AIAA SciTech Forum
7-11 January 2019, San Diego, California
AIAA Scitech 2019 Forum

10.2514/6.2019-1426



# Predictive Learning via Lookahead Simulation

Aris Kanellopoulos*, Kyriakos G. Vamvoudakis†
*Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA, 30332, USA*

Yorai Wardi‡
*School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, 30332, USA*

**In this paper, target tracking is achieved through a variable gain integrator for a system whose dynamics are unknown or uncertain. Towards that, a predictor-based dynamic controller is utilized which, in the case of dynamical systems, is implemented via a simulation subsystem which operates alongside the plant. To relax the need for complete knowledge of the system model by the simulator, we augment the model-based variable gain integrator with a learning approximator. The lookahead simulation defines the control input based on the current approximation of the system, which is improved as the approximator learns. Finally, in order to decrease the usage of the controller's resources, we implement an event-triggered control strategy. The efficacy of the approach is shown through simulation examples on nonlinear systems.**



## Nomenclature

| | | |
|---|---|---|
| $x \in \mathbb{R}^n$ | = | system states |
| $u \in \mathbb{R}^m$ | = | control input |
| $y \in \mathbb{R}^p$ | = | system output |
| $r \in \mathbb{R}^p$ | = | target reference trajectory |
| $W \in \mathbb{R}^{d\times n}$ | = | ideal approximator weights for the input dependent activation function |
| $\hat{W} \in \mathbb{R}^{d\times n}$ | = | current approximator weights |
| $T \in \mathbb{R}$ | = | lookahead horizon |
| $\sigma(x,u) : \mathbb{R}^{n+m} \to \mathbb{R}^d$ | = | nonlinear input dependent activation function |
| $h(x) : \mathbb{R}^n \to \mathbb{R}^d$ | = | filtered regressor vector |
| $l(x) : \mathbb{R}^n \to \mathbb{R}^n$ | = | approximation correction term |
| $f(x) : \mathbb{R}^n \to \mathbb{R}^n$ | = | drift dynamics |
| $k(x) : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ | = | input dynamics |
| $h_o(x) : \mathbb{R}^n \to \mathbb{R}^p$ | = | output equation |
| $\epsilon(x) : \mathbb{R}^n \to \mathbb{R}^n$ | = | function reconstruction error for the input dependent activation function |

## Introduction

A plethora of different applications require the tracking of a reference signal by a dynamical system [1–3]. The most common element that is used to successfully drive a system to a certain steady state is the integrator. However, the use of integrators alone as components of the control design is known to lead to instabilities. Integrators can be utilized alongside proportional controllers and differentiators in order to compensate for the instability issues.

A different approach to tracking via integrable action is the use of time-varying integrator gains. The aforementioned method was used in the control problem of power and instruction-throughput in multicore computer processors in

*PhD student, Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA, 30332, USA e-mail:ariskan@gatech.edu.

†Assistant Professor, Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA, 30332, USA e-mail:kyriakos@gatech.edu.

‡Professor, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, 30332, USA e-mail: ywardi@ece.gatech.edu.

[4],[5], [6]. In order to apply the principles of the methodology described above to systems governed by differential equations in [7], a predictor structure, simulating in real-time the future behavior of the system was introduced.

It is known that in real-world applications, complete knowledge of the system dynamics is not available. Extensions to the theory of neural networks for pattern recognition of static input-output maps have introduced adaptive estimation techniques to the problem of identifying a nonlinear dynamical system [8]. This has resulted in the development of learning-based control algorithms, many of which do not require knowledge of the system dynamics. However, in many applications, estimating the parameters, or even the structure, of the plant is useful. As a result, model identification techniques have been introduced jointly with learning-based algorithms.

### Related Work

Different approaches to nonlinear regulation have been introduced over the years. For example, techniques such as the Byrnes-Isidori regulator [9] or high-gain observers [10]. The principles of simulation-based control have also been employed by the research community of Model Predictive Control (MPC) [11]. In MPC, the controller utilizes models of the system dynamics to solve an optimal control problem at each time instant in order to derive the optimal input for a predefined prediction horizon. MPC methods do not take into account modeling errors or unknown terms in the dynamics explicitly, but they have been shown to be robust under certain deviations from the expected behavior.

The use of learning-based control, by means of model-based identifiers was introduced in [12]. Since then, various research angles have been explored. In [13], the authors propose a Q-learning based controller to optimally regulate a linear dynamical system without explicit use of its parameters. In [14], nonlinear regulation is achieved by means of two approximators, a critic and an actor, that solve the optimal control problem online and in a synchronous manner. However, this method requires knowledge of the system model. Two approaches for relaxing the requirement for full knowledge of the system's model have been investigated. In [15], the authors use the method of Integral Reinforcement Learning, which relaxes the need for knowledge of the drift dynamics. In [16], [17] and [18], the authors employ a third approximator that learns the system model online alongside the optimal control approximators.

### Contributions

The contributions of this paper are as follows. A variable integrator controller is employed in order to successfully track an arbitrary target trajectory. To account for unknown and uncertain parameters in the plant, we introduce a learning-based approximator that evaluates the dynamics of the system. The approximate dynamics are fed into the variable integrator controller to conduct a lookahead simulation that determines the control input to the plant. During the plant's operation, the weights of the approximator are updated and provide the simulation-subsystem with an increasingly more accurate model of the plant. The potency of the model-free controller is further highlighted by considering cases where the controller has limited resources and needs to operate in an event-triggered fashion. Numerical simulations are presented to show the efficacy and limitations of the proposed approach.

