PROCESS DESIGN AND CONTROL

Improving Set-Point-Following Performance of Industrial Controllers with a Fast Dynamic Inversion Algorithm

Antonio Visioli*,† and Aurelio Piazzi‡

Dipartimento di Elettronica per l'Automazione, University of Brescia, Via Branze 38, 25123 Brescia, Italy, and Dipartimento di Ingegneria dell'Informazione, University of Parma, Parco Area delle Scienze 181A, I-43100 Parma, Italy

In this paper, we propose a new method for the improvement of set-point-following performances of industrial controllers. Basically, this new approach consists of substituting the standard step signal to be applied to the closed-loop system (in which the controller has been previously selected) with a command input function that is determined by means of a dynamic inversion procedure. This procedure is based on a fast algorithm, and it is therefore suitable for application in an industrial context, where the ease of implementation is of major concern. Simulation results show the effectiveness of the technique.

1. Introduction

It is well-known that the ease of implementation is a major issue for an industrial controller, as it is important to achieve the best cost/benefit ratio. In fact, it is for this reason that proportional−integral−derivative (PID) controllers are still largely the most used controllers in industry, despite the fact that many effective control methodologies have been proposed in the past 50 years.

Thus, for a new methodology to be suitable for actual implementation in an industrial environment, it has to satisfy different requirements, in addition to achieveing high levels of performance. Basically, the extra tuning effort required from the operator should be kept as low as possible, the design procedure should have a clear physical meaning, and the computational effort should be minimized to allow for the low-cost implementation of the controller.

In this context, we propose a new technique for the improvement of the set-point-following performance of industrial controllers, which consists basically of using a determined command input function instead of a standard step signal.[1] In particular, we consider a single-loop control scheme in which the controller (typically of PID type, but no assumption has to be made regarding its structure) has already been fixed. A step signal is then applied to the set point, and the closed-loop system model is identified. Note that the use of a step-response-based model is widely adopted in the model predictive control framework, particularly in the commercially available dynamic matrix control algorithm.[2] At this point, a desired system output function has to be selected when a transition from one set-point value to another must be performed by the system. In this context, simple polynomial functions can be employed, as they guarantee a monotonic transition without overshoot and, further, they can be easily parametrized by the desired transition time.[3] Finally, a new command function, which substitutes for the standard step signal, is calculated by inverting the identified model, to generate the desired function at the system output. This determined input function (possibly scaled) is then used as any new output transition is required from the system.

The paper is organized as follows. In section 2, the methodology is presented in detail. In section 3, simulation results are given. Conclusions are drawn in the last section.

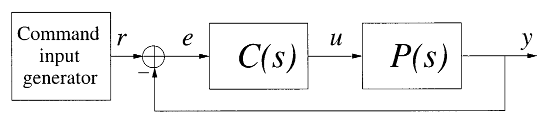

Figure 1. General control scheme.

2. Methodology

2.1. Generalities. We consider the general single-loop control scheme, shown in Figure 1, where $P(s)$ is the process and $C(s)$ is the controller. It is assumed that the closed-loop system is asymptotically stable. The aim of the devised method is to find the command function $r(t)$ that produces a desired system output transition from y_0 to y_1, starting from time $t = 0$, without assuming a priori knowledge on the process model. Despite the fact that the process and the controller are defined in the continuous-time domain, in the following, we will work with sampled data, as the use of microprocessors is the common practice in industrial environments. We assume that the sampling time T has been chosen by any standard technique.[4]

2.2. Step Response Model. An identification experiment can be easily performed by applying a step signal to the input of the closed-loop system. A system model

* Corresponding author. Phone: +39-030-3715460. Fax: +39-030-380014. E-mail: visioli@ing.unibs.it.
† University of Brescia.
‡ University of Parma. E-mail: aurelio@ce.unipr.it.

