{R}^n \), the following statements are equivalent:

1. The manifest variable \( s \) belongs to the controller behavior \( C \), i.e. \( C(d/dt)s = 0 \).
2. There exist an integer \( n_C \), a polynomial matrix \( \chi \in \mathbb{R}^{n_C \times n}[\xi] \) and a state \( \zeta^C \) such that \( \dot{\zeta}^C = \chi(d/dt)s \).

**Proof** See Rapisarda and Willems (1997, Theorem 6.2).

Note that the polynomial matrix \( \chi \) in Theorem 4.4 can be obtained by reduction of the kernel representation \( C(d/dt)s = 0 \) to a first-order representation. An efficient algorithm is proposed in Rapisarda and Willems (1997, Algorithms 1–3) to compute the polynomial matrix \( \chi \) using iteratively the shift-and-cut operation, defined as an operator \( \sigma_\chi : \mathbb{R}(\xi) \to \mathbb{R}(\xi) \) (see Rapisarda & Willems, 1997, Definition 5.1). As a simple illustration of this algorithm, for a system described by \( C(d/dt)s = 0 \) with \( C(\xi) = c_0 + c_1 \xi + \cdots + c_M \xi^M, c_M \neq 0, \forall \xi \in \mathbb{R}^n \), the \( \chi \) matrix and, hence the state \( \zeta^C \), is computed by an iterative application of the shift-and-cut operator,

\[
\zeta^C = \left[ \begin{array}{c}
(1 + c_2 \frac{d}{dt} + \cdots + c_M \frac{d^{M-1}}{dt^{M-1}}) s \\
( c_2 + \cdots + c_M \frac{d^{M-2}}{dt^{M-2}} ) s \\
\vdots \\
( c_M \frac{d}{dt} ) s \\
\end{array} \right] = \chi \left( \frac{d}{dt} \right) s.
\]  

(19)

We are now in a position to summarize the above results in the following algorithm.

**Algorithm** (control synthesis and its implementation)

**Data:** \( \mathcal{P}_m' \subseteq \mathcal{P} \) available at time \( t \).

**Result:** Online controller design and smooth interconnection in real-time.

1. For \( t < t_{inter} \), observe the experimental plant behavior \( \mathcal{P}_m' \) and synthesize the online controller \( C_t \) using Equation (7) given in Proposition 3.1.
2. For \( t_{inter} - t \leq t < t_{inter} \), compute the virtual manifest trajectory \( \bar{s} \), given in Proposition 4.2, and subsequently the corresponding virtual state \( \bar{\zeta} \) of the controller, with \( t \geq t_{settling} \) where \( t_{settling} \) is the settling time of the desired behavior \( D \).
3. At \( t = t_{inter} \), initialize \( \bar{\zeta}^{C'}(t_{inter}^-) \) as \( \bar{\zeta}^{C'}(t_{inter}^+) = \zeta^{C'}(t_{inter}^-) \) to achieve a smooth interconnection of \( \mathcal{P} \) and \( C_t \).
4. Stop.

**Remark 2** In step 2 of the above algorithm, the rationale behind considering a time interval \([t_{inter} - t, t_{inter}]\) for computing the virtual signals comes from the following: observe that the computation of the reference trajectory \( \bar{r} \) in Proposition 4.2, and hence the construction of \( \bar{s} \) should be done on all \( t < t_{inter} \) to correctly calculate at the switching time \( t_{inter} \) the value \( \bar{\zeta}^{C'}(t_{inter}^-) \) of the state of the controller to be switched in the loop. However, from a practical standpoint related to the computation of a “very good” estimate for \( \bar{\zeta}^{C'}(t_{inter}^-) \), it is quite sufficient to generate \( \bar{s} \) over a finite-time window larger than the settling time of the desired behavior \( D \).