## Mathematical Preliminaries and Problem Formulation

In this section we present the tracking controller which will be utilized in this work as was developed for systems with known parameters and dynamics in [7, 19].

### Discrete and Memoryless Systems

The concept of variable gain can be better understood in the context of memoryless systems, for which they were initially designed. In this section, we examine the use of variable gain integrators in discrete time systems and continuous memoryless plants.

Given a reference signal $r$, the objective is to have the system output $y$ asymptotically track the reference. The original discrete-time variable gain integrator had the following structure.

$$u_n = u_{n-1} + Q_n e_{n-1}, \tag{1}$$

where $u_k$ is the control input at the discrete time-step $k \in \mathbb{N}$, $e_k = y_k - r$ denotes the tracking error and $Q_n$, the integrator gain. If $Q_n$ is designed constant, then the controller is a traditional adder (the discrete-time version of an

integrator). Although different approaches can be considered, in this paper, the expression of the variable gain is inspired by a Newton-Raphson scheme implemented on the function $y_n = r$. Specifically, the memoryless system is formulated as $y_n = g(u_n)$, and the variable gain is,

$$Q_n = \left(\frac{\partial g}{\partial u}\right)^{-1}(r - g(u)). \tag{2}$$

In continuous time settings, the formulation of variable gain integrators can be easily understood for memoryless input-output maps. Consider the plant described by a function $g : \mathbb{R}^n \to \mathbb{R}^n$, such that $\forall t \geqslant 0$,

$$y(t) = g(u(t)).$$

Assuming that the function $g$ is continuously differentiable, and that the Jacobian matrix $\frac{\partial g}{\partial u}(u)$ is invertible $\forall u \in \mathbb{R}^n$, the continuous-time equivalent to (1),(2) is

$$\dot{u}(t) = \left(\frac{\partial g}{\partial u}(u(t))\right)^{-1}(r(t) - g(u(t))). \tag{3}$$

Consequently, the control input $u(t)$ moves in the direction defined by a Newton-Raphson algorithm with step size scaled by d$t$. It has been shown in [7], that the steady state error between the output and a time-varying reference signal can become arbitrary small.

**Lookahead Simulation**

In this section, we will show how a simulation subsystem can be utilized to extend the feedback controller in (3) in a natural way.

Initially, consider a plant which is affine in the control input, represented by a system of differential equations,

$$\dot{x}(t) = f(x(t)) + k(x(t))u(t), \ , t \geqslant 0, \tag{4}$$

Accordingly, the system output is given by

$$y(t) = h_o(x(t)),$$

The following assumption holds.

**Assumption 1** *The functions $f(\cdot)$, $k(\cdot)$ and $h_o(\cdot)$ are continuously differentiable.* □

In order to extend the control scheme from the memoryless case to the case of plants described by differential equations, we need to determine an appropriate structure for the function $g(u)$ used in (3). Towards that, we will introduce a simulation-based predictor that will evaluate the output of the system during a given time window.

Given a measurement $x(t)$ of the system state at time $t$, a lookahead simulator is utilized, which assumes constant input throughout the simulation period and computes the output. The result of the simulation, i.e., the predicted output is then utilized as the function $g(u)$.

Mathematically, the simulator is given by,

$$\begin{aligned}\dot{\xi}(\tau) &= f(\xi(\tau)) + k(\xi(\tau))u(\tau) \ , \tau \in [t, t+T],\\ \xi(t) &= x(t)\end{aligned}$$

where the input is considered constant, i.e., $u(\tau) = u(t), \ \forall \tau \in [t, t+T]$. The solution of this simulation is denoted as $\phi(\tau; x(t), u(t)) = \xi(\tau)$, which is implicitly a function of the current state measurement of the system (used as the initial condition of the predictor) and the current control input (considered constant in the simulation). Finally, the predicted output at time $t+T$, which will be used in the feedback loop as $g(u)$, is,

$$g(u) = h_o(\phi(t+T; x(t), u(t))).$$

Similar to (3), the desired input is computed based on the controller dynamics,

$$\dot{u}(t) = \left(\frac{\partial g}{\partial u}(u(t))\right)^{-1}(r(t+T) - g(u(t))). \tag{5}$$

We note that the extension of (3) to dynamical systems does not force the error between the current output and the current reference to zero, but rather the future expectation of this error.

In nonlinear system regulation, finding closed-form solutions to the lookahead simulation and the corresponding predicted output is impossible apart from a few simple cases. Therefore, the terms $g(u)$ and $\frac{\partial g(u)}{\partial u}$ in (5) have to be computed numerically.

The designer should take care when she picks the window of the simulation $T$. Since the control input will be chosen in order to drive the simulated future output to the reference, convergence of the actual output can be guaranteed by picking $T$ small. However, this can lead to large numerical errors in the predictor as well as instabilities in the closed-loop system.

## Learning-based approximator

In this section, we will develop an approximation scheme for the dynamics of the system utilized in the predictor. For ease of exposition, dependence of the state and control trajectories on time will be omitted where obvious. Initially, the following assumptions are needed.