10.1021/ie020734f CCC: $25.00
Published on Web 03/11/2003

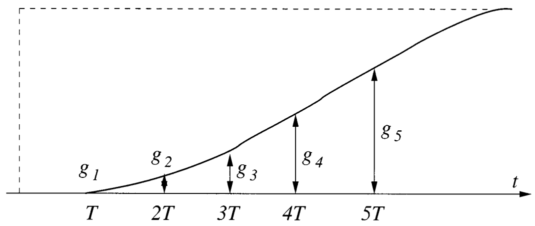

Figure 2. Step response model.

can then be obtained by considering the truncated response ($t \in \{T, 2T, ..., NT\}$)

$$y(t) = y_0 + g_{t/T}r(0) + \sum_{i=1}^{t/T-1} g_i[r(t-iT) - r(t-(i+1)T)] \quad (1)$$

where $g_i \equiv g(iT)$, $i = 1, \cdots, N$, are the sampled output values in response to a unit-step input (see Figure 2) and $r(t)$ is the system input. In the following discussion, the value of y_0 will be taken to be 0 without loss of generality. The number N of parameters has to be taken high enoughto allow a sufficiently accurate description of the system, but not so high that the computational effort of the control strategy is unmanageable. From a practical point of view, the sampling of the step response to obtain parameters g_i should stop when the process output remains close to its steady-state value for a sufficiently long time.

For the presented methodology, it is convenient to write expression 1 in matrix form

$$\mathbf{Y} = \mathbf{GR} \quad (2)$$

where

$$\mathbf{Y} = \begin{bmatrix} y(T) \\ y(2T) \\ y(3T) \\ \vdots \\ y(NT) \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} g_1 & 0 & 0 & \cdots & 0 \\ -g_1 + g_2 & g_1 & 0 & \cdots & 0 \\ -g_2 + g_3 & -g_1 + g_2 & g_1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 \\ -g_{N-1} + g_N & -g_{N-2} + g_{N-1} & -g_{N-3} + g_{N-2} & \cdots & g_1 \end{bmatrix}$$

and

$$\mathbf{R} = \begin{bmatrix} r(0) \\ r(T) \\ r(2T) \\ \vdots \\ r((N-1)T) \end{bmatrix}$$

Remark 1. It should be noted that, in many cases, it might not be necessary to perform an ad hoc identification experiment (i.e., to stop the normal process operations) to apply the devised methodology. In fact, as the model is obtained by evaluating a standard closed-loop step response, data taken from an output transition performed during normal process operations can be used. Obviously, it is important that the collected data be representative of a true step response (and therefore operations such as detrending might be necessary[5]) and, if an unmeasured load disturbance occurs during the transient response, then the data should not be used. In this context, it can be useful to adopt the method proposed in ref 6 to detect load disturbances.

2.3. Output Function Selection. To apply the dynamic inversion procedure, a desired output function has to be chosen to describe a transition from one set-point value to another. Although, in principle, any function can be applied, an effective choice is to adopt a "transition polynomial"[3] that guarantees a smooth and monotonic (i.e., without overshoot) transition between $y_0 = 0$ and y_1 and can be easily parametrized by the transition time τ that can eventually be selected to fully exploit the actuator capability (see remark 4). Formally, we define

$$y_d(t) = c_{2p+1}t^{2p+1} + c_{2p}t^{2p} + \cdots + c_1 t + c_0$$

The polynomial coefficients can be uniquely found by solving the following linear system, in which boundary conditions at the endpoints of interval $[0,\tau]$ are imposed

$$\begin{cases} y_d(0) = 0, & y_d(\tau) = y_1 \\ y_d^{(1)}(0) = 0, & y_d^{(1)}(\tau) = 0 \\ \vdots & \vdots \\ y_d^{(p)}(0) = 0, & y_d^{(p)}(\tau) = 0 \end{cases}$$

Thus, the desired output function can be expressed in closed form as

$$y_d(t;\tau) = \begin{cases} y_1 \dfrac{(2p+1)!}{p!\tau^{2p+1}} \displaystyle\sum_{i=0}^{p} \dfrac{(-1)^{p-i}}{i!(p-i)!(2p-i+1)} \tau^i t^{2p-i+1} & \text{if } 0 \le t \le \tau \\ y_1 & \text{if } t > \tau \end{cases} \quad (3)$$

If the algorithm described in subsection 2.4 is employed, then the order of the transition polynomial can be chosen arbitrarily. (Note that this does not apply when the dynamic inversion is performed analytically, based on the continuous-time transfer function of the system. In that case, the order of the polynomial has to be chosen depending on the relative degree of the system to guarantee the continuity of the command function.[3]) Indeed, the order of the polynomial can be chosen to handle the tradeoff between the need to decrease the rise time and the need to decrease the control effort, taking into account that the rise time decreases and the control effort increases as the order of the polynomial increases. In general, a good choice in this context is to select $p = 2$, i.e., to use the desired output function

$$y_d(t;\tau) = \begin{cases} y_1\left(\dfrac{6}{\tau^5}t^5 - \dfrac{15}{\tau^4}t^4 + \dfrac{10}{\tau^3}t^3\right) & \text{if } 0 \le t \le \tau \\ y_1 & \text{if } t > \tau \end{cases} \quad (4)$$

Expression 4 is employed in all of the simulation examples presented in section 3.