5. **Numerical example**

We present an academic example to demonstrate the effectiveness of the theory developed above. The control objective is to design an online controller for an unknown plant (i.e. where the polynomials \( n_p(\xi) \) and \( d_p(\xi) \) are not known) such that the \( y \)-trajectory follows the \( r \)-trajectory. The implementable desired behavior that captures the control objective is given by \( D(\xi)s = 0 \), where \( n_{r}(\xi) = 40, n_{r}(\xi) = 13.33(\xi^2 + 4\xi + 1), d_{r}(\xi) = \xi^2 + 14\xi + 40, s = [r^T y^T u^T]^T \). A controller having unknown parameters \( (a, b, c) \) is fixed with polynomials \( n_{i}(\xi) = (\xi + 1)(a\xi + b), d_{i}(\xi) = \xi(\xi + c) \) at the outset. Initially, we choose a controller with arbitrary parameters \( (a, b, c)_p = (53.33, 160, 44) \) which is interconnected with the unknown plant. An experimental plant behavior \( \mathcal{P}_m' \) is observed up to the time \( t_{inter} \). For simulation purpose, we consider \( t_{inter} = 5 \) s. Using system identification techniques as described in Isermann and Münchhof (2011), the polynomials in Equation (7) are computed using the online measurements and the given desired behavior.
Note that by the synthesis approach, the obtained parameters of the controller are such that when it makes an interconnection with the plant, the closed-loop achieves the desired specifications. The controller parameters are computed as \((a, b, c, T) = (13.33, 40, 14)\).

From the theory developed in Section 4, the fulfillment of the condition given in Lemma 4.3 is actually achieved through \(\tilde{z}^{C_{r}}(t_{\text{int}}^{C}) = \tilde{z}^{C_{r}}(t_{\text{int}}^{C})\). The differential equation describing the manifest behavior of the controller is given by \((d^2/dt^2)u + 14(d/dt)u = 13.33(d^2/dt^2)(\tilde{r} - y) + 53.33(d/dt)(\tilde{r} - y) + 40(\tilde{r} - y)\).

Using the shift-and-cut operation, we compute the state trajectory as

\[
\tilde{z}^{C_{r}} = \chi \left( \frac{d}{dt} \right) \tilde{s} = \left[ \begin{array}{c} \sigma_{+}(C(\xi)) \\ \sigma_{+}^{2}(C(\xi)) \end{array} \right] \tilde{s},
\]

\[
\tilde{z}^{C_{r}} = \chi \left( \frac{d}{dt} \right) \tilde{s} = \left[ \begin{array}{cc} 1 + 14 & 13.33 \\ 1 & 13.33 \end{array} \right] \tilde{s},
\]

where \(\sigma_{+}^{2}(\cdot)\) is the operator resulting from iterating twice \(\sigma_{+}(\cdot)\). Note that the above state representation does not distinguish between an input–output partition of the manifest variables. For example, assuming \((\tilde{r} - y)\) as the input and \(u\) as the output, the above equation can be written in classical state-space equation as

\[
\tilde{z} = \left[ \begin{array}{cc} 0 & 3 \\ 1 & \mp 4 \end{array} \right] \tilde{z} + \left[ \begin{array}{cc} 3 \\ -10 \end{array} \right] \left[ \begin{array}{c} u \\ y \end{array} \right],
\]

\[
u = \left[ \begin{array}{cc} 0 \\ 1 \end{array} \right] \tilde{z} + 13.33(\tilde{r} - y),
\]

while assuming \([u, y]^T\) as the input and \(\tilde{r}\) as the output, one obtains

\[
\tilde{r} = \left[ \begin{array}{cc} 0 \\ -1 \end{array} \right] \tilde{z} + \left[ \begin{array}{cc} 1 \\ \mp 13.33 \end{array} \right] \left[ \begin{array}{c} u \\ y \end{array} \right].
\]

Note that, as by a product of representation (21), we simultaneously generate the virtual state of the controller \(C_{r}\) and the reference trajectory \(\tilde{r}\) of the virtual closed-loop system. The closed-loop responses are illustrated in Figure 3.