**Assumption 2** *For continuous functions defined in a strictly connected compact domain $\mathbb{S}$, $f : \mathbb{S} \to \mathbb{R}^n$ and $k : \mathbb{S} \to \mathbb{R}^{n\times m}$ there exist ideal weights $W_f \in \mathbb{R}^{d\times n}$, $W_k \in \mathbb{R}^{d\times m}$ such that,*

$$f(x) = W_f^T \sigma_f(x) + \epsilon_f(x),$$
$$k(x) = W_k^T \sigma_k(x) + \epsilon_k(x),$$

*where $\sigma_f(x) : \mathbb{R}^n \to \mathbb{R}^{d_f}$, $\sigma_k(x) : \mathbb{R}^n \to \mathbb{R}^{d_g \times m}$ are activation functions, and $\epsilon_f(x) : \mathbb{R}^n \to \mathbb{R}^n$, $\epsilon_k(x) : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are the function reconstruction errors.* □

**Assumption 3** *The ideal weights have known bounds, i.e., $\exists \bar{W}_f, \bar{W}_k \in \mathbb{R}^+$, such that $\|W_f\| \leqslant \bar{W}_f$ and $\|W_k\| \leqslant \bar{W}_k$* □

The dynamics can be also written as

$$\dot{x} = -Ax + W_f^{\mathrm{T}} \sigma_f(x) + Ax + W_k^{\mathrm{T}} \sigma_k(x)u + \epsilon(x),$$

where $A = aI$, with $a > 0$ and $\epsilon(x) = \epsilon_f(x) + \epsilon_k(x)u$, the bounded reconstruction error combining the approximation errors for the drift and the input dynamics. According to [16], we state the following lemma to provide a filtered regressor.

**Lemma 1** *The solution of the nonlinear affine system (4) can be expressed as*

$$x = W^T h(x) + al(x) + \epsilon(x),$$

*where*

$$\dot{h}(x) = -ah(x) + \sigma(x,u) \tag{6}$$

*and*

$$\dot{l}(x) = -Al(x) + x, \tag{7}$$

*where* $\sigma(x,u) = \begin{bmatrix} \sigma_f(x)^T & u^T & \sigma_k(x)^T \end{bmatrix}^T$. □

Accordingly, the estimated dynamics of the system will be of the form

$$\dot{\hat{x}} = \hat{W}^{\mathrm{T}} h(x) + al(x) \tag{8}$$

We shall define the state estimation error as

$$e(t) = \hat{x}(t) - x(t)$$

The proposed learning algorithm to identify the nonlinear dynamics of the system is given as

$$\dot{\hat{W}} = -\Gamma h(x) e^{\mathrm{T}},$$

where $\Gamma > 0$ is a, commonly diagonal, matrix which determines the speed of convergence to the ideal weights.

It is known [20] that in order to ensure convergence of the approximator's weights, sufficient persistence of excitation conditions must be met. It is usual practice to achieve this by adding probing noise during learning. However, it is difficult to guarantee persistence of excitation *a priori*. Following the ideas of experience replay [21], we shall utilize past data collected throughout the trajectory of the system during learning to relax the need for probing and improve convergence of the weights. Consequently, the update law for the approximator weights $\hat{W}(t)$, will be,

$$\dot{\hat{W}} = -\Gamma h(x) e^{\mathrm{T}} - \Gamma \sum_{i=1}^{n_{\exp}} h(x_i) e_i^{\mathrm{T}}, \tag{9}$$

where $x_i = x(t_i)$ and $e_i = \hat{x}(t_i) - x(t_i)$ are derived based on trajectory data collected at times $t_1 < t_2 \cdots < t_{n_{\exp}}$.

**Theorem 1** *Consider the system* (4)*, equipped with the regressor dynamics* (6),(7)*. If the vector of collected past data* $[x_1\ x_2\ \ldots\ x_{n_{exp}}]$ *contains d linearly independent elements, then for bounded approximation error, the weight error* $\tilde{W} = \hat{W} - W$ *is uniformly ultimately bounded.*

*Proof.* The proof follows from [16]. ■

The pseudocode for the approximate lookahead tracking for unknown dynamics is provided in Algorithm 1 and the system's structure in Fig.1.

**Algorithm 1:** Simulation-based lookahead target tracking with learning-based approximator

**procedure**
  Give initial state $x(t_0)$, lookahead horizon $T$, initial approximator weights $\hat{W}(0)$ and set $l(0) = 0$, $h(0) = 0$.
  Utilize current state $x(t)$ as initial conditions for the lookahead simulation.
  Simulate the system utilizing (8) with constant control input $u(t)$.
  Derive approximate values of $g(u)$, $\frac{\partial g}{\partial u}$.
  Calculate control input utilizing (5).
  Propagate the plant utilizing (4), and update the approximator weights through (9), and the regressors through (6) and (7).
  Go to 3.
**end procedure**