2. 4. Dynamic Inversion. Once the desired output function has been selected, i.e., the array $\mathbf{Y}_d$ has been constructed, then the corresponding closed-loop system input $r(t)$ that causes $y_d(t;\tau)$ can easily be determined by simply inverting the system using eq 2. For matrix

$\mathbf{G}$ to be invertible by a standard numeric algorithm, it should be well-conditioned; for example, there must not be a row (or a column) where all of the elements are very small with respect to the elements of the other rows (or columns). This happens when the process has a true dead time or an apparent dead time (i.e., when the process is of high order), which causes some of the first sampled output values g_i of the step response to be null or almost null. Thus, we denote by k the number of first rows of $\mathbf{G}$ in which all of the elements are less than a selected threshold ϵ. Then, we obtain matrix $\hat{\mathbf{G}}$ by removing the first k rows and the last k columns from $\mathbf{G}$. Subsequently, by evaluating $y_d(t;\tau)$ at the first $N - k$ sampling time intervals, the array $\mathbf{Y}_d = [y_d(T;\tau)\ y_d(2T;\tau) \cdots y_d((N - k)T;\tau)]^T$ can easily be constructed. The first $N - k$ values of the command reference input are then determined by applying the expression

$$\hat{\mathbf{R}} = [r(T)\ r(2T) \cdots r((N-k)T)]^T = \hat{\mathbf{G}}^{-1}\mathbf{Y}_d \qquad (5)$$

In this way, the input function can be calculated simply by determining the inverse of a matrix, which can be performed by using different algorithms (see, e.g., ref 7).

Note that, if the sampling time T and the value of N have been selected appropriately, as well as the value of τ, then the last element of the array $\hat{\mathbf{R}}$ actually corresponds to the steady-state value of the input, and therefore, the value of $r((N-k)T)$ can be applied to the closed-loop system for $t > (N - k)T$.

Remark 2. Because the first k rows and the last k columns have been removed from matrix $\mathbf{G}$, the obtained output function is delayed by kT with respect to the desired output function. Actually, the dead time is removed in the model of the closed-loop system transfer function employed in the dynamic inversion.

Remark 3. Because the proposed methodology is capable of ensuring good set-point-following performance despite the tuning procedure adopted for the controller, it is convenient to design the controller to guarantee good load disturbance performance, even if this implies that the predicted closed-loop step response is unsatisfactory. In other words, in the case that the closed-loop system step response is too oscillatory, the replacement of the standard step signal by the determined command input function allows for a high-performance output transition to be obtained.

Remark 4. The choice of the value of the transition time τ can be left to the user, who can select it with the aim of minimizing the rise time subject to the actuator constraints.[3] Alternatively, to avoid increasing the tuning effort required of the operator, τ can be automatically fixed by applying some heuristics. For example, τ can be chosen to be equal to the rise time of the closed-loop step response obtained.

Remark 5. In typical process control applications, it is common practice to tune the controller to guarantee good load rejection performance, which is of major concern, and then to avoid excessive overshoots in the step response by filtering the set point.[8] With respect to this technique, the methodology presented in this paper has the great advantage of avoiding an increase in the rise time. In fact, in contrast to set-point filtering, the idea of the proposed method is to obtain the desired output transition independently of the closed-loop dynamics.