6. Conclusion

In the behavioral setting, solving a control problem is all about making an interconnection between the plant and the controller. In this paper, we have presented the design of an online controller using the real-time measurements generated by the plant in a closed-loop setting. After synthesizing the so-called data-driven controller, it should be interconnected to the plant by a real-time switching mechanism. Generally, this interconnection in real-time is not so smooth. We proposed a real-time algorithm that guarantees the smooth interconnection between the unknown plant and the online synthesized controller. Note that, in this paper, the synthesis problem was addressed under the unity feedback configuration. Nevertheless, from a theoretical viewpoint, synthesizing controllers under a more general type of feedback configuration, such as e.g. when the controller has two degrees of freedom, would be worth investigating within the behavioral system-theoretic framework.

References

Bellman, R., & Cooke, K. L. (1995). Modern elementary differential equations (2nd ed.). Mineola, NY: Dover Publications.

Fiaz, S., Takaba, K., & Trentelman, H. L. (2011). Tracking and regulation in the behavioral framework. *Automatica, 47*, 2343–2354.

Graebe, S. F., & Ahlén, A. (1996). Bumpless Transfer. In W. S. Levine (Ed.), *The Control Handbook* (pp. 381–388). Boca Raton, FL: IEEE Press.

Isermann, R., & Münchhof, M. (2011). *Identification of dynamic systems: An introduction with applications*. Berlin: Springer-Verlag Berlin.

Jain, T., Yamé, J., & Sauter, D. (2011). Synergy of canonical control and unfalsified control concept to achieve fault tolerance. In S. Bittanti, A. Cenedese, & S. Zampieri (Eds.), *Proceedings of 18th IFAC world congress* (Vol. 18, pp. 14832–14837). Milan, Italy: IFAC.

Jain, T., Yamé, J. J., & Sauter, D. (2012a). Model-free reconfiguration mechanism for fault tolerance. *International Journal of Applied Mathematics and Computer Science, 22*(1), 125–137.
Jain, T., Yamé, J. J., & Sauter, D. (2012b). On implementing on-line designed controller for smooth interconnection in the behavioral framework. In Bendtsen, J. D. (Ed.), 7th IFAC Symposium on Robust Control Design (ROCOND’12) (Vol. 7, pp. 424–429). Denmark: Aalborg.

Julius, A. A., Willems, J. C., Belur, M. N., & Trentelman, H. L. (2005). The canonical controllers and regular interconnection. Systems & Control Letters, 54(8), 787–797.

Kinnaert, M., Delwiche, T., & Yamé, J. (2009). State resetting for bumpless switching in supervisory control. Proceeding of the European Control Conference (EUCA’09), Budapest, Hungary, pp. 2097–2102.

Markovsky, I. (2010). Closed-loop data-driven simulation. International Journal of Control, 83(10), 2134–2139.

Markovsky, I., & Rapisarda, P. (2008). Data-driven simulation and control. International Journal of Control, 81(12), 1946–1959.

Polderman, J. W., & Willems, J. C. (1997). Introduction to mathematical systems theory: A behavioral approach. New York: Springer-Verlag.

Rapisarda, P., & Willems, J. C. (1997). State maps for linear systems. SIAM Journal on Control and Optimization, 35(3), 1053–1091.

Rocha, P., & Wood, J. (2001). Trajectory control and interconnection of 1D and nD systems. SIAM Journal on Control and Optimization, 40(1), 107–134.

Safonov, M., & Tsao, T. C. (1997). The unfalsified control concept and learning. IEEE Transactions on Automatic Control, 42(6), 843–847.

Trentelman, H. L., Yoe, R. Z., & Praagman, C. (2007). Polynomial embedding algorithms for controllers in a behavioral framework. IEEE Transactions On Automatic Control, 52(11), 2182–2188.

van der Schaft, A. (2003). Achievable behavior of general systems. Systems & Control Letters, 49(2), 141–149.

Willems, J. C. (1997). On interconnections, control, and feedback. IEEE Transactions on Automatic Control, 42, 326–339.

Yamé, J. J., & Kinnaert, M. (2007). On bumps and reduction of switching transients in multicontroller systems. Mathematical Problems in Engineering, 2007, 17 pp. http://dx.doi.org/10.1155/2007/54212