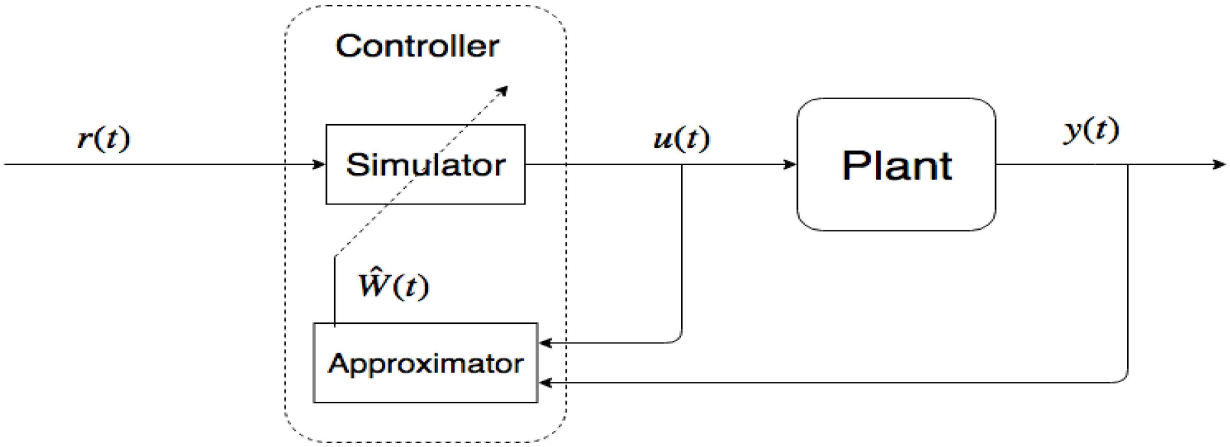



**Fig. 1 Learning-based control system.**

## Event-triggered control

Due to the ability of the variable integrator control law to operate on a general class of linear and nonlinear systems, we will be able to examine its use alongside different learning-control frameworks. We investigate the use of an

event-triggered structure to update the controller only when the plant's trajectory has deviated considerably, inspired by [22, 23]. More formally, the dynamics of the controller are,

$$\begin{aligned} \dot{u}(t) &= 0 \\ u(t^+) &= \left(\frac{\partial g}{\partial u}(u(t))\right)^{-1}(r(t+T) - g(u(t))), \text{ when } \mathcal{A}\big(e_x(t)\big) \geqslant \kappa\mathcal{B}\big(x(t)\big), \end{aligned} \tag{10}$$

where $\mathcal{A}(\cdot), \mathcal{B}(\cdot)$ are functions of class $\mathcal{K}_\infty$, and $\kappa \in \mathbb{R}^+$. The state error is $e_x(t) = x(t_j) - x(t)$, where $t_j$, $j \in \{0, 1, \dots\}$ is the time instant of the last control update. Through this design we can alleviate the computational effort that the simulator introduces to the closed-loop system as well as take into account the, potentially limited, resources of the system. We note that the design process of the learning-based approximator and the event-triggered controller are decoupled. Thus, the approximator will converge as long as the conditions of Theorem 1 are satisfied. The pseudocode that implements this procedure is provided in Algorithm 2.

**Algorithm 2:** Event-triggered learning-based lookahead simulation target tracking

**procedure**
  Give initial state $x(t_0)$, lookahead horizon $T$, initial approximator weights $\hat{W}(0)$ and set $l(0) = 0$, $h(0) = 0$, $u(0) = 0$.
  **for** $j = \{0, 1, 2, \dots\}$
    Sample current state $x(t)$ and compute $e_x(t) = x(t_j) - x(t)$
    **if** $\mathcal{A}\big(e_x(t)\big) \geqslant \kappa\mathcal{B}\big(x(t)\big)$
      Utilize current state $x(t)$ as initial conditions for the lookahead simulation.
      Simulate the system utilizing (8) with constant control input $u(t)$.
      Derive approximate values of $g(u)$, $\frac{\partial g}{\partial u}$.
      Update control input utilizing (10).
      Set $j = j + 1$
    **else**
      $\dot{u}(t) = 0$
    **end if**
    Propagate the plant utilizing (4), and update the approximator weights through (9), and the regressors through (6) and (7).
    Go to 4.
  **end for**
**end procedure**

## Simulation examples

Simulation results focused on the more complex case of nonlinear dynamical systems. Specifically, we designed a controller for the two-dimensional system of an inverted pendulum,

$$\begin{aligned} \dot{x}_1(t) &= x_2(t) \\ \dot{x}_2(t) &= a_p \sin(x_1(t)) - b_p x_2(t) + u(t) \end{aligned}$$

utilizing a predictive simulation with lookahead horizon $T = 0.1$s and arbitrary positive parameters $a_p$, $b_p$ which are considered unknown. The internal dynamics of the control utilize the current approximation of the system. We use a basis of $N = 6$ elements for each dimension of the system. Figure 2(a) shows the trajectory of the actual plant, as well as the reference trajectory. In Fig.2(b) emphasis is given in the time window where the controller's lookahead simulation operates without accurate approximation of the plant dynamics, i.e., before the approximator weights converge to the ideal ones. However, it should be noted that since we can guarantee convergence of the approximator weights to a compact set, and not at their ideal values, the steady state tracking error will be sensitive to the approximator's tuning parameters. Figure 3 shows the derived control input that drives the system to the desired trajectory. Finally, Fig.4 shows the evolution of the approximator weights.

The results for the event-triggered case can be seen in Fig.5, where the target and the plant's trajectories are shown, in Fig.6, where the control input is presented and in Fig.7, where we show the convergence of the approximator weights. Throughout the design process, it was observed that the event-triggered model-free controller was lead to instabilities for larger values of the lookahead horizon $T$. Thus, in this simulation the horizon was set to $T = 0.5$s. Finally, in Fig.8(a), the difference between the time-triggered and the event-triggered model-free controllers is shown while in Fig.9(b) we provide the evolution of the triggering condition and the corresponding triggering threshold. The more discrete nature of the event-triggered controller can be observed.