Remark 6. In the presence of measurement noise, as is always the case in practical applications, the proposed method can be successfully applied provided that the step response function employed for the identification of the closed-loop system model (eq 1) is appropriately filtered. Note that, for the proposed method, the required filtering can be performed off-line, and therefore, a zero-phase noncausal filter can be used to avoid phase distortion. Further, the presence of the noise has to be considered when matrix $\hat{\mathbf{G}}$ is constructed from matrix $\mathbf{G}$. Actually, because of the noise measurements, it is sensible to redefine parameter ϵ as a noise band NB, i.e., a threshold value that determines, as before, whether the sampled value g_i has to be discarded. Specifically, if $|g_i| <$ NB, then g_i is considered to be 0 in the construction of matrix $\hat{\mathbf{G}}$. The value of NB can easily be selected by monitoring that process output for a sufficiently long time when the process is at steady state. Note that the concept of a noise band has already been applied successfully in the implementation of autotuning procedures in industrial regulators.[9]

3. Simulation Results

Some simulation examples are presented to illustrate the proposed methodology and to demonstrate its effectiveness. For the sake of clarity, measurement noise will be taken into account only in the last example. For all of the presented examples, the controller has a PID structure described by the following expression

$$U(s) = K_p\left[E(s) + \frac{1}{T_i s}E(s) - T_d s Y(s)\right] \qquad (6)$$

where $U(s)$, $E(s)$, and $Y(s)$ are the Laplace transforms of the control variable, system error, and process output, respectively. The required output transition is from $y_0 = 0$ to $y_1 = 1$.

3.1. FOPDT Process. As a first example, we consider the first-order plus dead-time (FOPDT) process

$$P_1(s) = \frac{1}{10s + 1}e^{-5s} \qquad (7)$$

with the following values of the PID parameters: $K_p = 2.61$, $T_i = 10.05$, and $T_d = 2.51$. The sampling time is fixed at 0.5 s. To obtain the closed-loop system model, $N = 161$ samples of the step response are evaluated, and matrix $\mathbf{G}$ is constructed accordingly. By fixing $\epsilon = 0.01$, one obtains $k = 11$. Thus, the first 11 rows and the last 11 columns of $\mathbf{G}$ are removed, yielding matrix $\hat{\mathbf{G}}$ of dimensions 150×150. The desired output array is constructed by selecting $\tau = 5$ s. The resulting command function $r(t)$, obtained by applying the dynamic inversion, is reported in Figure 3. A comparison of the process outputs obtained by using $r(t)$ and the standard step signal is shown in Figure 4, and the corresponding control variables are plotted in Figure 5. It is evident that the use of the calculated command function provides a great improvement in the set-point-following performance. In any case, to better compare the two approaches, the resulting overshoot O (%), rise time T_r (defined as the interval time in which the process output passes from 10 to 90% of its steady-state output), settling time T_s (defined as the as the minimum time after that the regulated output remains within a 2% range of y_1), and integrated absolute error

$$\mathrm{IAE} = \int_0^{T_s} |e(t)|\ \mathrm{d}t$$

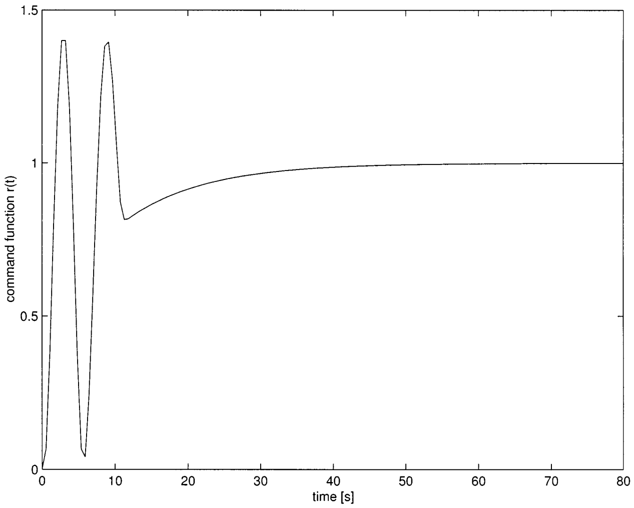

Figure 3. Command function for the example with $P_1(s)$.

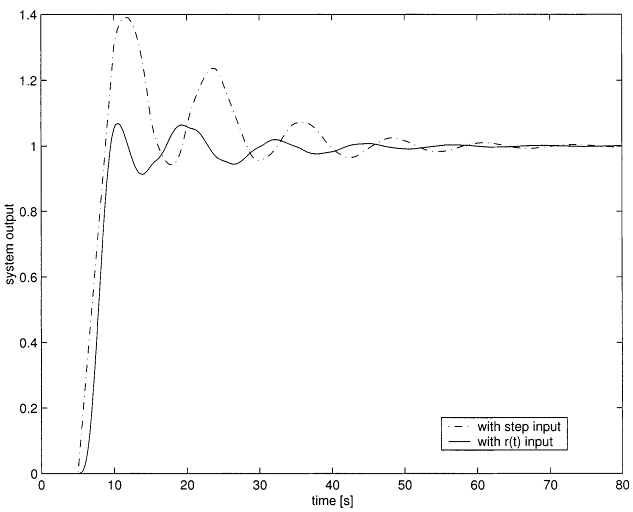

Figure 4. Process outputs for the example with $P_1(s)$.