## Conclusion

We utilize a lookahead algorithm to track a target reference trajectory. Considering unknown or uncertain plant dynamics, we employ a learning-based approximator. At each time step, the controller is dynamically updated based on simulation conducted on the current approximation of the system. As the actual plant evolves with the derived control input, we simultaneously train the approximator to provide a more accurate prediction of the system's evolution. In order to increase the speed of convergence and relax the need for persistence of excitation during learning, we implement an experience replay-based tuning algorithm that leverages previous stored data concurrently with current data. Furthermore, we propose an event-triggered framework based on which, the control input is updated in an intermittent fashion, when the trajectory of the system has deviated from the desired one.

Future work will focus on the case when both the approximator and the controller operate with limited resources, leading to both event-triggered control and intermittent learning. Also, the proposed model-free controller will be extended in the case of multi-agent systems and target pursuit problems.

## Acknowledgments

This work was supported in part by NATO under grant No. SPS G5176, by ONR Minerva under grant No. N00014 − 18 − 1 − 2160, by an NSF CAREER CPS-1851588.

## References

[1] Martin, P., Devasia, S., and Paden, B., "A different look at output tracking: control of a VTOL aircraft," *Automatica*, Vol. 32, No. 1, 1996, pp. 101–107.

[2] Devasia, S., Chen, D., and Paden, B., "Nonlinear inversion-based output tracking," *IEEE Transactions on Automatic Control*, Vol. 41, No. 7, 1996, pp. 930–942.

[3] Limón, D., Alvarado, I., Alamo, T., and Camacho, E. F., "MPC for tracking piecewise constant references for constrained linear systems," *Automatica*, Vol. 44, No. 9, 2008, pp. 2382–2387.

[4] Almoosa, N., Song, W., Wardi, Y., and Yalamanchili, S., "A power capping controller for multicore processors," *American Control Conference (ACC), 2012*, IEEE, 2012, pp. 4709–4714.

[5] Chen, X., Xiao, H., Wardi, Y., and Yalamanchili, S., "Throughput regulation in shared memory multicore processors," *High Performance Computing (HiPC), 2015 IEEE 22nd International Conference on*, IEEE, 2015, pp. 12–20.

[6] Chen, X., Wardi, Y., and Yalamanchili, S., "IPA in the loop: control design for throughput regulation in computer processors," *Discrete Event Systems (WODES), 2016 13th International Workshop on*, IEEE, 2016, pp. 141–146.

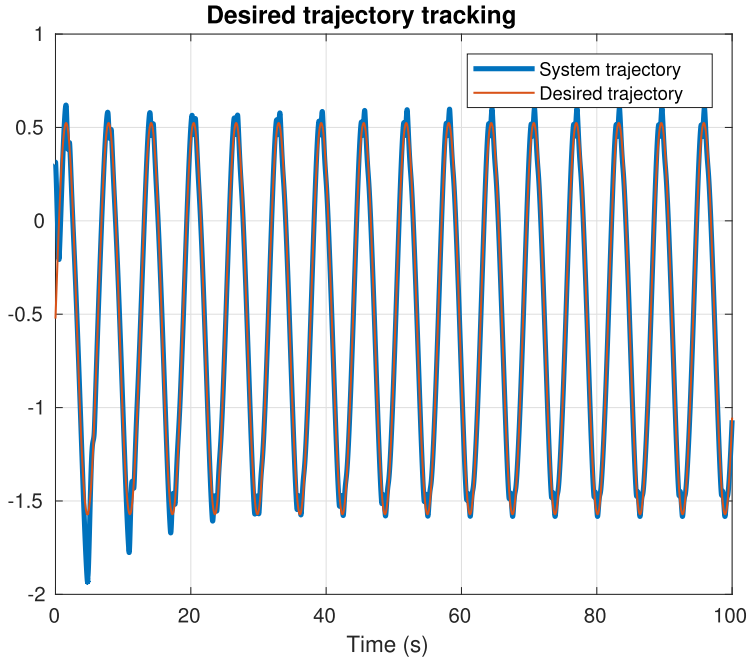
[7] Wardi, Y., Seatzu, C., Egerstedt, M., and Buckley, I., "Performance Regulation and Tracking via Lookahead Simulation: Preliminary Results and Validation," *Decision and Control, 2017., Proceedings of the 56th IEEE Conference on*, 2017.

[8] Kosmatopoulos, E. B., Polycarpou, M. M., Christodoulou, M. A., and Ioannou, P. A., "High-order neural network structures for identification of dynamical systems," *IEEE transactions on Neural Networks*, Vol. 6, No. 2, 1995, pp. 422–431.

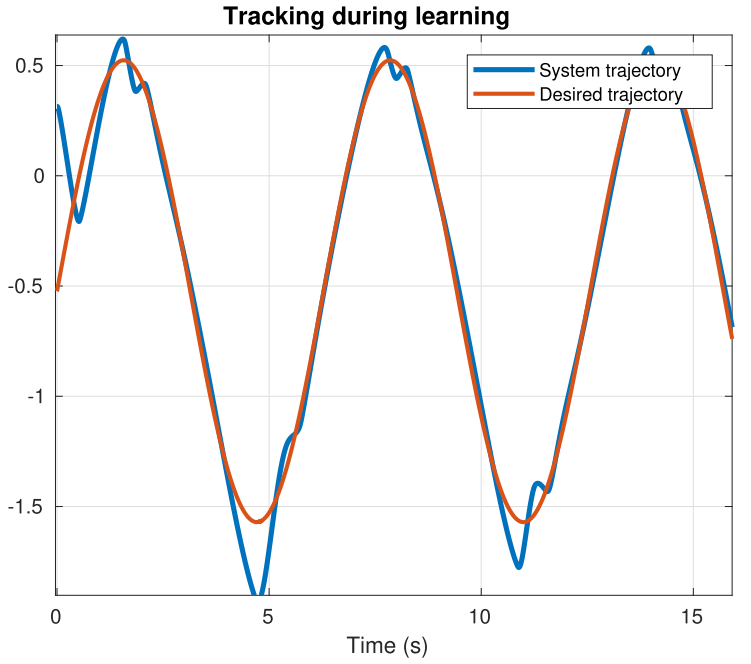
[9] Isidori, A., and Byrnes, C. I., "Output regulation of nonlinear systems," *IEEE transactions on Automatic Control*, Vol. 35, No. 2, 1990, pp. 131–140.

[10] Khalil, H. K., "On the design of robust servomechanisms for minimum phase nonlinear systems," *Decision and Control, 1998. Proceedings of the 37th IEEE Conference on*, Vol. 3, IEEE, 1998, pp. 3075–3080.

[11] Camacho, E. F., and Alba, C. B., *Model predictive control*, Springer Science & Business Media, 2013.




**(a) Tracking of the reference output for an inverted pendulum without complete knowledge of the system dynamics via lookahead simulation.**




**(b) Detail of the tracking evolution. Since the lookahead simulator operates with a learning approximator of the system, initially the tracking is less precise.**

**Fig. 2 Tracking of the target trajectory**

[12] Werbos, P. J., "Neural networks for control and system identification," *Decision and Control, 1989., Proceedings of the 28th IEEE Conference on*, IEEE, 1989, pp. 260–265.

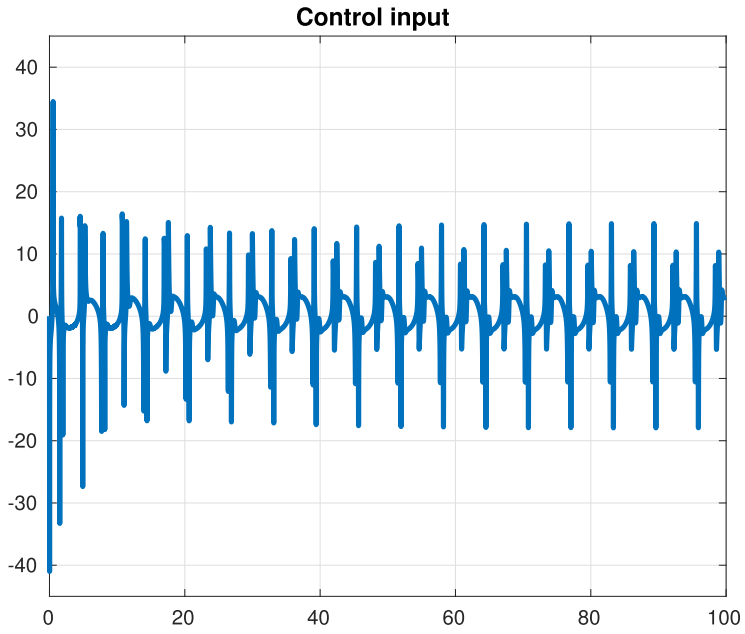



**Fig. 3 Control input to achieve tracking of the target trajectory.**

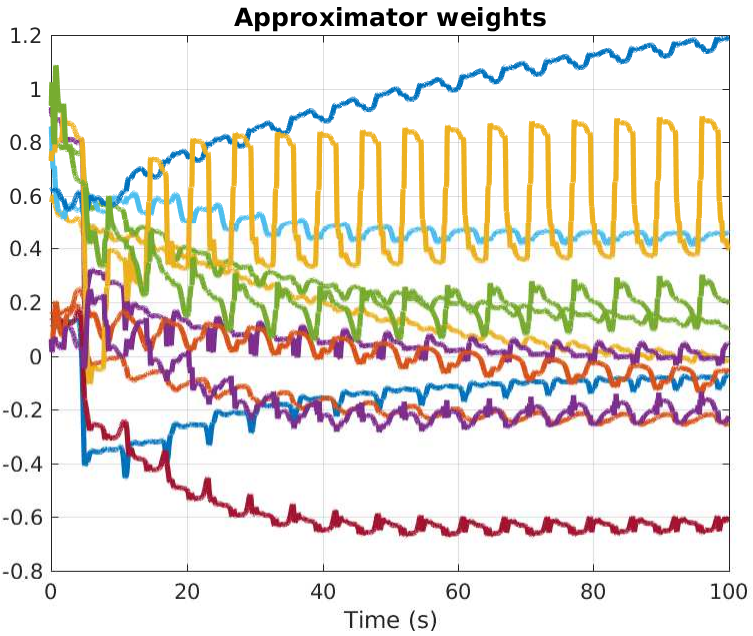



**Fig. 4 Evolution of the approximator weights as a function of time. Whereas the approximation of the dynamics is satisfactory, the weights only converge to a set around the ideal values.**

[13] Vamvoudakis, K. G., "Q-learning for continuous-time linear systems: A model-free infinite horizon optimal control approach," *Systems & Control Letters*, Vol. 100, 2017, pp. 14–20.