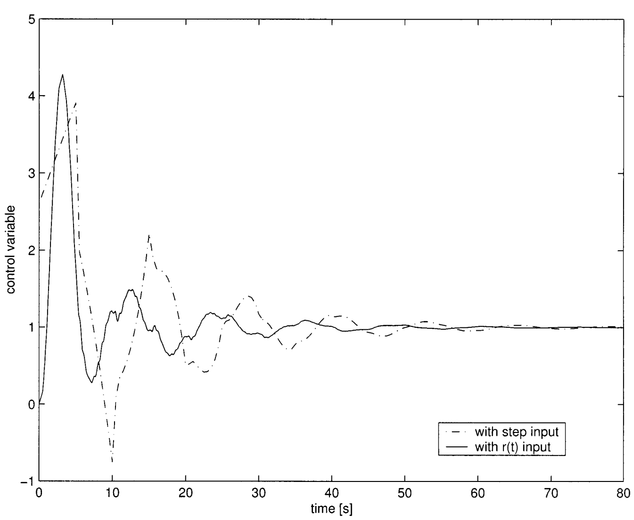

Figure 5. Control variables for the example with $P_1(s)$.

were calculated and are reported in Table 1. It appears that the overshoot is significantly reduced without a increase in the rise time, so that the settling time is greatly reduced as well.

Table 1. Summary of the Results in the Example with Process $P_1(s)$

	O (%)	T_r (s)	T_s (s)	IAE
step input	39	3.07	49.6	10.82
dynamic-inversion-based input	7	2.54	28.8	8.86

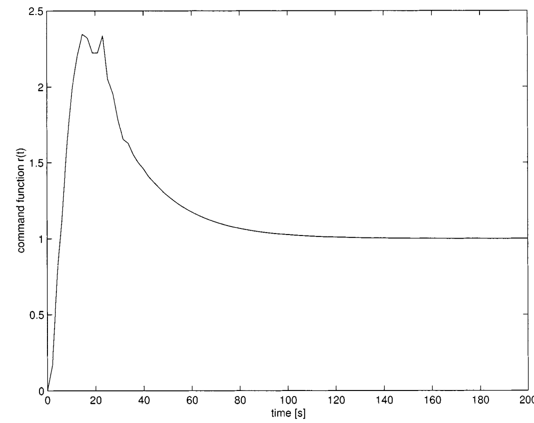

Figure 6. Command function for the example with $P_2(s)$.

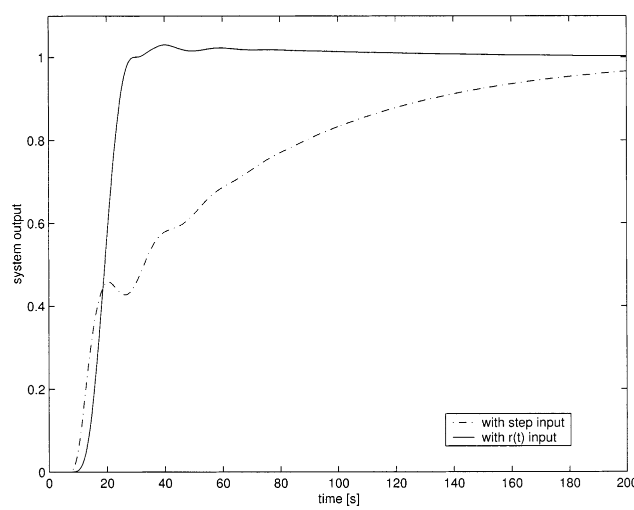

Figure 7. Process outputs for the example with $P_2(s)$.