[14] Vamvoudakis, K. G., and Lewis, F. L., "Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem," *Automatica*, Vol. 46, No. 5, 2010, pp. 878–888.

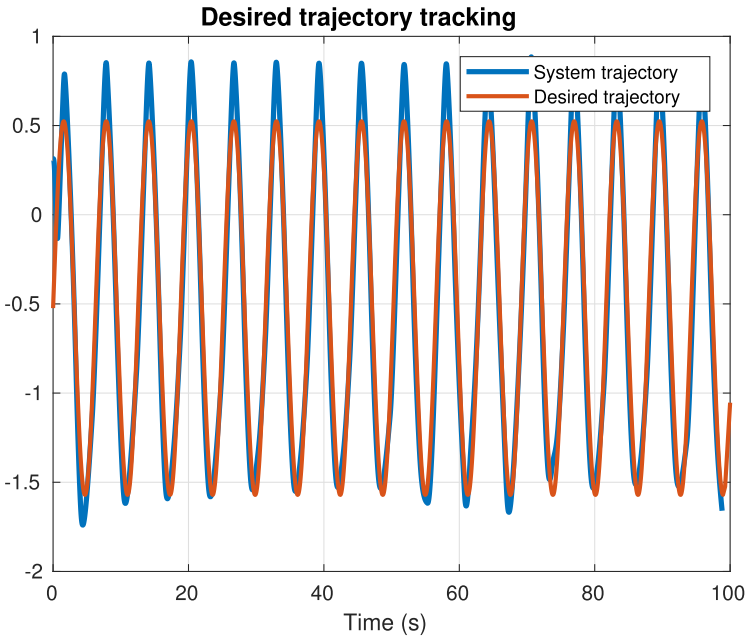



**Fig. 5 Tracking of the reference output by an event-triggered model-free simulation-based controller. To avoid instabilities, the lookahead horizon was increased. Coupled with the steady state learning errors, the system had large deviations from the reference.**

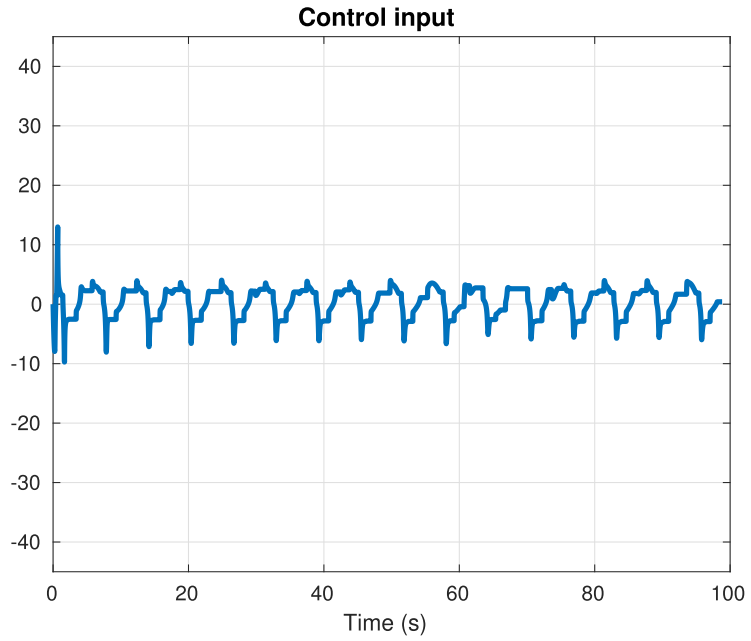



**Fig. 6 Control input in the case of the event-triggered controller.**

[15] Modares, H., Lewis, F. L., and Naghibi-Sistani, M.-B., "Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems," *Automatica*, Vol. 50, No. 1, 2014, pp. 193–202.

[16] Modares, H., Lewis, F. L., and Naghibi-Sistani, M.-B., "Adaptive optimal control of unknown constrained-input systems using

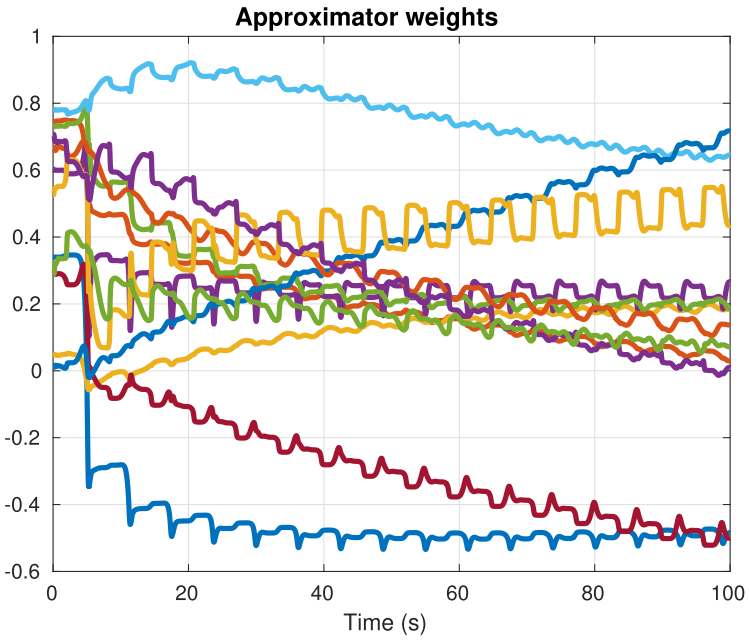



**Fig. 7 Evolution of the approximator weights as a function of time in the event-triggered controller. In this example, even when the controller is not updated, the approximator still learns.**



policy iteration and neural networks," *IEEE transactions on neural networks and learning systems*, Vol. 24, No. 10, 2013, pp. 1513–1525.