High-Order Process. As a second example, we consider the high-order process

$$P_2(s) = \frac{1}{(s+1)^8} e^{-5s} \tag{8}$$

with the following values of the PID parameters: $K_p = 0.37$, $T_i = 19.93$, and $T_d = 4.98$. In this case, the sampling time is fixed at 2 s, and a number $N = 101$ of output samples is used to model the closed-loop system. By again fixing $\epsilon = 0.01$, we obtain $k = 5$, and therefore, $\hat{\mathbf{G}}$ is of dimensions 96 × 96. With the selection of $\tau = 20$ s, the command input function $r(t)$ plotted in Figure 6 was determined. A comparison between the process outputs obtained with the standard and the new approach is shown in Figure 7, and the corresponding control variables are plotted in Figure 8. As in the previous example, the dynamic-inversion-based systems outperforms the one with the step signal. The results are summarized in Table 2.

3.3. Process with Lag-Dominant Dynamics. Processes with lag-dominant dynamics, which are frequently encountered in industry, especially in fluid processes, are likely to present a large overshoot in the

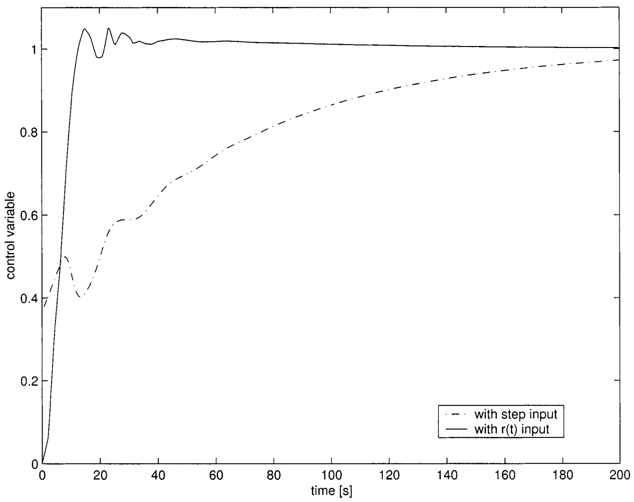

Figure 8. Control variables for the example with $P_2(s)$.

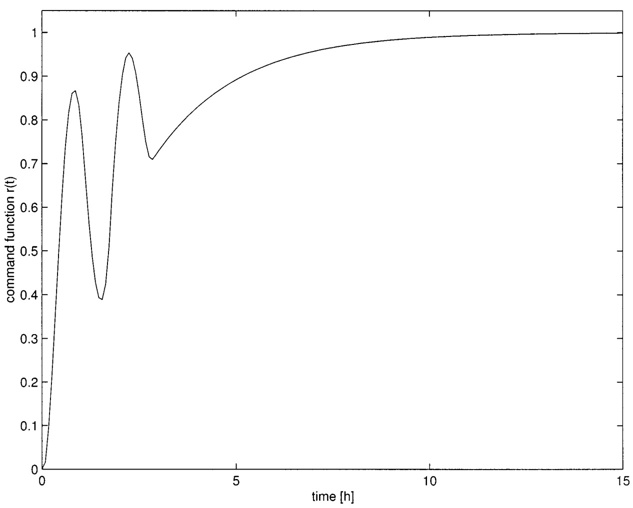

Figure 9. Command function for the example with $P_3(s)$.

Table 2. Summary of the Results in the Example with Process $P_2(s)$

	O (%)	T_r (s)	T_s (s)	IAE
step input	0	121.1	232.8	53.34
dynamic-inversion-based input	3	10.64	45.5	20.22

set-point response when the controller is tuned for good performance in load rejection. Thus, as a third example, we consider a distillation column in which the top composition (or temperature) is measured. The response to a change in reflux flow can be modeled as an FOPTD process in which the time constant is[10]

$$\theta = \frac{n^2 + n}{2}\theta'$$

where n is the number of trays and θ' is the time constant of a single part of the process, and the dead time can be expressed by $L = 0.14\theta$. If $n = 100$ and $\theta' = 5$ s, one obtains $\theta = 7$ h and $L = 0.98$ h. Thus, the following transfer function is considered

$$P_3(s) = \frac{1}{7s + 1}e^{-0.98s} \tag{9}$$

The following values of the PID parameters were selected: $K_p = 8.26$, $T_i = 2.01$, and $T_d = 0.5$. In this case the sampling time is fixed to 0.08 h, and $N = 126$

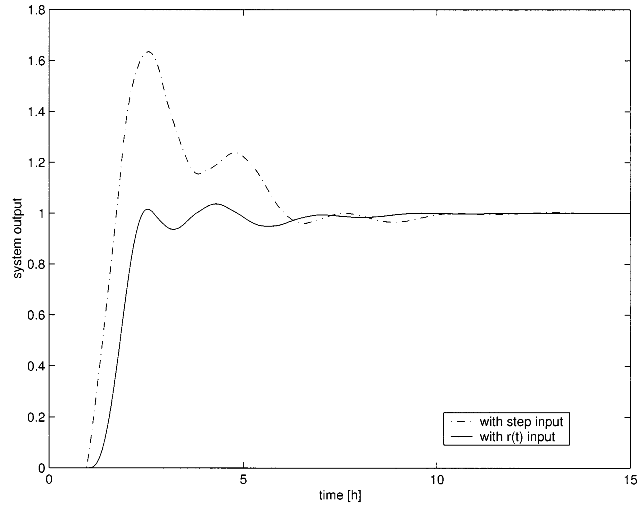

Figure 10. Process outputs for the example with $P_3(s)$.

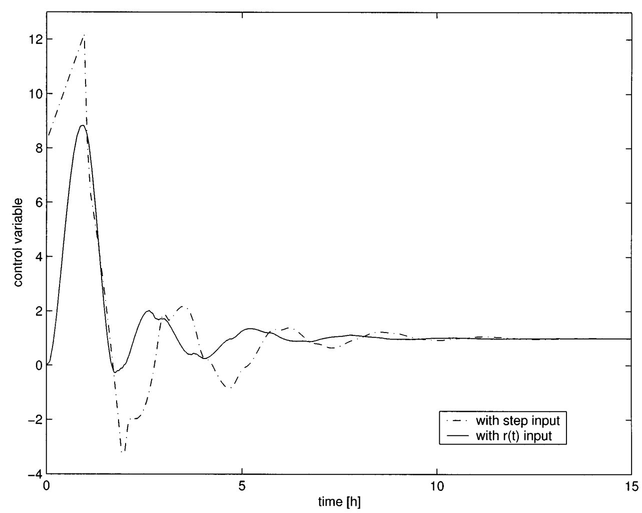

Figure 11. Control variables for the example with $P_2(s)$.

output samples are employed to model the closed-loop system. By again fixing $\epsilon = 0.01$, we obtain $k = 13$, and therefore, the resulting matrix $\hat{\mathbf{G}}$ dimensions are 113 $\times$ 113. Selecting $\tau = 1.6$ h, the command input function $r(t)$ shown in Figure 9 is determined. The process outputs obtained with both the standard and the new approach are plotted in Figure 10, and the corresponding control variables are plotted in Figure 11. The results are summarized in Table 3. It appears that, also in this case, the dynamic-inversion-based approach outperforms the one with the step signal.

3.4. Effects of Measurement Noise. To evaluate the effects of measurement noise on the performance obtained with the proposed dynamic-inversion-based technique, we again considered process $P_2(s)$ (see eq 8) with the output corrupted by white noise. In fact, in the simulations, a random number in the interval [−0.1, 0.1] was added to the process output at each sample time. The process output signal was filtered using a sixth-order Butterworth filter with a cutoff frequency of 0.06 Hz. (Note that the sampling time is 0.5 s in this case.) Then, NB was set to 0.1, and the same procedure as in the noise-free case was applied. The resulting command input signal is plotted in Figure 12, and the corresponding output is shown in Figure 13. It can be observed that measurement noise does not prevent the achievement of high performance, provided that appropriate

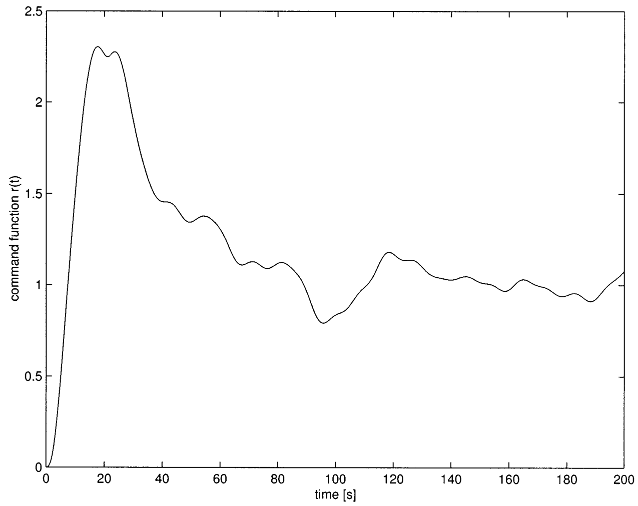

Figure 12. Command function for the example with $P_2(s)$ (in the presence of measurement noise).