[17] Bhasin, S., Kamalapurkar, R., Johnson, M., Vamvoudakis, K. G., Lewis, F. L., and Dixon, W. E., "A novel actor–critic–identifier architecture for approximate optimal control of uncertain nonlinear systems," *Automatica*, Vol. 49, No. 1, 2013, pp. 82–92.

[18] Kamalapurkar, R., Walters, P., and Dixon, W. E., "Model-based reinforcement learning for approximate optimal regulation," *Automatica*, Vol. 64, 2016, pp. 94–104.

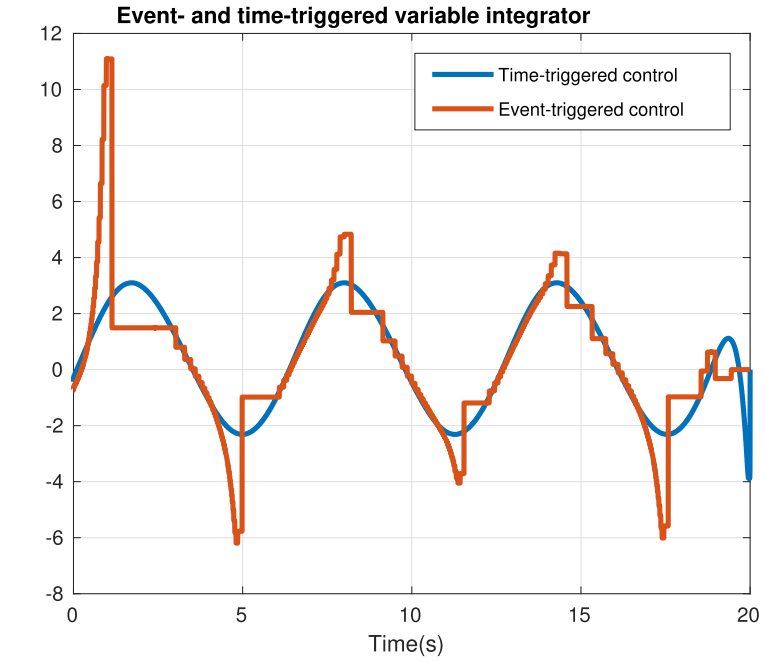
[19] Wardi, Y., Seatzu, C., and Egerstedt, M., "Tracking Control via Variable-gain Integrator and Lookahead Simulation: Application to Leader-follower Multiagent Networks," *Analysis and Design of Hybrid Systems 2018, Conference on*, 2018, to appear.

[20] Ioannou, P. A., and Sun, J., *Robust adaptive control*, Vol. 1, PTR Prentice-Hall Upper Saddle River, NJ, 1996.

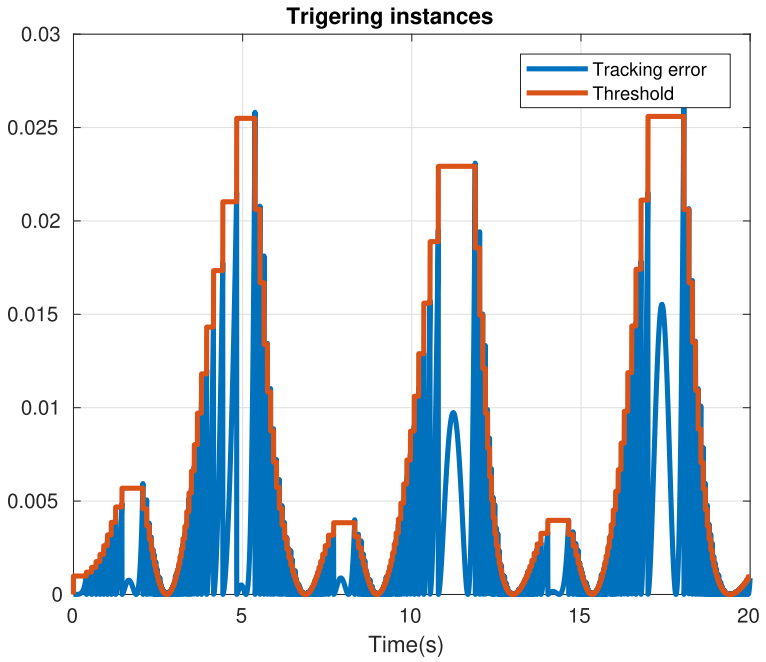
[21] Adam, S., Busoniu, L., and Babuska, R., "Experience replay for real-time reinforcement learning control," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 42, No. 2, 2012, pp. 201–212.

[22] Tallapragada, P., and Chopra, N., "On event triggered tracking for nonlinear systems," *IEEE Transactions on Automatic Control*, Vol. 58, No. 9, 2013, pp. 2343–2348.

[23] Tabuada, P., "Event-triggered real-time scheduling of stabilizing control tasks," *IEEE Transactions on Automatic Control*, Vol. 52, No. 9, 2007, pp. 1680–1685.




**(a) Comparison between the time-triggered and the event-triggered controllers.**




**(b) Evolution of the triggering condition and the threshold.**

**Fig. 8 Evolution of the event-triggered controller**