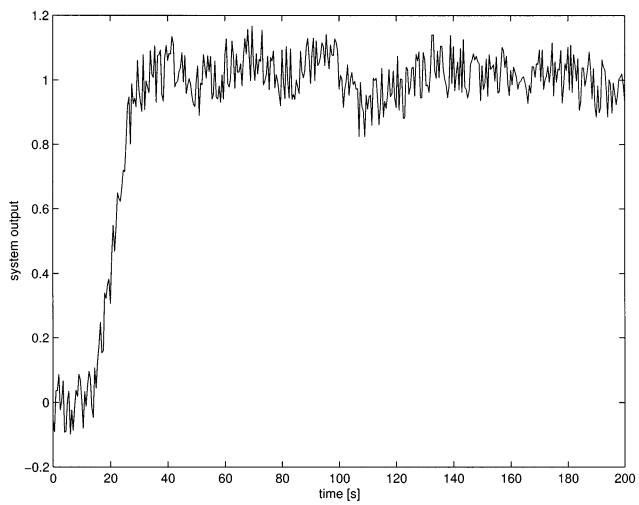

Figure 13. Process outputs for the example with $P_2(s)$ (in the presence of measurement noise).

Table 3. Summary of the Results in the Example with Process $P_3(s)$

	O (%)	T_r (h)	T_s (h)	IAE
step input	63	0.6	7.05	2.66
dynamic-inversion-based input	4	0.8	6.45	1.97

filtering is applied to the process output signal before the devised algorithm is employed.

4. Conclusions

In this paper, a dynamic-inversion-based methodology for improving the set-point-following performance of a control scheme has been proposed. The methodology relies on a fast and simple algorithm that is based on the calculation of the inverse of a matrix. Actually, because of its features, the method appears to be suitable for application as an add-on functionality in single-station industrial controllers and also in distributed control systems, as the performance of many control loops can be significantly improved by applying the determined command functions instead of the step signals without the need to retune the controllers.

Acknowledgment

This work was partially supported by MIUR scientific research funds.

Literature Cited

(1) Piazzi, A.; Visioli, A. Optimal inversion-based control for the set-point regulation of nonminimum-phase uncertain scalar systems. *IEEE Trans. Autom. Control* **2001**, *46*, 1654.

(2) Camacho, E. F.; Bordons, C. *Model Predictive Control*; Springer-Verlag: London, 1998.

(3) Piazzi, A.; Visioli, A. Optimal noncausal set-point regulation of scalar systems. *Automatica* **2001**, *37*, 121.

(4) Aström, K. J.; Wittenmark, B. *Computer-Controlled Systems: Theory and Design*; Prentice Hall: Upper Saddle River, NJ, 1997.

(5) Leva, A.; Cox, C.; Ruano, A. *Hands-on PID autotuning: A guide to better utilisation*; IFAC Professional Brief; Amsterdam, The Netherlands, 2001.

(6) Hägglund, T.; Aström, K. J. Supervision of adaptive control algorithms. *Automatica* **2000**, *36*, 1171.

(7) Press, W. H.; Teukolsky, S. A.; Vetterling, W. T.; Flannery, B. P. *Numerical Recipes: The Art of Scientific Computing*; Cambridge University Press: Cambridge, 1995.

(8) Shinskey, F. G. *Process Control Systems—Application, Design, and Tuning*; McGraw-Hill: New York, 1996.

(9) Aström, K. J.; Hägglund, T.; Hang, C. C.; Ho, W. K. Automatic tuning and adaptation for PID controllers—A survey. *Control Eng. Pract.* **1993**, *1*, 699.

(10) Shinskey, F. G. PID−deadtime control of distributed processes. Presented at the IFAC Workshop on Digital Control PID'00, Terrassa (E), Terrassa, Spain, Apr 5−7, 2000.

Received for review September 18, 2002
Revised manuscript received January 15, 2003
Accepted February 10, 2003

IE020734F